

Wireless Security

Determining applications and characteristics of encrypted wireless traffic.

Chris
CMPE 257
3/9/2011

Papers

- On Inferring Application Protocol Behaviors in Encrypted Network Traffic
- Inferring Speech Activity from Encrypted Skype Traffic

Introduction

- Several fundamental security mechanisms for restricting access to network resources rely on the ability of a reference monitor to inspect the contents of traffic as it traverses the network.
 - As an administrator you want to know the type of traffic to determine if it is acceptable or not.
 - As a user you may need to know that someone can figure out what you are doing even if encrypted.

Introduction

- Traditional packet inspection does not work if the contents are encrypted.
 - No port numbers or TCP flags to check.
 - Also valuable if not encrypted but is being disguised as another type of traffic.
- You can still view packet size, timing, and direction.

Experiment

- Gathered traffic from first 10 minutes of every quarter hour over a two-month period on George Mason Universities OC-3 link.
 - While not wireless this gave them more data to experiment with.
- Extracted packets with ports SMTP (25), HTTP (80), HTTP over SSL (443), FTP (20), SSH (22), and Telnet (23) inbound and SMTP and AIM outbound.
 - These ports were then marked with these labels without further inspecting the packets. Results compare back to this labeling.

Experiment

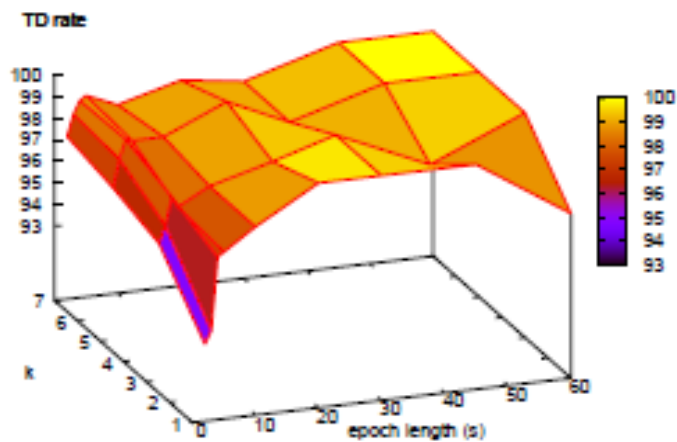
- To generate the data for the encrypted test, the packets were then encrypted with AES at 512 bits.
- The only thing left was the timing, size, and direction of the packets.

Traffic Classification

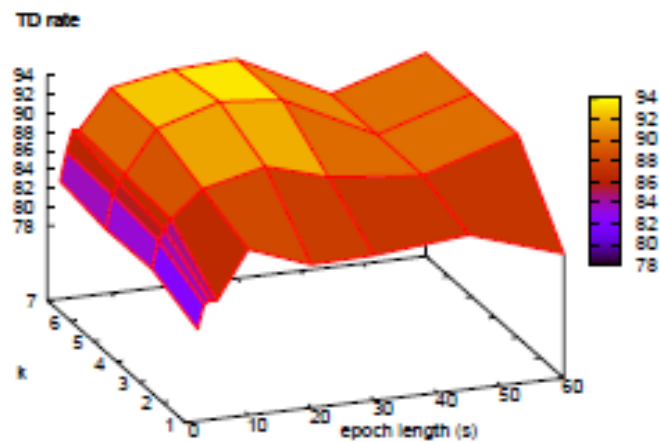
- For each trace, all connections for the given protocol sorted in order of arrival, then split into several smaller epochs of constant length s
- Count the number of packets of each type during each epoch
 - 4 types: small or not; inbound or outbound
- s on the order of several seconds.

Identifying Application Protocols in Aggregate Traffic

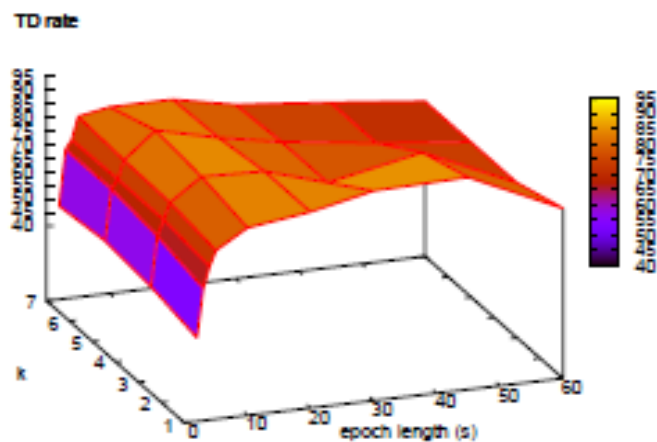
- Construct a k-Nearest Neighbor (k-NN) classifier to assign a label to each epoch based on number of packets of each type.
- To build the k-NN classifier a random day from the data set was used as a training set based on the earlier port classification.
- To classify each new epoch they use Kullback-Leibler distance to determine which vectors in the training set are “nearest” to the vector of counts for the given epoch.



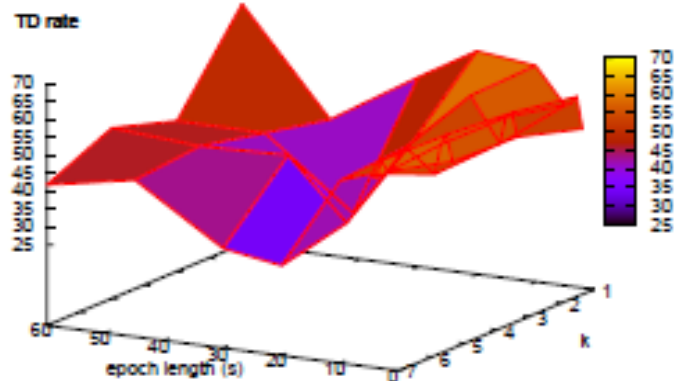
(a) HTTP



(b) HTTPS



(c) SMTP(out)



(d) SSH

Figure 1: Per-epoch recognition rates for HTTP, HTTPS, SMTP-out and SSH with varying values of s and k

Identifying Application Protocols in Aggregate Traffic

- s – epoch length
- k – how many nearest neighbors
- Recognition rates tend to increase with increase with both s and k .
- Increasing s gives a bigger sample, but do not allow for analyzing shorter traces where an adversary could hid in quickly.
 - Paper chose $s=10$ seconds as an acceptable tradeoff.
- Given the list of labels, the mode of traffic is the one with the most frequently occurring label.

Identifying Application Protocols in Aggregate Traffic

- Evaluate this classifier by applying it against another days traffic.

| protocol | 1-NN | | 3-NN | | 5-NN | | 7-NN | |
|----------|-------|------|-------|------|-------|------|-------|------|
| | TD | FD | TD | FD | TD | FD | TD | FD |
| HTTP | 100.0 | 00.0 | 100.0 | 00.0 | 100.0 | 00.0 | 100.0 | 00.0 |
| HTTPS | 100.0 | 00.0 | 100.0 | 01.2 | 100.0 | 01.2 | 100.0 | 03.6 |
| AIM | 91.7 | 00.0 | 91.7 | 00.0 | 91.7 | 00.0 | 83.3 | 00.0 |
| SMTP-in | 100.0 | 00.0 | 100.0 | 00.0 | 100.0 | 00.0 | 100.0 | 00.0 |
| SMTP-out | 100.0 | 03.6 | 91.7 | 03.6 | 91.7 | 03.6 | 75.0 | 03.6 |
| FTP | 100.0 | 03.6 | 100.0 | 01.2 | 100.0 | 01.2 | 100.0 | 02.4 |
| SSH | 75.0 | 00.0 | 75.0 | 00.0 | 75.0 | 00.0 | 75.0 | 00.0 |
| Telnet | 83.3 | 00.0 | 100.0 | 00.0 | 100.0 | 00.0 | 100.0 | 00.0 |

Table 1: Protocol detection rates for the k -NN classifier ($s = 10$ sec)

Identifying Application Protocols in Aggregate Traffic

- By using Kullback-Leibler distance to construct classifiers for short slices of time, they can be combined to build a classifier for longer traces which performs well on aggregate traffic where only a single application protocol is involved.

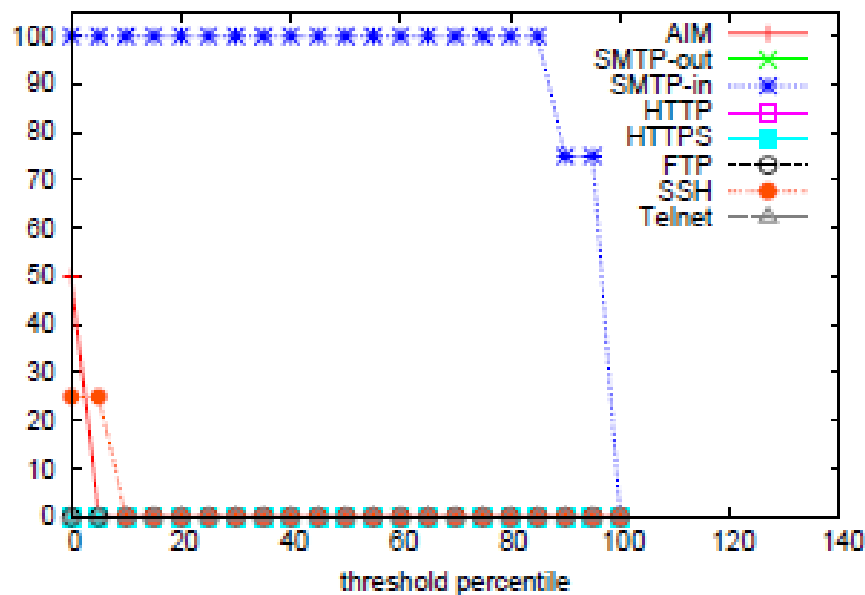
Multi-flow protocol detector

- Since we cannot assume all flows in the aggregate carry the same application, need to expand to a multi-flow protocol detector.
- In general a network administrator (or a hacker) is concerned with detecting the presence of a few specific protocols within the aggregate.

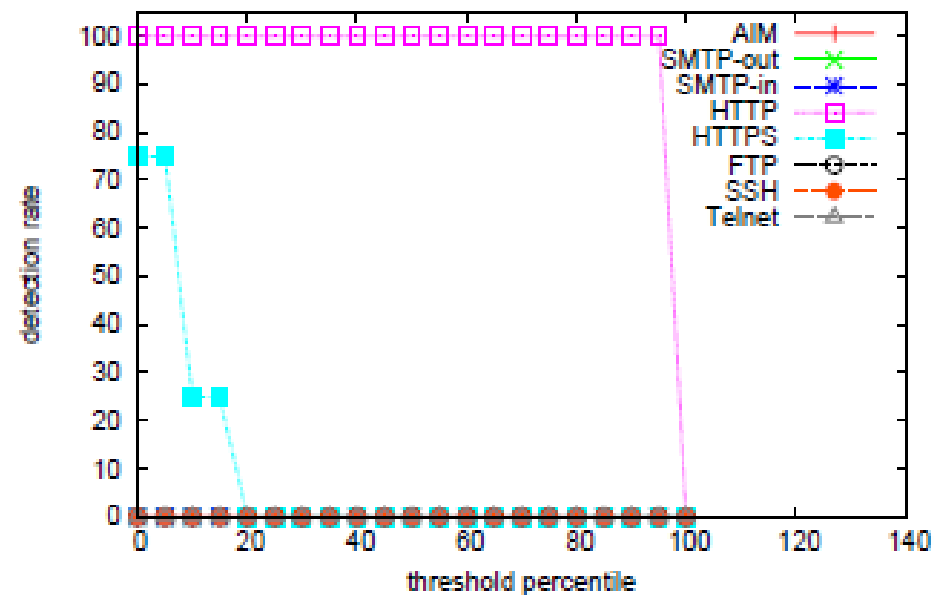
Multi-flow protocol detector

- Modify the k-NN classifier.
 - Label the vectors in the training set based on whether they contain an instance of the target protocol(s).
- Run experiment similar to before, but with aggregate traffic.
- Flag the aggregate as containing the target protocol if some* percentage of the epochs return True.
 - Tune by adjusting the percentage.

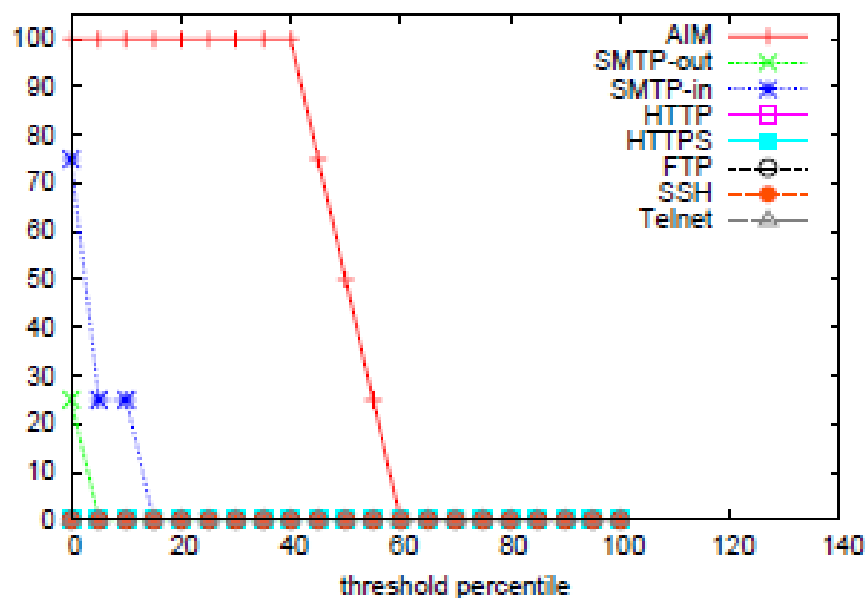
(a) SMTP(in) Detector - detection rates



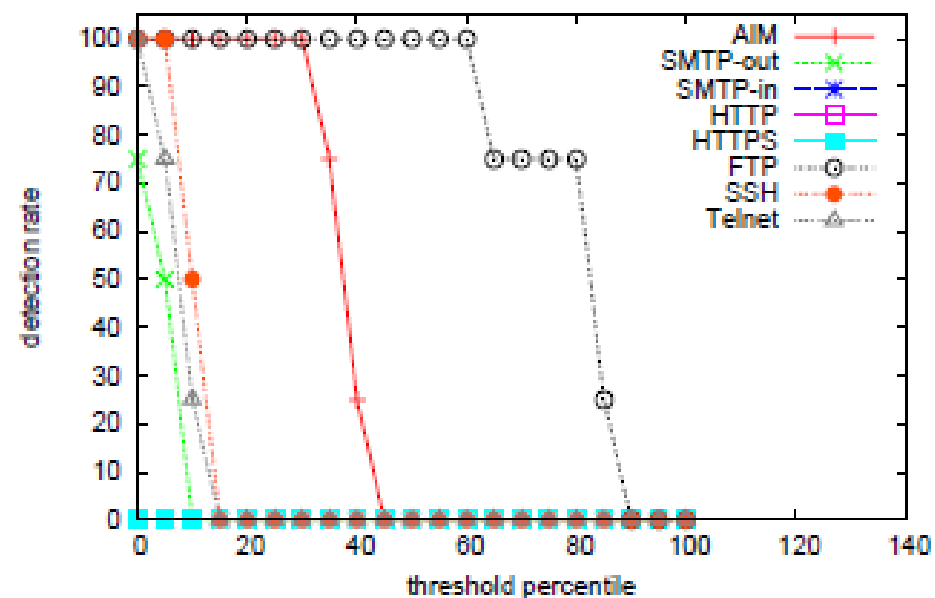
(b) HTTP Detector - detection rates



(c) AIM Detector - detection rates



(d) FTP Detector - detection rates

Figure 2: Detection rates for multi-flow protocol detectors ($k = 7, s = 10\text{sec}$)

Machine Learning Techniques

- Build statistical models for the sequence of packets produced by each protocol of interest, and then use these models to identify the protocol in use in new connections.
- Use techniques based on profile hidden Markov models.
 - Difficulty with protocols that have more than one typical behavior like SSH (SCP for bulk transfer and interactive) versus FTP (always bulk transfer).

Machine Learning Techniques

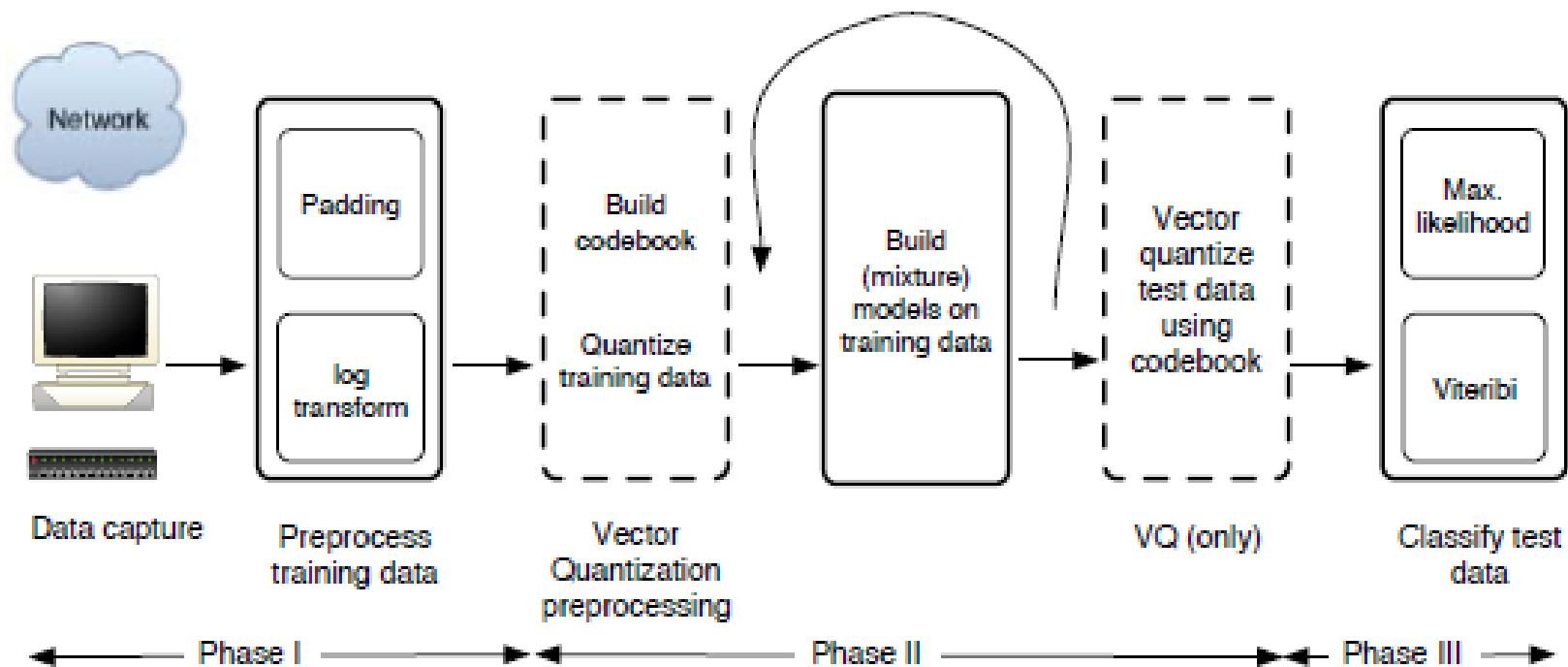


Figure 3: Process overview for construction of our Hidden Markov Model-based classifiers.

Profile Hidden Markov Models

- The profile HMM is best described as a left-right model built around two long parallel chains of hidden states. Each chain has one state per packet in the TCP connection, and each state emits symbols with a probability distribution specific to its position in the chain.

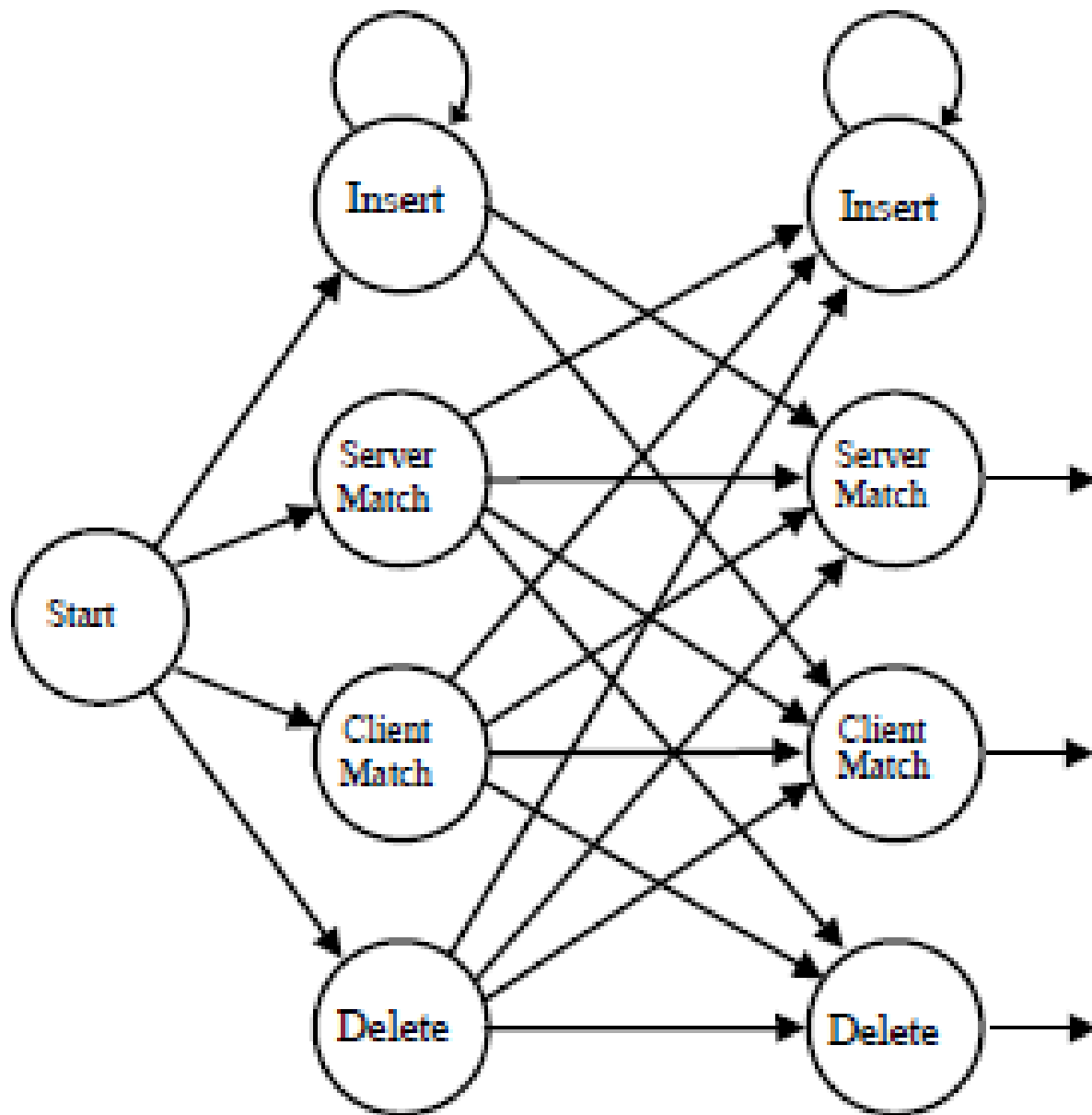


Figure 4: Profile HMM for TCP sequences

Profile Hidden Markov Models

- The addition of a second match state per position was intended to allow the model to better represent the correlation between successive packets.
- Server Match state matches only packets observed traveling from the server to the client.
- Client Match state matches packets traveling in the opposite direction.
- A transition from a Client Match state to a Server Match state indicates that a typical packet (for the given protocol) was observed traveling from the client to the server, followed by a similarly typical packet on its way from the server to the client.

Profile Hidden Markov Models

- Insert - allows for one or more extra packets “inserted” in an otherwise conforming sequence, between two normal parts of the session.
- Delete - allows for the usual packet at a given position to be omitted from the sequence.
- Transitions from the Delete state in each column to Insert state in the next column allow for a normal packet at the given position to be removed and replaced with a packet which does not fit the profile.
- In practice, the Insert states represent duplicate packets and retransmissions, while the Delete states account for packets lost in the network or dropped by the detector.

HMM-based Classifiers

- The Viterbi classifier finds each model's best explanation for how the packets in the sequence were generated (whether by normal application behavior, TCP retransmissions, etc.), represented by the Viterbi path, and the likelihood of each model's explanation (i.e., the Viterbi path probability). It then picks the model that provides the best explanation for the observed packets.

Single vs Multiple features

| protocol | micro-level | | equivalence class | |
|----------|-------------|------|-------------------|------|
| | TD | FD | TD | FD |
| AIM | 80.80 | 3.41 | 80.80 | 3.41 |
| SMTP-out | 73.20 | 3.07 | 80.10 | 1.82 |
| SMTP-in | 77.20 | 4.39 | 87.80 | 3.97 |
| HTTP | 90.30 | 2.10 | 96.70 | 1.47 |
| HTTPS | 88.50 | 3.24 | 94.40 | 2.72 |
| FTP | 57.70 | 2.01 | 57.70 | 2.01 |
| SSH | 69.10 | 2.93 | 71.00 | 2.88 |
| Telnet | 82.90 | 3.77 | 86.10 | 4.08 |

Table 2: Protocol detection rates for the Viterbi classifier, using packet sizes only

| protocol | micro-level | | equivalence class | |
|----------|-------------|------|-------------------|------|
| | TD | FD | TD | FD |
| AIM | 83.90 | 2.53 | 83.90 | 2.53 |
| SMTP-out | 74.40 | 2.24 | 79.70 | 1.60 |
| SMTP-in | 79.80 | 3.34 | 85.90 | 3.02 |
| HTTP | 78.00 | 1.09 | 92.90 | 0.62 |
| HTTPS | 87.20 | 3.74 | 91.10 | 1.88 |
| FTP | 58.20 | 1.81 | 58.20 | 1.81 |
| SSH | 76.30 | 8.37 | 77.80 | 7.90 |
| Telnet | 79.50 | 2.44 | 90.70 | 2.60 |

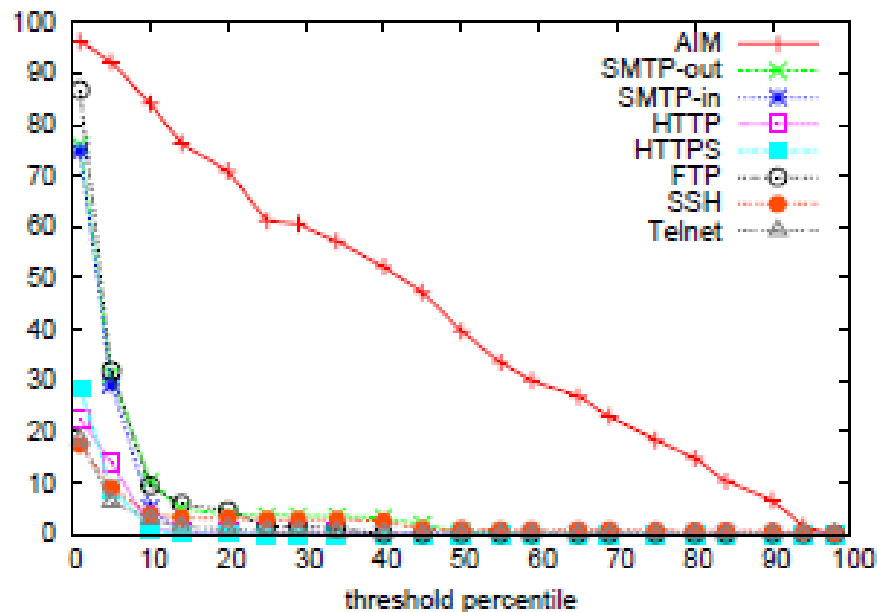
Table 3: Protocol detection rates for the Viterbi classifier with 140-codeword VQ

Protocol Detector for Single Flows

- Using the classifiers from before would be computationally intensive.
- Instead of building models for each protocol, build one for the target protocol and one for the “noise” in the network.
- Able to test 3200 connections in 15 seconds

Protocol Detector for Single Flows

(a) AIM Detector - detection rates



(b) HTTP Detector - detection rates

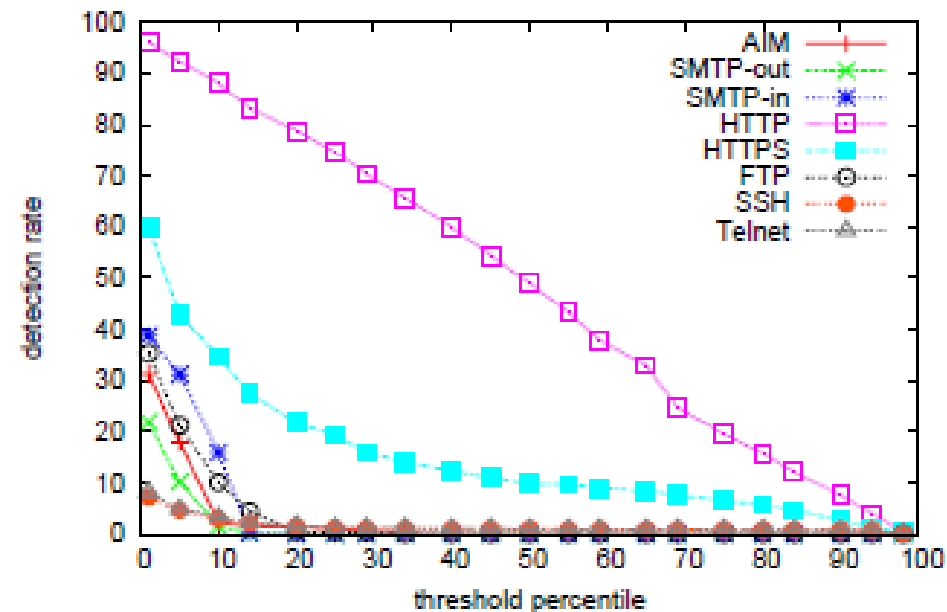


Figure 5: Detection rates for threshold-based protocol detectors for AIM and HTTP

Number of live connections

- What if the connections are aggregated and put in an encrypted tunnel?
 - Wireless users VPN back to work.
 - User trying to further hide traffic by putting it all in an SSH tunnel.
- A model-based technique which enables us to accurately track the number of connections in a network-layer tunnel which carries traffic for only a single application protocol.

Number of live connections

- Approach is founded on a few basic assumptions about the behavior of the tunneled TCP connections and their associated packets. These assumptions, while not entirely correct for real traffic, nevertheless allow us to employ simple and usable models which, as we demonstrate later, produce reasonable results for a variety of protocols.

Number of live connections

- Assumption 1 - The process N_t describing the number of connections in the tunnel is a Martingale, meaning that, on average, it tends to stay about the same over time.
- Assumption 2 - The process N_t describing the number of connections in the tunnel is a Gaussian process. That is, the number of connections N_t in each time slice t follows a Gaussian distribution.
- Assumption 3 - For each packet type m , each connection in the tunnel generates packets of type m according to a homogeneous Poisson process with constant rate γ_m , which is determined by the application protocol in use in the connection.

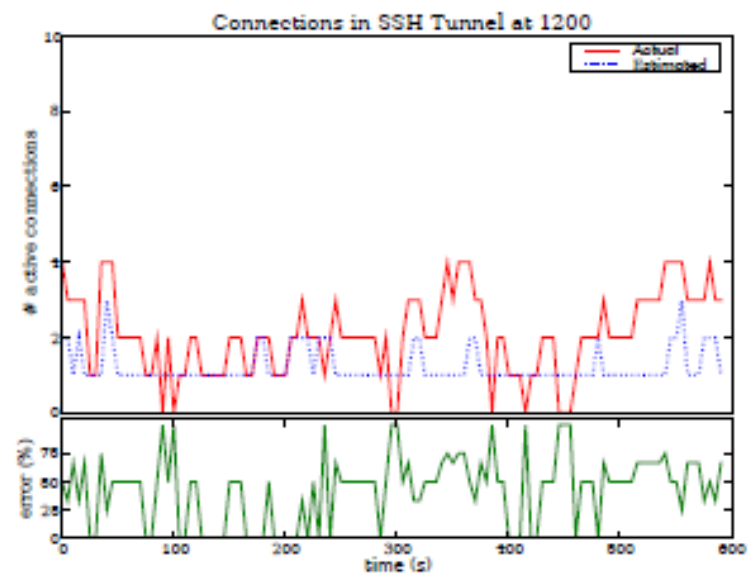
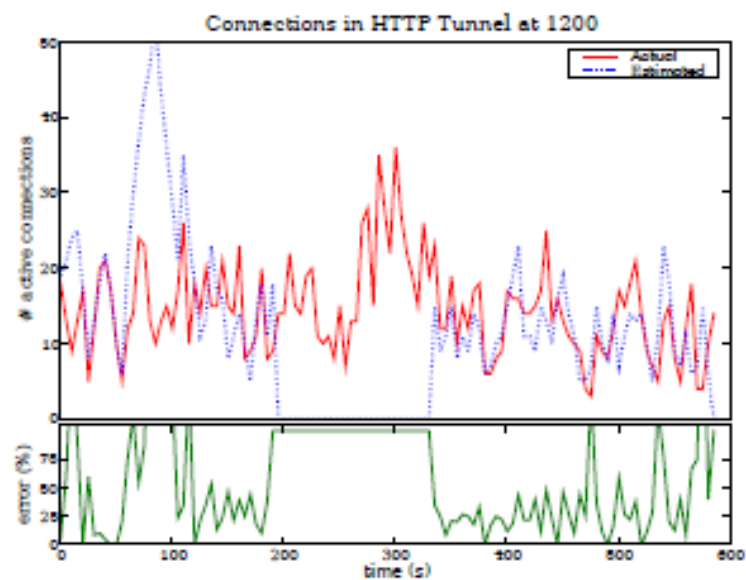
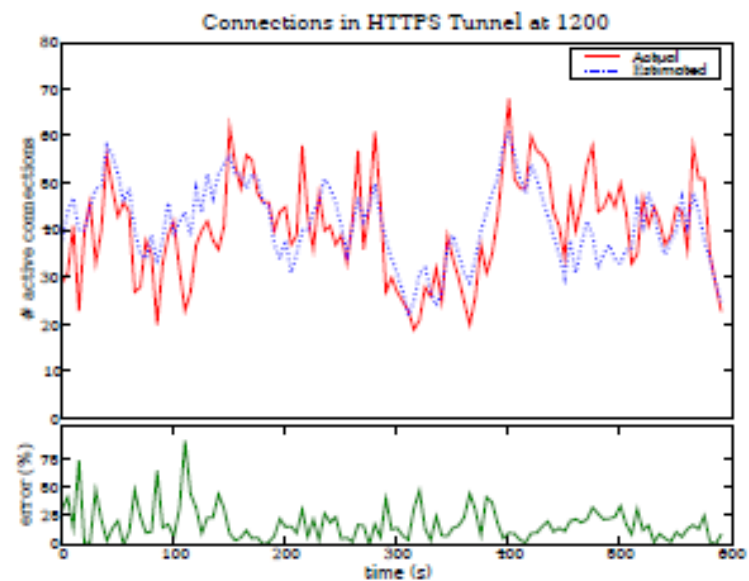
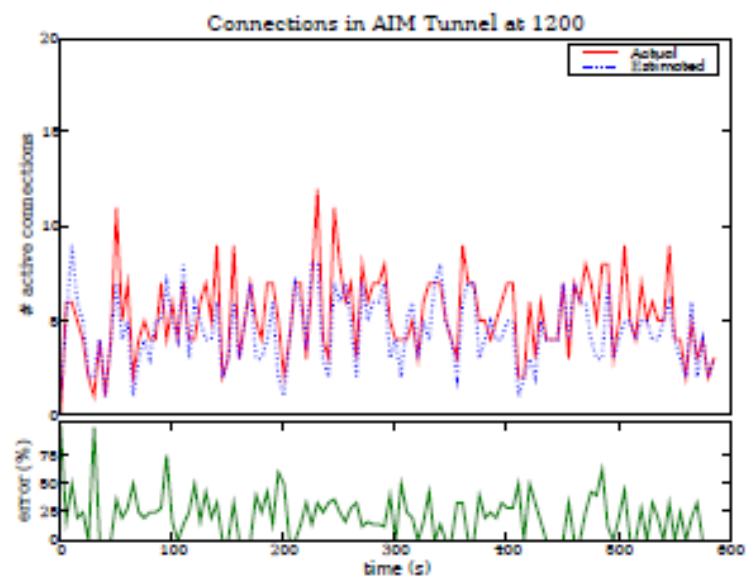


Figure 7: Actual and estimated number of connections in simulated tunnels for AIM, HTTP, HTTPS, and SSH in the 12:00 trace on the testing day

Inferring Speech Activity from Encrypted Skype Traffic

- This paper dealt with Skype traffic, but could be expanded to any VoIP traffic.
- In conjunction with the last paper, an adversary could find VoIP flows, then use this to determine the speech activity.
- In the future as cellular networks move to data only with VoIP this could easily give people a lot of information about what is happening even in encrypted traffic.

Inferring Speech Activity from Encrypted Skype Traffic

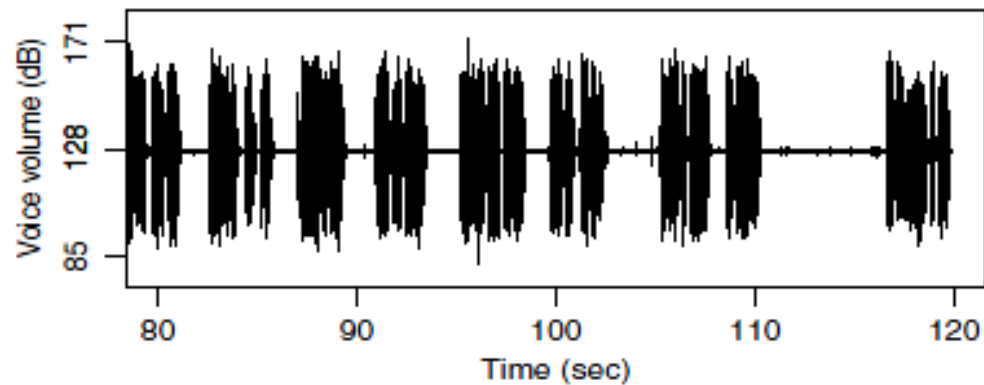
- Historical voice activity detection (VAD) is about detecting the presence or absence of human speech in audio signals.
- This paper focuses on speech detection in VoIP traffic instead of audio signals, called network-level VAD.
- Real-time network-level VAD algorithm to extract voice data from encrypted and non-silence-suppressed Skype traffic.

Inferring Speech Activity from Encrypted Skype Traffic

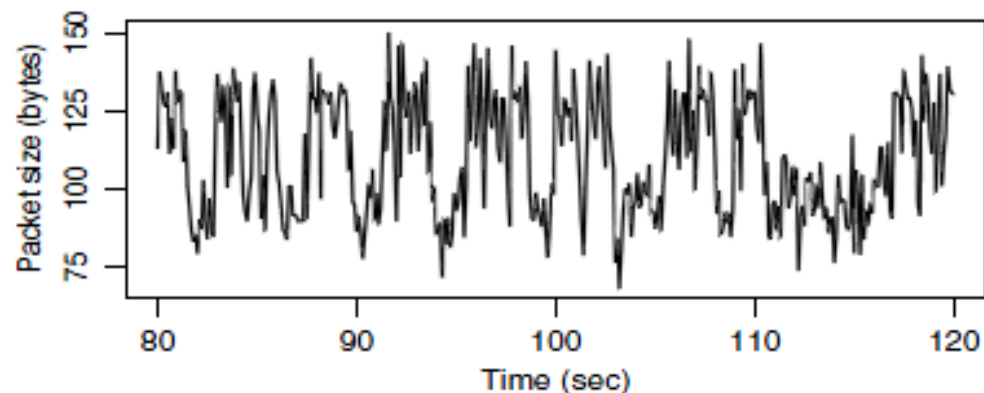
- VoIP flows are difficult to recognize due to proprietary protocols, nonstandard port numbers, and encrypted payloads, the *human conversation pattern embedded in the traffic can be a unique* signature of VoIP flows.
- Conversation activity between two people often occurs on a *multi-second time scale*, while the *traffic patterns of other* applications, e.g., web traffic or online game traffic, are usually on a sub-second time scale.

Inferring Speech Activity from Encrypted Skype Traffic

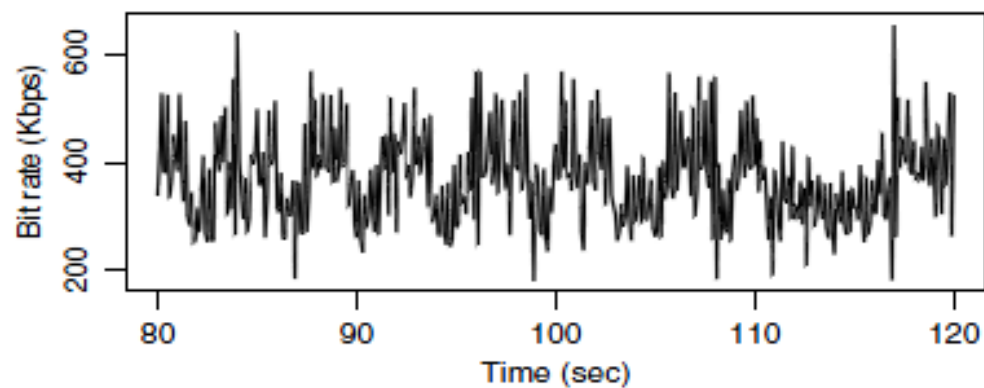
- The proposed scheme is based on our observation that, in Skype traffic, *speech activity is highly correlated to packet size, as more information will be encoded in a voice packet while a user is speaking.*



(a) Human speech recording



(b) Packet size process



(c) Bit rate process

Fig. 2. The volume process of a human speech recording and the packet size and bit rate process of the corresponding VoIP traffic.

Inferring Speech Activity from Encrypted Skype Traffic

- Network-level VAD is more difficult than traditional VAD because network traffic takes the form of audio signals compressed by an encoder, which may adjust the coding level, redundancy level, quantization levels, and packetization period at any time to achieve more efficient voice transmission.
- This produces more randomness in VoIP traffic than source VAD.

Experiment

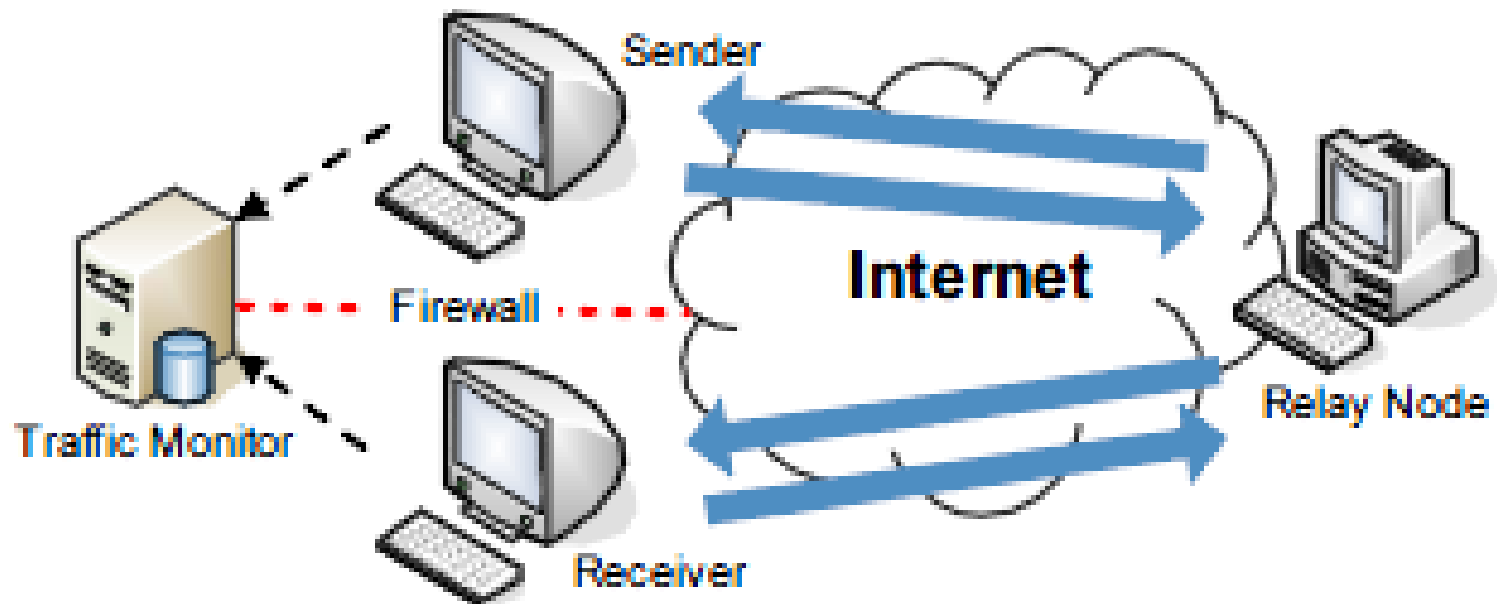


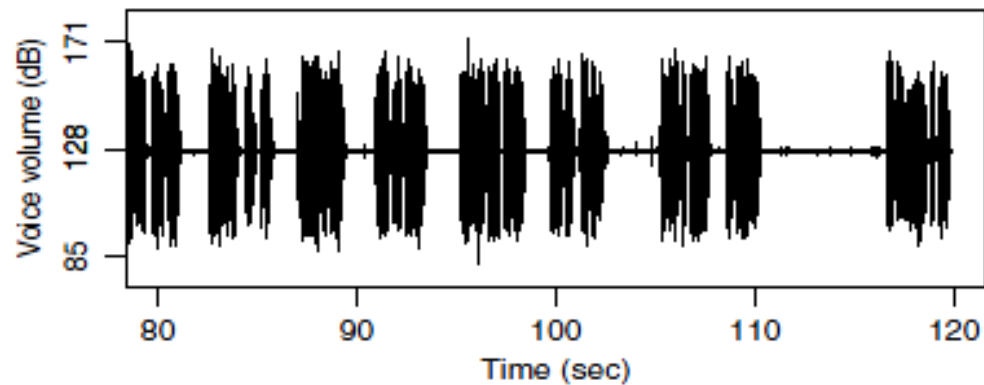
Fig. 1. Network setup for measuring processing delays at any relay node on the Internet.

Experiment

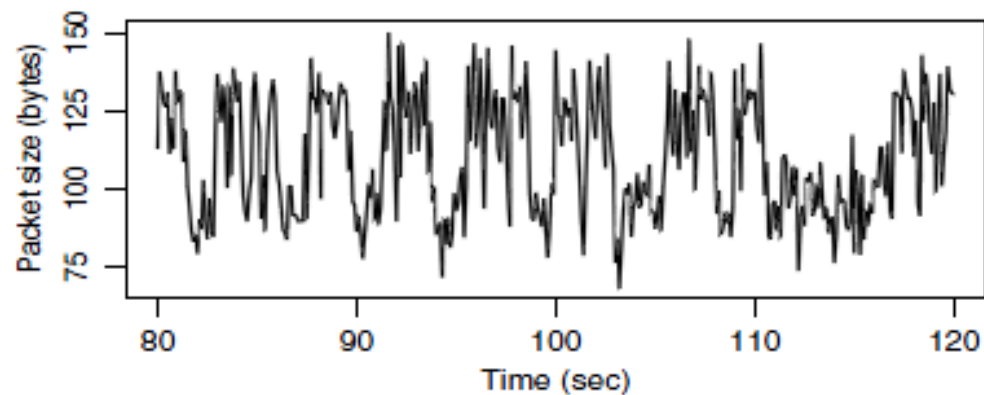
- Route call through internet, while capturing packets and raw speech data to compare results.
- Ran traces for two months generating 1839 calls.
- For comparison, a source level VAD was run on the speech signals captured.
 - Skipping the math, it defined an ON period for speech signals with volume over a threshold and OFF period otherwise.

Proposed Scheme

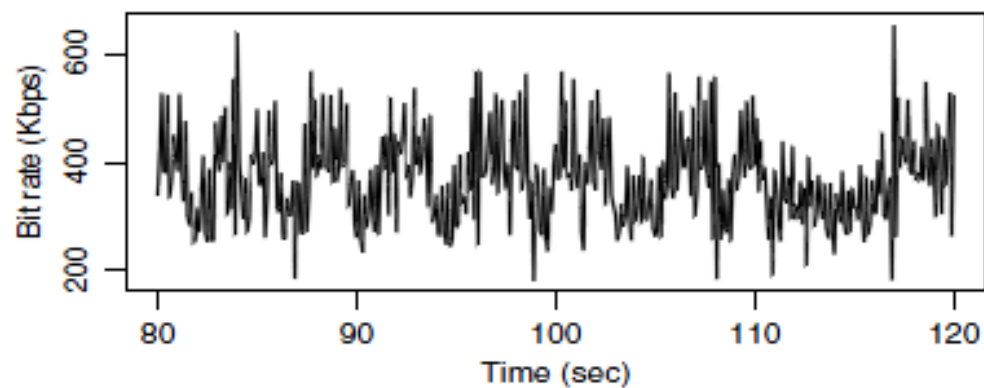
- Preliminary analysis shows that both the *packet size and the bit rate are indicators of speech activity*, even though the packet payload is encrypted.
- The result suggests that the packet rate is nearly independent of user speech. Although the bit rate exhibits a reasonable correlation with the volume process, it may contain randomness due to packet retransmission or congestion control mechanisms. Therefore, we adopted the packet size process as the basis for inferring voice activity.



(a) Human speech recording



(b) Packet size process



(c) Bit rate process

Fig. 2. The volume process of a human speech recording and the packet size and bit rate process of the corresponding VoIP traffic.

Proposed Scheme

- A static threshold would be nice, but the packet size process is not stationary as its mean may change over time.
 - Probably due to Skype adjusting the encoding bit rate, redundancy factor, and packetization delay on the host or congestion level.
- Apply smoothing to remove high-frequency variations, then apply an adaptive threshold.

Smoothing

- Use an exponentially weighted moving average.
- $P_i = \lambda Y_i + (1 - \lambda)P_{i-1}$
- Y_i denotes the observed packet size of the i th time unit (time unit of 0.1 second for experiment)
- P_i denotes the smoothed packet size in the i th time unit.
- Found that a weight λ of .2 achieves the best performance (.1 to .4 had similar performance)

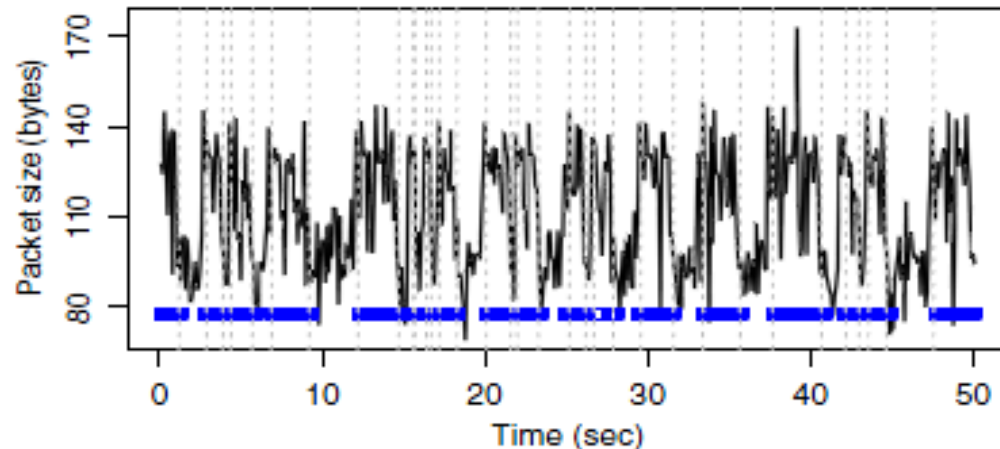
Adaptive Thresholding

- 1) Find all local max and min in window of 5 samples. If difference is greater than 25 bytes, then max is "peak" and min is "trough"
- 2) Denote each peak and trough as (t_i, s_i) where t is time and s is smoothed packet size. For each pair of adjacent troughs (t_a, s_a) and (t_b, s_b) , take the max peak between them and denote the size as s_p . Then draw a line from $(t_a, (s_a + s_p)/2)$ to $(t_b, (s_b + s_p)/2)$
- 3) Voice is ON if smoothed packet is greater than any thresholds defined at the time the sample was obtained.

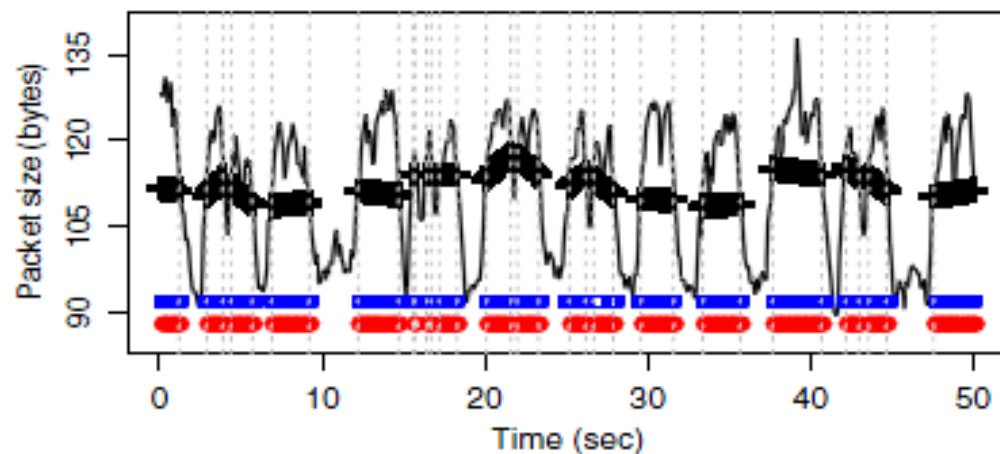
Evaluation

- On the upper graph, the blue squares mark the times a speech burst is present.
- On the lower graph, the lines formed by black crosses are the adaptive thresholds.
- The lower graph shows the level of agreement between the estimated speech activity (red circles) and the ground truth (blue squares).

Evaluation



(a) Observed packet size process



(b) EWMA smoothing process

Fig. 3. The packet size process and its smoothed process, along with the computed adaptive thresholds and the true and estimated speech activity.

Evaluation

- Number of Active Periods
 - Estimated ON periods/True ON periods
- Average Length of ON Periods
 - Mean length of estimated ON periods/Mean length of true ON periods
- State Correctness
 - Estimated state matches true state
 - Ranged 70%-90% with median around 85%.

Evaluation

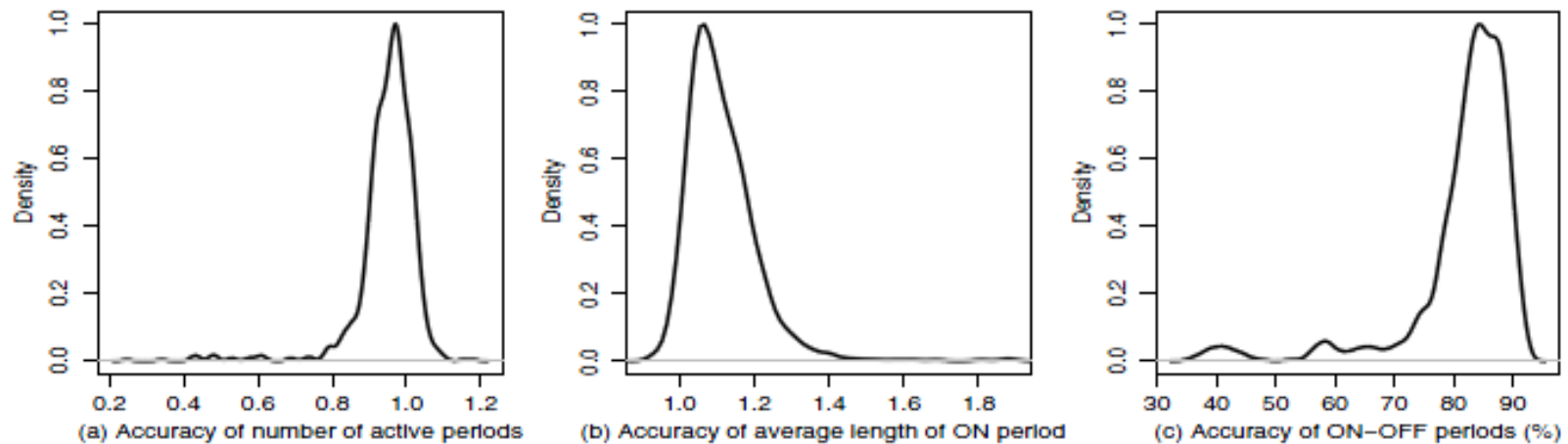


Fig. 4. Three figures show (a) The ratio of the number of active periods between true and estimated speech activity, (b) The ratio of the average length of active periods between true and estimated speech activity, and (c) The correctness of the inferred speech activity.

Conclusion

- Taking Skype as the subject of our study, we show that inferring voice activity from Skype traffic achieves a reasonable level of detection accuracy even there has been high degree of randomness in the network traffic.
- If you care about people not being able to detect what you are doing over a wireless connection, encryption alone is not effective.