BGP-lens: Patterns and Anomalies in Internet Routing Updates *

B. Aditya Prakash[#]
baditya@cs.cmu.edu

Nicholas Valler[+]
nvaller@cs.ucr.edu

David Andersen[#]
dga@cs.cmu.edu

Michalis Faloutsos[+]
michalis@cs.ucr.edu

Christos Faloutsos[#]
christos@cs.cmu.edu

[#]Computer Science Department, Carnegie Mellon University, USA
[+]Computer Science Department, University of California - Riverside, USA

ABSTRACT

The *Border Gateway Protocol* (BGP) is one of the fundamental computer communication protocols. Monitoring and mining BGP update messages can directly reveal the health and stability of Internet routing. Here we make two contributions: firstly we find patterns in BGP updates, like self-similarity, power-law and lognormal marginals; secondly using these patterns, we find anomalies. Specifically, we develop *BGP-lens*, an automated BGP updates analysis tool, that has three desirable properties: (a) It is *effective*, able to identify phenomena that would otherwise go unnoticed, such as a peculiar 'clothesline' behavior or prolonged 'spikes' that last as long as 8 hours; (b) It is *scalable*, using algorithms that are all linear on the number of time-ticks; and (c) It is *admin-friendly*, giving useful leads for phenomenon of interest.

We showcase the capabilities of *BGP-lens* by identifying surprising phenomena verified by syadmins, over a massive trace of BGP updates spanning 2 years, from the publicly available site `datapository.net`.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms, Measurement, Security

*This material is based upon work supported by the National Science Foundation under Grants No. CNS-0721736 and CNS-0721889, and also under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344 (LLNL-CONF-404625) and subcontracts B579447, B580840. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, or other funding parties.

KDD'09, June 28–July 1, 2009, Paris, France.

Keywords

BGP Monitoring, Anomalies, Patterns, Self-similarity

1. INTRODUCTION

The '*Border Gateway Protocol*' (BGP) is responsible for keeping route information up-to-date. Thus, each router sends BGP updates to its neighbors, to keep them current with the path information that it has. Path information changes, e.g., whenever a link goes down or whenever it comes back up again. In an ideal setting, there should be no BGP updates; in reality there are, due to link failures, node failures, router-maintenance shut-downs, misconfigured routers, bugs in the router software etc. Our goal here is two-fold: (a) we want to find how *normal* BGP updates look like and (b) how to automatically spot deviations and anomalies, which ultimately helps system administrators diagnose/repair undesirable network behaviors.

The volume and the complexity of BGP updates makes it practically impossible for a human to process all the update information at several different levels of granularity. For example, we studied about 18 million BGP updates spanning over 2 years from `datapository.net`.

In this paper, we present *BGP-lens*, a novel tool for *automatically* detecting patterns and anomalies in BGP updates at many different scales of observation. We also showcase its capabilities on massive traces involving millions of measurements. The key novelties of *BGP-lens* are:

a. It is effective: Applying it to real BGP data, we identify several subtle phenomena that may otherwise go unnoticed, such as a peculiar "clothesline" behavior, as well as prolonged high-activity periods with high, near-constant volume of updates for several hours. The tool builds on a carefully chosen set of algorithms, avoiding methods like FFT, auto-correlation, thresholding, which despite their popularity, would be unsuitable here due to the bursty and noisy nature of the data. Instead, *BGP-lens* builds on top of more sophisticated, lesser-known tools like wavelets and median filtering (see §3 and §6.)

b. It is scalable: The algorithms are linear on the number of time-ticks and thus *BGP-lens* can handle large datasets: the runtimes were in the order of minutes (see §4).

Moreover, we carefully designed our tool so that it is *admin-friendly:* *BGP-lens* works with zero user input. It automates the definition of surprising phenomena, provides reasonable defaults for all the required thresholds, and ranks

the phenomena according to their statistical significance. Finally, *BGP-lens* also provides the *leads* for an investigation or troubleshooting, by identifying the network entities (origin ASes, prefixes) that participate [1].

For expert users, the tool offers the additional capability to tune a few "knobs" that control the level of sensitivity. These knobs are intuitive and require only simple tuning (e.g. low, medium, high), without the need to understand the intricate details of the underlying data mining methods (see §5).

Finally, *BGP-lens* can help sysadmins identify surprising phenomena that may otherwise go unnoticed, as was the case in Section 6, with the Alabama Supercomputing Network, whose sys-admin confirmed the anomaly.

The rest of the paper is organized as follows: We review the related work in Section 2. The proposed strategies with motivating observations are discussed in Section 3; the algorithms of the tool with a discussion on user interface etc. are presented in Section 4 and Section 5 respectively. The experimental results and case studies are presented in Section 6. We then conclude the paper in Section 7.

2. RELATED WORK

In this section, we survey earlier BGP analysis, as well as time series analysis tools.

2.1 BGP Measurements and Analysis

There has been significant work in studying BGP phenomena, which can roughly be grouped into: (a) measurement and modeling studies [9] [10]; (b) studies of network-wide BGP dynamics [5]; and (c) attempts to troubleshoot and improve BGP [21].

The characteristics of BGP updates have been studied in detail by Labovitz, who presented canonical measurement studies on BGP anomaly and route instability detection [9] [10]. Modeling studies can be characterized by the work of Maennel and Feldmanm [12], who presented a *workload* model to capture the structure of BGP traffic. Furthermore, Feldmann et al. [5] present methods to detect various BGP anomalies that affect inter-domain routing.

Work closely related to our own is presented by Teoh et al. [22] and employs statistical and visualization methods to diagnose BGP anomalies. Our work differs in that they focus on novel visualization methods to detect BGP anomalies rather than data mining techniques. Additionally, Tseng et al. [23] detect routing changes and management actions of an AS by examining BGP related data, but do not necessarily detect anomalies or other surprising BGP phenomena.

2.2 Time Series Analysis Tools

Typical tools for time series analysis and pattern discovery include the Discrete Fourier Transform (DFT) (see, e.g., Oppenheim and Schafer [14]) and the family of wavelet transforms ([16] [3]), with the Haar, Daubechies-4, Morlet, and Gabor, among the most famous. Wavelets have been extensively used for analysis of real time series before, for e.g. see [6] and [25].

For time series indexing, earlier works have used the Fourier transform, wavelets, or piece-wise linear approximations [4] [17] [8].

[1]This can be considered as a first step towards identifying the source of an anomaly, which is a much harder problem [5].

Table 1: BGP-updates snippet; Washington Router

time	peerAS	originAS	prefix
2005-02-17 12:39:42	11317	1252	204.29.119.0/24
2005-02-17 12:39:43	10490	3464	204.29.80.0/24
2005-02-17 12:39:46	10490	3464	204.29.79.0/24
2005-02-17 12:39:49	10490	3464	204.29.118.0/22
2005-02-17 12:39:55	11317	776	204.29.78.0/24
2005-02-17 12:39:55	22388	7588	207.157.115.0/24
2005-02-17 12:39:56	1252	6677	192.211.42.0/24
2005-02-17 12:39:58	10764	2200	204.29.120.0/24
...	...	...	...

For time series forecasting, the typical method is linear forecasting, also known as *autoregression* (AR) methodology, or Box-Jenkins [1], with its numerous variations (ARIMA, seasonal ARIMA, Fractional Integration (ARFIMA) [15] etc.). Non-linear forecasting includes the *Delayed Coordinate Embedding* method [18] [2], as well as tools from chaos, fractals and self-similarity.

With few exceptions, several of the above tools do not work well for the bursty, self-similar sequences we have, because they assume ergodicity, smooth changes and Gaussian errors. As we show later (Figure 6), BGP update messages, are indeed self-similar and bursty. Typical tools for self-similar analysis employ the Hurst exponent (see, eg., the SELFIS tool [7]), and the *entropy plots* [26] (Figure 5).

Among the applicable tools from the list above, we propose to use the Haar wavelet transform. We describe it in more detail in Section 3.4.

3. TOOL COMPONENTS AND OBSERVATIONS

BGP-lens examines a given time-series of BGP updates to discover interesting phenomena, such as periodicities, and anomalies. Below we present a description of the data used in this analysis and detail the two complementary components of *BGP-lens* (temporal and frequency).

3.1 The Data

We examine BGP Monitor data containing 18 million BGP update messages over a period of two years (`09/2004` to `09/2006`) from the Datapository project [13]. The primary source of data is Abilene, an academic research network employing Juniper routers running a full-mesh of iBGP sessions. Abilene uses one Zebra monitoring router per point of presence (PoP) to collect BGP updates by establishing an iBGP session as a client. As Figure 1(a) shows, there are some significant gaps in the BGP update record, which the tools handle seamlessly.

A snippet of the data is provided in Table 1. A BGP update is basically an advertisement of path to some part of the network from a router to another router. Conceptually, a BGP update is a row with many fields (columns) each containing some piece of information of the update. For example, `time` gives the time the update was sent, `originAS` is the AS (Autonomous System) which sent the update, `prefix` is the network space for which the update is being sent. Other data, primarily used for traffic engineering, is not shown in the snippet. In this paper, we will focus on `time`, `originAS` and `prefix`.

Problem Definition.

Although there are many aspects of this data, we look at a time-series which gives us the number of updates received by a router every b seconds (called the *bin size*). After identification of target time periods, the `originAS` and `prefix` fields are used to find out which parties are involved in the suspected updates. We now describe the problem we are attacking precisely:

General Problem:

- *Given*: The raw data as before (e.g. Table 1)
- *Problem*: Find patterns and anomalies.

Specific Problem:

- *Given*: Time-series after converting the raw data using appropriate bin size (as described above). Also some associated auxiliary data (`originAS`, `prefix` fields).
- *Problem*: Find patterns and anomalies. Also report suspicious entities (paths, IPs).

3.2 Shortcomings of standard techniques

Given the data, what can we discover? Regardless of bin size b, a linear-linear plot of the BGP update time series emphasizes the very high values and obliterates the others (e.g. Figure 1(a)). Therefore, a visual inspection of that plot provides minimal information: it shows several high spikes with the vast majority of time-intervals having few BGP updates. But, there can be lots of patterns hidden: e.g., consider ellipse **E** (Figure 1(a)) and its corresponding magnification (Figure 1(b)); we clearly see a short duration spurt. Note that these patterns can't be obtained through simple thresholding as choosing a threshold when there are such huge variations in the data is near-impossible. For example, if we choose 10^3 as the threshold, it will miss Ellipse **E** completely. Other methods like FFT and auto-regression [1] (which assumes Gaussian errors) also don't work here because of the burstiness (for example see Figure 1(c)). We analyze the burstiness of the time-series later in Section 5.

3.3 Temporal Analysis - The "Clothesline" Effect

As we saw above, the challenge of employing temporal analysis on BGP updates stems from the burstiness of the updates. To overcome this challenge, we propose using the log-linear plot (we use the transformation $\log(x+1)$, to handle bins with x=0 updates), (e.g. see Figure 2(b) (bottom)), which emphasizes small values over high values. We refer to it as the *clotheslines* plot, for reasons that we explain next.

3.3.1 Multi-scale Analysis

The log-linear ('clotheslines') plot shows no striking outliers in the BGP update activity, with the obvious exception of the large gaps in the data due to missing values. But in this plot, the bin size plays an important role. For example, see Figure 2. The bottom figures show the clothesline plots for bin sizes 10sec and 600sec. While the 10sec plot does not show anything striking, the 600sec plot shows an unexpected phenomenon. The phenomenon is visually similar to bed sheets hanging from a clothesline, thus we refer to it as the "*clothesline*" phenomenon. Analysis of the clothesline phenomenon leads directly to the following observation:

OBSERVATION 1. *Depending on the bin size b, we may observe 'clotheslines', that is, near-consecutive bins with similar count of updates per bin.*

For example, for b=600sec, there are many, near-consecutive updates in the range of 50 updates per bin (henceforth "*50-clothesline*"). Similarly, there is another clothesline at approximately 100 updates per 600sec, (henceforth called the "*100-clothesline*").

An intuitive explanation of the clothesline phenomenon is simply a periodic stream of BGP update messages (both update and withdraw) over a prolonged time period. A likely explanation may be Route Flapping, as suggested by the sys-admins of the related networks (see Section 6).

Up to this point in our analysis, we have described the clothesline phenomenon and how to identify it through visual inspection. The logical follow-up question is, *rather than visual inspection of clothesline plots, how can BGP-lens spot clotheslines automatically*? To answer this, we leverage the power of the "marginal" distribution; it turns out that outliers in the "marginal" distribution usually correspond to clotheslines.

3.3.2 Marginals

Figure 2 (top figures) show the PDF (probability density function) of the volume of updates, i.e. it plots the number of times we see bins with volume v (within a given time period) versus the volume v. Here, we use log-log scales, expecting a power law. Indeed, the distribution of updates is skewed, with a power-law-like tail. However, the marginal plot for b=600sec has a very pronounced tilt and several smaller spikes. Thus, we have the observation w.r.t. this plot:

OBSERVATION 2. *The PDF of the update volume seems to be a mixture of lognormals. The dominating one spikes at volume v=50, which is also the mode of the distribution (most common value)*

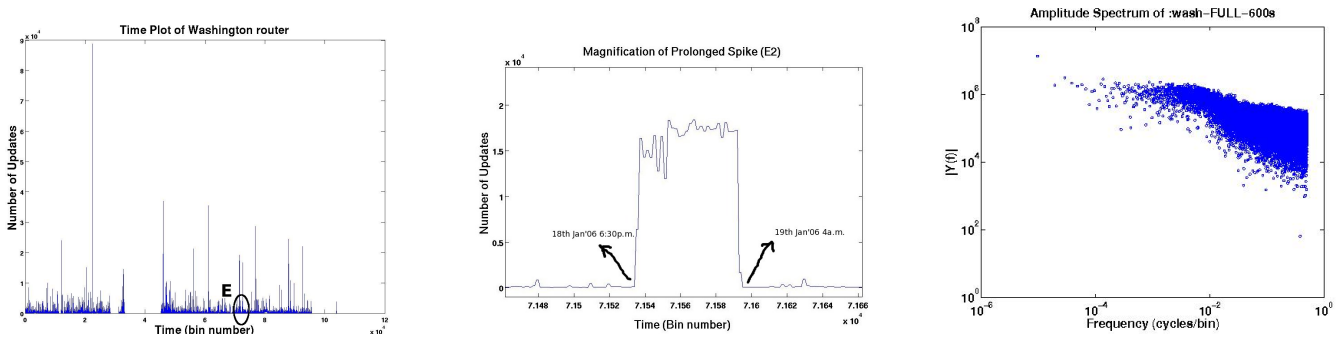
Closer inspection, e.g., of Figure 2(b) (top) shows that there is a spike at number of updates v=100 in addition to the one at around v=50; compared to the 'clothesline' plot (Figure 2(b) (bottom)), these are exactly the heights of the 'hanging bed sheets'. Thus, extremes in the marginal distribution helps us spot clotheslines.

3.4 Frequency Analysis - "Tornado" Plots

The challenge of frequency analysis of the BGP update message signal stems from the self-similar nature of the signal (see Section 5). To overcome this challenge, we chose a multi-resolution analysis tool, and specifically the Discrete Wavelet Transform (DWT), with Haar wavelets.

The wavelet analysis is similar to the Fourier analysis, in a multi-resolution way. In order to visually interpret the wavelet transform, we employ the *scalogram* (see Figure 3), which plots the (absolute) values of the wavelet coefficients in the scale-space domain. The horizontal axis is time; the vertical axis is scale (coarser scales correspond to lower frequencies, and are at the top); dark color indicates high absolute value of the corresponding wavelet coefficient. Thus, dark colors at a coarse scale indicate long, slow-moving periodicities. Dark colors at fine scales indicate short duration, fast-moving cycles.

Figure 3 shows scalograms (top) and the corresponding time-signals. We have marked the areas of interest with



(a) Time-plot (b) Magnification of ellipse E in (a) (c) Log-Log FFT of time plot

Figure 1: Plots for Washington Router (WASH) (Bin Size = b=600sec), 09/2004-09/2006. (a) Time-plot. (b) Magnification of a prolonged spike: ellipse E in (a) automatically discovered by *BGP-lens* , but otherwise *invisible*! (c) Log-Log FFT plot of WASH (b=600sec): conveys no information.

ellipses. The general idea is to look for high-energy (dark color) areas on the scalogram, and try to interpret the phenomenon. We refer to these observations as the "*tornado*" effects, and the corresponding scalogram as the "*tornado*" plot. Below we give specific observations that appeared in practice and their interpretation.

Because the signal consists of several spikes of varying power, the scalogram looks like a collection of tornadoes touching down. Figure 3(a) illustrates these concepts using a synthetic time-series. As demonstrated, a huge spike will have a tornado that will 'touch-down' i.e. there will be dark areas in all scales. Specifically in the data, see **E1** in Figure 3(b) which corresponds to 11th February 2005. The respective high energy wavelet coefficients are marked. Hence, we have:

OBSERVATION 3 (TORNADOS/SPIKES). *Pronounced spikes in the updates time series correspond to 'tornadoes' that touch down.*

Also, a larger spike will have a darker tornado (larger coefficients) than a smaller spike (compare the two spikes in the Figure 3(a)). However, the data scalogram has several tornado-like shapes, that do *not* touch-down. What is their interpretation? As exemplified in Figure 3(a), these are periods of near-constant sudden increased activity or "prolonged spikes" and show up as tornadoes which do *not* touch-down,

OBSERVATION 4 (PROLONGED SPIKES). *On the scalogram, a tornado-like shape that does not reach the highest frequencies, corresponds to a "prolonged spike".*

Notice that prolonged spikes are easily detectable on the scalogram (see, e.g., the high energy region indicated by ellipses **E2** and **E3** in Figure 3), while they can be invisible both in the time series plot, as well as the FFT. Specifically, a tornado that is *not* touching down, but stops at level i, corresponds to a prolonged spike of duration roughly 2^i time-ticks.

For example, *BGP-lens* helped us detect a period of sustained activity on 18^{th} January, 2006 that lasted about 8 hrs (ellipse **E2**). The tornado plot shows a dark tornado and detail plots show consistent spikes at coarser scales. A magnification of the time series (see Figure 1(b)) for this period also clearly shows consistent high traffic ($> 10^4$). We discuss more of such observations in Section 6.

3.5 Prolonged Spikes vs. Clotheslines

Prolonged spikes are high-intensity short (duration of *hours*) bursts while clotheslines are more sustained (duration of *months*) low-intensity activities. For example, Figure 2(b) (bottom) shows the 100-clothesline and 50-clothesline in bold red lines, while Figure 1(b) shows a 8 hour burst of approximately 15000 updates per 600sec. Hence we need to apply different methods for them. In addition, the probable networking reasons for these two phenomena can also be very different (ranging from router restarts to malicious behavior).

4. AUTOMATING THE DISCOVERY

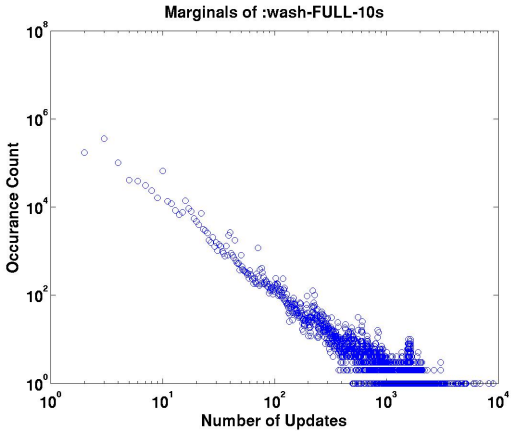
How could we automate the discovery of clotheslines and prolonged spikes? How could we also identify leads on potential sources of instability? This is exactly what we describe below.

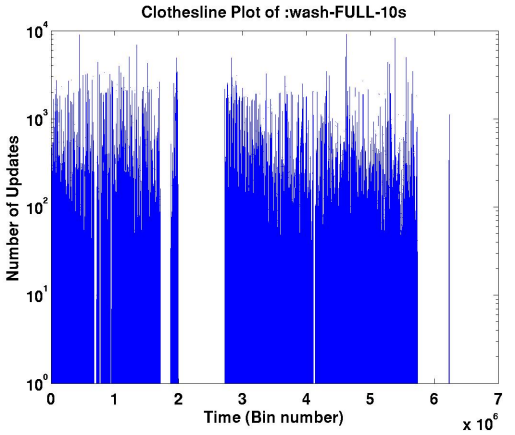
4.1 Clothesline Detection

There are three parts to this problem - first, finding possible clothesline values; second, identifying the time periods of such behavior and third, locating the most common origin AS*es* and prefixes in such time periods. Note that this has to be done across multiple bin sizes b (aggregation times) as we do not *a priori* know where a clothesline would exist. (Compare Figures 2(a) and (b)).

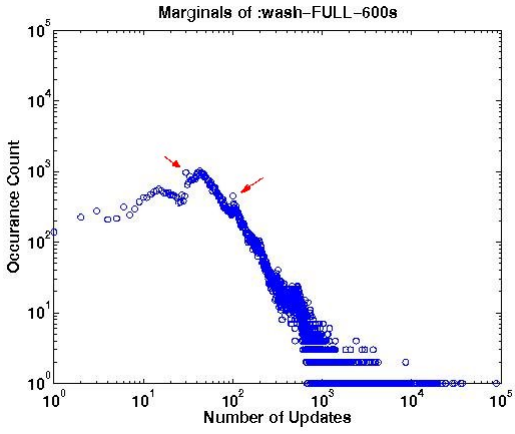
For a given bin size, we use the median filtering approach [24] for identifying unusually frequent update values. This involves using a sliding window over the marginals plot and plotting the median value over this window. In the resulting plot we choose those points that are over a relative threshold from the window-median line. Figure 2(b) (top) shows the two updates found using this approach, marked by red arrows.

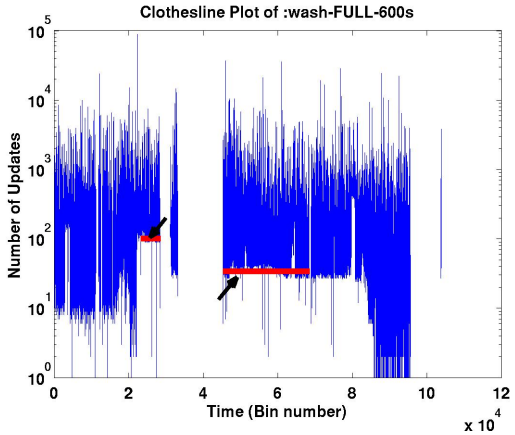
Next we try to find the longest time-interval where the number of updates are consistently above the found values (hence, the clothesline). After a simple linear pass to do this, we pick out those origin AS*es* and prefixes in such time periods who are most *persistent*. Hence, we first create a time series (for only the intervals found before) corresponding to each entity which represents the number of updates sent for/by the entity in each time bin. The most consistent entities and therefore most likely contributing to the clothesline





(a) Bin Size 10s: Marginal Plot (top) and Clothelines Plot





(b) Bin Size 600s: Marginal Plot (top) and Clothelines Plot

Figure 2: Clotheslines and aggregation bin sizes (a) Bin Size 10sec (b) Bin Size 600sec. Clotheslines marked with bold red lines (automatically discovered by *BGP-lens*)

would be the ones whose time series' have the least variance. These can now serve as an initial lead for the sysadmin. In short the procedure is:

1. For each time bin size b=2^i, derive the corresponding marginals plot.
2. For each marginals plot use the median filtering approach to determine 'outliers'; Rank them according to their deviations, and pick the top N.
3. For each of the previous outliers, find the longest time-interval from the corresponding clothesline plot.
4. For each time interval found, report the most consistent IPs/AS*es* etc.

4.2 Prolonged Spike Detection

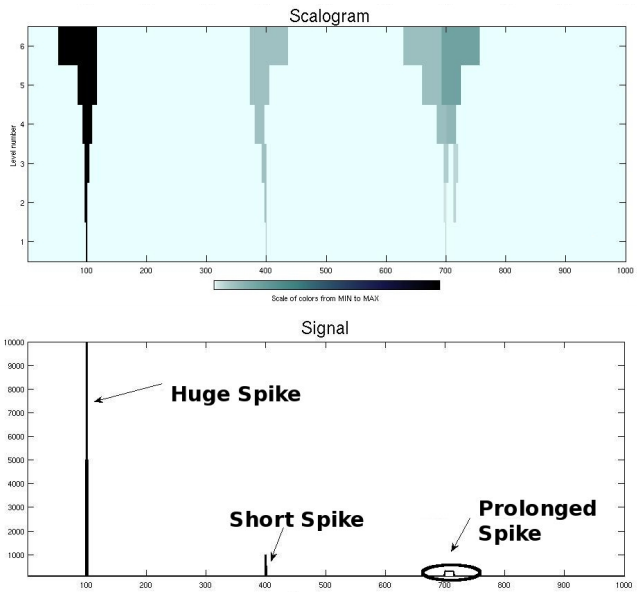
The problem of finding a prolonged spike can be stated as one to find tornadoes in the scalogram which don't 'touch-down'. The suspected time-period would be the scale of the level at which the tornado stops. The entire algorithm is shown in Algorithm 1. It outputs a set of smallest time intervals containing the prolonged spikes.

Note that it takes in two parameters, τ_{sens} and $\tau_{duration}$ corresponding to user sensitivity for a spike's strength and duration. *BGP-lens*, by default, is set it to 60% and 8 respectively - this means that *BGP-lens* will report spikes

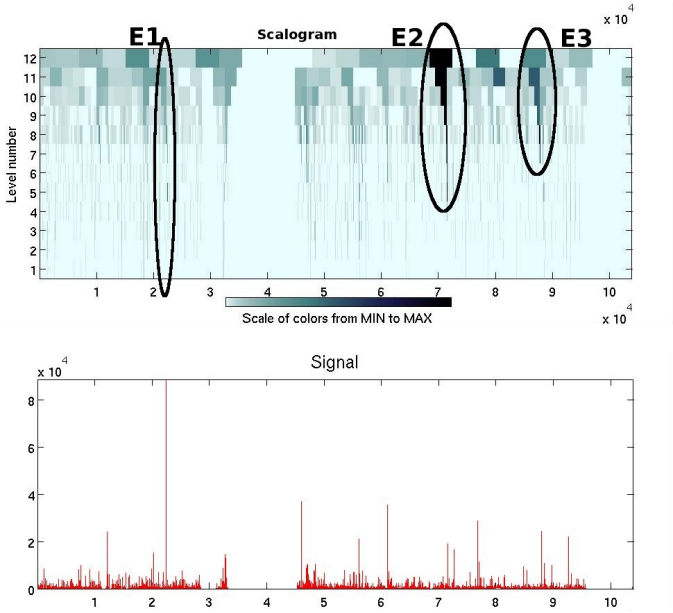
Algorithm 1 Prolonged Spike

Require: Timeseries T, τ_{sens}, $\tau_{duration}$
1: M = Wavelet transform of T
2: len = maximum wavelet level in M
3: **for** $l = \tau_{duration}$ to len **do**
4: $c_{max} = max(M(l,:))$
5: **for all** coefficients c in $M(l,:), c \geqslant \tau_{sens} * c_{max}$ **do**
6: $intr_c$ = time interval corresponding to c
7: best_interval = find_tornado($intr_c, intr_c, l$)
8: **print** "Prolonged spike found in" best_interval
9: **end for**
10: **end for**

– *find_tornado*($intr, intr_{best}, l$)
1: **if** $l > len$ **then**
2: **return** $intr_{best}$
3: **end if**
4: $c_{max} = max(M(l,:))$
5: **if** $\exists$ only 1 coefficient c in $M(l,:), c \geqslant \tau_{sens} * c_{max}$ **then**
6: $intr_c$ = time interval corresponding to c
7: **return** find_tornado($intr, intr_c, l+1$)
8: **else**
9: **return** $intr_{best}$
10: **end if**



(a) Synthetic: scalogram (top) and time-plot



(b) Real: scalogram (top) and time-plot

Figure 3: Time-series and their wavelet scalograms: (a) Synthetic series (b) Real series (WASH)

whose corresponding wavelet coefficients are within 60% of the maximum and whose duration is at least $2^{len-8+1}$ (where 2^{len} is the duration of our time series). Having spotted the time-intervals of interest, *BGP-lens* must acquire all relevant updates for each time interval and find the associated heavy hitters with respect to origin AS*es*, prefixes etc. These are the initial leads for a sysadmin to follow, to determine the cause of the events.

4.3 Scalability

How scalable are our algorithms? This is a natural question as the *BGP-lens* has been designed to run on large body of updates. We have given a running time vs. number of months of updates plot in Figure 4. It plots the running time of our tool for discovering the top-5 anomalies versus updates gathered in 1, 3, 6, 12, 24 months for the Washington router. The experiments were performed on commodity hardware having two AMD Opteron dual-core 2.4GHz CPUs (4 cores), 48G memory and the OS as Fedora Core 5. The running time are averages over 3 runs. Clearly, we grow linearly - in addition, even for running over 2 years worth of data (> 18 million updates) we take less than 4 mins. This makes *BGP-lens* attractive to be deployed actively on real networks.

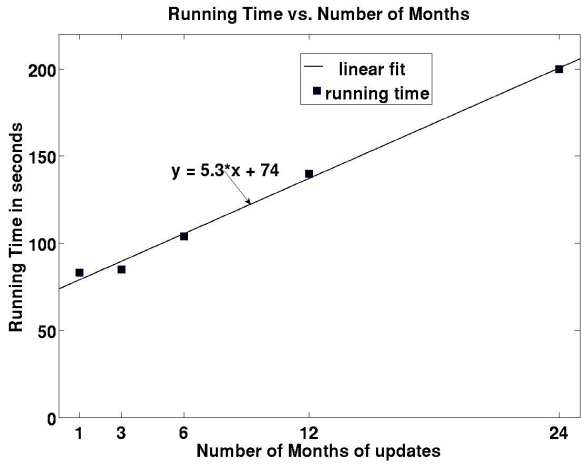


Figure 4: Scalability Results: Plot of Running Time vs. Number of months of updates. Detecting top-5 anomalies.

5. DISCUSSION

In this section, we discuss the self-similarity of BGP-updates and expand on our human interface design principle.

5.1 Burstiness Analysis

As we mentioned before in Section 3, tools like DFT etc. are ill-suited to such time-series due to the inherent burstiness. While analyzing this, we discovered that our time-series, is in fact, self-similar. We first give a little background on entropy and self-similarity and then note our findings.

Entropy. There are several ways to measure self-similarity (Hurst exponent, variance plot etc). Among them, we choose the so-called *entropy plot* which is more robust and leads to an intuitive interpretation with the *b-model* [26].

In more detail, the entropy plot $H_s(P)$ of a sequence P is defined as the *entropy* H_s as a function of *scale* s. We elaborate next: Let p_t be the fraction of packets at time-tick t (so that they sum up to 1). We divide the time sequence

into 2^s disjoint, equal intervals, and we define s as the *scale*. Let $p_{t,s}$ be the fraction of packets at interval t and scale s.

The entropy $H_s(P)$ of the time sequence $P = p_1, \ldots, p_t, \ldots$ at scale s is defined by Shannon's entropy formula [20]:

$$H_s(P) = -\sum_{t=0}^{2^s} p_{t,s} \log_2 p_{t,s}$$

and it clearly increases with the scale s. If our traffic is self-similar for some range of scales (s_1, s_2), then the entropy plot is linear for this range, and the slope is by definition the *information fractal dimension* D_1 [19].

A generator that gives self-similar sequences and linear entropy plots is the so-called *b-model* [26]. A "b"-model with *bias parameter* b generates activity recursively: If the total number of packets is, say, N, during the full interval of observation, and b=0.8 (80-20 law), then the first half of the time interval receives b=80% fraction of the activity, and the second half receives the remaining 20%; and so on, recursively, for the quarters, eighths, etc. Figure 5(a) illustrates the first few steps of the recursive generation of such bursty traffic. Figure 5(b) plots the generated traffic, with bias factor b=0.8, after 2^{10} subdivisions. For traffic generated by a b-model, the slope s of the entropy plot, and the bias factor b obey the equation $s = -b\log_2 b - (1-b)\log_2(1-b)$ (see [26] for the proof).

Findings in our Data. Several network-related traffic sequences exhibit self-similar behavior [11]. Is this the case here? Figure 6 repeats the time-plot and also gives the entropy plot for the full sequence (Washington router, 2 years duration, b = 600sec). We use the entropy plot that we described earlier, and indeed we see that the plot is a straight line with slope 0.83 (Figure 6(b)). This approximately corresponds to a b-model of 75-25. This motivates the need to use a multi-resolution technique like wavelets rather than DFT.

5.2 User Interface

As developed, *BGP-lens* was a series of command line tools. In order to improve usability, we spent time designing an admin-friendly graphical user interface. We describe the details next and show an example screenshot (Figure 7).

BGP-lens is ready to be used without any manual configuration: it will scan the data at multiple thresholds using default values, identify and rank statistical deviations, and report them to the user in order of statistical significance.

To increase the usefulness of the tool, we also provide "basic" and "advanced" modes of operation. In more detail, our components for identifying prolonged spikes and clotheslines contain user knobs for better control over the exploration. Broadly, they can be classified into 'sensitivity' knobs and 'duration' knobs. The sensitivity knobs control the number of 'suspicious' events that the sys-admin is willing to look into: the higher the sensitivity, the more events we return. The duration knobs control the length of the events that *BGP-lens* will check (e.g., daily versus monthly, versus yearly disturbances). The important point is that these knobs are (a) optional and (b) they have settings like 'low', 'medium', 'high', thus hiding all the details of the underlying algorithms.

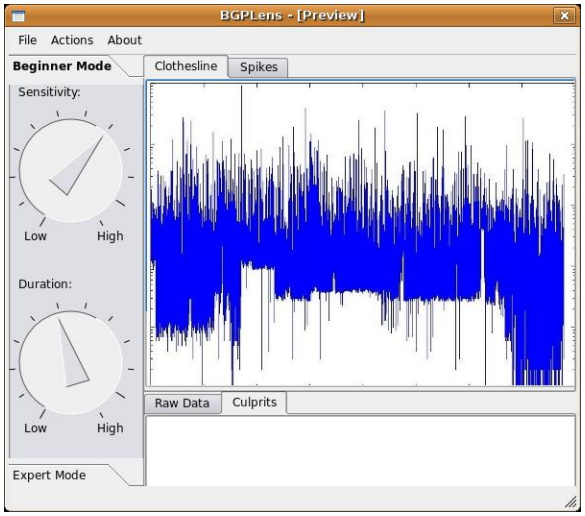


Figure 7: *BGP-lens* GUI Example

6. BGP-LENS AT WORK

In this section, we demonstrate the impact that *BGP-lens* can have when used as a network administration tool. First, *BGP-lens* identifies phenomena that may escape the radar of sys-admins or other monitoring tools. Second, it provide leads as to where to look for the origin of the observed phenomena.

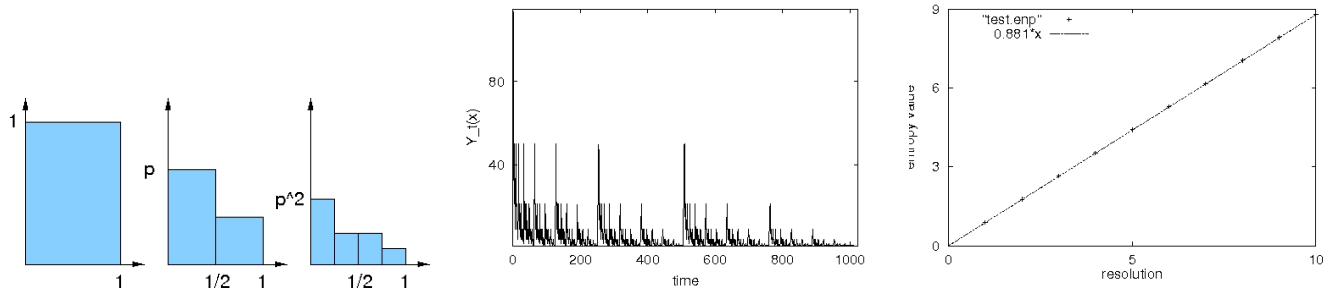
6.1 Clotheslines

As mentioned earlier, *BGP-lens* detected two distinct *clothesline* phenomena, the *50-clothesline* and the *100-clothesline*, both at bin size b=10 minutes. The former was observed from approximately late-August to late-September 2005. The latter clothesline lasted approximately 1 month, from February to March 2005.

Digging deeper: a success story. While simply identifying an anomaly is interesting, *BGP-lens* goes further and presents potential *leads* in the form of the origin AS(es) and prefixes most commonly observed contributing to clothesline effects. Table 2 indicates the AS origin and prefixes that contributed to the periodic 50-clothesline. Note that our interest is not to claim that all BGP routers see this clothesline behavior, but the capability of the tool to identify such peculiar behaviors automatically within a very large dataset.

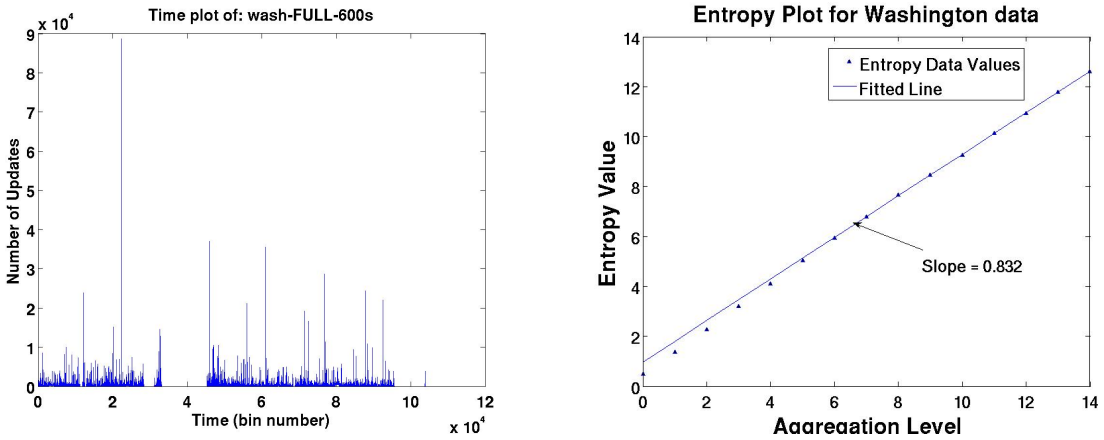
BGP-lens pointed to the education network of the state of Alabama (AL Supercomputer Net) as a potential source of this phenomenon. We contacted the administrators of the network who attributed the anomaly to changes while transitioning address space causing IGP route flapping, so that "*the route for 207.157.115.0/24 was appearing and disappearing in [the] IGP routing table ... [which] may have caused BGP to flap.*" We contacted other network administrators, but in many cases we did not get a response.

Note that this particularly anomaly shows up so strongly using *BGP-lens*, but it went undetected and unresolved, despite its *30 day duration*, in a professionally managed network. We believe that this incident highlights the need for automated, parameter-free anomaly detection for routing events.



(a) generation (b) synthetic data (c) entropy plot

Figure 5: Illustration of the *b-model*: (a) the recursive 80-20 procedure in its first three iterations (b) the generated synthetic activity *(eg., number of updates, over time)* (c) its *entropy plot* (entropy versus scale - see text) Because the synthetic input traffic is self-similar, the entropy plot is linear, that is, *scale free*. Its slope is 0.881, much different than 1.0, which would be the uniform distribution (50-50)



(a) Time plot (b) Entropy Plot

Figure 6: Time plots are bursty. Number of updates over time for an Abilene node (WASHINGTON), 09/2004-09/2006. (a) time plot in linear-linear axes (b) entropy plot. Notice that the time-plot is bursty, and that the entropy plot is linear, with slope 0.83, which implies self-similarity - no characteristic scales.

6.2 Prolonged Spikes

BGP-lens detects prolonged spikes in the number of BGP update messages across multiple different time-scales as we explained earlier. Spikes are directly related to route instability and the overall *health* of inter-domain routing.

We detected three examples of prolonged BGP update spikes that would go unnoticed with most previous techniques. The first spike was observed on May 12, 2006 with a duration of approximately 5 hours. Table 3 shows the observed number of updates and leads as to the origin AS and prefix contributing the most update messages to the spike. In fact, our analysis of this spike attributes the primary sources of BGP update to primary and middle schools in the city of Guangzhou, China. Unfortunately, despite numerous attempts, we did not receive a response from the the Guangzhou network administrators.

Two more spikes were observed on January 18-19, 2006 lasting approximately 8 hours, and on August 1, 2005 lasting approximately 3 hours. *BGP-lens* provides again the starting points of where to look for the cause of these spikes, but the results are omitted due to space limitations.

7. CONCLUSIONS

In this paper, we develop *BGP-lens*, a novel, admin-friendly tool for *automatically* detecting surprising patterns and anomalies of BGP updates at many different scales of observation. The key characteristics of our approach are:

1. It is *effective*, spotting subtle phenomena like the 'clotheslines' and 'prolonged spikes'.
2. It is *admin-friendly*, requiring no parameters and providing leads to network administrators, like most frequent IP addresses and paths, in the phenomenon of interest.
3. It is *scalable*: All its algorithms are linear on the number of time-ticks, and thus *BGP-lens* can handle huge datasets.

In addition we discover surprising aspects of the data:

- Marginals that are mixture of log-normals with a power-law tail.
- Self-similarity corresponding to a 75-25 b-model (= slope of 0.83 in the entropy plot).

Table 2: 50-Clothesline Results, 22-Aug to 25-Sept-2005

Origin AS	Median #Updates	Comments
4788	235	TM Net, Malaysia
3464	21	AL Supercomp. Net, US
10036	134	C&M Comm., Korea
9768	109	KT, Korea
Prefixes	**Median #Updates**	**Comments**
207.157.115.0/24	14	AL Supercomp Net, US
192.211.42.0/24	14	AL Ind. Dev. Training, US
216.109.38.0/24	14	AL Supercomp. Net, US
192.94.104.0/22	14	U. of NE Medical Center

Table 3: Prolonged Spike Results, 12-May-2005

Origin AS	#Updates	Comments
4538	229960	CERNET, China
9406	4976	CERNET, China
23911	1516	CERNET, China
Prefixes	**#Updates**	**Comments**
222.200.236.0/23	1314	CERNET, China
222.203.64.0/24	1311	CERNET, China
222.202.96.0/24	1311	CERNET, China

Future work will focus on making the algorithms incremental and "any-time", so that we can deploy it as a non-stop monitoring tool.

8. REFERENCES

[1] G. E. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, Englewood Cliffs, NJ, 3rd edition, 1994.

[2] D. Chakrabarti and C. Faloutsos. F4: Large-scale automated forecasting using fractals. *CIKM 2002*, Nov. 2002.

[3] I. Daubechies. *Ten Lectures on Wavelets*. Capital City Press, Montpelier, Vermont, 1992. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.

[4] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proc. ACM SIGMOD*, pages 419–429, Minneapolis, MN, May 25-27 1994. 'Best Paper' award; also available as CS-TR-3190, UMIACS-TR-93-131, ISR TR-93-86.

[5] A. Feldmann, O. Maennel, Z. M. Mao, A. Berger, and B. Maggs. Locating Internet Routing Instabilities. *SIGCOMM Comput. Commun. Rev.*, 34(4):205–218, 2004.

[6] D. Field. Scale-invariance and self-similar 'wavelet' transforms: an analysis fo natural scenes and mammalian visual systems. In M. Farge, J. Hunt, and J. Vassilicos, editors, *Wavelets, Fractals, and Fourier Transforms*, pages 151–193. Clarendon Press, Oxford, 1993.

[7] T. Karagiannis, M. Molle, and M. Faloutsos. A User-Friendly Self-Similarity Analysis Tool. In *ACM Computer Communication Review*, volume 33, pages 81–93, 2004.

[8] E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD Conference*, Santa Barbara, CA, 2001.

[9] C. Labovitz, G. R. Malan, and F. Jahanian. Internet routing instability. In *SIGCOMM '97: Proceedings of the ACM SIGCOMM '97 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 115–126, New York, NY, USA, 1997. ACM.

[10] C. Labovitz, G. R. Malan, and F. Jahanian. Origins of internet routing instability. Technical Report CSE-TR-368-98, 1998.

[11] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the self-similar nature of ethernet traffic. *IEEE Transactions on Networking*, 2(1):1–15, Feb. 1994. (earlier version in SIGCOMM '93, pp 183-193).

[12] O. Maennel and A. Feldmann. Realistic BGP Traffic for Test Labs. *SIGCOMM Comput. Commun. Rev.*, 32(4):31–44, 2002.

[13] H. B. N. Feamster, D. Andersen and F. Kaashoek. Bgp monitor - the datapository project, http://www.datapository.net/bgpmon/.

[14] A. V. Oppenheim and R. W. Schafer. *Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, N.J., 1975.

[15] S. Papadimitriou, A. Brockwell, and C. Faloutsos. Adaptive, hands-off stream mining. *VLDB*, Sept. 2003.

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.

[17] D. Rafiei and A. O. Mendelzon. Similarity-based queries for time series data. In *SIGMOD Conference*, pages 13–25, Tucson, AZ, 1997.

[18] T. Sauer. Time series prediction using delay coordinate embedding. In A. S. Weigend and N. A. Gershenfeld, editors, *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley, 1994.

[19] M. Schroeder. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. W.H. Freeman and Company, New York, 1991.

[20] C. E. Shannon and W. Weaver. *Mathematical Theory of Communication*. University of Illinois Press, 1963.

[21] L. Subramanian, M. Caesar, C. T. Ee, M. Handley, M. Mao, S. Shenker, and I. Stoica. HLP: A Next Generation Inter-Domain Routing Protocol. *SIGCOMM Comput. Commun. Rev.*, 35(4):13–24, 2005.

[22] S. T. Teoh, K. Zhang, S.-M. Tseng, K.-L. Ma, and S. F. Wu. Combining Visual and Automated Data Mining for Near-Real-Time Anomaly Detection and analysis in BGP. In *VizSEC/DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 35–44, New York, NY, USA, 2004. ACM.

[23] S.-M. Tseng, S. F. Wu, X. Zhao, and K. Zhang. Reverse Engineering the Management Actions from Observed BGP Data. In *IEEE Workshop on Automated Network Management, INFOCOM 2008*, 2008.

[24] D. Vernon. *Machine Vision: Automated Visual Inspection and Robot Vision*. Prentice-Hall International (UK) Ltd., 1991.

[25] K. Wang and S. Shamma. Spectral shape analysis in the central auditory system. *NNSP*, Sept. 1993.

[26] M. Wang, T. Madhyastha, N. H. Chang, S. Papadimitriou, and C. Faloutsos. Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic. *ICDE*, Feb. 2002.