Network Traffic Behaviour in Switched Ethernet Systems

Tony Field, Uli Harder & Peter Harrison
Department of Computing
Imperial College of Science, Technology and Medicine
Huxley Building, 180 Queen's Gate, London SW7 2BZ
England
{ajf|uh|pgh}@doc.ic.ac.uk

Abstract

Measurements on a high-performance Ethernet are shown to match well a truncated Cauchy probability distribution, with a much better fit over smaller file/request sizes than the commonly used Pareto distribution. We observe self similar characteristics in the traffic at both file servers and at a CPU server elsewhere in the network, which targets, predominantly, file and web servers. This suggests propagation of self similarity. A simulation model of a single server with Poisson arrivals and Cauchy service demands yields a departure process that follows a power law and matches closely the observed traffic. This suggests a link between file/request size distribution and self similarity in traffic, leading to the possibility of using conventional queueing network performance models with processor sharing queueing discipline. This idea is further supported by an additional simulation experiment and suitable models are proposed.

1 Introduction

In preparation for the building of performance models for network performance we have been measuring and analysing network traffic at various parts of an academic departmental network, specifically the Computing Department of Imperial College, London. The motivation is to provide a better understanding of networking issues and to provide a body of data that can be used to validate future models of networks and network traffic.

Network traffic has been measured by many researchers and been analysed in many different ways ever since the seminal papers of Leland et al. and Eramili et al. [8, 9]. There are many papers debating the reason for the apparent self-similarity in modern communications traffic. The explanations put forward range from ON/OFF models of heavy-tail distributions [8, 9], through the file size distribution in file systems and web servers [15, 6] to user behaviour, higher level network protocols, back-off algorithms in the Ethernet [12], buffers in routers and the TCP congestion avoidance algorithms [14, 31, 26, 27, 7, 22].

Like many earlier papers the main motivation here is to obtain a better understanding of the characteristics of network traffic and to explain the sources of self-similarity. In addition, however, we want to build accurate models of network traffic, together with the web-servers with which it interacts. This requires that such models should be able to reconstruct the type of self-similar patterns that are observed in real networks. To this end, we develop a simple simulation reference model, carefully parameterised using insights from various observations of real traffic, which is shown to recreate very effectively the power laws observed in practice.

A key property that the simulation reproduces is the distribution of both web server and file server response sizes, in terms of the number of Ethernet packets required to transmit them. It is well known that these often exhibit power laws [15, 6] but previous work has tended to match them to Pareto distributions. Our analysis suggests that the data may be better represented by a (truncated) Cauchy distribution and the success of the simulation model in reproducing observed traffic patterns bears this out. The paper thus adds to the growing body of evidence linking heavy-tailed distributions with self-similarity, here backed up with a supporting model.

Curiously the Cauchy distribution appears elsewhere in our analysis, specifically when we consider the *changes* in the packet transmission rates over time. This type of measurement is commonplace in the analysis of stock prices, for example [21, 32], but not, to date, for network traffic. Interestingly stock price changes have also been found to be well approximated by a Cauchy distribution.

The measurements we have performed are cheap and easy to reproduce at other sites. In particular, in our discussions we promote the use of `/proc/net/dev` files when analysing over very long time periods. This is both economical on resources and can be done without system privileges. A comparison of `/proc/net/dev` and `tcpdump` -based measurements is presented later in the paper.

The rest of the paper is organised as follows: Section 2 discusses the two main mechanisms for real-time data capture within the Linux operating system. Section 3 describes the architecture of the network in question and the type of monitoring that has been performed. Section 4 describes the analysis techniques that have been applied to the captured traces and Section 5 presents the results of applying that analysis. The simulation model is detailed in Section 6 and the interpretation of its output in relation to our measurements and queueing models is discussed in Section 7. The paper concludes in Section 8, summarising the main contributions and indicating future research directions.

2 Data Capture

This paper refers to data measured in two different ways. One uses the `tcpdump` program that monitors traffic at the packet level. The other method reads the contents of the file `/proc/net/dev` periodically.

2.1 `tcpdump`

The `tcpdump` [29] utility attaches itself to the network socket in the Linux kernel and makes a copy of each network packet that arrives at the network interface. We have used the program to obtain for every frame that is transmitted:

- A timestamp indicating when the packet reaches the kernel;
- Source and destination IP addresses and port numbers;
- The size of the frame (only the user data is reported, the headers for various protocol layers have to be added on to recover the actual size of the frame);
- The traffic type (`tcp`, `udp`, `icmp`, etc).

The timestamp is the time the kernel first "sees" the packets rather than the time it seen by the Ethernet card.

2.2 `/proc/net/dev`

The Linux operating system keeps track of the number of packets and bytes sent and received in a virtual file called `/proc/net/dev` using counters. The counters get reset at machine reboot, the values are modulo 2^{32}. We have used a `PERL` program to query this file at regular intervals of 1 second and record the value of the counters and a time stamp. Occasionally this process fails to record the counter within 1 second. In the processing phase we then linearly interpolate the missing values and get a time series of the counter values for every second. The data is similar to the aggregated data of the previous section although the time resolution is by no means as good. This method has three advantages over `tcpdump` : it usually requires no special operating system privileges, it can be run for a much longer time as only summary data is collected, and it incurs lower intrusion overheads. The tradeoff between resolution and efficiency, both space and time, is addressed later in this paper.

All raw data discussed in this paper has been anonymised and is freely available for inspection and use in other research—see [25] for the url.

3 Network Architecture and Monitoring

The monitored system is a departmental switched Ethernet. We focus attention on three components of this network for the purposes of this paper: the router, which connects the network to the outside world, an arbitrarily chosen CPU server (named MOA), and the departmental web server which services both internal and external web page requests.

3.1 Router

The department is connected to the Internet via a Black Diamond router from Extreme Networks [10]. The Black Diamond is used as both a top-level *switch* for the internal switched Ethernet and as a router for all outgoing traffic. Those two functions are completely separate and implemented by different pieces of hardware. All outgoing packets from machines in the department are duplicated at media access control (MAC) level to the second network interface card (NIC) of a dedicated monitoring machine (ORWELL[1]). Log files are transferred to a different machine (GAUSS) for analysis. We have used `PERL` scripts to turn the `tcpdump` output into text files. In this paper we concentrate on data gathered over a two hour interval on 22 March 2002 between 12pm and 2pm.

When interpreting the results, the reader should keep in mind what time scales are involved. A 100Mbits/sec network manages to transmit about 13 Bytes per microsecond. Important Ethernet data sizes and their transmission times for a 100Mbps network are the *inter frame gap*: 12 Bytes or 0.92 μseconds, the *smallest frame*: 64 Bytes or 4.9 μseconds and the *largest frame*: 1500 Bytes or 115

[1]The Black Diamond provides a link of 1 Gbit/s. The monitoring machine runs Linux 2.2.x, has 256 Mbytes RAM and 4 SCSI disks used in rotation for the log files. ORWELL can monitor at a rate of 100Mbit/s

μseconds. The inter frame gap is the minimum gap between two consecutive Ethernet frames. The inter-arrival time histograms show peaks at about 6 and 120 μseconds, as these are the most likely inter arrival times on a busy network.

We also measured the outgoing traffic as seen by the monitoring machine using the `/proc/net/dev` file starting at 6 March 2002 at 18:48:03. We compare the two hour periods measured by both methods in Section 5.

3.2 CPU server

On one of the departmental CPU servers, MOA, we ran the `/proc/net/dev` monitor for twelve days at a measurement interval of one second. The measurement started at 1 February 2002 at 16:24:47.

3.3 Web Server

Web server traffic was filtered so as to collect only outgoing data generated by external requests. The web server itself provides information on the individual web requests, in particular bytes shipped per request. This data is sent to the outside world via the router (Black Diamond) and individual (www) packets are monitored as they pass through the same router. We thus obtain the measured request size distribution and a time series representing the instants the corresponding stream of Ethernet packets pass through the router. Again we use the interval on 22 March 2002 between 12pm and 2pm.

4 Analysis Methods

We concentrate here on the point processes formed by packet departure events happening at times t_i, $i \in I \subseteq I\!N$. The event that occurs at time t_i is called E_{t_i}. We assume that there are $n \in I\!N$ events (or observations of events), the first happening at t_1 and the last one at t_n. The observation period may begin before the first event and end after the last, so we define it to be $T = [t_0, t_{n+1}] \subset I\!R$ with $t_0 \leq t_1 \leq \ldots \leq t_n \leq t_{n+1}$, for arbitrary t_0, t_{n+1} bounding the set of event-instants.

The inter-event times, Δt_i, $1 \leq i \leq n-1$, are defined as

$$\Delta t_i = t_{i+1} - t_i$$

For a finite observation of a point process we can easily generate a histogram that approximates the probability density function (pdf) of inter-event times.

4.1 Power Laws

The probability density function $p(x)$ is said to follow a power law if

$$p(x) \propto \beta x^{\gamma}$$

as $x \to \infty$, for $\beta > 0, \gamma < -1$. When investigating the existence or otherwise of a power laws we use exponentially growing bin sizes for the histograms. Apart from the histogram, we also compute the mean and variance of the inter-event times which are useful for distribution fitting.

4.2 Aggregation

Starting from a point process one can investigate the behaviour of the corresponding *aggregated* time series. The observation period T is divided into N contiguous intervals of size $T_N = T/N$. In each of these intervals, we count the number of events or, if suitable, we aggregate the properties of the events. So the time series consists of N values

$$a_i = \Big|\{E_t | t_0 + iT_N \leq t < t_0 + (i+1)T_N\}\Big|.$$

for $i = 1, 2, \ldots, N$. Sometimes it is preferred to use the quantity $A_i = a_i / T_N$.

4.3 Self Similarity

An aggregated time series can be subjected to many analyses, one of which is its *scaling behaviour*. This is also known as testing self-similarity. Two of the first investigations of the statistical nature of network traffic were [8, 9]. The authors found evidence that the observed traffic was distinctly non-Poisson and thus not amenable to conventional traffic modelling techniques.

They instead used methods that had been developed earlier by Hurst who was investigating the "ideal" size of reservoirs (a good summary of Hurst's work is given in [20, 11]). In particular, Hurst introduced the rescaled range statistic R/S which gives an idea of the self-similarity or long-range dependence of a time series. Many other statistics, which are all proved or conjectured to be related to the Hurst parameter, have since been introduced. A good review of the estimators and their relationships can be found in [23, 28, 30]. The next section is based on the material found in those papers. In this investigation, we will use the *power spectrum* to analyse the correlation of the monitored data and inter-event interval histograms to analyse the inter-departure time distribution.

4.4 Power Spectrum

For a time series $X(t)$ with zero mean the auto-correlation function at lag τ is defined as

$$C(\tau) = \lim_{T \to \infty} \frac{1}{2T} \int_{-\infty}^{\infty} dt X(t+\tau) X(t).$$

In fact, it is usually easier to work with the Fourier transform of the auto-correlation function. By the Wiener-Khintchine theorem [33, 18], under certain assumptions, this is the same as the power spectrum (density) of the time series signal [2]

$$S(f) = \lim_{T \to \infty} \frac{1}{4\pi T} \left| \int_{-T}^{T} dt X(t) e^{-i2\pi f t} \right|^2 .$$

Note that this expression can be discretised for discrete time series. Since the actual point process tends to be rather sparse it is best to turn the time series into an aggregate time series of counts. In our investigation we have used 10ms bins for the aggregation in line with previous research [8, 9]. Again one is looking for power laws where the power spectrum $S(f)$ behaves like $S(f) \propto 1/f^{\alpha}$, where f is the frequency. The exponent α turns out to be 0 for white noise and 2 for a Brownian motion. From the relation of the power spectrum to the auto-correlation function

$$C(\tau) \propto |\tau|^{\alpha - 1} \quad \text{for } 0 < \alpha < 1$$

it also follows that an exponent α close to but smaller than 1 corresponds to long term correlations.

The power spectra were computed using standard methods published in the *Numerical Recipes in C* with overlapping windows [24]. In addition we averaged the values of the power spectra in logarithmic intervals to reduce the noise that is usually seen for higher frequencies. When interpreting a power spectrum it is useful to keep in mind that the leftmost (highest) frequency is determined by sampling rate and corresponds to the shortest time, while the rightmost (lowest) frequency is determined by the sampling length and corresponds to the longest time.

Other methods to investigate the correlation of the data are the rescaled range statistic, the log-variance plot, [20, 11, 5] detrended fluctuation analysis, wavelet transformations [1], the Fano factor and the Allan factor [23, 28, 30, 1].

4.5 Scaling Behaviour and Power Laws

Suppose a time series $X(t)$ exhibits the scaling law $X(t\alpha) = g(\alpha)X(t)$ for some function $g(\alpha)$. Then it is known that $X(t) = bg(t)$ and $g(\alpha) = \alpha^c$, for real constants b, c, is the only non-trivial solution to the above functional equation. This property is related to what is known as self-similarity. The exponent c is related to the Hurst parameter and, in fact, there is a firm belief – even a mathematical proof in some cases – that all exponents retrieved from power laws discovered in statistics of self-similar or fractal time series are related. So, the choice of the method used to analyse a time series is, up to a point, a matter of taste and ease of use.

If one assumes that a time series is generated by a stochastic process, such as Brownian motion, for example, one can investigate the distribution of the *changes* of the a_i from one time step to the next, i.e. the differenced time series $\{\Delta a_i = a_{i+1} - a_i | 1 \leq i \leq N - 1\}$. Alternatively one can work out the distribution of the a_i as they are the changes of the underlying counting process. Depending on the statistical nature, one also expects a different kind of scaling law for these distributions with respect to the length of the aggregation interval T_N.

5 Measurements

5.1 Outgoing Router Traffic

We begin with a routine analysis of the outgoing traffic at the Black Diamond router. We expect no surprises here and find, as has been observed numerously elsewhere, that there is strong evidence of a power law in underlying time series for packet departure times over the measurement interval, as measured using `tcpdump`. In particular it is clear that the inter-event times (figure 1) are distinctly non-exponential as evidenced in the figure. Rather, they have two humps corresponding to the shortest and longest packet size. The remainder of the distribution could resemble an exponential. The power spectrum shows a power law, as indicated by the straight line in figure 1. The underlying time series is an aggregation of the original one in 0.01 second bins, counting the packets. More interestingly the data gathered provides the opportunity to compare data obtained from `tcpdump` with that of `/proc/net/dev` . Whilst gathering the `tcpdump` data, we also used `/proc/net/dev` to measure network activity over the same period as reported in Figure 2. Similarly the distribution in the changes of the packet rates are very similar for both measurement types. Aggregated over two minute intervals, the packet

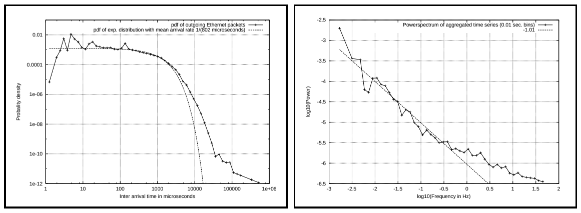

Figure 1. The left plot shows the inter arrival time histogram for the outgoing network traffic on a log-log scale. For comparison, we have drawn an exponential pdf with the same mean as that measured. The right plot shows the power spectrum of the outgoing network traffic time series.

[2]We should note that this is not a very rigorous way of defining the power spectrum, as the time series has to fulfil certain criteria for the integral to be well defined, for instance.

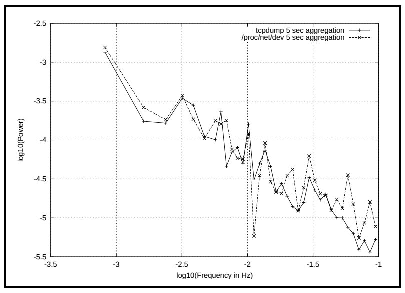

Figure 2. The plot compares power-spectra resulting from `/proc/net/dev` and `tcpdump` measurements.

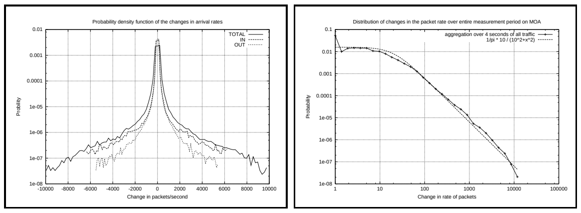

Figure 3. The left plot shows the probability distribution function of the changes in the arrival rates of packets. The right plot shows the probability distribution function of the changes in the arrival rates of packets of the entire traffic compared to a Cauchy distribution.

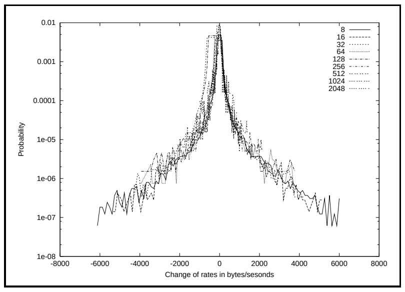

Figure 4. Probablity distribution function of the changes in the arrival rates of packets of the entire traffic at different aggregation levels.

rates are almost identical for both methods. The power spectrum obtained by `/proc/net/dev` is rather jerkier due to the smaller number of points as the bin size is now 5 seconds. Still, we can observe a power law behaviour similar to the one observed with `tcpdump` see figure 2, where both measurements are compared and aggregated to the same bin size of 5 seconds. The results show that data obtained by `/proc/net/dev` provides an excellent alternative to `tcpdump` particularly when monitoring over very long time periods. `/proc/net/dev` is used exclusively in the analysis of the CPU server which was monitored continuously over a twelve-day period.

Looking at the changes in the packet rates, Δ_i, we find that they are not Normally distributed. The distributions instead show a power law in a double-logarithmic plot.

5.2 CPU Server

The power spectrum of a twelve-day observation period clearly shows peaks corresponding to the daily cycles, and also has a peak between one and two hours, an interval that will correspond to typical session lengths of student users during term time. The main interest lies in the plots which show the *changes* in the observed packet rate. We see a distribution that is distinctly leptokurtic and certainly non-Gaussian, see figure 3and 4. In fact the plot 3 shows that the asymptotic power law has a gradient of -2. And the plot 4 shows that the distributions at different aggregation levels fall into one master curve when plotted on top of each other as they should do for a Cauchy distribution, see p. 92 [32] and p. 71 [21].

These plots were inspired by related work in the analysis of stock prices where changes in prices have been shown to follow similar patterns, see for example [21, 32]. In order to trace possible causes of this behaviour we next study the characteristics of the file sizes stored at the file server and the request sizes generated by the web server, as requests to these two servers constitute the main source of network traffic to/from the CPU server.

5.3 File Size and Webserver Request Distribution

Many authors have repeatedly shown that these distributions follow Zipf's law to a good approximation, which says that

$$P(\text{file or request size} > x) \approx \frac{1}{x}.$$

One pdf that can exhibit this behaviour is the Pareto distribution

$$p(x) = \alpha k^{\alpha} x^{-\alpha-1},$$

where $\alpha, k > 0$ and $x \geq k$. If $\alpha = 1$ the Pareto distribution shows the behaviour of the Zipf law for large x. In a double logarithmic plot, this distribution is a straight line with gradient $-(1+\alpha)$. In [6] the authors use the *log-log complementary distribution* plots to estimate the distribution for request sizes.

We have measured both the file size distribution on several Linux/UNIX machines (using an adapted script from [15]) and the distribution of the request size of external requests made to the departmental web server, on 22 March 2002 between 12.00 - 14.00 (which is the period we measured in detail earlier on). In doing so, we adopt a different approach by plotting the histogram of the measured pdf using exponentially increasing bin sizes. The results are shown in Figures 5.These measurements are distinctly

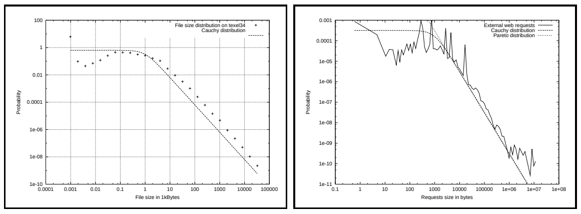

Figure 5. File size distributions on a departmental LINUX machine on the local disk and external request size distribution.

non-Pareto but we find that they are extremely well approximated by a Cauchy distribution. The symmetric Cauchy distribution has a pdf given by

$$p(x) = \frac{1}{\pi} \frac{s}{s^2 + x^2}$$

where $s > 0$. In our case we only use the positive half of the distribution and therefore have to multiply the equation above by a factor of 2. For large x the distribution behaves like $1/x^2$ so the cumulative distribution function (cdf) therefore obeys Zipf's law. The distribution is distinguished from the Pareto distribution by its behaviour for small x. This is easily missed when linearly sized bins are used to create the histogram – hence the value of choosing exponentially scaled bin sizes. Although qualitatively the approximation for small file sizes is not brilliant, it is significantly better than the Pareto assumption as can be clearly seen in figure 5. A Pareto distribution does not describe the request sizes smaller than 1 Kbyte at all.

6 Modelling the Traffic

We have now characterised the measured network traffic with its inter-event time histogram, the power spectrum of the aggregated time series and the change in the rates of packets seen on the network.

In this section we describe a simulation study that makes simple assumptions in line with our measurements. We are going to look at a single server queue where jobs arrive according to a Poisson process and where the service requirement is distributed according to a (truncated) Cauchy distribution. The server itself removes work from the queue in blocks (Ethernet frames) and waits after each frame for a short period (inter-frame gap). The blocks are all of the same size, i.e. we only have frames of one size.

This very simple model should reflect some the features seen in the Ethernet traffic of file or web server. We do not take the TCP connection build-up, close down or acknowledgements into account. This may be a serious deficiency of our model. However, the advantage is that we can attempt to model the traffic of both UDP-based (like NFS) and TCP-based (like web servers) systems.

Our main focus will be the inter-departure time histogram and the power spectrum or the auto correlation function of the departure process of this model. We use our earlier measurements to parameterise this process.

6.1 Simulation

In this section we present a very simple model of network traffic exhibiting scaling features (see figure 6). In our

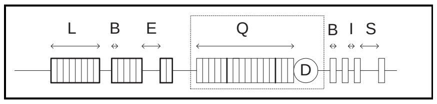

Figure 6. A sketch of our model.

model, requests of size L arrive at webserver. Their sizes L, measured in units of blocks B of 1518 bytes, are distributed according to a Cauchy distribution. The time between requests E is exponentially distributed. The server D emits each block after a service time corresponding to the network speed. The minimum time between blocks after service is the inter frame gap I of twelve bytes. The time between blocks can be larger than I, namely S, if the server was idle. The queue-length Q is measured as the number of blocks awaiting service. The choice of the request size distribution was motivated by the observation in Figure 5. The Cauchy distribution seems to describe the features of the empirical distribution well and has the advantage of being analytically tractable and easy to implement. The full Cauchy distribution has pdf

$$\tilde{p}(x) = \frac{1}{\pi} \frac{s}{s^2 + x^2} \text{ where } s > 0$$

but here we use a truncated Cauchy distribution whose pdf is given by:

$$p(x) = \begin{cases} \tilde{p}(x)/C & 0 \leq x \leq x_{\max} \\ 0 & \text{else} \end{cases}$$

where C is a normalisation constant

$$C = \int_0^{x_{\max}} \tilde{p}(x) dx.$$

The truncation of the Cauchy distribution gets rid of its usually prohibiting features like infinite moments. Neglecting all other factors like disk and CPU access caused by a request, the blocks requested get added to the input queue of a server that represents the network interface. We assume that all blocks leave the server at a given network speed. The server spends the same amount of time serving each block and then waits before the next block is processed, similar to the effect of the inter-frame gap on an Ethernet. The model has been implemented in JAVA.

We measure the cumulative queue length at the server to make sure the system is in equilibrium. Each departure from the system is logged and the time series of departures is analysed like those of real network traffic in previous sections. For the power spectra, we aggregated packet counts in 0.01 second bins.

To parametrise the model we used the log file of the departmental webserver covering the period from 12pm to 2pm on 22 March 2002. As the reader will recall, we have already presented an analysis of the traffic going through the Black Diamond router. Figure 5 shows the request size distribution for that period and compares it to a Cauchy distribution with $s = 1500$. If we truncate this distribution at 10,000 blocks we get a mean of 11.3 blocks per request and a standard deviation of 112 blocks. This compares to a mean 16 blocks and a standard deviation of 248 blocks with the measurements. The main reason for the discrepancy is likely to be the bad fit of the empirical distribution for requests larger than 1,000 kbytes, see figure 5. Empirically there were 4.6 external requests per second and in the simulation we used 5 requests per second. We make the assumption that the requests are Poisson, partly motivated by the desire to keep the model as simple as possible, given that we cannot tell from the logfile when the requests arrived, only when their service finished. However, there are also sound theoretical reasons for this assumption, see Section 7.2.

We ran the simulation 11 times with a simulation time of about 4.5 days for three different network speeds: 1Mbit/sec, 10Mbit/sec and 100Mbit/sec for the same arrival process specified above. In each case it took at least hours if not days to reach equilibrium. The average queue lengths were 1258 ± 47, 41.6 ± 1.4 and 3.97 ± 0.15 blocks respectively.

We analysed the last two hours of each simulation run similarly to the analyses of real network traffic before. We ran the simulation with different network speeds, the increase in network speed appears to shift the frequency interval governed by a power spectrum to the higher frequencies, see figure 7. This is presumably caused by the blocks being processed at a higher rate. The shift is approximately one 'decade' (order of magnitude) for each speed increase by a factor of ten. The slope of the power law is not affected at all by the network speed.

To compare our simulation at the network level with the measurements made at the Black Diamond router, we filtered the traffic trace discussed in section 5 for packets originating from an internal webserver which are bigger than one Kbyte (to coincide with the model). In total there were 556,907 packets fulfilling these criteria. Their mean inter arrival time was 13 msec with a standard deviation of 68 msec. The simulations generally produce around 410,000 events and have inter-departure times with mean around 20 msec and standard deviations of 50 to 80 msec, depending on the network speed. When we compare the power spectra, we find that the simulation corresponding to a network speed of 10Mbps fits best, see figure 7. Still, the simulation shows a much more distinct power law. This is likely to be related to factors that our model has neglected, like CPU, I/O waits and competition with other network traffic. Competition with other network traffic might also account for the fact that the empirical data agrees more with the 10Mbps simulation rather than the 100Mbps even though the webserver does have a 100Mbps network connection. However, the slope of the power law in the power spectrum based on the simulation and the empirical data are very similar.

To demonstrate the effect of the power law in the distribution of the request sizes, we ran our simulation with an exponentially distributed request size. The mean request size was 16 blocks per request. While the model manages to reproduce the mean inter arrival time well, it fails to capture the power spectrum, see figure 7. In the simulation, the power law behaviour of the request size distribution can also be seen in the distribution of changes in the packet rate. The results are similar to the measurements we made at the CPU server, figure 3. This similarity supports the hypothesis that the self-similarity seen at the CPU server is largely caused by the file size distribution.

The inter-departure time histogram of the simulation has a spike at the shortest time between transmissions, i.e. one block plus the inter frame gap, caused by the busy periods. The remainder is an exponential distribution caused by the gaps between an idle period and the next request arrival.

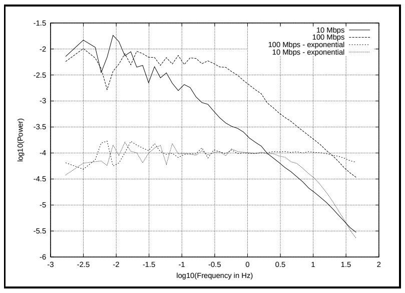

Figure 7. This plot compares the power spectra of the simulation using Cauchy distributions to ones using exponential distributions for the request sizes at different network speeds.

7 Interpretation

7.1 Comparison with archive data

In addition to data collected locally, we also analysed the Ethernet data sets in the Internet traffic archive [2]. This is the data that has been measured and discussed in [8]. We used the following files: pAug.TL.Z, pOct.TL.Z, OctExt.TL.Z and OctExt4.TL.Z. The first two are shorter traces of one million arrivals measured on the internal Ethernet, the latter two are longer traces measuring the external arrivals only. For a more detailed description see [8].

The time-stamps in the files have microsecond granularity although accuracy is limited to around 10 microseconds by the hardware clock (four microseconds) and bus contention. However, this is still (just) enough to distinguish between packets as there is around a 10 microsecond silence between packets on a 10Mbps Ethernet and the smallest packet lasts about 60 microseconds.

We applied the same methods used to analyse our own data to the LBL data investigated in [8]. Looking at the inter-departure time histograms in figures 1 and 8, the internal traces are more similar to our data. This data also exhibits the same two peaks corresponding to the smallest and largest packet size on Ethernets. Note, though, that the peaks are shifted to the right by an order of magnitude due to the different speed of the measured Ethernets. This seems to suggest that the measurement errors for the time stamps are not as bad as the authors of [8] thought. It also suggests an interesting way to measure the speed of an unknown network.

Just like our data, the internal traces do not provide a power law in the inter-departure time distribution. There is also a similar picture when we look at the power spectra in

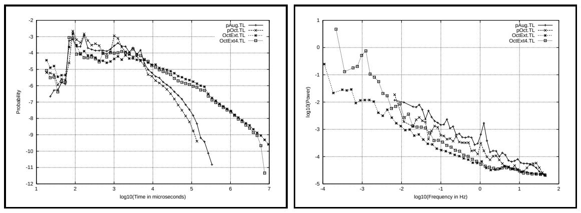

Figure 8. This plot shows the inter arrival time histograms for the four LBL traces. The power spectra of the four LBL traces, based on an aggregation of 10 msec.

figures 8 and 1. The range of the power laws are similar. The external traces seem to provide a better power law than the internal ones. There is, however, a difference in the data when we look at the changes in the packet rates. Figure 9 shows a linear plot for the internal sources. In contrast to

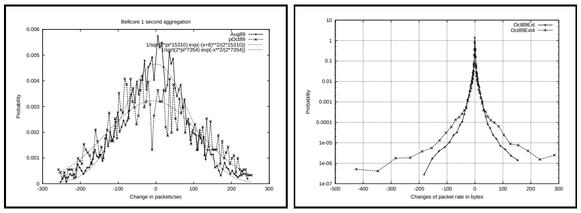

Figure 9. The change in the packet rates for the internal traces as a linear plot. The two lines are normal distributions with the same mean and average as the measured data. The change in the packet rates for the external traces as a log-linear plot.

the data we measured, see figures 3 and 4, the internal data shows no sign of a power law for this metric. However, the external traces plotted in figure 9 show a similar leptokurtic behaviour. The slope in the log-log plot is different to ours; in the Bellcore data it is approximately -2.4.

In summary the data we have gathered shows a slightly different quantitative behaviour compared to the Bellcore data, especially in the change of the packets' rates. However, there are many qualitative similarities, for example in the inter-departure time histograms and the power spectra.

7.2 Analytical modelling

The comparison between the properties of the traffic generated by the simulation and that observed at the Black Diamond router suggest that file size distribution can be a sufficient cause of power laws, self-similarity and long range correlation. This is the case even for Poisson external arrivals, which were used in the simulation. Although we have not tested the hypothesis that external request instants approximate Poisson streams, this has been found to be the case for many decades in a wide range of teletraffic systems. In fact, the hypothesis has sound theoretical foundations. It can be proved that a superposition of arrival processes is asymptotically Poisson under appropriate conditions; for example a large number of independent renewal processes of which none is dominant, i.e. has a much higher rate than the average – see [19, 13] for example.

The output traffic from a server comprises (a) a sequence of constant-size packets separated by the small interframe gap when the server is busy (with *any* work); (b) a null stream, when the server is idle. Put in queueing terms, the output is an on-off process with constant inter-event times during the on-periods, which are the *busy periods* of the queue, and exponentially distributed off-periods, with parameter equal to the arrival rate to the queue. Clearly, for heavy-tailed service times, the busy period must also be heavy tailed, although the precise calculation is complex.

We therefore set up another simulation experiment to test the hypothesis directly that on-off traffic with Cauchy (to characterise the heavy tail) on-periods and exponential off-periods follows a power law. We used the same Cauchy distribution as in the simulation of Section 6.1. Thus, at low queue utilisations, the output of this simulation must be close to that of the simulated queue in Section 6.1 since busy periods would be the same as service times with high probability. Consequently, in this case, the output would also match the observed traffic. As expected the simulation output did follow a power law for various exponential distribution parameter values.

Whilst this does not fully justify our conjecture – we would need to determine the actual busy period distribution for this – it provides considerable support. If true, it would mean that many internet sub-networks could be modelled by conventional queueing networks with Poisson external arrivals and general (here Cauchy) service times. Moreover, servers rarely operate first-come-first-served (FCFS) and processor sharing (PS) is a better representation. Thus, by the BCMP Theorem [4], product-form solutions exist for these sub-networks, from which performance measures like mean queue length, throughput and mean response time follow via recursive algorithms that can be implemented efficiently. Moreover, such models can easily accommodate multiple classes of traffic (with different demands at each node and different paths through the network). In some sense, one of the conclusions of our analysis in this paper is that we need not have bothered, product-form queueing networks are fine! However, we have gained more insight and backed up our conjecture by systematic experimentation. In fact, the actual internal processes are important for other purposes, such as jitter analysis and computation of higher moments than the mean of response time. Furthermore, it is not always the case that PS queueing discipline is faithful to reality; if it is not and, for example, FCFS is appropriate, the product-form solution is lost and with it the efficient computation of performance measures. The same applies if servers are correlated, as is often observed.

8 Conclusion

The following main contributions have been made by this paper:

- `/proc/net/dev` has been proposed as an effective and efficient alternative to `tcpdump` for monitoring network traffic. This was supported by comparing observations made using both methods.
- The results of measurements taken from three locations on a state-of-the-art switched Ethernet have been investigated. These are publicly available via the world-wide web.
- Measured traffic at these locations was shown to fit closely various (truncated) Cauchy distributions, which have heavy tails.
- Changes in packet rates (first difference of the packet-density time series) were also shown to conform to a Cauchy distribution, revealing a possible link with the observations of the server traffic.
- Simulation of a specific M/G/1 queue supports the hypothesis that transmission of a (truncated) Cauchy-distributed number of Ethernet packets by a fixed-rate communication device produces a time series of (aggregate) packet densities whose power spectrum matches closely that of observed network traffic.
- By the nature of the server-nodes, traffic conforming to the observed power law should be generated by an on-off process with exponentially distributed off periods and on periods distributed as the busy time of the server. We postulated that this busy time should be heavy-tailed and a simulation experiment based on this did indeed produce traffic with the right kind of power law.
- This suggests a possible internet modelling strategy based on processor sharing and conventional (product-form) queueing networks.

In our future work we will investigate better fitting methods for the distributions we have discussed here. We would also like to make our simulation model more realistic without introducing unnecessary complications.

Also, we would like to investigate how we can incorporate results in [31, 26, 27, 7, 22], which indicate that the congestion avoidance algorithm of TCP [16] introduces power laws, into to our model. Perhaps combining our findings with a model based on dynamical systems will shed more light on the nature of network traffic. In mathematical and theoretical physics self-similarity has been studied widely over the last few decades, motivated by the widespread occurrence of $1/f$ noise in natural phenomena. A possible interpretation of $1/f$ noise is the notion of self-organised criticality (SOC) [3, 17]. The word criticality is borrowed from physics where a critical state of a system is related to an infinite correlation length and the system going through a phase transition. There are many areas in science where $1/f$ has been seen: see for instance Jensen's book [3, 17] for a good overview of the topic.

Acknowledgements

The authors would like to thank the Computer Support Group in helping with the data capture. We would also like to thank Jörn Davidsen of the Chemical Physics Theory Group (Department of Chemistry, University of Toronto) for fruitful discussions on self-similarity and criticality, and Will Knottenbelt and David Thornley for stimulating conversations. The research was funded by EPSRC (research grant QUAINT, GR/M80826).

References

[1] P. Abry and D. Veitch. Wavelet analysis of long-range-dependent traffic. *IEEE Transactions on Information Theory*, 44(1):2–15, 1998.

[2] ACM/SIGCOMM. The internet traffic archive. *http://www.acm.org/sigcomm/ITA/*.

[3] P. Bak, C. Tang, and K. Wiesenfeld. Self organised criticality: an explanation of 1/f noise. *Physical Review Letters*, 59:381, 1987.

[4] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Closed and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, April 1975.

[5] J. Beran. *Statistics for Long-Memory Processes*. Chapman & Hall, 1994.

[6] M. E. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846, 1997.

[7] A. B. Downey. The structural cause of file size distributions. *SIGMETRICS/Performance*, pages 328–329, 2001.

[8] W. E. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1):1–15, February 1994.

[9] A. Erramilli, O. Narayan, and W. Willinger. Experimental queueing analysis with long-range dependent packet traffic. *IEEE/ACM Transactions on Networking*, 4(2):209–223, 1996.

[10] ExtremeNetworks. http://www.extremenetworks.com/.

[11] J. Feder. *Fractals*. Plenum, 1988.

[12] K. Fukuda et al. Origin of critical behaviour in ethernet traffic. *cond-mat/0007435*.

[13] P. Harrison and N. Patel. *Performance Modelling of Communication Networks and Computer Architectures*. Addison-Wesley, 1993.

[14] T. Huisinga et al. A microscopic model for packet transport in the internet. *Physica A*, 294:249–256, 2001.

[15] G. Irlam. Unix file size survey. http://www.base.com/gordoni/ufs93.html.

[16] V. Jacobson. Congestion avoidance and control. *ACM Computer Communication Review Proceedings of the Sigcomm '88 Symposium in Stanford, CA*, 18(4):314–329, August 1988.

[17] H. Jensen. *Self-Organised Criticality*. CUP, 1998.

[18] A. Khintchine. Korrelationtheorie der stationären prozesse. *Mathematische Annalen*, 109:604, 1934.

[19] H. Larson and B. Shubert. *Probabilistic models in engineering sciences*. Wiley, New York, 1979.

[20] B. Mandelbrot. *The Fractal Geometry of Nature*. Freeman, 1982.

[21] R. N. Mantegna and H. E. Stanley. *An Introduction to Econophysics*. CUP, ISBN 0 521 62008 2, 2000.

[22] D. Newman, N. D. Sizemore, B. Carreras, and V. Lynch. Growth and propagation of disturbances in a communication network model. *Hawaii International Conference on Systems Sciences*, January 2002.

[23] B. Pilgram and D. T. Kaplan. A comparison of estimators of $1/f$ noise. *Physica*, D 114:108–122, 1998.

[24] W. Press et al. *Numerical Recipes in C 2nd ed.* CUP, 1993.

[25] QUAINT. Measurement data. *http://www.doc.ic.ac.uk/~uh/QUAINT/data/*.

[26] H.-P. Schwefel. Behavior of tcp-like elastic traffic at a buffered bottleneck router. *http://citeseer.nj.nec.com/451117.html*.

[27] B. Sikdar and K. S. Vastola. The effect of tcp on the self-similarity of network traffic. *http://citeseer.nj.nec.com/sikdar01effect.html*.

[28] M. Taqqu, V. Teverovski, and W. Willinger. Estimators for long-range dependence: an empirical study. *Fractals*, 3(4):785–798, 1995.

[29] Tcpdump. http://www.tcpdump.org/.

[30] S. Thurner et al. Analysis, synthesis, and estimation of fractal-rate stochastic point processes. *Fractals*, 5(4):565–595, 1997.

[31] A. Veres and M. Boda. The chaotic nature of TCP congestion control. *INFOCOM*, (3):1715–1723, 2000.

[32] J. Voit. *The Statistical Mechanics of Financial Market*. Springer, ISBN 3 540 41409 6, 2001.

[33] N. Wiener. Generalized harmonic analysis. *Acta Mathematica*, 55:117, 1930.