Statistical Protocol IDentification with SPID: Preliminary Results

Erik Hjelmvik & Wolfgang John

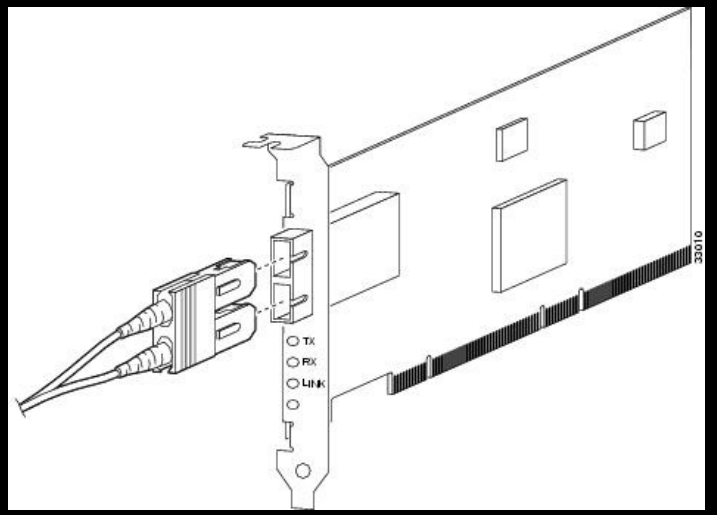
SNCNW'09

Network Traffic Classification

- Identify the application layer protocol
- Traffic classification is needed for:
  - QoS assignment and traffic shaping
  - Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS)
  - Deep Packet Inspection (DPI)
  - Network Forensics

Current Classification Approaches

- Port numbers
- Pattern-matching in payload
- Connection patterns
- Flow properties



Automatic traffic classification is difficult!

Validation Results of SPID

Overall results:

- Recall: 91.1%
  (few missed sessions)
- Precision: 100%
  (no false positives)

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

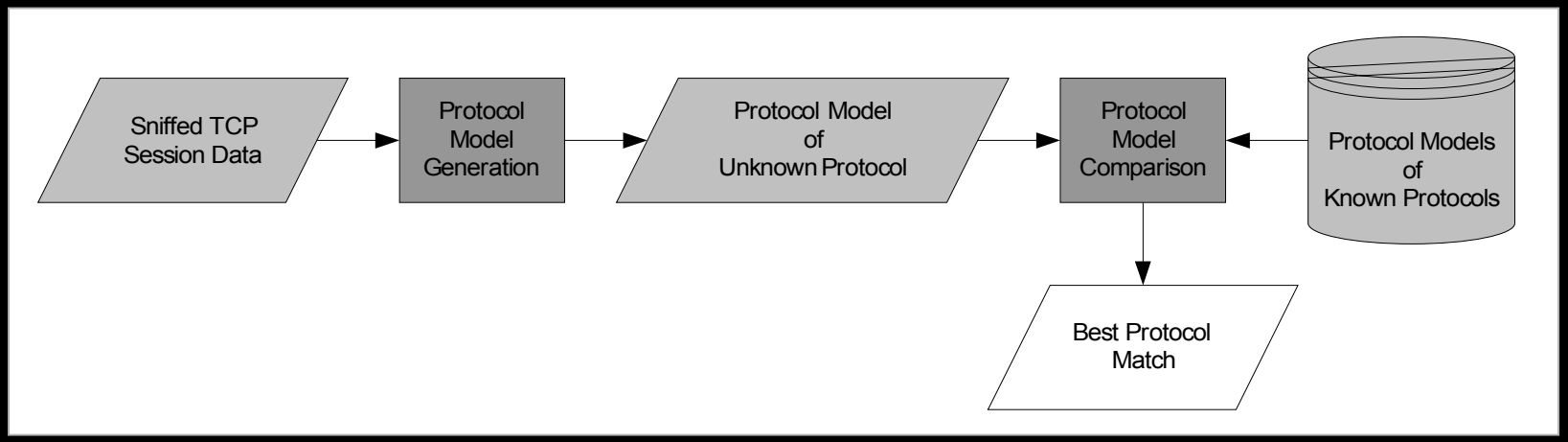
$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Results per protocol:

- BitTorrent: 98.1% recall
- eDonkey: 77.6% recall
- HTTP: 97.0% recall
- SSH: 100% recall
- SSL: 86.7% recall

The SPID Algorithm

Protocol identification based on statistical measurements of various protocol attributes



Design goals:

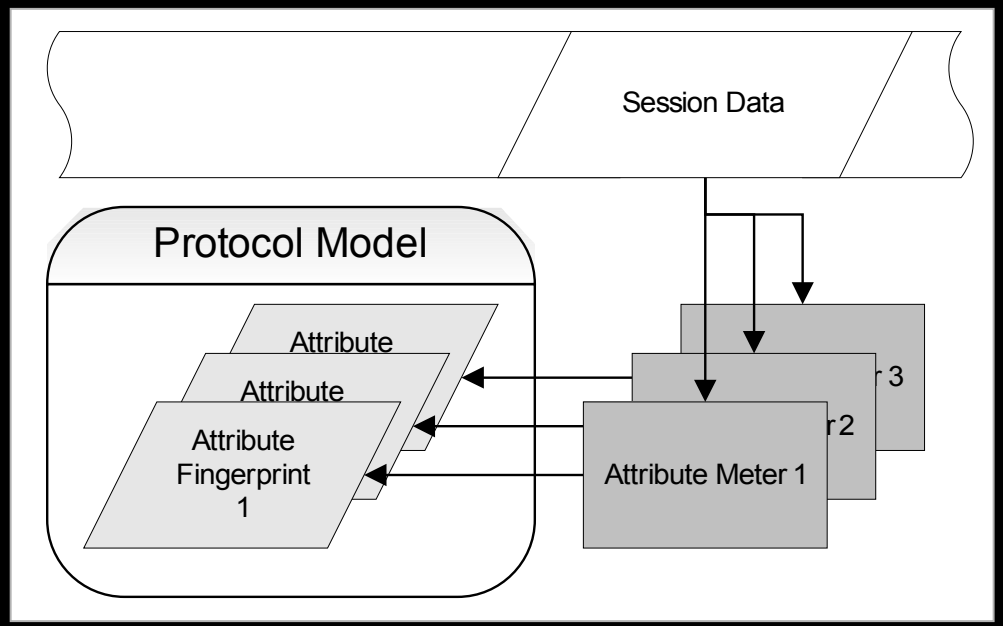
- No manual fingerprint generation
- Low time complexity
- Early protocol identification

Attribute Meters - The Heart of SPID

http://spid.wiki.sourceforge.net/AttributeMeters

An Attribute Meter is a function that provides measurements of a specific property (attribute)

A Protocol Model contains statistics from over 30 attribute meters

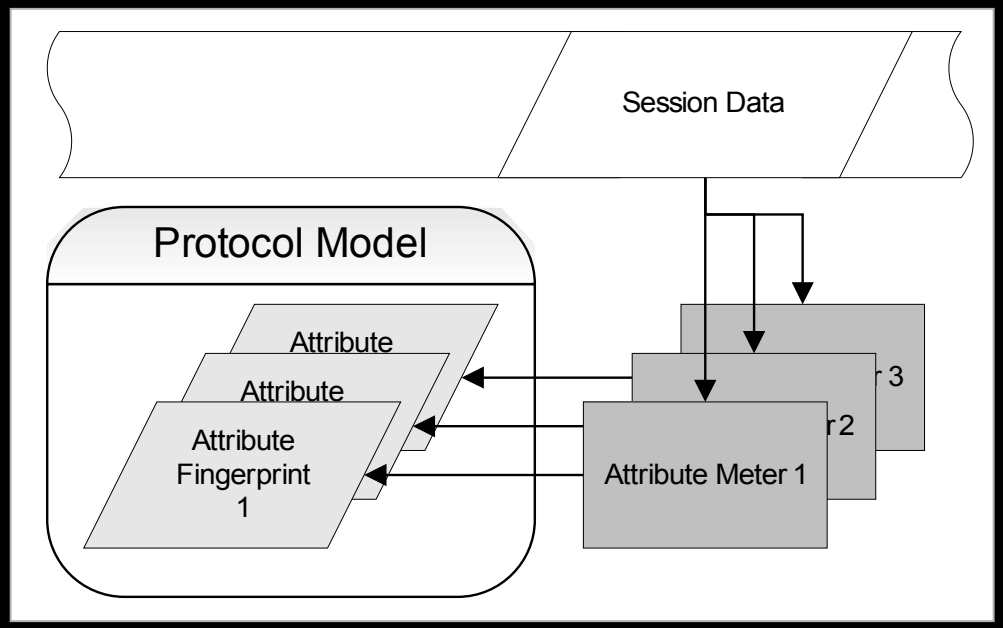


Attribute Meters - The Heart of SPID

http://spid.wiki.sourceforge.net/AttributeMeters

Properties can be:

- statistical flow features
- byte frequencies
- byte sequences
- offsets for common byte-values



Attribute Meters - Example1 (payload)

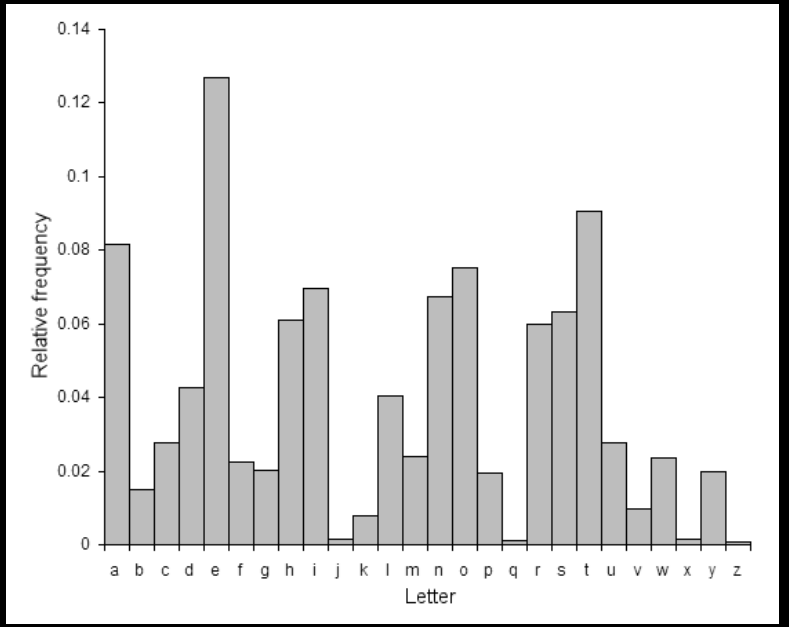
http://spid.wiki.sourceforge.net/ByteFrequencyMeter

ByteFrequencyMeter:

- frequency of byte values in application layer data

HTTP has a high frequency of:

- 0x20 [space]: 5.4%
- 0x65 'e': 4.0%
- 0x74 't': 3.1%



eDonkey has a high frequency of:

- 0x00 [null]: 11.0%
- 0x20 [space]: 8.4%
- 0x57 'W': 8.6%

Attribute Meters - Example 2 (flow)

http://spid.wiki.sourceforge.net/DirectionPacketLengthDistributionMeter

DirectionPacketLengthDistributionMeter:

- packet sizes and packet directions

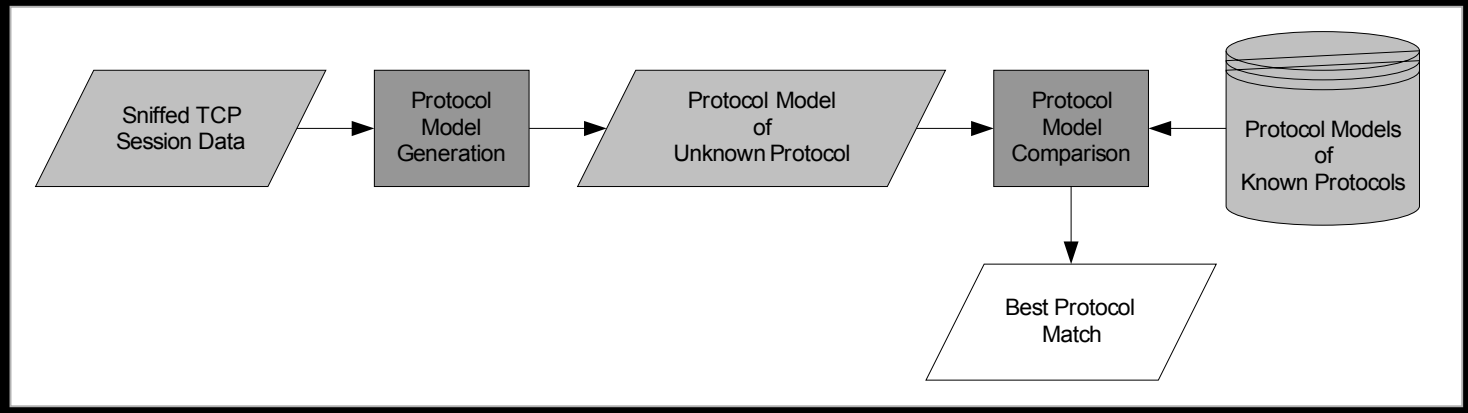
Large SSL packets (>1400 bytes):

- Client to Server: 13%
- Server to Client: 34%

Large IRC packets (>1400 bytes):

- Client to Server: 0%
- Server to Client: 10%

Comparing Protocol Models



Kullback-Leibler divergence (relative entropy):

- P = observed session's probability distribution
- Q = protocol model's probability distribution

$$D_{KL}(P_{attr}||Q_{attr,prot}) = \sum_i P_{attr}(i) * log_2 \frac{P_{attr}(i)}{Q_{attr,prot}(i)}$$

Small K-L divergence = good match

Future Work

More training data:

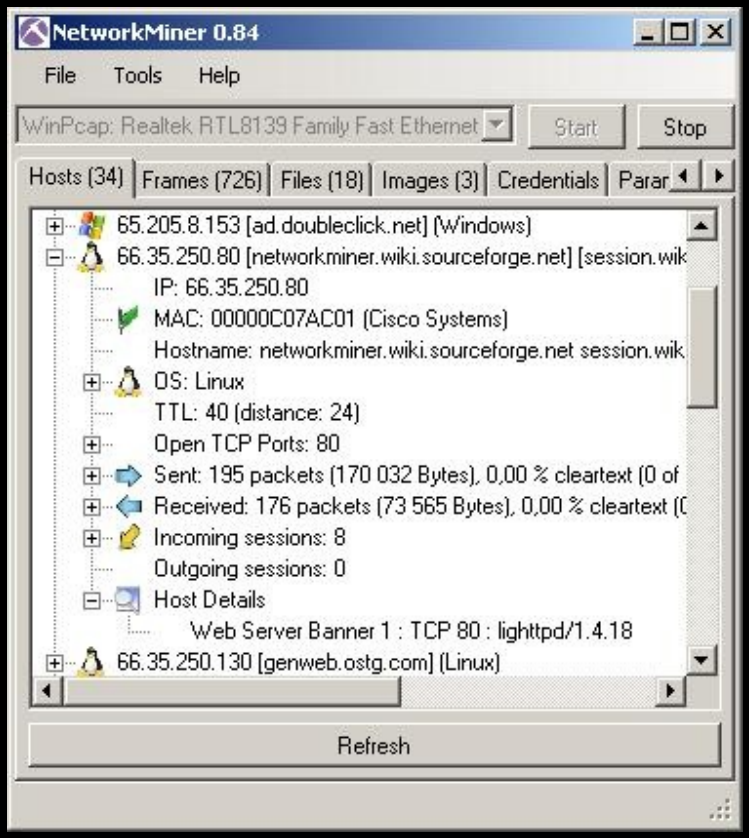
- Improve existing protocol models
- New protocol models

Reduced set of attribute meters

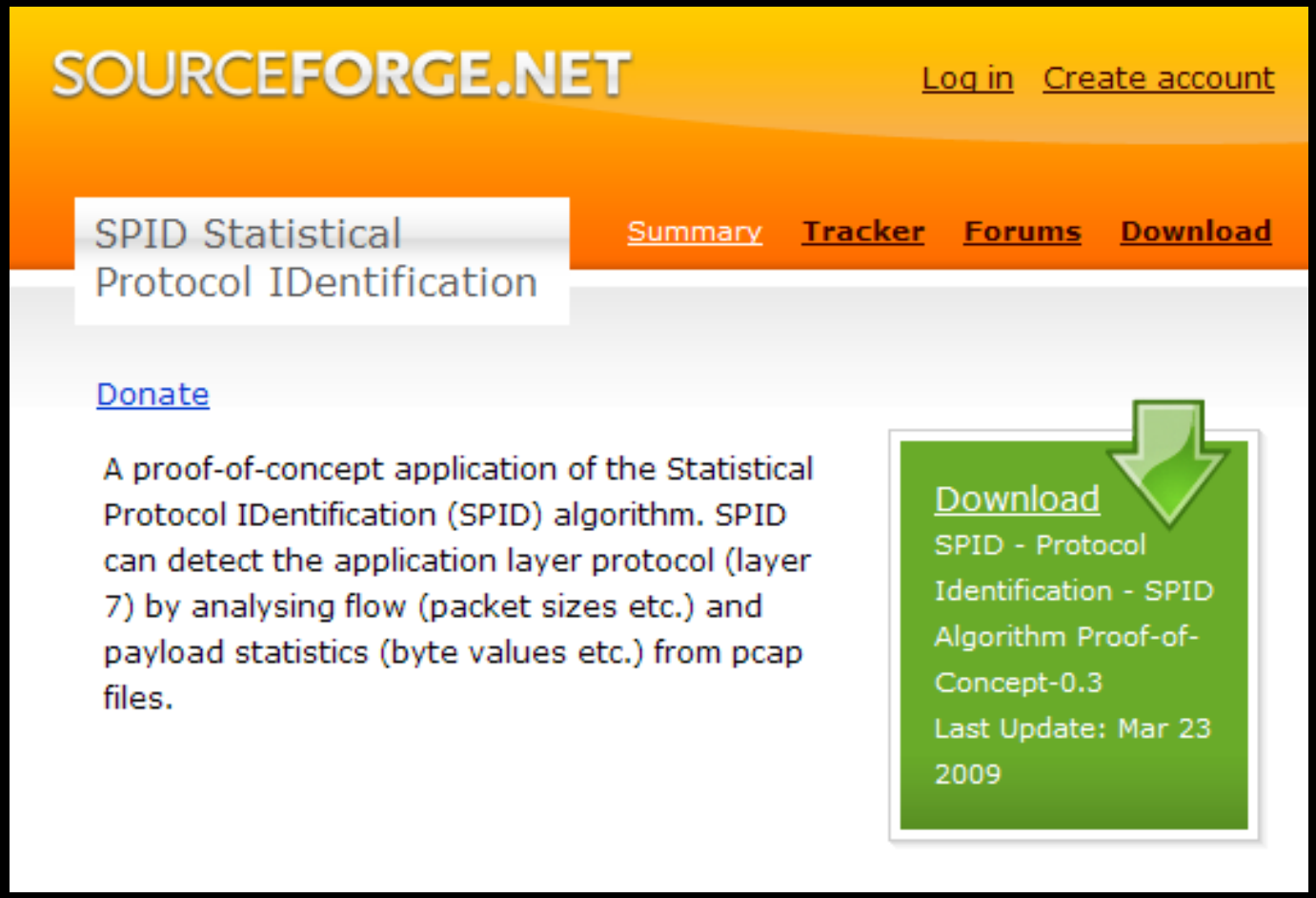
Effect of different network types:

- LAN data
- Backbone data

Implement SPID in NetworkMiner
http://networkminer.sourceforge.net/



Try out SPID for yourself!



http://sourceforge.net/projects/spid (SPID download)
http://spid.wiki.sourceforge.net/ (SPID wiki-pages)