Class Discussion of “BLINC: Multilevel Traffic Classification in the Dark” and related works

Feb. 24, 2006

Original paper by:
T. Karagiannis, K. Papagiannaki, and M. Faloutsos
in Proceedings of ACM SIGCOMM, 2005

Discussion Moderator: Charles V Wright
cvwright@jhu.edu
BLINC - Contrast to *Profiling Backbone Traffic*

- **BLINC** --  
  Supervised Learning: Classification  
  - Given *labeled examples* of relevant classes, assign labels to new, unlabeled examples

- *Profiling Backbone Traffic* --  
  Unsupervised Learning: Clustering  
  - Given a bunch of unlabeled data, find the *dominant subgroups* of similar examples
BLINC – Payload Classification: Good or Bad?

Some comments were positive

- I liked ... the clean approach of testing the implementation against a full payload inspection scheme...

Some were more dubious

- ... the validity of their BLINC methodology is completely dependent on their initial payload-based classification... I think a strong look should be taken at this ...
BLINC – Payload Classification: Good or Bad?

- Note that some flows are classified *without* any actual payload analysis (!)
- They're essentially using the *same assumptions* on which the BLINC method is founded to set the baseline for BLINC's evaluation.
BLINC – Privacy?

- Claim: inspecting only headers is good for privacy

- Comment: I'm quite sure that given just packet headers someone could determine the real juicy stuff: what websites you're going to, where you get your streaming video from --- all those things you don't want your wife to know.
BLINC – Privacy?

- Why can't we protect privacy – for real!
  - Can we?
  - Implications for DETER, etc.
  - Anonymization techniques?

R Pang, M Allman, V Paxson and J Lee, *The Devil and Packet Trace Anonymization.*
BLINC -- Extensions: Inspecting Actual Flows

- Take into account the amount of incoming and outgoing traffic.

- I see [BLINC] as being a secondary test for traffic after it has been attempted to be classified using more detailed application layer analysis.

- Why not experiment with adding the recent 'novel statistical approaches' ... to see if completeness and accuracy can be further increased ...
A Different Perspective: Analysis of Individual Flows

Different unit of analysis

- Instead of the whole network, let's look at one flow at a time
- Does this give us a better idea of what's going on?

Complementary to yesterday's techniques
In Broad Daylight: Payload-based Classification

- Use the actual contents of packets to determine what the flow is doing
- This is basically just text classification
- Nevertheless, there are a lot of papers using this kind of approach
  - Others are still trying
  - BLINC uses its own new method
In Broad Daylight: Payload-based Classification

- Problem: **Encryption**
  - We don't send everything in the clear anymore

- Problem: **Privacy**
  - Requires reading over everyone's shoulders
Do Internet protocols “look” different on the wire?

in the dark

YES!


Some relevant features:

- Duration
- Bytes transferred
- Packet interarrivals
- Connection interarrivals
### V. Paxson, *Empirically-Derived Analytic Models of Wide-Area TCP Connections*

<table>
<thead>
<tr>
<th>Proto.</th>
<th>Variable</th>
<th>Model</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>telnet</td>
<td>originator bytes</td>
<td>log(_2)-extreme (Eqn 1; § 3.2)</td>
<td>(\alpha \approx \log_2 100; \beta \approx \log_2 3.5)</td>
</tr>
<tr>
<td></td>
<td>responder bytes</td>
<td>log(_2)-normal, 80-100%</td>
<td>(\bar{x} = \log_2 4500; \sigma_x = \log_2 7.2)</td>
</tr>
<tr>
<td></td>
<td>duration secs.</td>
<td>log(_2)-normal</td>
<td>(\bar{x} = \log_2 240; \sigma_x = \log_2 7.8)</td>
</tr>
<tr>
<td></td>
<td>resp. / orig.</td>
<td>log(_2)-normal</td>
<td>(\bar{x} = \log_2 21; \sigma_x = \log_2 3.6)</td>
</tr>
<tr>
<td></td>
<td>resp. / dur.</td>
<td>exponential, 0-90% resp.</td>
<td>(\lambda \approx 1/30)</td>
</tr>
<tr>
<td></td>
<td>resp. / dur.</td>
<td>log(_2)-normal, 90-100% resp.</td>
<td>(\bar{x} = 5.3; \sigma_x = 1.5);</td>
</tr>
<tr>
<td>nntp</td>
<td>originator bytes</td>
<td>log(_2)-normal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(\bar{x} \approx 11.5; \sigma_x \approx 3;)</td>
<td></td>
</tr>
<tr>
<td>smtp</td>
<td>originator bytes</td>
<td>log(_2)-normal + 300B, 0-80%;</td>
<td>(\bar{x} \approx 10; \sigma_x \approx \log_2 2.75)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>log(_2)-normal + 300B, 80-100%</td>
<td>(\bar{x} \approx 8.5; \sigma_x \approx \log_2 3)</td>
</tr>
<tr>
<td>ftp</td>
<td>connection bytes</td>
<td>log(_2)-normal</td>
<td>(\bar{x} \approx \log_2 3000; \sigma_x \approx 4)</td>
</tr>
<tr>
<td></td>
<td>session bytes</td>
<td>log(_2)-normal</td>
<td>(\bar{x} = 15; \sigma_x = 4)</td>
</tr>
<tr>
<td></td>
<td>burst bytes</td>
<td>Pareto (Eqn 2), 95-100%</td>
<td>(\alpha \approx 1; k \approx 10^{5.5})</td>
</tr>
</tbody>
</table>
At Dusk:
TCP header-based classification

- Look at the 40 bytes of TCP and IP headers in each packet to determine what the flow is doing
- More realistic
- Privacy-friendly
- Good results
At Dusk:
TCP header-based classification


- Uses Naive Bayes with modifications
- Uses info from TCP headers:
  - Flow duration
  - TCP port
  - Payload size stats (mean, variance, ...)
  - Interarrival time
A.W. Moore and D. Zuev, Internet Traffic Classification Using Bayesian Analysis Techniques

- Naive Bayes:
  - Classes $C = \{c_1, c_2, \ldots, c_k\}$
  - Observed flow $y$
  - For each class $c_j$ in $C$, calculate

$$p(c_j \mid y) = \frac{p(c_j)f(y \mid c_j)}{\sum_{c_j} p(c_j)f(y \mid c_j)}$$

- Pick the class with the highest $p(c_j \mid y)$
A.W. Moore and D. Zuev, Internet Traffic Classification Using Bayesian Analysis Techniques

Results (compared to hand-classified data)

- Naive Bayes: 65.26% of flows
- With extensions: 96.29% of flows

Still using port numbers

- Vin Diesel doesn't use port numbers
- Why should we?
At Dusk: TCP header-based classification


- *Uses a Decision Tree Classifier to identify traffic from 5 application protocols*

- *Unit of analysis is a sliding window of packets, over which average values are calculated for packet size, interarrival time, and TCP flags*
Early et al., Behavioral Authentication of Server Flows

- Sliding window technique
  - Looks at a sliding “window” of packets, calculates average values of packet size, interarrival time, TCP flags, etc

- Example:

\[ W_1 \]

Whew! They dodged a bullet with this one!

E Keogh, et al., Clustering of Time Series Subsequences is Meaningless. ICDM'03
Early *et al.*, *Behavioral Authentication of Server Flows*

- Sliding window technique
  - Looks at a sliding “window” of packets, calculates average values of packet size, interarrival time, TCP flags, etc

- Example:

  ![Image](image)

  \[ W_2 \]

  Whew! They dodged a bullet with this one!

E Keogh, *et al.*, *Clustering of Time Series Subsequences is Meaningless*, ICDM'03
Early et al., *Behavioral Authentication of Server Flows*

- Sliding window technique
  - Looks at a sliding “window” of packets, calculates average values of packet size, interarrival time, TCP flags, etc

- Example:

  $W_3$

Whew! They dodged a bullet with this one!

E Keogh, *et al.*, *Clustering of Time Series Subsequences is Meaningless*, ICDM'03
Early et al., *Behavioral Authentication of Server Flows*

- Decision Tree Classifier (C5.0 Algorithm)
  - automatic feature selection
  - automatically partition the parameter space to achieve maximum information gain on the training set

- Procedure:
  - Classify each window of packets
  - Give the whole flow the label most often assigned to its component windows
Early et al., *Behavioral Authentication of Server Flows*

- The decision tree algorithm finds that the *most distinguishing feature* of HTTP traffic is the TCP “push” flag (!)
- Recognition rates generally > 90% on synthetic and real-world data
- SMTP is harder to distinguish from FTP and Telnet
  - Multi-modal behaviors and similar-looking protocols can make recognition difficult
It's Getting Dark...

- What if we restrict our analysis to info available at the network layer?

- We're left with
  - Packet Size
  - Direction
  - Interarrival Time

  to guide us in making our decisions
It's Getting Dark...


Unsupervised technique: uses $k$-means clustering to group flows together based on

- Packet size statistics (min, max, quartiles)
- Interarrival statistics
- Byte counts
- Duration
- Idle time
It's Getting Dark...

C. Wright, F. Monrose, and G. Masson, HMM Profiles for Network Traffic Classification (Extended Abstract) in DMSEC'04.

- Very “lean” data: uses only packet size, direction, and interarrival time
- Key assumption: where in the stream a given packet occurs tells us what it should look like
where a given packet occurs tells us what it should look like

http://www.cs.jhu.edu/~cwright/traffic-viz
where a given packet occurs tells us what it should look like
where a given packet occurs tells us what it should look like.

http://www.cs.jhu.edu/~cwright/traffic-viz
where a given packet occurs tells us what it should look like

http://www.cs.jhu.edu/~cwright/traffic-viz
Profile HMMs
Profile HMMs: Empirical Evaluation

- Ideally, we'd train on one network (GMU), and test on another (JHU? LBL?)
  - And we will! Soon!
- In the mean time, we use data from several days spread over a month
  - Train on one, Test on the others, Repeat
- Therefore, model construction must be highly automated
  - Parameters and thresholds are derived from training data
Profile HMMs: Challenges

- Multi-Modal Behaviors
  - Example: SSH and SCP
  - Solution: mixture models (?)

- Long-Lived Connections

- Non-Linear Behaviors
  - Solution: better topology (?)
Practical Application: Protocol Detectors
It always gets darkest... in a **tunnel**

- What if we can't tell which packets belong to the same flow?
  - The simplest case: one protocol, many connections passing through one tunnel
  - The realistic case (IPSec): one tunnel, a handful of protocols, many connections
one protocol, one tunnel, many connections

- We can handle this case too
  - Chop the sequence of tunnel packets into many small slices
  - Count up how many packets of each type arrive during each slice of time
  - Use a simple $k$-Nearest Neighbor classifier

- What's more, we can even count the number of connections in the tunnel
one protocol, one tunnel, many connections

- Simplifying assumptions:
  - (see scribe notes)
one protocol, one tunnel, many connections