Detecting ICMP Rate Limiting in the Internet

USC/ISI Technical Report ISI-TR-717 April 2017

Hang Guo (hangguo@isi.edu) John Heidemann (johnh@isi.edu)

***Abstract*—Active probing with ICMP is the center of many network measurements, with tools like ping, traceroute, and their derivatives used to map topologies and as a precursor for security scanning. However, rate limiting of ICMP traffic has long been a concern, since undetected rate limiting to ICMP could distort measurements, silently creating false conclusions. To settle this concern, we look systematically for ICMP rate limiting in the Internet. We develop a model for how rate limiting affects probing, validate it through controlled testbed experiments, and create *FADER*, a new algorithm that can identify rate limiting from user-side traces with minimal requirements for new measurement traffic. We validate the accuracy of FADER with many different network configurations in testbed experiments and show that it almost always detects rate limiting. Accuracy is perfect when measurement probing ranges from 0 to $60\times$ the rate limit, and almost perfect (95%) with up to 20% packet loss. The worst case for detection is when probing is very fast and blocks are very sparse, but even there accuracy remains good (measurements $60\times$ the rate limit of a 10% responsive block is correct 65% of the time). With this confidence, we apply our algorithm to a random sample of whole Internet, showing that *rate limiting exists* but that *for slow probing rates, rate-limiting is very, very rare*. For our random sample of 40,493 /24 blocks (about 2% of the responsive space), we confirm 6 blocks (0.02%!) see rate limiting at 0.39 packets/s per block. We look at higher rates in public datasets and suggest that fall-off in responses as rates approach 1 packet/s per /24 block (14M packets/s from the prober to the whole Internet), is consistent with rate limiting. We also show that even very slow probing (0.0001 packet/s) can encounter rate limiting of NACKs that are concentrated at a single router near the prober.**

I. Introduction

Active probing with pings and traceroutes (ICMP echo requests) are often the first tool network operators turn to to assess problems, and widely used tools in network research [11], [12], [17], [19], [21]. Studies of Internet address usage [11], [7], [29], [20], [13], path performance [17], outages [21], [25], Carrier-Grade NAT deployment [24] DHCP churn [19] and topology [12], [26], [6], [16], [18] all depend on ICMP.

An ongoing concern about active probing is that network administrators rate limit ICMP. If widespread, rate limiting could easily distort measurements, possibly silently corrupting results. Researchers try to avoid rate limiting by intentionally probing slowly and selecting targets in a pseudo-random order [11], [14], but recent work has emphasized probing as quickly as possible [8]. For IPv4 scanning, the Internet Census (2008) sends 1.5k probe/s [11], IRLscanner (2010) sends 22.1k probe/s [15], ZMap (2013) sends 1.44M probes/s [8], or 14M probes/s in their latest revision [5]. Assuming about 3 billion target addresses and pseudorandom probing, these rates imply a probe arrives at a router handling a given /16 every 0.003 to 30 seconds. Interest in faster probing makes rate limit detection a necessary part of measurement, since undetected rate limiting can silently distort results.

Although rate limiting is a concern to active probing and has been studied briefly in some papers that consider active probing [26], [11], [10], [6], [13], we know only two prior study explicitly looking for rate limiting in the general Internet [23], [9]. The work from Universite Nice Sophia Antipolis detect and characterize rate limit to ICMP Time exceeded replies in response to expired ICMP echo requests. [23]. However their detection is expensive, requiring hundreds of vantage points and 17 probing rates to cover 850 routers in the Internet. More importantly, they never look at rate limiting to ICMP echo requests on forward path. Google studied traffic policing of TCP protocol from server side traces [9], Their detection depended on server-side traffic analysis of billions of packets in Google's CDN. Like those prior works, we want to study rate limiting of ICMP in global scale, but our goal is to do so in a lightweight manner that does not require intensive traffic probing or extensive sever-side data. Lightweight methods to detect rate limiting will help researchers by preventing their results from being distorted silently, while not adding too much extra complexity and cost to their research.

Our first contribution is to provide a new lightweight algorithm to detect ICMP rate-limiting and estimate rate limit across the Internet. Our approach is based on two insights about how rate-limiting affects traffic: first, a rate-limiting will cause probe loss that is visible when we compare slower scans with faster scans, and second, this probe loss is randomized. As a result, we can analyze two ICMP scans taken at different rates to identify rate limiting at any rate less than the faster scan.

Our second contribution is to re-examine existing

public data for signs of ICMP rate limiting in the whole Internet. We examine two different groups of data. First, we use random samples of about 40k /24 blocks to show that *ICMP Rate limiting is very rare in the general Internet* for rates up to 0.39 packets/s per /24: only about 1 in 10,000 /24 blocks are actually rate limited. Second, we look at higher rate scans (up to 0.97 packets/s) and shows the fall-off of responses in higher probing rates is consistent with rate limits at rates from 0.28 to 0.97 packets/s per /24 in parts of the Internet. Finally, although low-rate scans do not usually trigger rate limiting, we show that rate limiting explains results for error replies when Internet censuses cover non-routed address space.

II. Problem Statement

Rate limiting is a facility provided in all routers to allow network administrators to control access to their networks. In most routers, rate limiting can be configured in several ways. Administrators may do *traffic policing*, limiting inbound ICMP (or TCP or UDP) to prevent Denial-of-Service (DoS) attacks against internal networks. Routers also often *rate-limit generate of ICMP error messages* (ICMP types 3 and 11, called here ICMP NACKs) to prevent use of the router to attack others (an attacker generate a stream ICMP NACK-generating traffic, spoofing a victim's address to amplify the attack's effects). ICMP rate limiting in either direction matters to researchers. Limits on the forward path affect address usage and outage studies [11], [23], while limits on the reverse path affect studies that use traceroute-like mechanisms [12], [26].

When a rate limit is reached, the router can simply drop packets over the limit, or it can generate an error reply (ICMP type 3). Most routers simply drop traffic over the rate limit, but Linux IP tables can also generate NACKs [4]. Dropping traffic over the rate limit matches the typical goal of protecting the network from excessive traffic, since generating NACKs adds more traffic to the network.

Our paper develops FADER (Frequent Alternation Availability Difference ratE limit detector), an algorithm that can detect and estimate rate limits in the forward path. Our goal is to estimate rate limits while minimizing network traffic and infrastructure: our approach works from a single vantage point, and requires two scans at different rates, detecting rate limits that take any value between those rates. This goal is challenging for two reasons: First, the amount of information conveyed through two-rate probing from single vantage point is very limited. Second, active probing data can be distorted by potential events at target IP blocks like DHCP [19], diurnal [22] and outages [21].

III. Modeling Rate Limited Blocks

Our detection algorithm is based on models of rate limiting in commercial routers.

A. Rate Limit Implementations in Commercial Routers

We examined router manuals and two different router implementations; most routers, including those from Cisco [1] and Juniper [2], implement ICMP rate limiting with some variation on a token bucket.

With a token bucket, tokens accumulate in a "bucket" of size B tokens at a rate of L tokens/s. When the bucket size is exceeded, extra tokens are discarded. When a packet arrives, it consumes one token and is forwarded, or the packet is discarded if the token bucket is empty (we assume 1 token per packet, although one can use tokens per bytes). Ideally (assuming smooth traffic), for incoming traffic at rate P packets/s, if $P < L$, the traffic is below rate limit and will be passed by token bucket without loss. When $P > L$, initially all packets will be passed as the bucket drains, then packet loss and transmission will alternate as packets and tokens arrive and are consumed. In the long run, when $P > L$, egress traffic exits at rate L packets/s.

B. Modeling Availability

We assume a block of addresses is behind a rate-limited router, with all sharing a common IPv4 prefix of some length. When not otherwise specified, we assume blocks have /24 prefixes and use n_B to represent number of IP in a /24 block: 256. We first model the *availability* of that block—the fraction of IPs in target block that respond positively to our probing. We consider both the *true* availability (A), ignoring rate limiting, and also the *observed* availability ($\hat{A}$) as affected by rate limiting.

Two observations help model availability. From subsection III-A, recall that L packet/s pass (the rate imit), when P packet/s are presented to the token bucket. Therefore L/P is the proportion of probes that are passed. Second, if N IPs in target block are responsive, a non-rate-limited ping hits a responsive IP that replies with probability N/n_B. Putting above two observations together gives us the model of rate limited block's availability:

$$A = \frac{N}{n_B} \quad and \quad \hat{A} = \begin{cases} A(L/P), & \text{if } P > L \\ A, & \text{otherwise} \end{cases} \tag{1}$$

C. Modeling Response Rate

Response rate is the positive responses we receive from target block per second. In our model Equation 2, we consider both the *true* value (R) ignoring rate limit and the *observed* value ($\hat{R}$) affected by rate limit.

$$R = \frac{N}{n_B} P \quad and \quad \hat{R} = \begin{cases} R(L/P), & \text{if } P > L \\ R, & \text{otherwise} \end{cases} \tag{2}$$

D. Modeling Alternation Count

Response Alternation is defined as the transition of an address from responsive to non-responsive or the other way around. For instance, if a target IP responds positively twice in a row, then omits a response, then responds again, it alternates responses twice (from responsive to non-responsive and back). Response alternation is important to distinguish between rate limits and other sources of packet loss—rate limits cause *frequent alternation* between periods of packet response and drops as the token bucket fills. Frequent alternation helps distinguish rate limiting from networks outages, since outages are long lives while rate-limits show as intermittent failures. Frequent alternation is, however, less effective in distinguishing rate limiting from transient network congestion because congestion losses are randomized (mostly due to our probes are randomized) and create frequent alternation. An extra re-probing could ensure the detection result are robust against potential false positives caused by transient network congestion.

We model the count of observed response alternations, $\hat{C}$, both accurately and approximately. The accurate model fits measured values very precisely over all values of P/L, but requires enumerating the probabilities and response alternation counts (represented by p_n and c_n respectively in the model) for all 2^r possible cases for r rounds of observation. Since r is quite large for full datasets we study in section V (our data has $r = 1800$ iterations), this enumeration is not computable. The accurate model is:

$$\hat{C} = \begin{cases} N \sum_{n=1}^{2^r} p_n c_n, & \text{if } P > L \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

for a rate limit L, probing rate P and N ideal responsive IPs.

The approximate model, on the other hand, provides single expression covering all r but fits only when $P \gg L$ (so that consecutive packets from same sender are never passed by token bucket). We use it in our evaluation since it is computable when $r = 1800$. It is:

$$\hat{C} = 2(L/P)Nr, \quad \text{when } P \gg L \tag{4}$$

IV. Detecting Rate Limited Blocks

The models (section III) assist our detection algorithm.

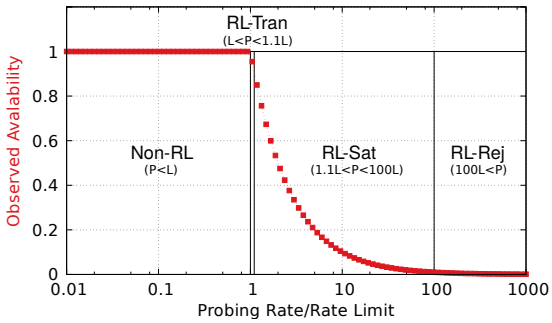


Figure 1: Four phases of ICMP rate limiting, with $\hat{A}$ as a function of P/L.

A. Input for Detection

Our detection algorithm requires *low-* and *high-rate* measurements as input.

Low-rate measurements must be slower than any rate limit that are detected. They therefore capture the true availability (A) of target blocks. While they require that we guess the range of possible rate limits, we observe that most routers have a minimum value for rate limits (for example, 640 kb/s for Cisco ME3600-A [1]), and we can easily be well under this limit. An example of a suitable low-rate scan is the ISI Internet censuses at 0.0001 pings/s per /24 block [11].

High-rate measurements must exceed the target rate limit. It is more difficult to guarantee we exceed rate limits because we do not want measurements to harm target blocks with too much traffic. However, selecting some high rate allows us to detect or rule out rate limiting up to that rate; the high-rate sets the upper bound for FADER's detection range.

In addition, high-rate measurements must be repeated to use alternation detection portion of our algorithm Algorithm 1. Validation shows that 6 repetitions is sufficient for alternation detection (subsection VI-A), but our existing datasets include as many as 1800 receptions, and our algorithms can detect rate-limiting candidates without alternation count, although with a high false-positive rate. An example high-rate measurement is the ISI Internet surveys that repeated scan many blocks about 1800 times at 0.39 pings/s per /24 block.

Both low- and high-rate measurements need to last a multiple of 24 hours to account for regular diurnal variations in address usage [22].

B. Four Phases of ICMP Rate Limiting

The models from section III allow us to classify the effects of ICMP rate limiting on block availability into four phases (Figure 1). Defined by the relationship between probing rate P and rate limit L, these phases guide our detection algorithm:

1) Non-RL (Non-Rate-Limited): when $P < L$, before rate limiting has any effect,
2) RL-Tran (Rate Limit Transition): when $L < P < 1.1L$, rate limiting begins to reduce the response rate with alternating responses. $\hat{A}$ starts to fall but is not distinguishing enough for detecting.
3) RL-Sat (Rate Limit Saturation): when $1.1L < P < 100L$, rate limiting and frequent alternation are common, and $\hat{A}$ falls significantly.
4) RL-Rej (Rate Limit Rejection): when $P > 100L$ occurring at threshold $T_{rej} = P/L = 100$. Here $\hat{A} < 0.01N/n_B$, most packets are dropped and response alternations become rare.

These phases also help identify regions where no algorithm can work: rate limits right at the probing rate, or far above it. For rate limits that happen to lie in the RL-Tran Phase ($L < P < 1.1L$)—here there is not enough change in response for us to identify rate limiting over normal packet loss. Fortunately, this region is narrow. In addition, no algorithm can detect rate limits in RL-Rej phase because such rate limited block will look dark ($\hat{A} < 0.01N/n_B$) and give too little information (at most one response for every one hundred probes sent) for anyone to know if it is a heavily rate limited block or a non-rate-limited gone-dark block.

In subsection VI-B we show that our algorithm is correct in the remaining large regions (Non-RL and RL-Sat), provided $P < 60L$.

C. Detecting Rate Limited Blocks

FADER is inspired by observations that the RL-tran phase is narrow, but we can easily tell the difference between the non-RL and RL-Sat phases. Instead of trying to probe at many rates, we probe at a slow and fast rate, with the hope that the slow probes land in the non-RL phase and with the goal of bracketing the RL-Tran phase. If the target block shows much higher availability in slow probing, we consider the block a rate limit *candidate* and examine if its traffic pattern look like rate limiting: consistent and randomized packet dropping and passing.

We first introduce *Frequent Alternation Test* Algorithm 1 to distinguish rate limiting from other types of packet loss.

This subroutine identifies the consistent and randomized packet dropping caused by rate limiting (by looking for large number of responses alternations). Threshold $(2\hat{N}_L r)/T_{rej}$ is derived from our approximate alternation count model Equation 4. As low-rate measurement is assumed non-rate-limited, we have $\hat{N}_L$ (number of responsive IPs observed in low-rate measurement) $= N$ (the number of responsive IP when non-rate-limited). Recall that we do not try to detect rate limits in RL-Rej phase (subsection IV-B), we have $P < T_{rej}L$. Substituting both into alternation count model gives us, for a rate limited block, there must be at least $(2\hat{N}_L r)/T_{rej}$ response alternations.

Algorithm 1 Frequent Alternation Test

Input:
$\hat{C}$: observed response alternation count in high-rate measurement
r: number of probing rounds in high-rate measurement
$\hat{N}_L$: number of responsive IPs observed in low-rate measurement
$\hat{N}_H$: number of responsive IPs observed in each round of high-rate measurement (where responsive IPs observed at ith round is $\hat{N}_{H_i}$)
Output:
O_{fat}: results of frequent alternation test

```
1: if Ĉ > (2N̂_L r)/T_rej and NotDirTmpDn(N̂_H, N̂_L, r) then
2:     O_fat ← Passed   /* has frequent alternations */
3: else
4:     O_fat ← Failed   /* no frequent alternations */
5: end if

6: function NotDirTmpDn(N̂_H, N̂_L, r)
7:     for i = 1 to r do
8:         if N̂_{H_i} ≥ N̂_L then
9:             return false
10:        end if
11:    end for
12:    return true
13: end function
```

Function NotDirTmpDn filters out diurnal and temporarily down blocks, which otherwise may be misinterpreted (false positives) because their addresses also alternate between responsive and non-responsive. NotDirTmpDn checks if any round of the high-rate measurement looks like the daytime (active period) of diurnal block or the up-time of temporarily down blocks, satisfying $\hat{N}_{H_i} = \hat{N}_L$ or even $\hat{N}_{H_i} > \hat{N}_L$

Next, we describe our detection algorithm FADER (Algorithm 2).

FADER first detects if target block is rate-limited (line 1-19), producing "cannot tell" for blocks that are non-responsive when probed or respond too little to judge. No active measurement system can judge the status of non-responsive blocks; mark such block as cannot-tell rather than misclassifying them as rate limited or not. In our experiments we see cannot-tell rates of 65% when probing rate is 100 times faster than rate limit and in average only 2.56 IPs respond in each target block Figure 11a; these rates reflect the fundamental limit of any active probing algorithm in an Internet with firewalls, rather than a limit specific to our algorithm.

Once target block is detected as rate limited, FADER estimates its rate limit (line 20-22). Note that threshold $\hat{N}_L < 10$ used in line 3 is empirical, but chosen because very sparse blocks provide too little information to reach

Algorithm 2 FADER

Input:
$\hat{A}_L$: measured block availability in low-rate measurement
$\hat{A}_H$: measured block availability in high-rate measurement
$\hat{N_L}$: number of responsive IPs in low-rate measurement
T_{rej}: lower bound of RL-Rej phase
O_{fat}: result of frequent alternation test
Output:
O_{fader}: detection result of FADER
$\hat{L}$: estimated rate limit (if detect rate limit)

1: **if** $\hat{A}_L = 0$ **or** $\hat{A}_H = 0$ **then** /* target block down */
2: | $O_{fader} \leftarrow$ Can-Not-Tell
3: **else if** $\hat{N_L} < 10$ **then** /* block barely responsive */
4: | $O_{fader} \leftarrow$ Can-Not-Tell
/* if significant lower availability in faster probing */
5: **else if** $(\hat{A}_L - \hat{A}_H)/\hat{A}_L > 0.1$ **then**
6: | **if** $\hat{A}_H/\hat{A}_L < 1/T_{rej}$ **then** /* RL-Rej phase */
7: | | $O_{fader} \leftarrow$ Can-Not-Tell
8: | **else**
9: | | **if** $O_{fat} =$ Passed **then**
10: | | | $O_{fader} \leftarrow$ Rate-Limited
11: | | **else** /* no frequent alternations (most */
12: | | | /* likely target block temp down) */
13: | | | $O_{fader} \leftarrow$ Can-Not-Tell
14: | | **end if**
15: | **end if**
/* if no significant availability drop in faster probing */
16: **else if** $0.1 > (\hat{A}_L - \hat{A}_H)/\hat{A}_L > -0.1$ **then**
17: | $O_{fader} \leftarrow$ Not-Rate-Limited
18: **else** /* $-0.1 > (\hat{A}_L - \hat{A}_H)/\hat{A}_L$ */
19: | $O_{fader} \leftarrow$ Not-Rate-Limited
20: **end if**
/* estimate rate limit if detected */
21: **if** $O_{fader} =$ Rate-Limited **then**
22: | $\hat{L} \leftarrow \frac{n_B \hat{A}_H P_H}{\hat{N_L}}$
23: **end if**

definite conclusions. Test $(\hat{A}_L - \hat{A}_H)/\hat{A}_L > 0.1$ in line 5 is derived by substituting $P > 1.1L$,the lower bound of RL-Sat phase (recall that we intentionally give up detecting rate limits in RL-Tran Phase $L < P < 1.1L$), into availability model Equation 1. Test $\hat{A}_H/\hat{A}_L < 1/T_{rej}$ (line 6) is derived by substituting $P > T_{rej}L$ (the threshold that we give up detection), into availability model Equation 1. Estimating the rate limit (line 21) inverts our availability model (Equation 1).

V. Results: Rate Limiting in the Wild

We next apply FADER to existing public Internet scan datasets to learn about ICMP rate limiting in the Internet. (We validate the algorithm later in section VI.)

blocks studied	40,493	(100%)	
not-rate limited	24,414	(60%)	
cannot tell	15,941	(39%)	
rate limited	111	(0.27%)	*(100%)*
false positives	105	(0.25%)	*(95%)*
true positives	6	(0.015%)	*(5%)*

Table II: Application of FADER to it71w census and survey.

A. How Many Blocks are Rate Limited in the Internet?

We first apply FADER to find rate limited blocks in the Internet, confirming what we find with additional probing.

Input data: Rather than do new probing, we use existing Internet censuses and surveys as test data [11]. Reusing existing data places less stress on other networks and it allows us to confirm our results at different times subsection V-B. Table I lists the datasets we use in result section and they are available publicly [27], [28].

Censuses and surveys define the low- and high-rates that bound rate limits detected by our algorithm. A census scans at at 0.0001 pings/s per block, while surveys send 0.39 pings/s per block. These two rates provide the low- and high-rates to test our algorithm. We could rerun FADER with higher rates to test other upper bounds; we report on existing higher rate scans in subsection V-C.

Both censuses and surveys are intentionally slow and randomized, to spread out load on the Internet and avoid abuse complaints. Surveys cover about 40k blocks (30k of which are randomly selected and 10k randomly selected drawn from specific levels of responsiveness), and they probe those blocks about 1800 times over two weeks, supporting Frequent Alternation detection. With a 2% of the responsive IPv4 address space, randomly chosen, our data provides a representative of the Internet. Censuses cover almost the entire unicast IPv4 Internet, but we use only the part that overlaps the survey.

Initial Results: Here we apply FADER to it71w, the latest census and survey datasets, in Table II. We find that *most blocks are not rate limited* (60%), while a good number (39%) are cannot tell, usually because they are barely responsive and so provide little information for detection. However, *our algorithm classifies a few blocks* (111 blocks, 0.27%) as apparently rate limited.

Validation with additional probing: To confirm our results, we next re-examine these likely rate-limited blocks, We re-probe each block, varying probing rates from 0.01 to 20 ping/s per block to confirm the actual rate limiting. Our additional probing is relatively soon (one month) after our overall scan; it is unlikely that many network blocks changed use in that short time.

Figure 2 shows this confirmation process for one example block. Others are similar. In these graphs,

Start Date (Duration)	**Size** (/24 blocks)	**Alias**	**Full Name**
2016-08-03 (32 days)	14,460,160	it71w census	`internet_address_census_it71w-20160803`
2016-08-03 (14 days)	40,493	it71w survey	`internet_address_survey_reprobing_it71w-20160803`
2016-06-02 (32 days)	14,476,544	it70w census	`internet_address_census_it70w-20160602`
2016-06-02 (14 days)	40,493	it70w survey	`internet_address_survey_reprobing_it70w-20160602`
2013-09-17 (32 days)	14,477,824	it56j census	`internet_address_census_it56j-20130917`
2013-11-27 (33 days)	14,476,544	it57j census	`internet_address_census_it57j-20131127`
2014-01-22 (29 days)	14,472,192	it58j census	`internet_address_census_it58j-20140122`

Table I: Datasets used in this paper

	Response Rate	**Availability**	**Rate Limit** (ping/s per blk)	
/24 Block	(measured, pkts/s)	($\hat{A}_L$, %)	(measured)	(estimated)
124.46.219.0	0.009	9.77	0.09	0.09
124.46.239.0	0.08	53.13	0.15	0.12
182.237.200.0	0.06	58.98	0.10	0.12
182.237.212.0	0.04	27.34	0.15	0.10
182.237.217.0	0.06	49.61	0.12	0.13
202.120.61.0	0.35	17.58	1.99	0.32

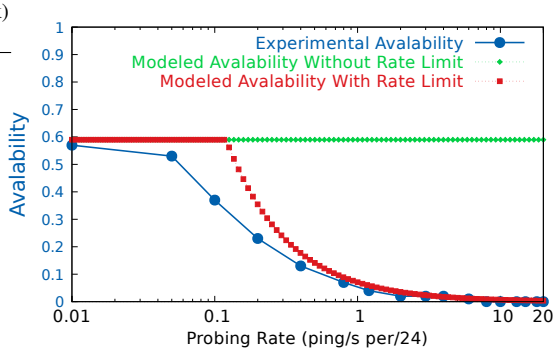
Table III: True rate limited blocks in the it71w Census and Survey.

red squares show modeled availability and response rate, assuming the block is rate limited (given the rate limit estimation from our algorithm in Table III). Green line with diamonds show the availability and response rate if the block is not rate limited. For a rate limited block, its measured availability and response rate would match the modeled values with rate limiting. As Table III) shows, this block's measured availability (blue dots) and response rate (cyan asterisks) tightly matches the modeled value with rate limiting (red squares) while diverging from theoretical values without rate limiting (green diamonds). This data shows that this block, 182.237.200.0/24, is rate limited.

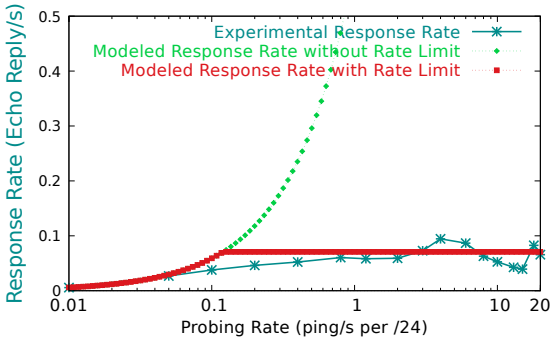
Although this example shows a positive confirmation, we find that most of the 111 blocks are false positives. Only the 6 blocks listed in Table III are indeed rate limited. These high false positive rate are by design: FADER strives to avoid missing rate-limited blocks (by using necessary conditions: significant lower availability in faster scan and frequent response alternations in fast scan, to detect rate limitings). We use additional verification to confirm true positives. Among the 6 rate limited blocks, five belong to the same ISP: Keumgang Cable Network in South Korea, while the last block is from Shanghai Jiaotong University in China. We have contacted both ISPs to confirm our findings, but they did not reply.

Our first conclusion from this result is *there are ICMP rate-limited blocks, but they are very rare*. We find only 6 blocks in 40k, less than 0.02%.

Second, we see that each of FADER's steps rule out about 95% of all the blocks entering that rule. As in Table IV, 2,088 out of 40,403 (5.2%) passed



(a) Availability



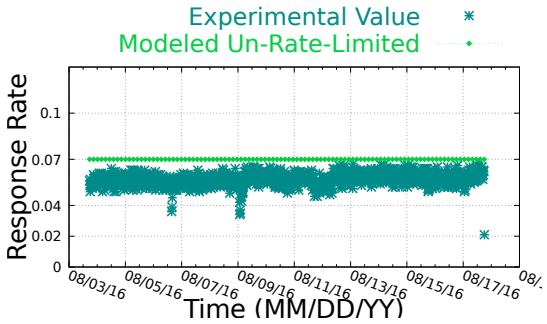
(b) Response Rate

Figure 2: Confirming block 182.237.200/24 is rate limited with additional probing.

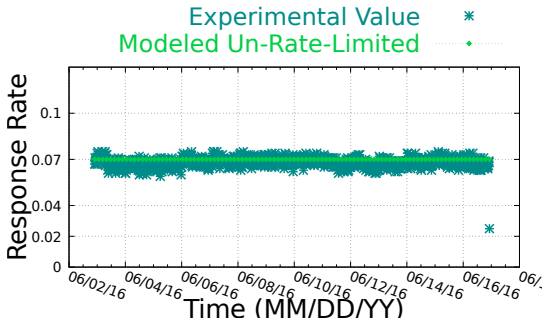
phase 1 (Availability Difference Test) and 111 out of 2,088 (5.3%) passed phase 2 (Frequent Alternation Test). However, even after two phases of filtering, there is still a fairly high false positive rate in the remaining blocks, since only 6 of 111 (5.4%) are finally confirmed as rate limited.

Finally, we show that *when* we detect rate limiting, our *estimate of the rate limit* are correct in general. Table III shows this accuracy: five out of six rate limits observed in re-probing closely match FADER's estimates.

However our rate limit estimation (0.32 ping/s per block) for block 202.120.61/24 is 5 times smaller than the rate limit (1.99 pings/s) that we observe when we re-probe. (We compute the rate limit when re-probing by measuring $\hat{R}$ (the response rate), $\hat{A}_L$ and inverting



(a) It71w Survey



(b) It70w Survey

Figure 3: Response rate (reply/s) of 202.120.61/24: measured every 1000 s (cyan asterisks), compared to expected when not rate-limited (green diamonds)

Test Name	Number of Blocks (Ratio)		
	Input	Passed	Filtered
Availability Difference	40,403	2,088 (5.2%)	38,315 (94.8%)
Frequent Alternation	2,088	111 (5.3%)	1,977 (94.7%)
Re-probing Validation	111	5 (4.5%)	106 (95.5%)

Table IV: Results of rate-limit detection on it71w.

our response-rate model Equation 2.) When we review the raw data, we believe that the rate limit for this block *changed* between our measurements.

B. Verifying Results Hold Over Time

To verify our approach works on other datasets, we also apply FADER to it70w census and survey data. This data is taken two months before it71w and sharing 76% of the same target blocks. Detection results of it70w data agrees with our previous conclusion, resulting in about the same number of blocks identified as rate limited (0.3%, 138 of 40,493) , and the same fraction as actually limited (0.012%, 5). Of blocks that we confirm as rate limited after re-probing, four also are detected and confirmed in it71w. The fifth, 213.103.246/24, is from ISP Swipnet of Republic of Lithuania and is not probed in it71w.

We observe inconsistencies between it70w and it71w for two blocks, 124.46.219/24 and 202.120.61/24. These blocks are detected as rate limited blocks in it71w, but are classified as Can-Not-Tell and Not-Rate-Limited respectively in it70w. We believe one block is hard to measure, and the other actually changed it use between the measurements. Block 124.46.219/24 is only sparsely responsive, with only 25 addresses responding (9.8%). Most of our probes to this block go to non-responding addresses are dropped, making it difficult to detect rate limiting (as shown in subsection VI-D). For the 202.120.61/24 block, we believe it is *not* rate limited in it70w even though it is in it71w. Figure 3b shows its response in the it70w survey; this measured response-rate closely matches what we expect without rate limiting. In comparison, measured response rate in it71w is strictly below expected value without rate limiting as shown in Figure 3a, matching our expectation of reduced response rate under rate limiting.

C. Is Faster Probing Rate Limited?

Having shown that rate-limited blocks are very rare, at least when considering rates up to 0.39 packets/s, we next evaluate if *faster* probing shows signs of rate limiting. We study ZMap TCP-SYN probing datasets from 0.1M to 14M packet/s [5] (0.007 to 0.97 packets/s per /24 block as estimated in subsubsection V-C2) and show rate limiting could explain the drop-off in response they see at higher rates.

ZMap achieves an probing rate of 14.23M packets/s allowing a full scan of IPv4 in about 4 minutes [5]. To evaluate these very fast rates, they perform a series of 50-second experiments from 0.1M to 14M packets/s Each experiment targets a different random sample of an IP pool of about 3.7 billion addresses. Their probing results show overall availability (the fraction of positive responses of all hosts that are probed) is roughly stable for probing rates up to 4M packets/s. However, when probing rates exceed 4M packets/s, the availability starts to decline linearly as the probing increases. At 14M packets/s they see availability that is only 67% of the availability of measurements at 4M packet/s. They state that they do not know the exact reason for this decline. Figure 4, which is a copy of Figure 2 in ZMap paper [5], visualize this linear availability drop from 4M to 14M packets/s.

We believe the cause of this drop is rate limiting—once rate limits are exceeded, as the packet rate increases, availability drops. We also believe that there are roughly the same amount of rate limiting at each packet rate between 4M and 14M packets/s (0.28 to 0.97 packets/s per /24 as estimated in subsubsection V-C2) in the Internet, causing the overall availability drop to be linear.

We would like to apply FADER to ZMap probing results. Unfortunately we cannot because there is no way to recover the exact IPs that are probed in each

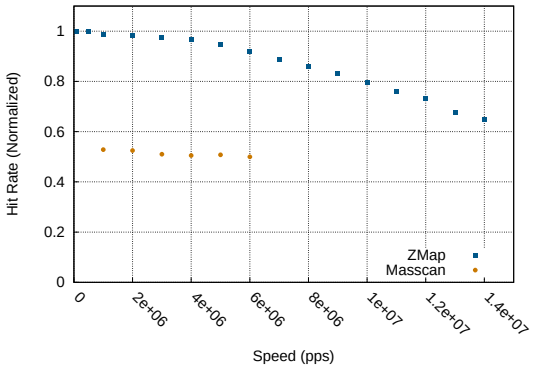


Figure 4: The Original ZMap 50-Second Experiment Availability (Hit-Rate) Chart from Figure 2 in ZMap paper [5]

experiment, so we cannot compare observed availability against actual (they probe in a pseudorandom sequence, but do not preserve the seed of this sequence). In addition, each of their experiments is a one-time run, so we can not look for response alternation.

Instead, we create a model of their measurement process and show rate limiting can cause the same drop in availability as the probe rate increases subsubsection V-C1. We show availability of many ZMap target blocks match our expectation of availability of rate limited blocks by statistically estimating the number of IPs probed in each block and the block availability subsubsection V-C2

1) Rate Limiting Can Explain ZMap Probing Drop-Off: To support our hypothesis that rate limiting is the cause of the linear drop in availability in ZMap probing results, we create a model of their measurement process and show rate limiting can cause the same probing results.

Our model, shown in Figure 5, simulates the whole measurement process of ZMap 50-second experiments. More specifically, we model a simplified network topology that just captures rate limiting and the traffic generated by ZMap 50-second experiments.

In our modeled network topology, there is one prober and 100 rate-limiting routers whose rate limits are 40k ($4M/100$) packets/s, 41k ($4.1M/100$) packets/s, all the way up to 139k ($13.9M/100$) packets/s. We use 100 routers, each with a rate limit 1k packets/s faster than the one before, to match our assumption that there are same amount of rate limiting at every probing rate from 4M to 14M packets/s. Each router covers roughly 1/100th of the 3.7 billion IP addresses 1.08% of which are responsive to TCP-SYN probing. (1.08% is the availability before linear drop in ZMap data) In each experiment, prober probes a random sample of this 3.7 billion IP pool with P packets/s for 50 seconds. As a consequence, there are roughly $50/100P$ target IPs behind each router in each experiment.

It is reasonable to assume at least one probe is sent to each target /24 block because the set of ZMap experiments we care about (from 4M to 14M packets/s) send in average 14 to 48 probes to each target /24 block as the probes are uniform random.

Availability (Figure 6a) and response rate (Figure 6b) produced by this model (red square in the charts) matches those in ZMap probing results (blue rounds in the charts) closely. The similarity of the results of our model to their experimental results suggests that our simple model can provide a plausible explanation for the packet loss observed at their fast probing. We know that our model is limited—the Internet is not 100 routers each handling 1/100th of ZMap traffic. However, it shows that it is possible to explain the fall-off in ZMap response with rate limiting. The precision of our model isn't very surprising because we set up this model to simulate ZMap data. If we could determine the exact IPs that are probed and carry out multiple passes, we could apply the complete FADER and draw a clear picture of rate limitings in fast probing.

2) Availability of ZMap Target Blocks Shows Signs of Rate Limiting: Observing that rate limits are consistent with the drops in response of ZMap at high speeds, we next apply FADER to ZMap data to look for specific blocks that appear to be rate limited.

Input data: A challenge in using ZMap data is there is no easy way to recover what specific addresses are probed in an incomplete run—we know the order is pseudo-random, but they do not provide the seed. We address this gap by statistically estimating the number of IPs probed in each block, assuming pseudorandom probes into the same 3.7 billion IPv4 pool. Assuming uniform sampling, about same number of IP will be sampled from each /16 block in the pool. (Here we look at /16 blocks instead of /24 blocks because larger blocks decrease the statistical variance.)

As a consequence, for a 50-second ZMap scan of P packets/s, approximately $50P/(3.7 \times 10^9) \times 2^{16}$ IPs are probed in each /16 block, given $50P/(3.7 \times 10^9)$ as the fraction of addresses probed in 50s, against a target 2^{16} addresses in size. We then estimate availability of each /16 block as the fraction of target IPs that respond positively to probes. We estimate probe rates by substituting 2^8 for 2^{16}, finding that ZMap probing rates are 0.007 to 0.97 packets/s per /24 block.

Initial Results: We next apply FADER to detect rate limiting. (We cannot apply Frequent Alternation Test with single-round ZMap data so we assume all blocks pass Frequent Alternation Tests.) For each ZMap target

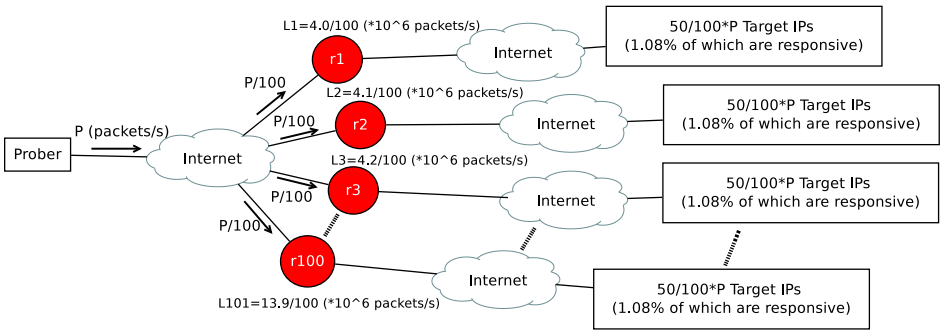
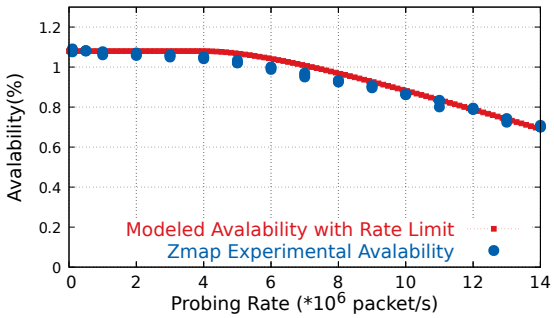
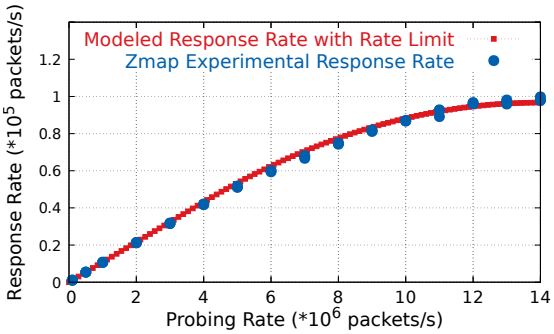


Figure 5: Rate limiting model for ZMap data



(a) Availability Comparison



(b) Response Rate comparison

Figure 6: Our modeled availability and response rate (Red) closely matches the experimental values in ZMap probing results (Blue, 3 trials at each probing rate)

block, we use slowest 50 s scan (0.1M packets/s) as the low-rate measurement and test each of the other 15 faster ZMap scans as high-rate measurement. This gives us 15 test results (each at a different high rate), for each target block. We consider a block as potentially rate limited if it is detected as rate limited in at least one test. We do not consider the other blocks (cannot tell or not-rate limited) further.

Table V shows detection results. Most ZMap target blocks (53,149 blocks, 93.99%) are cannot tell in all

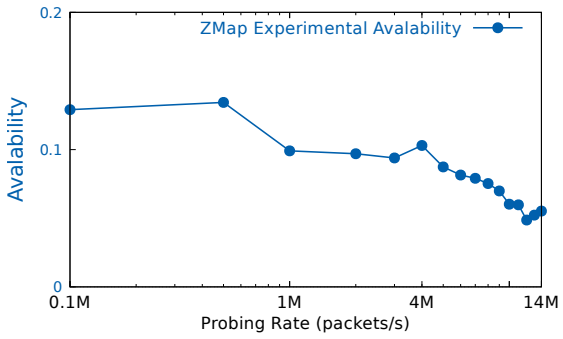
blocks studied	56,550	(100%)	
0 rate limited	53,460	(94.54%)	*(100%)*
15 cannot tell	53,149	(93.99%)	*(99.42%)*
15 not-rate limited	311	(0.55%)	*(0.58%)*
others	0	(0%)	*(0%)*
at least 1 rate limited	3,090	(5.46%)	*(100%)*
at least 13 rate limited	2,153	(3.81%)	*(69.68%)*
less than 13 rate limited	937	(1.66%)	*(30.32%)*

Table V: Applying 15 FADER Tests to Each of ZMap Target /16 Blocks

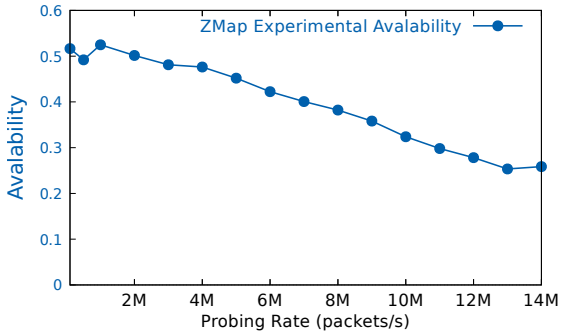
15 FADER tests (43,067 of them due to target block went completely dark during low-rate measurement and provide no information for detection). A good number of them (3,090 blocks, 5.46%) are classified as rate-limited in at least one FADER test and are thus considered potentially rate-limited. It's worth noting that most (69.68%) of those potentially rate-limited blocks are consistently classified as rate-limited in most FADER tests (at least 13 out of 16 tests), supporting our claim that those blocks are potentially rate-limited.

Confirmation with Additional Examinations: Our algorithm is optimized to avoid false negatives, so we know many of these potential rate limited blocks are false positives. Since we cannot do Frequent Alternation Tests, to further filter out false positive and false negative in detection results, we manually check a 565 (1%) random sample of 56,500 ZMap target blocks. Of these sample blocks, 31 are detected as rate-limited in at least one FADER test and are considered potentially rate-limited. Among the other 534 blocks (that are detected as rate limited in zero test and are considered cannot tell or not-rate limited), 532 are classified as cannot tell in all 15 FADER tests while 2 are classified as not-rate limited in all 15 tests.

We find the 534 cannot tell or not-rate limited blocks to be true negative. They either has almost 0 $\hat{A_L}$ or $\hat{A_H}$ (providing no information for detection) or become more available at higher probing rate (opposing our expectation of reduced availability at faster scan)



(a) 125.182/16 in Log Scale



(b) 50.62/16 in Linear Scale

Figure 7: Two ZMap Target Blocks Showing Multiple Rate Limits

Of those 31 potential rate-limited blocks, they all show reduced availability at higher probing rates, matching our expectation of rate limited blocks. We also find 7 of them appears to have more than one rate limits. For example, block 125.182/16 in Figure 7a looks like a superposition of $\hat{A}$ curves of two rate limits: one at 0.5M packets/s, the other at 4M packets/s (recall the ideal $\hat{A}$ curve a of rate limited block in Figure 1). Block 50.62/16 in Figure 7b, on the other hand, show nearly linear drops in availability as probing rates get higher, suggesting it consists of multiple rate limits (reasons are similar as in subsubsection V-C1). We manually check each /24 blocks in those two /16 blocks, and it appears that those /24 blocks indeed have multiple rate limits. This observation supports our claim that different parts of the /16 have different rate limits.

D. Rate Limiting of Response Errors at Nearby Routers

Although we have shown that probing rates up to 0.39 pings/s trigger rate limits on almost no *target* blocks, we next show that even slow probing can trigger rate limits when *traffic to many targets is aggregated at a nearby router*.

In the it57j census we see this kind of reverse-path aggregation because a router near our prober generates ICMP error messages for packets sent to unrouted IPv4 address space. We examine millions of NACK replies in this census and see they are all generated by the same router. We confirm this router is near our prober with traceroute and based on the hostname.

This router sees about 500 ping/s, about one-third of census traffic. The router was configured to generate ICMP error message (NACKs), but it had NACK rate limiting of about 80 NACK/s.

To better understand this procedure, we visualize responses from one of the target block in it57j census. Figure 8 shows block 103.163.18/24, one of the unreachable blocks behind this router. It56j census Figure 8a shows the whole block is non-responsive and so is the first half of It57j census. Figure 8b However in the middle of it57j census (16/12/2013 GMT), this router began to generate NACK feedback for probes targeting unreachable IPs. Rather than each probe drawing a NACK, we instead see a roughly constant rate of 1.64 NACK/day. Similar NACK traffic is also seen in the it59j Census Figure 8c. This response is consistent with a rate limited return path.

VI. Validation

We validate our model against real-world routers and our testbed, and our algorithm with testbed experiments.

A. Does the Model Match Real-World Implementations?

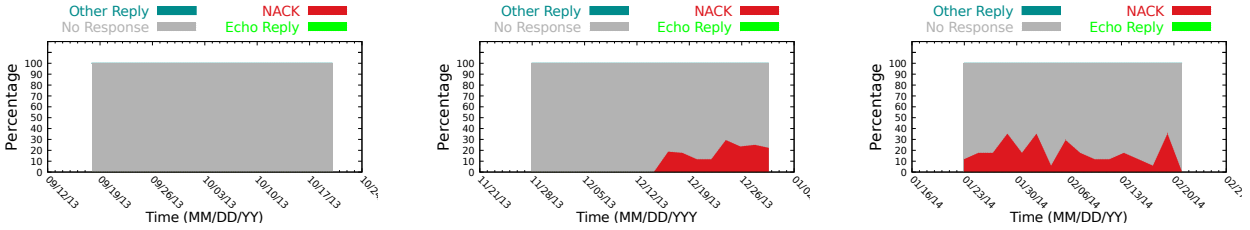
We next validate our models for availability, response alternation, and response rate of rate-limited blocks. We show they match the ICMP rate limiting implementations in two carrier-grade, commercial routers and our testbed.

Our experiments use two commercial routers (Cisco ME3600-A and Cisco 7204VXR) and one Linux box loaded with Linux filter iptables as rate limiters. Our measurement target is a fully responsive /16 block, simulated by one Linux box loaded with our customized Linux kernel [3]. In each experiment, we run a 6-round active ICMP probing from below rate limit to at most $7500\times$ the rate limit, stressing our models to extremes.

We begin with validating our *availability model* from Equation 1. Figure 9 shows model predicted availability (the red line with squares) closely matches router experiments (blue line with dots on the left graph) and testbed experiments (blue line with dots on the right graph) from below to above the rate limit.

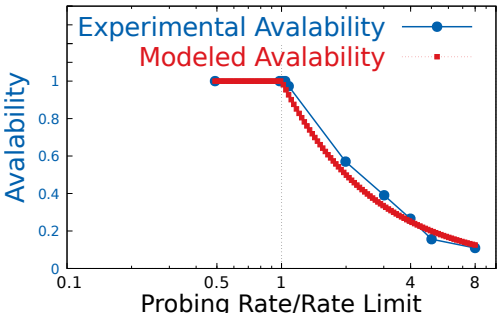
We validate our *response rate model* from Equation 2. We omit this data due to space limitations, but our response rate model is accurate from a response rate of 0.01 to $90\times$ the rate limit.

We next validate our models of alternation counts (Equation 3 and Equation 4). Figure 10a shows precise model fits perfectly from below the rate limit up to $7500\times$ the rate limit Figure 10b shows the abstract model (defined in Equation 4) fits when $P \gg L$. In our case,

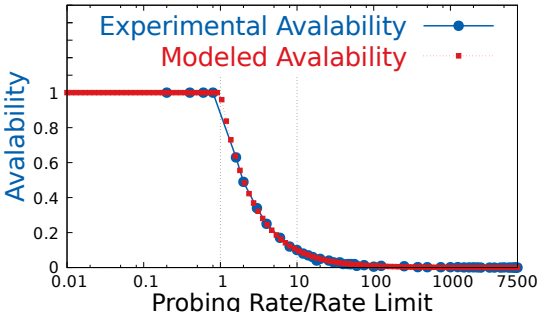


(a) It56j Census (b) It57j Census (c) It58j Census

Figure 8: Responses from 103.163.18/24 over time (every 1.85 day)

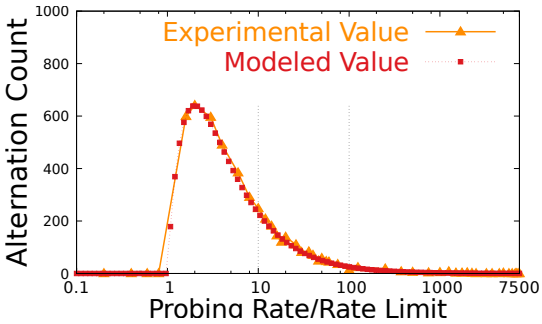


(a) Router Experiment: Up To 8 Times Rate Limit

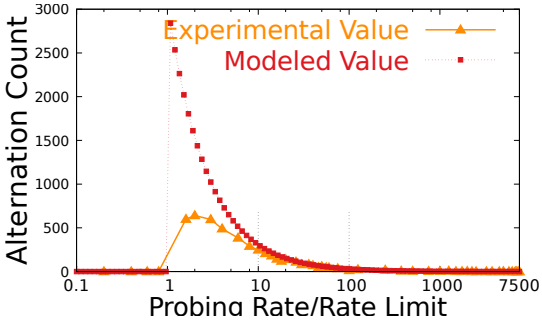


(b) Testbed Experiment: Up To 7500 Times Rate Limit

Figure 9: Validating the availability model.



(a) Precise Model



(b) Approximate Model

Figure 10: Validation of the Alternation Count Model, up to 7500× rate limit.

with 6 rounds of active probing, the approximate model fits when $P > 10L$.

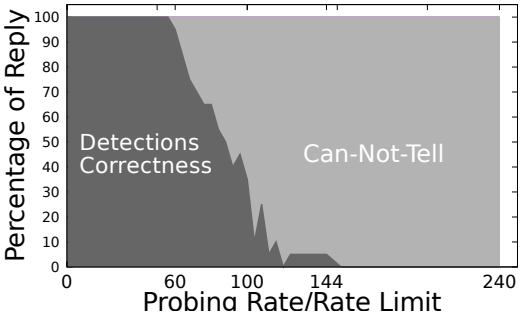
We are unable to validate alternation count model with commercial routers; the routers are only available for a limited time. But we believe testbed validations shows the correctness of our alternation counts models since we have already shown rate limiting in testbed matches that of two commercial routers.

B. Correctness in Noise-Free Testbed

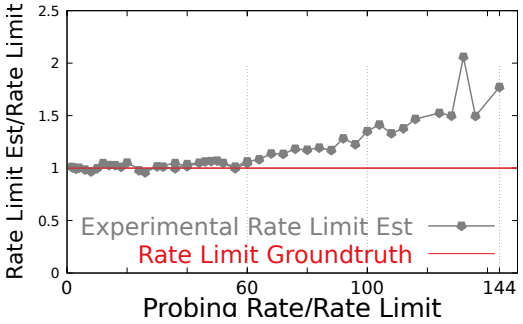
We next test the correctness of FADER in a testbed without noise. For our noise-free experiment, we run our high-rate measurement probing from $1.6L$ all the way to $240L$ stressing FADER beyond its designed detecting range $P < 60L$.

Figure 11a shows that FADER detection is perfect for $P < 60L$. However, as we exceed FADER's design limit ($60L$), it starts marking blocks as can-not-tell. The fraction of can-not-tell rises as P grows from $60L$ to $144L$. Fortunately, without packet loss, even when the design limit is exceeded, FADER is *never incorrect* (it never gives a false positive or false negative), it just refuses to answer (returning can-not-tell).

In addition to detecting rate limiting, FADER gives an estimate of what that rate limit is. Figure 11b shows the precision of its estimation, varying P from L to $144L$, The rate limit estimate is within 7% (from -4.2% to $+6.9\%$) when $P < 60L$, and it drops gradually as the design limit is exceed.

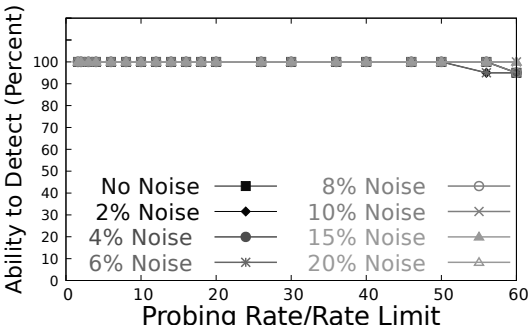


(a) Detection Correctness

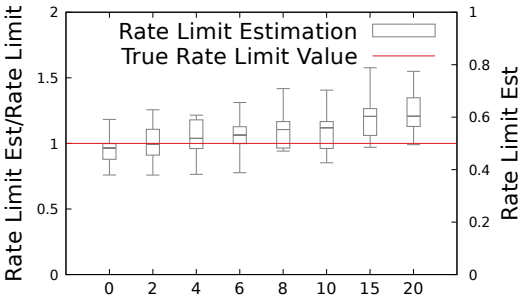


(b) Rate Limit Estimation

Figure 11: FADER detection in a noise-free environment.



(a) Detection Correctness



(b) Rate Limit Estimation When $P = 26L$

Figure 12: FADER detection with packet loss.

C. Correctness in the Face of Packet Loss

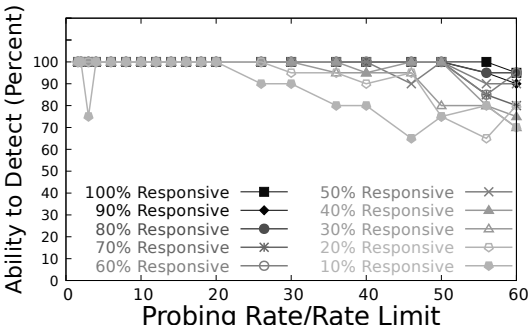
We next consider FADER with packet loss, one form of noise. Packet loss could be confused with loss due to rate limiting, so we next we vary the amount of random packet loss from 0 to 60%.

Figure 12a shows FADER detection as packet loss increases. There is almost no misdetection until probe rates become very high. At the design limit of $P = 60L$, we see only about 4% of trials are reported as cannot tell.

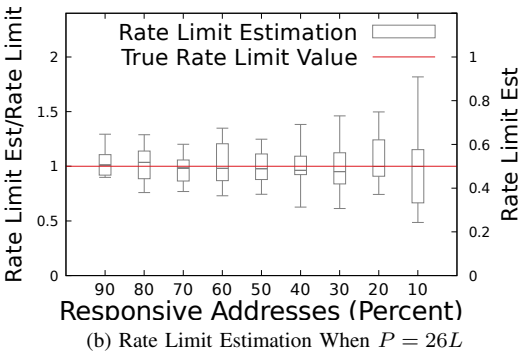
While ability to detect is insensitive to noise, our estimate of the rate limit is somewhat less robust. Figure 12b shows that packet loss affects our estimate of the value of the rate limit (here we we fix $P = 26L$, but we see similar results for other probe rates). Error in our rate limit is about equal to the dropping rate (at 20% loss rates, the median estimate of rate limit is 20.72% high).

D. Correctness with Partially Responsive Blocks

We next consider what happens when blocks are only partially responsive. Partially responsive blocks are more difficult for FADER because probes sent to non-responsive addresses are dropped, reducing the signal induced by rate limiting. Here we vary probe rate for different density blocks. (We hold other parameters fixed and so do not add packet loss.)



(a) Detection Correctness



(b) Rate Limit Estimation When $P = 26L$

Figure 13: FADER detection with partially responsive target blocks.

In Figure 13a we vary the relative probing rate and plot separate lines for each level of block responsiveness. shows as target blocks become less responsive. In general, the number of can-not-tell responses increase as block responsiveness falls, but only when the probe rate is also much greater than the rate limit. In the worst case, with only 10% of IPs responding at a probe rate $60\times$ the rate limit 35% of tries report can-not-tell. Fortunately, even in these worst cases, the algorithm reports that it cannot tell rather than silently giving a wrong answer.

Figure 13b shows the rate limit output by FADER as the block density changes. We show median and quartiles with box plots, and minimum and maximum with whiskers. The median stays at the true value, but the variance increases, as shown by generally wider boxes and whiskers. Here $P = 26L$; we see similar results at other probing rates.

E. Correctness in Other Network Conditions

FADER is designed for the general Internet, but we consider how blocks that use DHCP or for mobile networks might affect its accuracy.

DHCP: Addresses turnover in a DHCP block might affect FADER: long-term changes may affect it's comparison of availability (line 5, 16 and 18 in Algorithm 2), and short-timescale turnover might appear to be frequent alternation (Algorithm 1). When a DHCP allocates addresses sequentially from large block (multiple /24s), some of its /24 components may switch from busy to completely unutilized. When empty, FADER's availability comparison will trigger, but not frequent alternation.

FADER's frequent alternation test will be fooled by short-term changes in DHCP address use (around minutes). However, DHCP studies suggest that typical churn occurs on timeframes of 5 to 61 hours [19], so DHCP churn will not usually effect FADER.

Mobile Networks: Mobile networks (telephones) may have higher packet loss than typical due to wireless fading. However as validation subsection VI-C shows, random loss does not affect FADER's detection precision, although it does gradually decrease precision of rate limit estimation as loss increases.

VII. Related Work

Two other groups have studied the problem of detecting rate limits in the Internet.

Work from Universite Nice Sophia Antipolis studies router rate-limiting for traceroutes [23]. Specifically, they study ICMP, Type 11, Time exceeded replies on reverse paths.

They detect rate limit by launching TTL-limited ICMP echo requests from 180 vantage points, varying the probing rate from 1 to 4000 ping/s. Their algorithm looks for constant response rates as a sign of rate limits. They studied 850 routers and found 60% to do rate limiting. Our work has several important differences. The overall result is quite different: they find 60% of reverse paths are rate limited in their 850 routers, measured up to 4000 ping/s, while we find only 0.02% of forward paths in 40k /24 blocks are rate limited, with measurements up to 0.39 pings/s.

We believe that both their results and ours are correct. Many routers have reverse-path rate limiting on by default, consistent with their results. Our approach provides much broader coverage and generates less additional traffic since we reuse existing data at a lower rate. Our work uses a different signal (availability difference between fast and slow probing), and we add detection of frequent alternation to filter known false positives. Finally, we concentrate on the forward path, so our results apply to address allocation information, while they focus on reverse path, with results that apply to fast traceroutes.

Google recently examined traffic policing, particularly in video traffic [9]. There analysis uses sampled measurement from hundreds of Google CDNs to millions of users of YouTube. They provide a thorough analysis on the prevalence of policing and the interaction between policing and TCP. They also provide suggestions to both ISP and content providers on how to mitigate negative effect of traffic policing on user quality of experience. Their focus on TCP differs from ours on ICMP rate-limiting. Their coverage is far greater than ours, although that coverage is only possibly because Google is a major content provider. They find fairly widespread rate limiting of TCP traffic, but their subject (TCP video) is much faster than ours (ICMP) that such differences in results are not surprising.

VIII. Conclusion

Undetected rate limiting can silently distort network measurement and bias research results. We have developed FADER, a new, light-weight method to detect ICMP rate limiting. We validated FADER against commercial routers and through sensitivity experiments in a testbed, showing it is very accurate at detecting rate limits when probe traffic is between 1 and $60\times$ the rate limit.

We applied FADER to a large sample of the Internet (40k blocks) on two separate dates. We find that a only a tiny fraction (0.02%) of Internet blocks are ICMP rate limited up to 0.39 pings/s per /24 block. We also examined public high-rate datasets (up to about 1 ping/s per /24 block) and showed their probing results are consistent with rate limitings. We only see significant rate limiting on the reverse path when routers near the prober see a large amount of traffic. We conclude that low-rate ICMP measurement (up to 0.39 ping/s

per block) are unlikely to be distorted while high-rate measurement (up to 1 ping/s per block) risks being rate limited.

Acknowledgments

Hang Guo and John Heidemann's work is partially sponsored by the Department of Homeland Security (DHS) Science and Technology Directorate, HSARPA, Cyber Security Division, BAA 11-01-RIKA and Air Force Research Laboratory, Information Directorate under agreement number FA8750-12-2-0344 and via contract number HHSP233201600010C. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views contained herein are those of the authors and do not necessarily represent those of DHS or the U.S. Government.

References

[1] Cisco Manual For Configuring Traffic Policing. http://www.cisco.com/c/en/us/td/docs/switches/metro/me3600x_3800x/software/release/15-3_2_S/configuration/guide/3800x3600xscg/swqos.html#wp999715.

[2] Juniper Manual For Configuring Traffic Policing. http://www.juniper.net/techpubs/en_US/junos14.2/topics/concept/policer-types.html.

[3] rejwreply: a linux kernel patch that adds echo-reply to feedback type of iptable REJECT rule. https://ant.isi.edu/software/rejwreply/index.html.

[4] Ubuntu User Manual for IPtables. http://manpages.ubuntu.com/manpages/natty/man8/iptables.8.html.

[5] David Adrian, Zakir Durumeric, Gulshan Singh, and J. Alex Halderman. Zippier ZMap: Internet-wide scanning at 10 Gbps. In *Proceedings of the USENIX Workshop on Offensive Technologies*, San Diego, CA, USA, August 2014. USENIX.

[6] Robert Beverly. Yarrp'ing the internet: Randomized high-speed active topology discovery. In *Proceedings of the ACM Internet Measurement Conference*, Santa Monica, CA, USA, November 2016. ACM.

[7] Alberto Dainotti, Karyn Benson, Alistair King, kc claffy, Michael Kallitsis, Eduard Glatz, and Xenofontas Dimitropoulos. Estimating Internet address space usage through passive measurements. *ACM Computer Communication Review*, 44(1):42–49, January 2014.

[8] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. Zmap: Fast internet-wide scanning and its security applications. In *Proceedings of the 22Nd USENIX Conference on Security*, SEC'13, pages 605–620, Berkeley, CA, USA, 2013. USENIX Association.

[9] Tobias Flach, Pavlos Papageorge, Andreas Terzis, Luis Pedrosa, Yuchung Cheng, Tayeb Karim, Ethan Katz-Bassett, and Ramesh Govindan. An Internet-wide analysis of traffic policing. In *Proceedings of the ACM SIGCOMM Conference*, pages 468–482, Floranopolis, Brazil, 2016. ACM.

[10] Mehmet H. Gunes and Kamil Sarac. Analyzing router responsiveness to active measurement probes. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 23–32, Berlin, Heidelberg, 2009. Springer-Verlag.

[11] John Heidemann, Yuri Pradkin, Ramesh Govindan, Christos Papadopoulos, Genevieve Bartlett, and Joseph Bannister. Census and survey of the visible internet. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 169–182, New York, NY, USA, 2008. ACM.

[12] Bradley Huffaker, Daniel Plummer, David Moore, and k claffy. Topology discovery by active probing. In *Proceedings of the IEEE Symposium on Applications and the Internet*, pages 90–96. IEEE, January 2002.

[13] Youndo Lee and Neil Spring. Identifying and aggregating homogeneous IPv4 /24 blocks with hobbit. In *Proceedings of the ACM Internet Measurement Conference*, Santa Monica, CA, USA, November 2016. ACM.

[14] Derek Leonard and Dmitri Loguinov. Demystifying service discovery: Implementing an internet-wide scanner. In *Proceedings of the ACM Internet Measurement Conference*, pages 109–123, Melbourne, Victoria, Australia, November 2010. ACM.

[15] Derek Leonard and Dmitri Loguinov. Demystifying service discovery: Implementing an internet-wide scanner. In *Proceedings of the ACM Internet Measurement Conference*, pages 109–123, Melbourne, Victoria, Australia, November 2010. ACM.

[16] Matthew Luckie, Amogh Dhamdhere, Bradley Huffaker, David Cla rk, and kc claffy. bdrmap: Inference of borders between IP networks. In *Proceedings of the ACM Internet Measurement Conference*, Santa Monica, CA, USA, November 2016. ACM.

[17] Harsha V. Madhyastha, Tomas Isdal, Michael Piat ek, Colin Dixon, Thomas Anderson, Arvind Krishnamurthy, and Arun Venka taramani. iPlane: An information plane for distributed services. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation*, pages 367–380, Seattle, WA, USA, November 2006. USENIX.

[18] Alexander Marder and Jonathan M. Smith. MAP-IT: Multipass accurate passive inferences from traceroute. In *Proceedings of the ACM Internet Measurement Conference*, Santa Monica, CA, USA, November 2016. ACM.

[19] G. C. M. Moura, C. Gañán, Q. Lone, P. Poursaied, H. Asghari, and M. van Eeten. How dynamic is the isps address space? towards internet-wide dhcp churn estimation. In *2015 IFIP Networking Conference (IFIP Networking)*, pages 1–9, May 2015.

[20] Ramakrishna Padmanabhan, Amogh Dhamdhere, Emile Aben, kc cla ffy, and Neil Spring. Reasons dynamic addresses change. In *Proceedings of the ACM Internet Measurement Conference*, Santa Monica, CA, USA, November 2016. ACM.

[21] Lin Quan, John Heidemann, and Yuri Pradkin. Trinocular: Understanding Internet reliability through adaptive probing. In *Proceedings of the ACM SIGCOMM Conference*, pages 255–266, Hong Kong, China, August 2013. ACM.

[22] Lin Quan, John Heidemann, and Yuri Pradkin. When the Internet sleeps: Correlating diurnal networks with external factors. In *Proceedings of the ACM Internet Measurement Conference*, pages 87–100, Vancouver, BC, Canada, November 2014. ACM.

[23] R. Ravaioli, G. Urvoy-Keller, and C. Barakat. Characterizing icmp rate limitation on routers. In *2015 IEEE International Conference on Communications (ICC)*, pages 6043–6049, June 2015.

[24] Philipp Richter, Florian Wohlfart, Narseo Vallina-Rodriguez, Mark Allman, Randy Bush, Anja Feldmann, Christian Kreibich, Nicholas Weaver, and Vern Paxson. A multi-perspective analysis of carrier-grade NAT deployment. In *Proceedings of the ACM Internet Measurement Conference*, Santa Monica, CA, USA, November 2016. ACM.

[25] Aaron Schulman and Neil Spring. Pingin' in the rain. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 19–28, New York, NY, USA, 2011. ACM.

[26] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring ISP topologies with Rocketfuel. In *Proceedings of the ACM SIGCOMM Conference*, pages 133–145, Pittsburgh, Pennsylvania, USA, August 2002. ACM.

[27] USC/LANDER project. Internet address census dataset, predict id usc-lander/internet_address_census_it71w-20160803/rev5468 and usc-lander/internet_address_census_it70w-20160602/rev5404 and usc-lander/internet_address_census_it56j-20130917/rev3704 and

usc-lander/internet_address_census_it57j-20131127/rev3745 and usc-lander/internet_address_census_it58j-20140122/rev3912. web page http://www.isi.edu/ant/lander.

[28] USC/LANDER project. Internet address survey dataset, predict id USC-LANDER//internet_address_survey_reprobing_it70w-20160602/rev5417 and usc-lander/internet_address_survey_reprobing_it71w-20160803/rev5462. web page http://www.isi.edu/ant/lander.

[29] Sebastian Zander, Lachlan L. H. Andrew, and Grenville Armitage. Capturing ghosts: Predicting the used IPv4 space by inferring unobserved addresses. In *Proceedings of the ACM Internet Measurement Conference*, pages 319–332, Vancouver, BC, Canada, November 2014. ACM.