Using Signal Processing to Analyze Wireless Data Traffic*

Craig Partridge, David Cousins, Alden W. Jackson,
Rajesh Krishnan, Tushar Saxena, and W. Timothy Strayer
BBN Technologies
10 Moulton Street, Cambridge, MA 02138
{craig, dcousins, awjacks, krash, tsaxena, strayer}@bbn.com

ABSTRACT

Experts have long recognized that theoretically it was possible to perform traffic analysis on encrypted packet streams by analyzing the timing of packet arrivals (or transmissions). We report on experiments to realize this possiblity using basic signal processing techniques taken from acoustics to perform traffic analysis on encrypted transmissions over wireless networks. While the work discussed here is preliminary, we are able to demonstrate two very interesting results. First, we can extract timing information, such as round-trip times of TCP connections, from traces of aggregated data traffic. Second, we can determine how data is routed through a network using coherence analysis. These results show that signal processing techniques may prove to be valuable network analysis tools in the future.

Categories and Subject Descriptors

C.2.0 [**Computer Communication Networks**]: General - Security and Protection; C.4 [**Performance of Systems**]: Measurement Techniques

General Terms

Algorithms, Security, Measurement

Keywords

traffic analysis; signal processing; encryption; wireless networks

1. INTRODUCTION

Network security experts have long known that examining even subtle timing information in a traffic stream could, in theory, be exploited to achieve effective traffic analysis [15]. Consider the packet arrival pattern in a TCP flow. The pattern is a function of a number of key network parameters such as round-trip times, send rates, and various TCP and MAC layer timeouts, as well as the values for all other flows that share network links with the flow in question [18]. In theory, therefore, a trace of packet arrivals should be a possibly noisy composite of all of these patterns.

The problem of extracting characteristics from an otherwise noisy environment is very similar to the extraction of features from sonar data. Sonar signals are passed through sophisticated signal processing filters to identify the signals that have structure not otherwise visible.

The key idea, then, is to convert packet traces into signals, and then examine the signals to identify prominent recurring frequencies and time-periods. With an effective signal encoding, many well-known frequency analysis techniques from the signal processing literature can be applied. We use the frequency analysis techniques to perform traffic analysis to reconstruct the network topology or extract network traffic parameters.

In this paper, we consider the use of techniques similar to those employed in acoustics processing to do traffic analysis in the presence of noise, whether the noise is inherent in the traffic stream or placed there intentionally to camouflage the interesting traffic flows. We take packet traces of streams and convert them into signals suitable for signal processing. We then show examples of the kind of information that can be extracted from the signals using two techniques: Lomb Periodograms and Coherence.

2. DESIRED RESULTS

There is a wide range of questions that one might ask a traffic analysis system to answer. We, however, had particular types of results in mind when we began our work with signal processing techniques.

We assumed an environment in which senders seek to mask or hide their traffic using techniques such as tunneling, traffic aggregation, false traffic generation, and data padding. Tunneling hides the original source and ultimate destination and uses security gateways as the endpoints as traffic traverses hostile networks. Traffic aggregation works with tunneling under the theory of protection in numbers—many traffic flows all sharing the same tunnel may mask any one particular flow's characteristics. If there is not enough aggregated traffic to hide individual flows, false traffic can be generated to help hide the traffic of interest. Data padding tries to hide information that can be extracted from the

*This work was sponsored by the Defense Advanced Research Projects Agency (DARPA) under contract No. MDA972-01-C-0080. Views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied.

WiSe'02, September 29, 2002, Atlanta, Georgia, USA.

packet length.

We then sought techniques which answered one or more of the following questions:

- Who is talking to whom? Ideally, we would be able to identify each individual application endpoint. However, a very useful result would be to determine, for instance, how many different sites are sending their traffic over the same IPsec tunnel.
- What path is traffic taking over the network? This question is of particular interest in wireless networks (where determining how traffic is routed is difficult), but may also be useful in multi-tunnel environments such as Onion Routing [20].
- What types of application data are being sent? Are we seeing interactive applications or file transfer applications or both?
- Can we associate transmissions with a particular flow? For instance, if we determine that five concurrent flows are underway over an IPsec tunnel, can we (with high probability) determine which IPsec packets are associated with which flow? If we could break aggregate flows into their components, we could potentially use additional traffic analysis tools that are tuned to single flows (e.g., the password inference technology developed by [22]).

3. RELATED WORK

Signal processing has been used to analyze the nature of aggregate network traffic, and to develop accurate models of traffic consisting of asymptotically large number of flows, such as the traffic on a large intranet, or on the Internet backbone [4, 2, 16]. It has been shown that aggregate traffic on the Internet is self-similar, or shows long-range dependence [16]. Self-similarity means that no single time-scale completely captures the rich behavior of the aggregate network traffic. This observation implies that one needs to describe the evolution and steady progression of characteristics (such as the number of active TCP connections or the distribution of IP packet interarrival times) of aggregate network traffic across all scales, because no single scale can describe all of the fluctuations and variations [2, 9]. This observation has led to work on long-term memory models, self-similar models and models with fractal features, where signal processing tools such as the Wavelet transform are especially applicable because of their ability to capture frequency responses at various scales simultaneously [12, 3].

Though the work on the nature of aggregate network traffic is relevant to the material presented in this paper, the general focus of our work is not to model aggregate traffic, but rather the inverse problem, but to *deconstruct* the traffic into individual flows, or sessions.

Another related area is that of *network tomography* [24, 6]. Network tomography is concerned with identifying network bandwidth, performance and topology by taking measurements, either actively from the network nodes, with their cooperation [7, 8], or passively using measurements from preexisting traffic [23, 5]. Most network tomography work has also dealt exclusively with network monitoring and inference of wired networks such as the Internet ([6]). Moreover, traditional network tomography relies on the ability of the measuring agents to be able to participate in the communications, possibly at the network layer. The participation may either be in the form of the ability to take measurements, or even the ability to explicitly transmit packets to other nodes in the network.

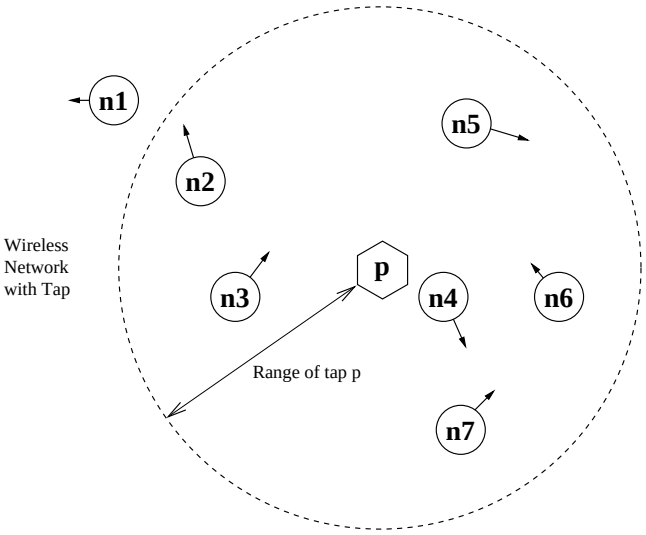

Figure 1: Wireless network with nodes (n1-n7) and tap (p)

However, in some scenarios, such as in adversarial wireless networks, we cannot assume that the measuring agents can participate on the network. Indeed, in many military domains, the nature of the network protocols deployed on the adversary's network may not even be known. As a result, the work in this paper makes far more conservative assumptions about what a measuring agent may do. Our aim is to discover network topology purely from the raw transmission traces.

4. NETWORK AND TAP MODEL

Our goal in this work is to make the traffic analysis techniques broadly applicable. To that end, we make as few assumptions about the network and the observed traffic as possible.

We assume that there is some *network* over which discrete pieces of data are transmitted by *senders*. The transmission of these pieces of data cause network *events*. An *event* is individually detectable or distinguishable—that is, a listening device can tell when an event is over and will not combine concurrent events from multiple senders into one event. It is important to note that an event need not perfectly correspond to a data packet. An event may represent the transmission of part of a packet (e.g., a frequency hop), or multiple packets (say two packets contained in a single wireless burst transmission).

A sender in this model is the device that caused the event. The sender is not necessarily the device that actually originated the data that caused the event.

We assume that there are one or more *traffic taps* within the network. A tap seeks to observe traffic on as much of the network as is possible from the tap's location. This broad definition is chosen to accommodate the difference between a tap on a wire or fiber, where the tap is restricted to data

placed on the wire, and a wireless tap, which is observing some (potentially very large) fraction of the wireless spectrum, and thus may see transmissions from a wide range of sources. This range is shown in Figure 1.

A tap collects event information in a *trace*. For most of the work discussed in this paper, the trace is assumed to contain only the *time* the event was seen and the *identity* of the sender of the event.

The concept of identity used here is intentionally vague—the identity could be the IP address of an IPsec gateway, the location of a radio transmitter, the upstream or downstream transmitter on a point-to-point link, or simply "the same sender as the one that sent these other events." The identity of a sender must be unique among all senders known to the tap (or set of cooperating taps); we assume the data collection process is setting identity and maintaining the uniqueness property.

We assume each tap has access to a clock used to record when when each event was heard. In a wireless network, this time of detection may be the middle of the transmission due to propagation or other effects such a frequency hopping. The granularity of the clock used to record time must be sufficiently small that two consecutive events on the same channel will be given different timestamps.

We note that there is no assumption about knowledge of the length of the event, the destination of the data corresponding to the event, signal strength, or any insight into the contents of the event, even though, in many cases, this and other additional information may be available. How this additional information might be used is discussed in later sections.

A tap may not capture all traffic. For instance, reception on a wireless network may be variable due to environment, noise, transmission power, or jamming such that a tap is unable to observe some transmissions. Furthermore, a tap may occasionally make an error and mistakenly believe it has seen an event when no event was sent (e.g., due to noise on the wireless network).

There are some other characteristics of taps worth commenting on:

Multiple taps: Multiple taps may be used together to develop a more complete picture of the network traffic.

Resource limitations: A tap (or a network of taps) must be capable of storing all the transmissions it detects for a sufficient amount of time for analysis to take place. For example, the round-trip time of a transport layer flow cannot be determined if the history that can be stored at taps is less than one round-trip time. The total volume of data that must be stored depends on the capacity of the channel and the maximum round-trip time of flows seen on the channel.

In the wireless environment, a tap may also be limited by the amount of spectrum it can examine in any given time. Indeed the spectrum range covered by the tap may be different from the spectrum range used by the sender, with the result that some events are not observed.

Mobility: Nodes may move around the network. Thus senders may move in and out of the range of one or more taps. We assume that senders typically dwell in the range of one or more taps long enough for events to be heard, and the senders identified and recorded.

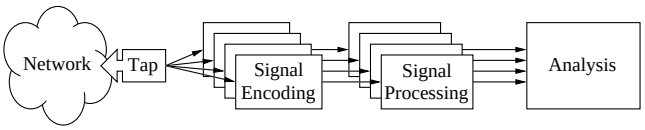

Figure 2: Model of Analysis

5. A NOTE ABOUT DATA SOURCES

Even though the techniques described in this paper have all been tested on real wireless data, the examples presented here all use simulated network data. We chose to present simulated data for two reasons.

The first reason is that, so far, we have not had the equipment to collect the kinds of wireless traces we need. Rather, we have taken existing traces and attempted to adapt them. So, for instance, one wireless data set we have used is a *tcpdump* trace of the wireless data and lacks the MAC layer ACK and RTS signals, and has deleted any errored packets. As a result, some of the key frequency information is lost. (One paradoxical consequence is that real data actually makes some results look better than they should because confusing signals have been edited from the traces).

The second reason is that no real trace, so far, has come with all the required "ground truth" data needed to cross-check results. So real data often involves making guesses about the meaning of results.

Simulation data does not suffer these limitations. We have all the signals and can present them in all their complexity. And if we cannot explain a result from a simulation, it represents a serious challenge in interpretation, not the lack of the necessary supporting data. So, for the purposes of clear exposition, we have used simulation data.

6. SIGNAL ENCODING

Figure 2 shows our traffic analysis processing model. Traffic is captured from the network via taps. The traces from the taps are encoded into signals. The signals are then processed, using various signal processing techniques and the final result is analyzed. This is precisely the same model of analysis used in signal processing of acoustic data.

The first step in producing a signal is acquiring the samples. Signal processing makes a distinction between whether the samples are gathered by a uniform or non-uniform sampling process. The type of signal produced must be appropriate for the target signal processing algorithm. With data traffic, the major concern is that the sampling frequency allow the separation of meaningful events. We assume the sampling process meets event separation criterion. Given separation, we can convert a trace into an event stream that is appropriate for any target signal processing algorithm.

The trace represents a set of discrete events $x(n)$, logged at times t_n, for $n = 0, \ldots, N$, where N is the number of events in the trace. The general approach to producing a uniformly sampled signal representing the time of arrival of event $x(n)$ is to pick an appropriate time quantization interval T, bin time into increments at that quantization mT, where m is a integer, and then place a marker in the bin representing the nearest time to t_n when the event $x(n)$ was detected. That is, $mT = q(t_n)$, where q is the quantization function such as the floor or the ceiling function. The

```
Time     Duration T    R O D Description
-----------------------------------------------
3.582728 0.004336 2 /* 3 0 3 tcp frame 1084 B */
3.587075 0.000152 3 /* 2 3 2 MAC ctrl frame   */
3.587417 0.000176 3 /* 2 3 2 MAC ctrl frame   */
3.587604 0.000152 2 /* 3 2 3 MAC ctrl frame   */
3.587807 0.000496 3 /* 2 3 0 ack frame 124 B  */
3.588313 0.000152 2 /* 3 2 3 MAC ctrl frame   */
3.588596 0.000176 3 /* 2 3 2 MAC ctrl frame   */
3.588783 0.000152 2 /* 3 2 3 MAC ctrl frame   */
3.588986 0.000496 3 /* 2 3 0 ack frame 124 B  */
3.589492 0.000152 2 /* 3 2 3 MAC ctrl frame   */
3.589934 0.000176 2 /* 3 2 3 MAC ctrl frame   */
3.590121 0.000152 3 /* 2 3 2 MAC ctrl frame   */
3.590324 0.002384 2 /* 3 1 3 udp frame 596 B  */
3.592719 0.000152 3 /* 2 3 2 MAC ctrl frame   */
3.593041 0.000176 3 /* 2 3 2 MAC ctrl frame   */
3.593041 0.000176 2 /* 3 2 3 MAC ctrl frame   */
3.593736 0.000176 1 /* 2 1 2 MAC ctrl frame   */
3.593923 0.000152 2 /* 1 2 1 MAC ctrl frame   */
3.594125 0.002384 1 /* 2 1 3 udp frame 596 B  */
3.596520 0.000152 2 /* 1 2 1 MAC ctrl frame   */
3.597082 0.000176 2 /* 3 2 3 MAC ctrl frame   */
3.597268 0.000152 3 /* 2 3 2 MAC ctrl frame   */
3.597471 0.002384 2 /* 3 1 3 udp frame 596 B  */
3.599866 0.000152 3 /* 2 3 2 MAC ctrl frame   */
3.600169 0.000176 2 /* 0 2 0 MAC ctrl frame   */
3.600355 0.000152 0 /* 2 0 2 MAC ctrl frame   */
3.600558 0.000496 2 /* 0 3 0 ack frame 124 B  */
3.601064 0.000152 0 /* 2 0 2 MAC ctrl frame   */
3.601327 0.000176 2 /* 3 2 3 MAC ctrl frame   */
```

Figure 3: Excerpt of trace capturing transmissions from four nodes of Figure 5. There is an FTP flow between nodes 0 and 3 and a pair of UDP flows between nodes 1 and 3. All traffic is routed through node 2. The *Time* and *Duration* of the transmissions, and the transmitter (*T*) node id, are captured by the tap. The extra information within (/* ... */) is listed here purely to give the reader an insight into the trace dynamics, and is not known to or captured by the tap. The extra info includes the receiver id. (*R*), the global origin (*O*) and destination (*D*) of the packet contained in this transmission, and a *Description* of the packet contents.

Nyquist limit provides the means for determining the size of the time increment; we aim to minimize the number of bins and yet meet the Nyquist limit. This process is known as *resampling*. Due to the errors introduced by quantizing the time of arrival, some information contained in $x(n)$ may be lost in the resultant encoding.

To produce a non-uniformly sampled signal representing the time of arrival of events $x(n)$, markers are placed only at times t_n. Since there is no resampling, no quantization error is introduced into the encoded signal.

The trace may be rich with information that can be encoded as a signal. Consider a function g as the encoding function. For a binary, or impulse, representation of time of arrival, $g(mT)$ is 1 when $mT = q(t_n)$, and 0 otherwise. A sign encoding function $(+1, -1)$ can be used to indicate which end of a wire the signal came from. A weighted encoding function can represent the transmission duration or signal strength. Additional parameters for each event can be represented in the signal by refining the encoding function g.

When multiple events are occurring simultaneously (i.e., within the same sample period) and would be set to the same time bin mT, we jitter the time of the conflicting events into empty adjacent sample times in order to keep data from being obscured.

While it is possible to encode the events of multiple senders into a single signal, better signal processing results usually come when one generates a separate signal representation for each sender. Recall that the sender is the most recent transmitter of the data that caused the event—it is not the originator of the data. Thus, a single sender's trace may contain the data of multiple flows (e.g., when the sender is a router). The idea here is simply to split the traces as much as possible before processing.

An example of a trace captured by a tap monitoring transmissions in a wireless network is shown in Figure 3. As discussed earlier, a duration-weighted sign encoding function can be used to encode the captured transmissions into a signal appropriate for signal processing. Figure 4 shows an encoding of transmissions from nodes 2 and 3, which can be used to analyze communications that span these nodes.

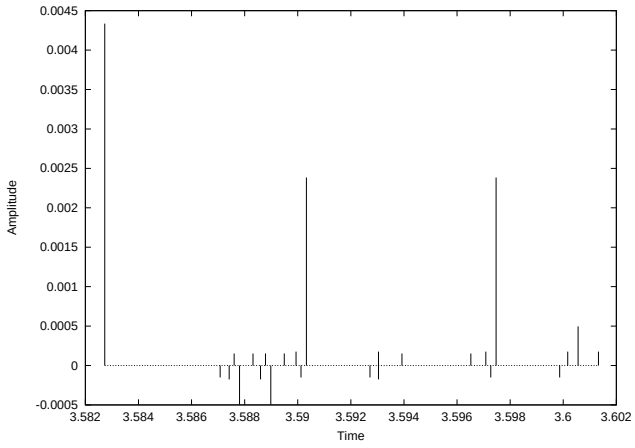

Figure 4: A non-uniformly sampled signal representation of the trace in Figure 3. $f = (1 \times$ duration$)$ for transmissions of node 2 and $g = (-1 \times$ durations$)$ for transmissions from node 3.

7. SIGNAL PROCESSING AND ANALYSIS

Given an encoded signal, we can make use of a wide range of signal processing algorithms to try to extract traffic information. In this section, we will describe some signal processing techniques which we have found useful for trace analysis.[1]

Most spectral processing techniques use the standard Discrete Fourier Transform (DFT) to compute the spectral power densities. The DFT requires that the signal be uniformly sampled.

The DFT of a uniformly sampled signal $x(n)$ (with $M = q(t_N)/T$ samples) provides an M-point discrete spectrum

[1]Unless otherwise noted, more information about these techniques can be found in signal processing textbooks such as [14].

$X_M(k)$, where

$$\begin{aligned} X_M(k) &\equiv \sum_{n=0}^{M-1} x(n)\left(cos(\frac{2\pi kn}{M}) - j\ sin(\frac{2\pi kn}{M})\right) \\ &\equiv \sum_{n=0}^{M-1} x(n)e^{-j2\pi kn/M} \\ &\equiv \text{DFT}\{x(n)\} \end{aligned} \quad (1)$$

is the M-point DFT. The values of k correspond to M equally spaced frequency bins of the sampling frequency of x.

The resulting spectrum $X(k)$ is a vector of complex numbers. The peak values in $X(k)$ correspond to frequencies of event times of arrival. The magnitudes of the peaks are proportional to the product of how often the arrival pattern occurs and the weighting of the data performed by encoding the signal. The phase of the peaks shows information on the relative phases between arrival patterns. The Fast Fourier Transform (FFT) is a computationally efficient decomposition of Equation 1, made possible when M is a product of powers of small integers, though powers of two are the most commonly used.

If the characteristics of the signal (due to variations in the traffic flow) vary markedly during the DFT analysis, then the resulting spectrum can be misleading, since the resolved peaks may be present for only part of the time in the signal. Also, it is often the case with signal representations that the spectral content contains many harmonically related peaks.[2] In these situations, the spectral peaks of interest may not be readily visible due to the overlap of the various harmonic peaks, causing the spectra to look like noise. Thus, the examination of the spectrum given by the DFT can provide visualization of flows in the form of characteristic peaks, the DFT, when used alone, can give spectra that are insufficient for further detailed analysis. In the remainder of this section we describe signal processing techniques which address this deficiency.

Periodograms, or *Power Spectral Density* (PSD) estimators, are spectral analysis techniques that are used to compute and plot the signal power (or spectral density) at various frequencies. A periodogram can be examined to identify those frequencies that have *high power*, that is, power above a certain predetermined threshold. As a consequence, periodograms are useful for identifying *important* or *key* frequencies, even in the absence of any prior knowledge about the nature of the signal.

Another important characteristic of periodogram techniques is that they work very well even in the presence of noise or interference. This is fortunate for analyzing network traffic because a flow of interest is often embedded in an aggregation of other traffic. In this case, from the perspective of the flow of interest, all other traffic contributes to the interference.

When signals are expected to be noisy (i.e., they have a high degree of randomness associated with them due to corruption by noise or consisting of random processes themselves), conventional DFT/FFT processing does not provide a good unbiased estimate of the signal power spectrum.[3]

[2]For example, the spectral content of a square pulse is the fundamental frequency of the pulse, plus all the odd numbered harmonics.

[3]That is, processing larger sets of data does not make the answer converge to a good result.

A better estimate of the signal periodogram, $P_{xx}(k)$, may be obtained with the *Welch Averaged Periodogram* [25, 14] which utilizes averaging in order to reduce the influence of noise. It uses windowing to account for the aperiodic nature of the signal. The periodogram is generated by averaging the power of K separate spectra $X_M^{(r)}(k)$, computed over K different segments of the data, each of length L ($\leq M$):

$$P_{xx}(k) = \frac{1}{KU}\sum_{r=0}^{K-1}\left|X_L^{(r)}(k)\right|^2 \quad (2)$$

where

$$\begin{aligned} X_L^{(r)}(k) &= \text{DFT}\left[w(n)x_r(n)\right] \\ U &= \frac{1}{L}\sum_{n=0}^{L-1} w^2(n) \end{aligned}$$

where the windowed data $x_r(n)$ is the r^{th} windowed segment of $x(n)$, $w(n)$ is a windowing function[4] used to reduce artifacts caused by the abrupt changes at the endpoints of the window, and U is the normalized window power. The value of the number of samples L within each segment depends on the window function, $w(n)$. The result can be interpreted as a decomposition of the signal into a set of discrete sinusoids (at frequencies $2\pi k/M$) and an estimation of the average contribution (or power) of each one. While the spectrum, $X(k)$, obtained by the DFT was complex valued, the peaks in $P_{xx}(k)$ are real valued, they also correspond to frequencies of event times of arrival. Similar to the DFT, the power of the peaks is proportional to the product of how often the arrival pattern occurs and the weighting of the data performed by encoding the signal. In addition to this similarity to the DFT, the Welch Averaged Periodogram permits the computation of confidence bounds on the peaks.

7.1 Flow Analysis using Lomb Periodograms

Recall that DFT-based periodograms require uniform samples, which requires resampling of the original trace and may lead to loss of information. In this section, we discuss a technique which overcomes this hurdle.

Packet arrivals in computer networks are inherently unevenly spaced, naturally resulting in a signal encoding that is *non-uniformly* sampled. Lomb, Scargle, Barning, Vaníček [17, 19] developed a spectral analysis technique specifically designed for data that is non-uniformly sampled. The Lomb method computes the periodogram by evaluating data only at the times for which a measurement is available. Although the Lomb method is computationally more complex than the DFT ($O(N\log N)$), this property makes it an especially appropriate PSD estimator for examining event arrival traces. Moreover, since only the event arrivals need to be stored in the time series (no resampling, as discussed in Section 6, is required), the Lomb method has an added advantage that the input data is sparse and consumes less storage memory.

[4]The term windowing or shading refers to the time-wise multiplication of the data stream $x(n)$ by a smoothing function $w(n)$. Many typical smoothing functions are used (e.g., Hamming, Kaiser-Bessel, Taylor), all of which reduce spectral background noise and clutter levels at the cost of some smearing of the peak energies in the frequency domain.

So, at the cost of increased CPU requirements, but decreased memory requirements, the Lomb method offers all the attractions of periodograms, such as confidence intervals for various peaks, with the added advantage of a more precise power density computations for non-uniform time series.

The Lomb method estimates a power spectrum for N points of data at any arbitrary angular frequencies. The power density (P_N) at a frequency f Hz or angular frequency ω ($= 2\pi f$) radians/second is:

$$P_N(\omega) \equiv \frac{1}{2\sigma^2}\left\{\frac{\left[\sum_n (h_n - \bar{h})\cos\omega(t_n-\tau)\right]^2}{\sum_n \cos^2\omega(t_n-\tau)} + \frac{\left[\sum_n (h_n - \bar{h})\sin\omega(t_n-\tau)\right]^2}{\sum_n \sin^2\omega(t_n-\tau)}\right\} \quad (3)$$

Where

$$\bar{h} \equiv \frac{1}{N}\sum_{n=0}^{N-1} h_n$$

$$\sigma^2 \equiv \frac{1}{N-1}\sum_{n=0}^{N-1}(h_n - \bar{h})^2$$

$$\tau = \frac{1}{2\omega}\tan^{-1}\left(\frac{\sum_n \sin 2\omega t_n}{\sum_n \cos 2\omega t_n}\right)$$

Also, h_n $(n = 0, \ldots, (N-1))$ are the N unevenly spaced samples of the signal at times t_n. The Lomb periodogram is equivalent to least-squares fitting a sinusoid of frequency ω to the given unevenly spaced data. In case t_n are evenly spaced (i.e., the signal is uniformly sampled), the Lomb periodogram reduces to the standard squared Fourier transform.

Note that while analyzing network traces, it may sometimes be more convenient to work with time periods rather than angular frequencies. We will see this in the next section, where we take specific networks and illustrate the use of the Lomb method. The power density at a time period X can be easily computed since it is simply equal to $P_N(\omega = 2\pi/X)$.

7.1.1 Wireless Network Analysis

In wireless networks, we model taps as nodes that can detect transmissions above a certain signal strength threshold, and uniquely identify (and tag) each signal reception with its transmitting node. Consequently, a tap may only hear a subset of nodes in the network. Moreover, we do not assume that the taps participate (or, indeed, even know about the MAC layer) in the network. They only detect the lowest level physical transmissions.

Consider the four node wireless network in Figure 5. We simulated this network in *ns-2*, with an 802.11b MAC layer, and a 2Mb/s transmission bandwidth (we used the *ns-2* settings for Lucent WaveLAN). The nodes were deliberately placed in a configuration so that any traffic from nodes 0 or 1 to node 3 has to be routed through node 2, because node 3 is too far away and cannot directly hear nodes 0 and 1. Therefore, the wireless link between nodes 2 and 3 is the bottleneck link. Three flows were set up: One FTP flow from node $0 \rightarrow 3$, one CBR flow from node $1 \rightarrow 3$ and one CBR flow from $3 \rightarrow 1$.

We then place the tap p in the network such that it can only detect transmissions from nodes 0 and 3. The tap does

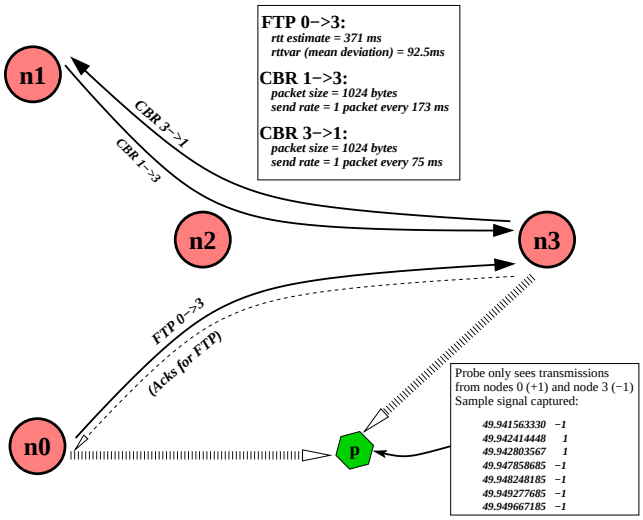

Figure 5: A wireless network with one FTP flow and two CBR flows. The network is configured to route traffic from nodes 0 and 1 to node 3 (and vice versa) via node 2. The tap is placed such that it only hears transmissions from nodes 0 and 3, and creates a simple signal encoding.

not hear any transmission from nodes 1 and 2 because node 1 is too far away, and node 2 is both far away and has low signal strength.

A simple signal encoding is created from the trace by assigning the amplitude $+1$ to all receptions from node 0, and -1 to all receptions from node 3. A small snapshot of this signal is shown in the box in Figure 5.

This simulation was run in *ns-2* for 300 seconds using the Dynamic Source Routing (DSR) protocol [13] to maintain connectivity in the ad hoc network. The CBR flow from $1 \rightarrow 3$ was configured to send packets of 1024 bytes each, at an average transmission rate of one packet every 173 ms. The CBR flow from $3 \rightarrow 1$ was also configured to send packets of 1024 bytes each, but at a rate of one packet every 75 ms. The statistics reported by *ns-2* for the FTP $0 \rightarrow 3$ was: round trip time (`rtt_`) of 371 ms, with a mean deviation (`rttvar_`) of 92.5 ms.

It should be noted that the trace produced by the tap in this network is complex and noisier than the trace would be on a wired network. This difference is not simply due to transmission media, but in the kinds of support traffic used in wireless networks. For instance, the events received at the tap include the DSR routing updates, which do not correspond to any end-to-end flow. Furthermore, due to the nature of 802.11b, the packet transmissions are interspersed with the corresponding RTS, CTS and MAC layer ACK transmissions [1]. Also, due to the nature of wireless networks, and the hidden-node problem, there are collisions which are resolved at the MAC layer, leading to retransmissions. Finally, there is interference in the signal from transmissions at node 3 that are not intended for node 1.

We are interested in identifying the characteristics of the various flows, so after collecting the signal from tap p, we compute the Lomb periodogram of that signal. Inspection of the Lomb periodogram plot shown in Figure 6 reveals that its three most prominent peaks correspond to each of

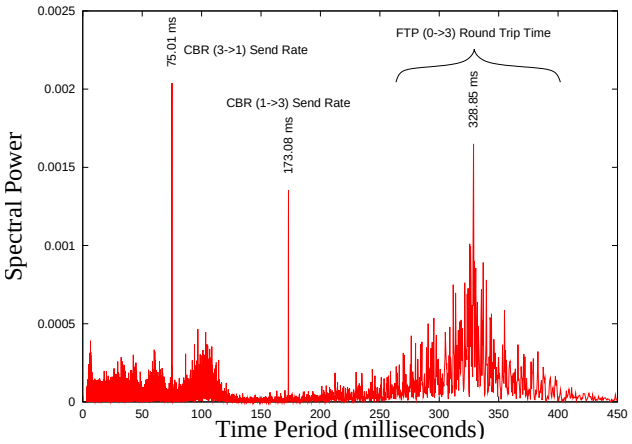

Figure 6: The Lomb periodogram for the wireless network of Figure 5 reveals all three flows involving four nodes, even though the tap only hears nodes 0 and 3. The $0 \rightarrow 3$ FTP is identified by the peaks spread near its RTT (328.85 ms).

the three flows.

Both CBR flows are revealed by the peaks very close to their transmission rates. The transmission intervals for CBR $3 \rightarrow 1$ and CBR $1 \rightarrow 3$ from Figure 5 were 75 ms and 173 ms, respectively, whereas the peaks are found at 75.01 ms and 173.08 ms, respectively.

The FTP flow from $0 \rightarrow 3$ can be identified by the peaks spread around 328.85 ms, which correspond to the round-trip time for this TCP flow. This value is well within the standard deviation of the measured round-trip time (the deviation and RTT were reported to be 92.5 ms and 371 ms by *ns-2*).

Observe that the plot is able to show the effects of both CBR flows, even though it does not receive any signal from node 1, an end-point for both these flows. The fact that we can see CBR from $1 \rightarrow 3$ is even more interesting because not only can the tap not hear the transmissions of node 1 (or node 2), but there is no way for the tap to know when node 3 receives a packet either. So effectively, the tap never hears any transmission *directly* related to this CBR flow, yet its peak is one of the most prominent peaks.[5]

This example is a good illustration of the Lomb periodogram's utility in extracting useful information for detection of conversations even in complex wireless networks where the trace may be quite noisy (due to the routing traffic, for example), incomplete (due to the limited range of taps), and complex (due to an inherently complex MAC layer transmissions). In this example, the Lomb method is able to identify the key timing parameters of the flows, and thus reveal all three IP flows.

7.1.2 Discussion

This example shows the promise of Lomb's technique for revealing key flow information, even when the signal did not explicitly contain data from transmissions related to some of those flows. Work with other traces, some simulated, some real, have confirmed this promise.

At the same time, there are challenges in using Lomb.

The first major challenge is finding ways to explain each peak in a graph. Even with simulated traffic (where presumably we know or can find all the time constants), there are peaks that sometimes elude understanding (such as the small peaks at 100 and 66 ms in Figure 6). Also, we have found that the Lomb periodogram technique identifies different network characteristics for different networks. It is able to identify the round-trip times of the FTP flow in Figure 6, but in a similar experiment using a wired network highlighted the transmission intervals rather than the round-trip time. For our purposes, the Lomb periodogram is not yet a refined tool.

Finally, the biggest challenge is to scale the Lomb periodogram method to larger networks. We have applied this technique to some large publicly available *tcpdump* traces, and found that even though there are some prominent peaks, it is difficult to identify the key timings that they represent. Moreover, despite the fact that Lomb periodogram works well in the presence of noise, we have found that the noise in large network traces can overwhelm this method by reducing the confidence in prominent peaks. Developing techniques to further reduce the effects of noise in large networks is an important challenge for reducing this approach to practice.

7.2 Tracking Network Dynamics using Time Varying Spectra

Until now, we have limited ourselves to collecting the entire trace for the full duration of a flow, and analyzing the aggregate signal using a one-dimensional (description of the signal only as a function of the frequency) representation of its spectra. However, these spectral techniques (e.g., Lomb Periodogram), are only valid when the underlying process that generated the signal is wide sense stationary,[6] i.e., its frequency content does not change with time. These techniques are still valuable when the signal statistics vary slowly enough such that they are nominally constant over an observation period which is long enough to generate good estimates. That is why it was appropriate to use Lomb periodograms for the analysis of round-trip times or the send rates of flows on networks whose nodes are static. On these networks (which includes most of the Internet), the RTT and mean send rates remain rates remain stable and relatively constant over the duration of individual flows.

However, in many scenarios, the network and flows are more dynamic in nature. For example, in mobile ad hoc networks, the nodes are mobile and the topology changes with time. Or, even in a static network, the objective may be to analyze the evolution of flows over time (to detect TCP stabilization times etc.). Such scenarios where the network or the flow characteristics dynamically change require techniques that can track changes in the spectra with time – or can develop a *time-varying spectral representation* of the signal. Such two-dimensional representations permit a description of the signal characteristics that involves *both* time and frequency, and provide an indication of the specific times

[5]We speculate that this relationship is caused by a form of imprinting. The CBR flow from $1 \rightarrow 3$ shares part of its path with the FTP and the interactions between the FTP data and the CBR flow causes the timing of the CBR flow to be reflected in the FTP acknowledgements.

[6]Wide sense stationary (WSS) usually requires that the mean and autocorrelation (and in the case of multiple streams, cross correlation) functions of the process are constant with respect to the the time and duration of observation.

at which certain spectral components of the signal are observed.

Processes whose spectra changes with time, are known as *nonstationary* processes [10]. Many (linear and quadratic) techniques have been developed for nonstationary signal processing, but of special importance for us are two linear techniques: (1) the *Short Term Fourier Transform*, or STFT [11], which is a natural extension of the Fourier transform that employs shifting temporal windows to divide a nonstationary signal into components over which stationarity can be assumed, and (2) the *Wavelet Transform* [21], which is more complex than the STFT, but offers better time-frequency resolution by trading off time resolution for frequency resolution and vice versa.

In this paper, we use temporal windows, similar to those in the STFT. In Section 7.3, we will use the windowing technique to track topology changes in a network with mobile nodes. Our general approach for analyzing dynamic networks using windowing is as follows.

The tap trace is divided up into temporal windows of a constant duration and spectral estimates are computed for each window. Often the windows are overlapped by a fixed percentage to ensure smooth boundary transitions from one window to the next. The output vector from spectral analysis (which can be cepstrum, coherences, cross-spectral-densities, or indeed power spectral densities computed using Lomb Periodograms) of each window is stacked together as columns of a two dimensional matrix, forming an image with time along the horizontal axis and the estimated parameter (such as amplitude or spectral density) along the other. This kind of representation is often known as a *spectrogram*. In the simplest form, a spectrogram is simply the squared modulus of the Short Term Fourier Transform of a nonstationary signal. Since spectrogram effectively plot the spectra, as it varies in time, it is useful for discovering variations in flow and network characteristics in a dynamically evolving traffic scenario.

Recall that the Lomb method, which is relatively new, permits the analysis of non-uniformly sampled data, at the cost of increased computational complexity. However, there are a multitude of classical signal processing techniques that are applicable to uniformly sampled data only. In order to exploit these techniques we will use uniformly sampled signals to analyze the time-varying spectra.[7]

7.3 Link and Path Discovery using Coherence

The previous sections focused upon the analysis of one signal stream. We now move to the analysis of signals from multiple trace files in order to relate transmissions in one location with those at another. We will use the windowing technique to capture variations in these signal relationships.

The idea is to look for relationships between time windows at different locations or between time windows for traffic from different sources. For instance, if we find a strong relationship between a time window for source 1 and a slightly later time window from source 2, we can infer that some of the traffic from source 1 is being forwarded through or acknowledged by source 2. Expressed in signal processing terms, if there is enough periodicity in a trace file to show spectral peaks, and if the transmissions of one source are forwarded or answered by another source at some layer of the network (such as with ACKs in TCP or via the MAC protocols in a wireless network), then we can compute (using a classical signal processing technique called *coherence*) the degree that the two different signals are related.

For the rest of this section, we use time-varying windows and coherence to identify all active (one-hop, or MAC layer) links between the various nodes in a network. Moreover, we will now work in a *mobile ad hoc wireless network*. Such ad hoc networks require our technique to recognize that links are transient because the nodes are mobile.

The multiple input extension of the periodogram in Equation 2 is *Cross Spectral Density* (CSD) which is essentially the cross spectrum (the spectrum of the cross correlation) $P_{xy}(k)$ of two random sequences. The formula is

$$P_{xy}(k) = \frac{1}{KU} \sum_{r=0}^{K-1} \left[X_N^{(r)}(k)\right] \left[Y_N^{(r)}(k)\right]^* \qquad (4)$$

where $[\,]^*$ denotes the complex conjugate. The resulting CSD shows how much the two spectra $X(k)$ and $Y(k)$ have in common. If two signals are randomly varying together with components at similar frequencies, and stay in phase for a statistically significant amount of time, then their CSD will show peak at the appropriate frequencies. Two independent signals do not give peaks. CSD may be complex valued, so the magnitude of the CSD is generally used in the same way the magnitude of the PSD is.

One can compute a version of the CSD known as *coherence*, whose value is mapped between 0 and 1. The formula is

$$C_{xy}(k) = \frac{|P_{xy}(k)|^2}{P_{xx}(k)P_{yy}(k)} \qquad (5)$$

This formulation is useful in situations where the typical dynamic range of spectra would cause scaling problems, such as in automated detection processing. Since the coherence is nicely bounded, it allows easier automation. However, as we lose the absolute levels of $P_{xy}(k)$, $P_{xx}(k)$, and $P_{yy}(k)$, it should still be used in conjunction with the CSD rather than as a replacement. CSD and coherence may also be presented in gram form in a manner identical to that discussed above.

CSD and coherence answer the question: what was the power of the conversation between any two sources in the network during a certain time-slice? Furthermore, if we encode transmission durations to amplitude, then the power of the peaks would give a sense of the bandwidth of the communications between the nodes. We have found this technique quite useful for discovering routing topology in wireless networks.

First we demonstrate the Coherence technique without the added complication of mobility. Figure 7 shows the results of analyzing 30 seconds of trace data for coherence. The data is taken from a simulated wireless network with a topology similar to Figure 5. Two simple flows are present. An FTP from $0 \rightarrow 3$ by way of node 2, and a CBR from $1 \rightarrow 3$, also by way of node 2. The figure shows one coherence plot for each pair of nodes in the lower diagonal of the matrix of nodes. Each coherence plot is labeled Coherence_{xy} and shows the coherence between nodes x and y. Plots with visible peaks indicate stronger coherence, which suggests two-way transactions (hence a conversation). Furthermore, the shapes of the peaks also provides information which may allow us to differentiate the types of data transfers (FTP vs.

[7]We are currently exploring ways to extend Lomb's method to analyze time-varying spectra using windows.

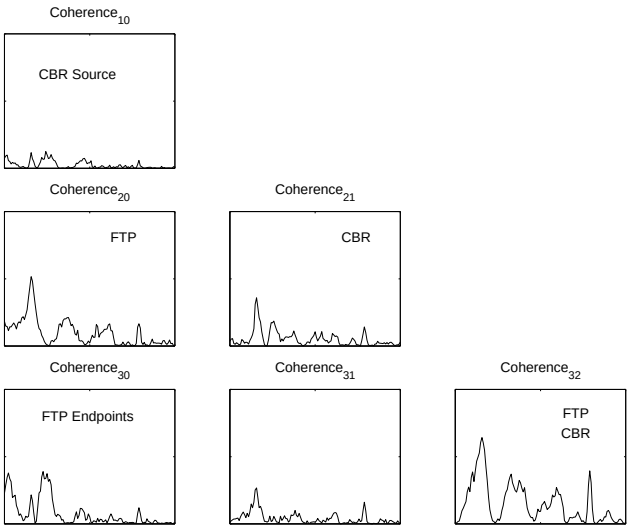

Figure 7: Coherence Between Nodes in the Wireless Network from Figure 5.

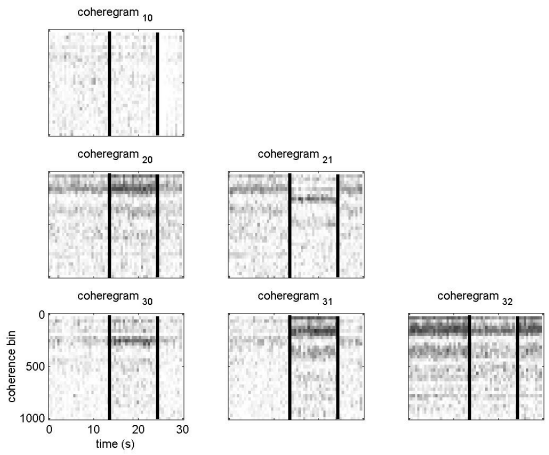

Figure 8: Coheregrams Showing Time Varying Coherence Between Nodes in the Wireless Network from Figure 5, due to a mobile node 1. Link/Routing changes are observed at 14 seconds and 25.5 seconds.

CBR, etc.).

One can see that strong peaks occur between node pairs 2 and 0, 2 and 1, 3 and 0, and 3 and 2. The links $2 \leftrightarrow 0$ and $3 \leftrightarrow 2$ are carrying the FTP, and links $2 \leftrightarrow 1$ and $3 \leftrightarrow 2$ are carrying the CBR. The peaks in Coheregram_{30} do not correspond to a link, but instead are due to the fact that the FTP transfer between nodes 0 and 3 cause those nodes to interact in a strongly periodic pattern due to the ACK feedback of TCP. There is a lack of coherence between nodes 0 and 1 because they do not share any information. We speculate that the coherence between nodes 3 and 1 is due to the traffic periodicity pattern of the FTP being affected by the UDP transmission, but we have not confirmed this.

Next, we demonstrate our solution to the problem of not only discovering the topology, but tracking topology and routing changes, in mobile networks.

Figure 8 shows a *coheregram* generated by analyzing another 30 seconds of trace data taken from the same wireless network in Figure 5, except that now node 1 moves around node 2 at a constant speed (while it moves), stopping for a short duration first between nodes 0 and 2, and then between nodes 2 and 3. This motion causes rerouting to occur twice, first at 14 seconds into the run, and again at 25.5 seconds. Initially, traffic from $1 \rightarrow 3$ is routed through node 2, until at time 14 seconds, node 1 gets close enough to node 3 to route directly. This continues until 25.5 seconds, when node 1 has circled far enough away from node 3 to resume routing through node 2.

Coherence spectra were computed for each 512 ms interval and displayed as a two-dimensional time-frequency gram where intensity is proportional to power at that time and frequency (white = low level to black = high level). The result is a gram plot for each pair of nodes (laid out exactly as in Figure 7). When the coherence remains similar from one interval to the next, peaks resolve as horizontal lines in the plot. However, when the network reroutes at 14 seconds and node 1 begins to communicate directly with node 3, the coherence peaks change visibly in Coheregram_{21} and Coheregram_{31}. At 25.5 seconds, they coherence peaks change visibly, and remain such until the network resume their old form. Such a change could be detected by automated means.

8. CONCLUSIONS

There's something very tantalizing about finding a new way to look at data traffic. For instance, the experience of seeing coherence techniques map the path a flow's traffic took through the network, and to recognize changing communication patterns in a mobile ad-hoc network was extremely exciting.

We started this paper with four questions we hoped signal processing techniques might address.

Clearly the coherence techniques give us insights into who is talking to whom, and the paths traffic take. We are currently working on refining these techniques to larger and more complex networks.

The Lomb periodogram gives us some insight into determining how many flows are traveling over a particular path: the peaks in the periodogram can be used to reveal features of individual flows. But we are a long way from using that data to determine which particular applications are in use or which individual events correspond to a particular flow.

At the same time the results reported in this paper obviously raise more questions than they answer. There are a number of opportunities to substantially refine algorithms, including:

- How best to encode a trace as a signal? Encoding is a key part of the analysis process and yet we've only just begun to explore the issues. It seems likely that different encodings will give different results, and perhaps highlight different aspects of a trace.
- How to separate wheat from chaff in the results? The Lomb periodogram is a good example. Even for modest amounts of traffic, it reveals a number of heavily used frequencies. How do we identify the frequencies we most care about?

- As mentioned in Section 7.2, often network traffic produces nonstationary processes, which require specialized techniques such as windowing and the Welch Average Periodogram described in Section 7. However, even these techniques also work well only if the signal statistics vary slowly enough, at least within the observation time covered by the window. Another alternative (which we are exploring) is to develop techniques which do not require the signal to be wide sense stationary at any time scale. Wavelets analysis is a relatively new tool in signal processing, developed only in 1980s [21], and they are applicable to completely nonstationary signals. We are exploring the use of such techniques for discovering time varying network properties.

- Finally, given that these techniques are beginning to work, what can we do to hide traffic patterns from them? What (possibly new) techniques should we use to make traffic less vulnerable to this sort of traffic analysis?

ACKNOWLEDGMENTS

We are indebted to Steve Kent, Greg Troxel, Chip Elliott, Alex Snoeren, and Paul Kolodzy for their suggestions for directions and reviews of early drafts.

9. REFERENCES

[1] *IEEE Std 802.11b — Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification: Higher-Speed Physical Layer Extension in the 2.4 GHz Band*. IEEE, 1999.

[2] Abry, P., Baraniuk, R., Flandrin, P., Riedi, R., and Veitch, D. Multiscale nature of network traffic. *IEEE Signal Processing Magazine 19*, 3 (2002), 28–46.

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

[4] Cappe, O., Moulines, E., Pesquet, J.-C., Petropulu, A., and Yang, X. Long-range dependence and heavy-tail modeling for teletraffic data. *IEEE Signal Processing Magazine 19*, 3 (2002), 14–27.

[5] Coates, M., Castro, R., Nowak, R., Gadhiok, M., King, R., and Tsang, Y. Maximum likelihood network topology identification from edge-based unicast measurements. *Proc. ACM SIGMETRICS* (2002), 11–20.

[6] Coates, M., III, A. H., Nowak, R., and Yu, B. Internet tomography. *IEEE Signal Processing Magazine 19*, 3 (2002), 47–65.

[7] Duffield, N., and Grossglauer, M. Trajectory sampling for direct traffic observation. *Proc. ACM SIGCOMM* (2000), 271–282.

[8] Duffield, N. G., and Presti, F. L. Multicast inference of packet delay variance at interior network links. *Proc. INFOCOM* (2000), 1351–1360.

[9] Feldmann, A., Gilbert, A., Willinger, W., and Kurtz, T. The changing nature of network traffic: Scaling phenomena. *ACM Computer Communication Review 28*, 2 (1998), 5–29.

[10] Haykins, S. *Communication Systems*, 3rd ed. John Wiley and Sons, Inc., 1994.

[11] Hlawatsch, F., and Boudreaux-Bartels, G. F. Linear and quadratic time-frequency signal representations. *IEEE Signal Processing Magazine 9*, 4 (1992), 21–67.

[12] Huang, P., Feldmann, A., and Willinger, W. A non-intrusive, wavelet-based approach to detecting network performance problems. *Proc. ACM SIGCOMM Internet Measurement Workshop* (2001), 213–227.

[13] Johnson, D. B., and Maltz, D. A. Dynamic source routing in ad hoc wireless networks. vol. 353. Kluwer Academic Publishers, 1996.

[14] Kay, S. M. *Modern Spectral Estimation: Theory and Application*. Prentice-Hall, Englewood Cliffs, NJ, 1988.

[15] Kent, S. Encryption-based protection for interactive user/computer communication. *Proc. Fifth Data Communications Symposium* (1977), 5–7 – 5–13.

[16] Leland, W. E., Taqq, M. S., Willinger, W., and Wilson, D. V. On the self-similar nature of Ethernet traffic. *Proc. ACM SIGCOMM* (1993), 183–193.

[17] Lomb, N. R. Least-squares frequency analysis of unequally spaced data. *Astrophysics and Space Science 39* (1976), 447–462.

[18] Low, S. H. A duality model of tcp flow controls. In *ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management* (September 2000). http://netlab.caltech.edu.

[19] Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. *Numerical Recipes in C*, 2nd ed. Cambridge Univ. Press, 1995.

[20] Reed, M. G., Syverson, P. F., and Goldschlag, D. M. Anonymous connections and onion routing. *IEEE Jour. Selected Areas in Communication 16*, 4 (May 1998), 482–494.

[21] Rioul, O., and Vetterli, M. Wavelets and signal processing. *IEEE Signal Processing Magazine 8*, 10 (1991), 14–38.

[22] Song, D. X., Wagner, D., and Tian, X. Timing analysis of keystrokes and timing attacks on ssh. *Proc. 10th USENIX Security Symposium* (2001), 337–352.

[23] Tsang, Y., Coates, M., and Nowak, R. Passive unicast network tomography based on tcp monitoring. In *Rice University, ECE Department Technical Report TR-0005*.

[24] Vardi, Y. Network tomography: estimating source-destination traffic intensities from link data. *Jour. American Statistical Assoc. 91* (1996), 365–377.

[25] Welch, P. D. The use of fast fourier transform for estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Trans. on Audio Electroacoustics AU-15* (1967), 70–73.