Entropy-based event detection

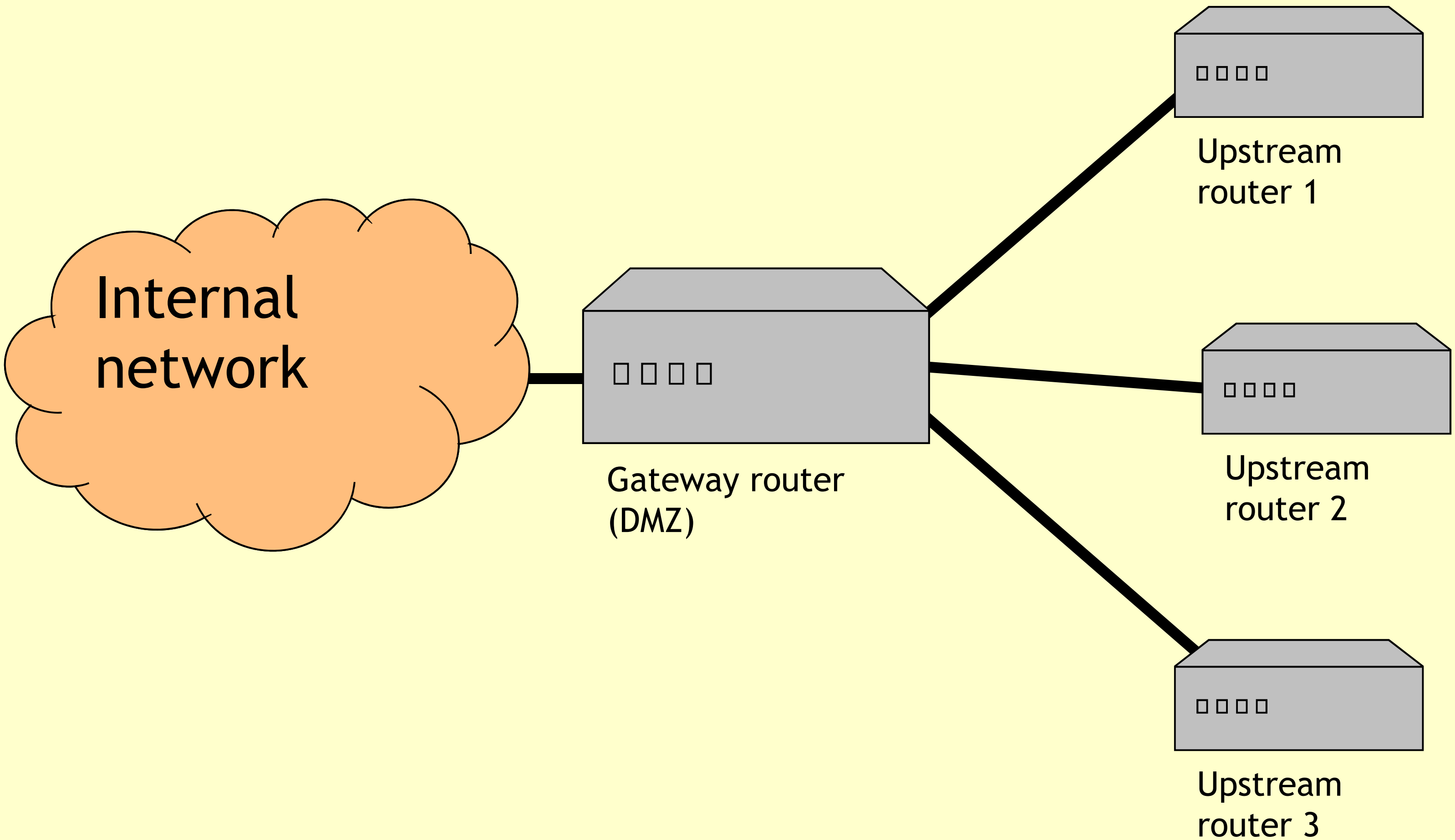
Ulrich Speidel
Department of Computer Science
The University of Auckland
ulrich@cs.auckland.ac.nz

Joint work with Raimund Eimann

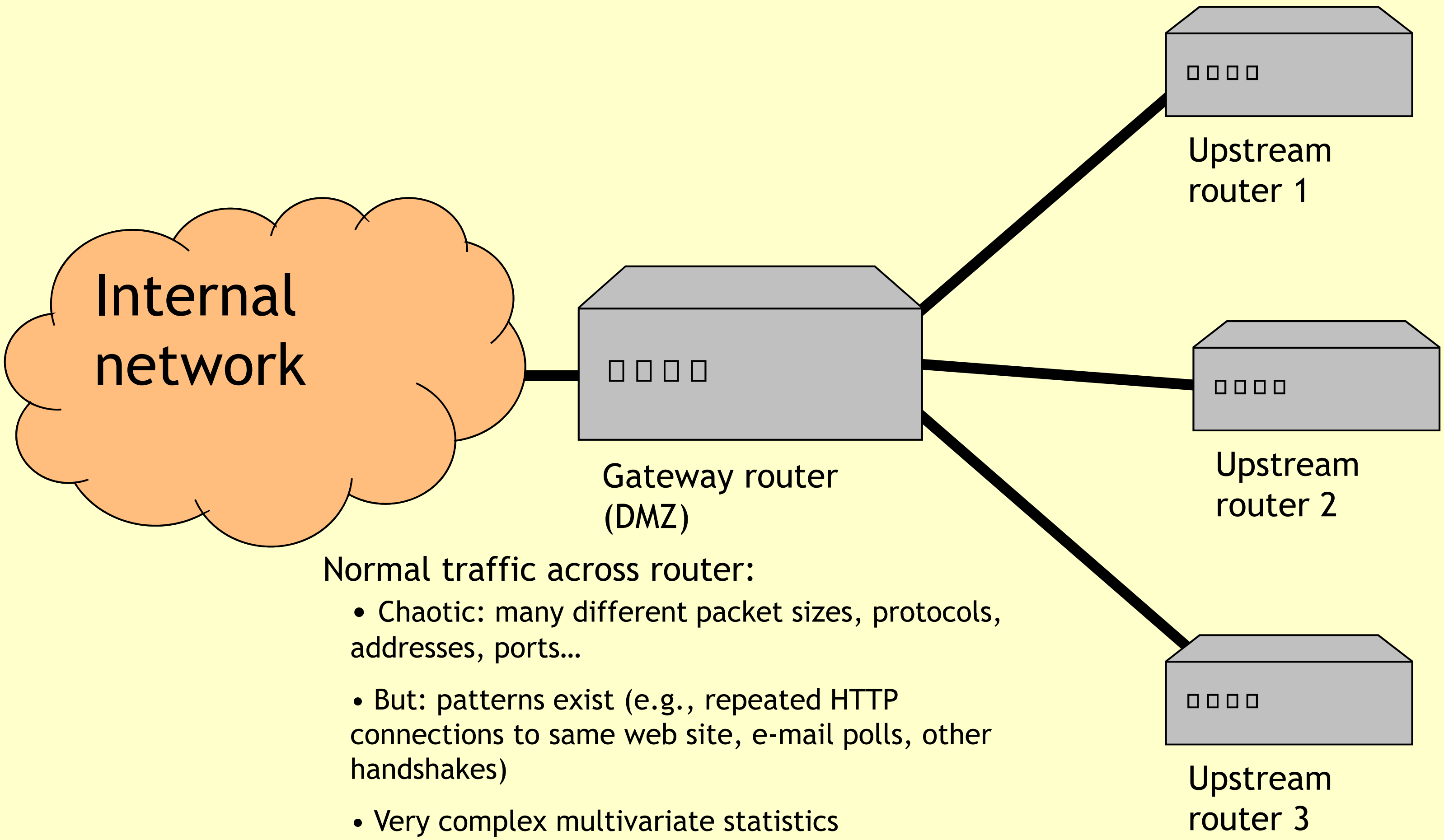
What's the problem?

- Complex systems – such as computer networks - have observables that yield multivariate time series data
- Chaotic behaviour is actually *normal* (to an extent)
- So: How does one detect significant system events and how does one distinguish them from normal behaviour?
- What if we don't know very well in advance what effect the event will have on the observables?

Scenario: Computer networks



Scenario: Computer networks

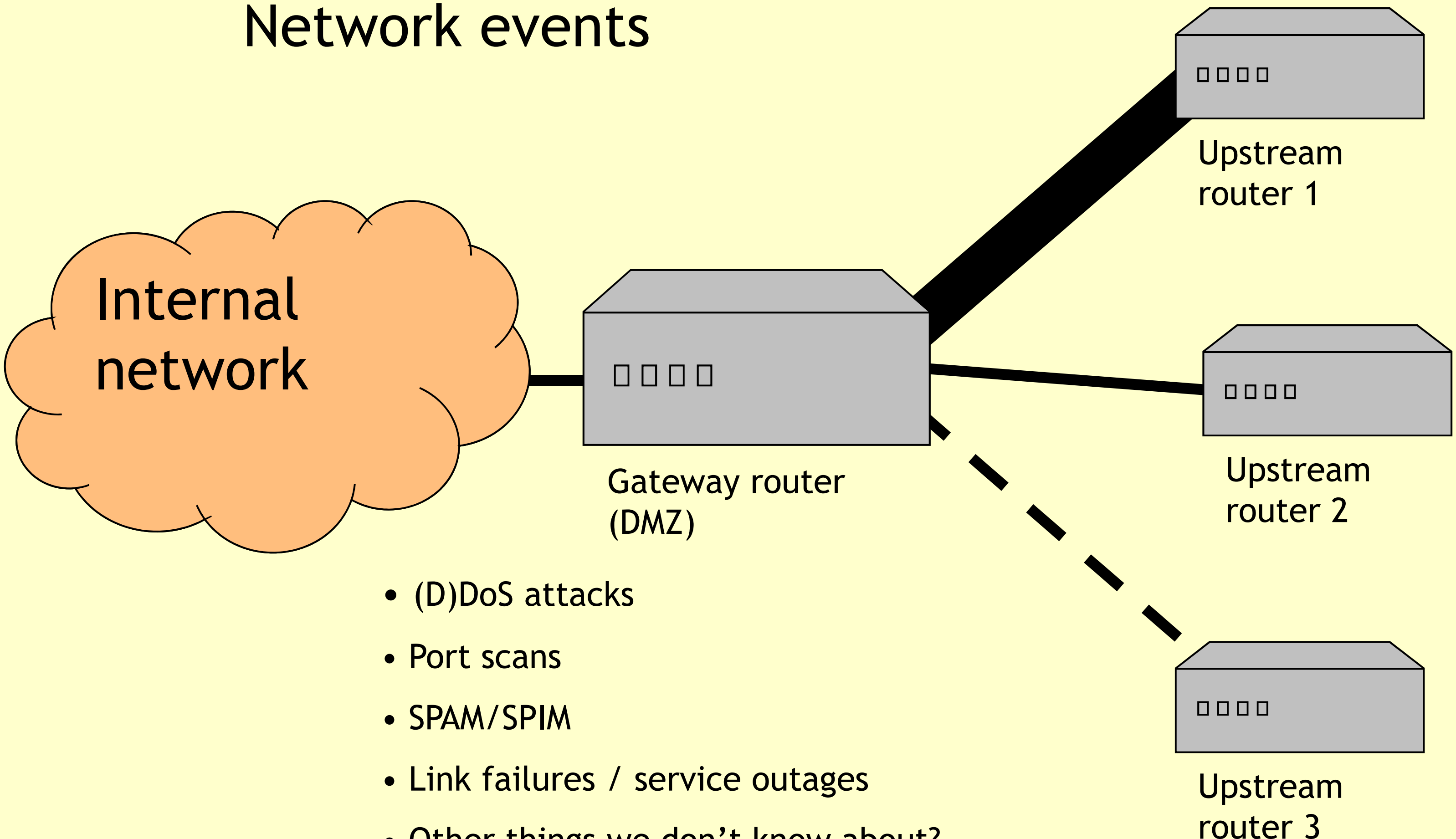


Normal traffic across router:

- Chaotic: many different packet sizes, protocols, addresses, ports…
- But: patterns exist (e.g., repeated HTTP connections to same web site, e-mail polls, other handshakes)
- Very complex multivariate statistics

Scenario: Computer networks

Network events



- (D)DoS attacks
- Port scans
- SPAM/SPIM
- Link failures / service outages
- Other things we don't know about?

Other examples

- Industrial process data (e.g., sensor data in smelter processes)
- Medical data (e.g., ECG/EEG/BP data)
- Traffic monitoring
- Airline data

Need to detect anomalies in order to find out what causes them

Conventional approaches

- E.g., monitor packet rate/interarrival times - not useful if router hits saturation during normal operation

Conventional approaches

- Monitor packet size distribution - complex diagram (histogram), fluctuates significantly with time, does not detect some events (e.g., port scans or link failures may go undetected)

Conventional approaches

- Monitor individual protocols, ports, or payload - generally too selective and complex to monitor – information is hard to aggregate and events are easily missed (especially new ones)
- Currently one of the more popular techniques, though

Conventional approaches

- Most if not all conventional approaches are complex and rather narrowband
- Focus is on a single observable, not aggregate
- *Patterns* do not play a major role

Patterns

- IP network traffic contains patterns
- E.g., handshakes, request/response packets in protocols such as HTTP etc.
- E.g., certain ports and IP addresses are seen more often than others, and tend to occur in close temporal proximity
- Permits a certain degree of predictability (low entropy)
- In other words: certain possible patterns occur much more often than others

Entropy monitoring

- Entropy = information rate (f.t.p.o.t.t.*)
- Postulate: Entropy of network traffic changes as patterns in the traffic change
- Network events cause change in patterns and hence change the observed entropy
- Not in itself a new concept:
  - Kulkarni, Bush, and Evans (2002): approximate entropy by LZ compression
  - Feinstein, Schnackenberg, Balupari, and Kindred (2003): use Shannon entropy
  - Wagner and Plattner (2005): also use compression-based monitoring

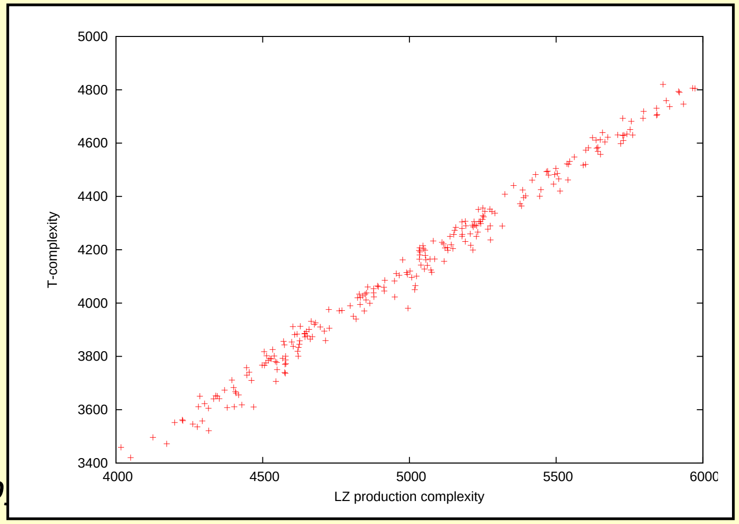
*for the purposes of this talk

Entropy measurement

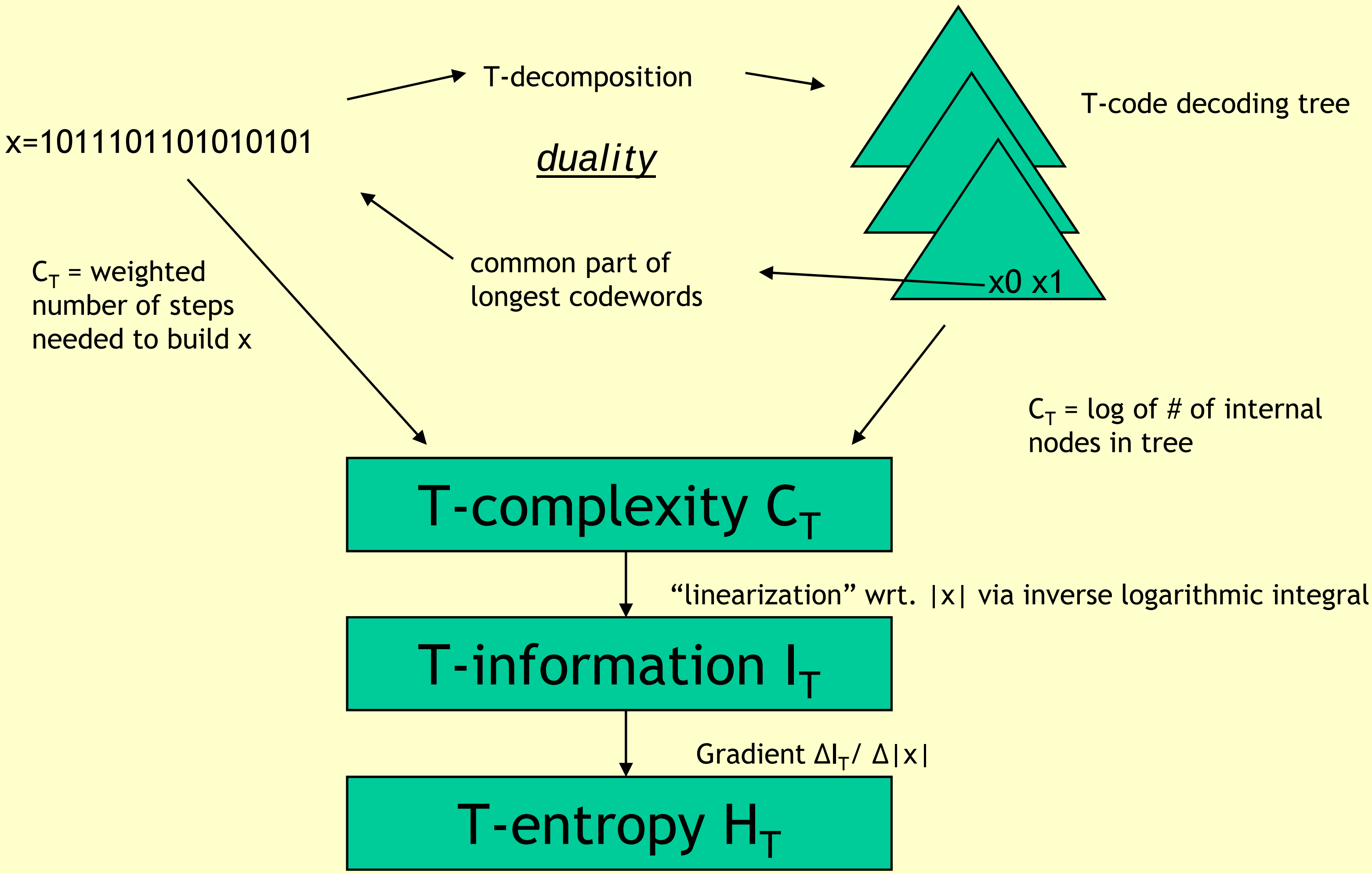
- Kulkarni, Bush, and Evans (2002): Quality of entropy-based detection depends on having a good entropy measure
- Fundamental problem: computability
- We can't measure, but we can estimate
- Classical estimators: Statistical/Shannon (bad), but also Lempel-Ziv algorithms (better, 1976 production complexity, LZ77, LZ78)
- Fundamental problem: Overestimation or time/space complexity

Possible alternative: T-Entropy

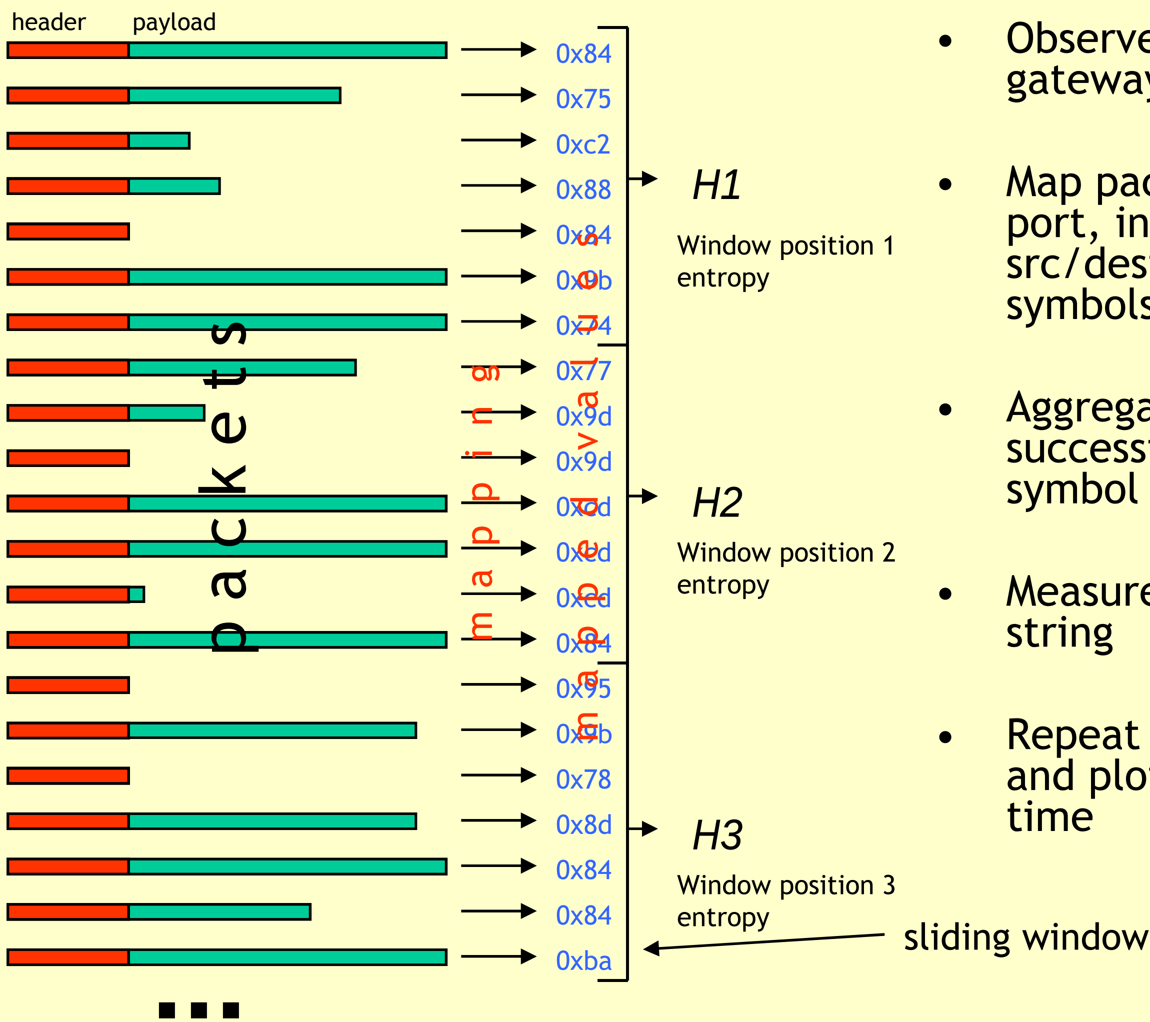
- Entropy measure developed by Mark Titchener in the late 1990's
- Based on the duality between finite strings and a family of recursively constructed variable-length code sets called *T-codes*
- Can be implemented to run in O(N log N) [Speidel and Yang 2005]
- Seems to be more sensitive for short strings than LZ-based estimators but correlates well with the latter [Speidel 2009]



T-entropy: conceptual overview



Network event detection: Methodology in principle

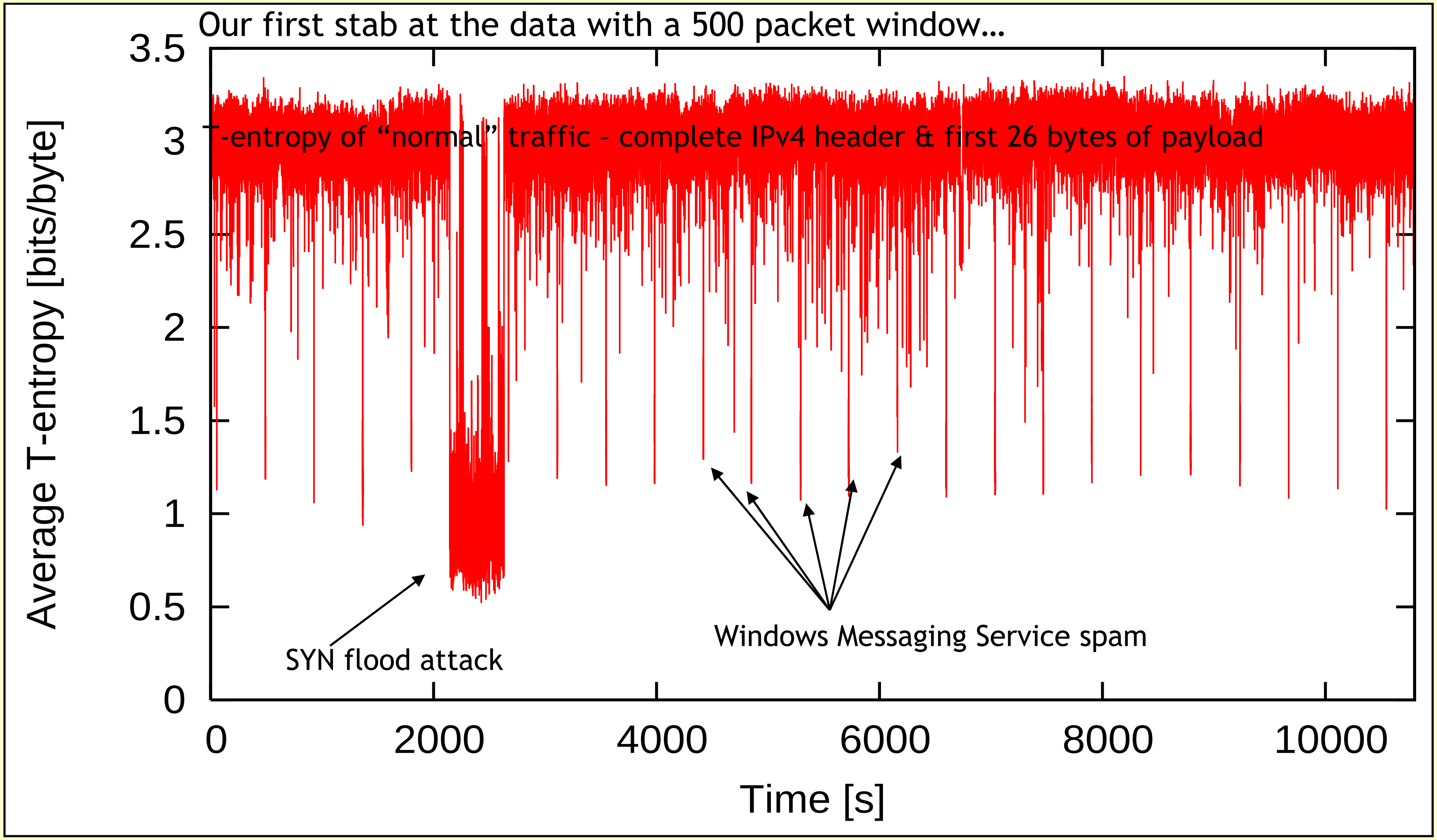


- Observe packets, e.g., at border gateway
- Map packet properties (e.g., length, port, interarrival time, protocol, src/dest address) into binary 8-bit symbols (one or several per packet)
- Aggregate several hundred or more successive mapped packets into a symbol string (sliding window)
- Measure average T-entropy of that string
- Repeat for sliding window over time and plot T-entropy values against time

Experimental results

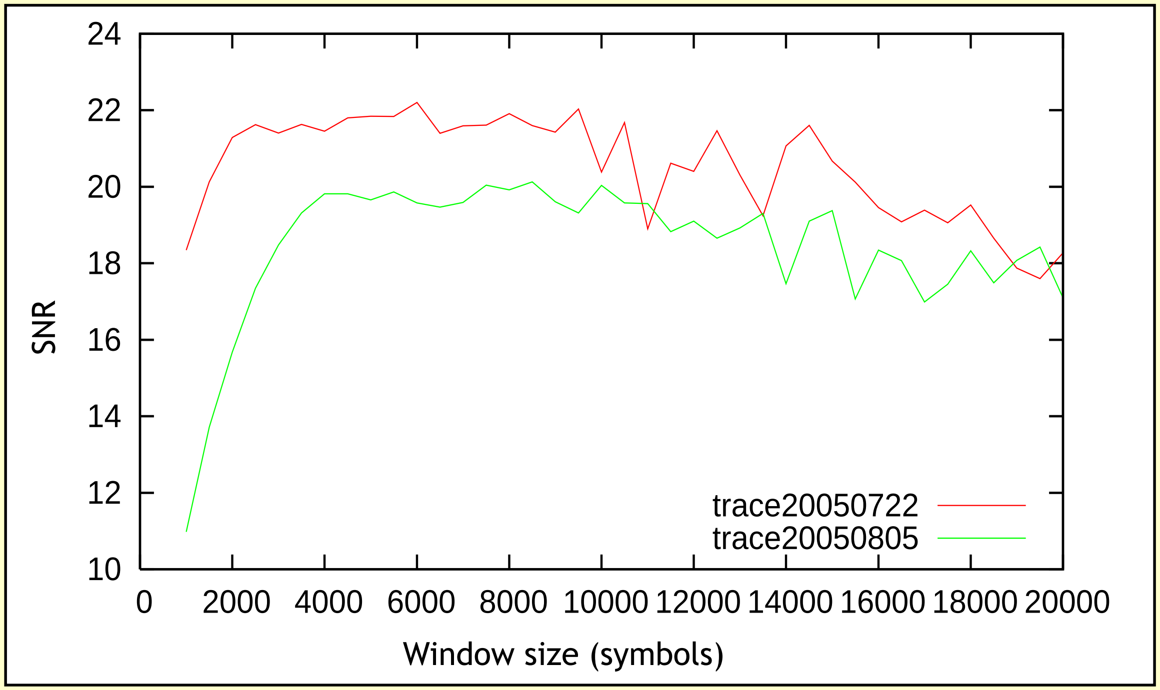
- Three hour IP datagram traces from U of Auckland's DMZ gateway
- Typical datagram rate about 8000 datagrams per second
- Processing time for a three hour trace file: 45 minutes on a normal state-of-the-art PC (2006)
- Various mappings and filters were applied
- The ones shown here today use the *full IPv4 information + 48 bytes of the payload* and use a 5000 packet window shifting by 0.675 seconds at a time

Experimental results



Experimental results

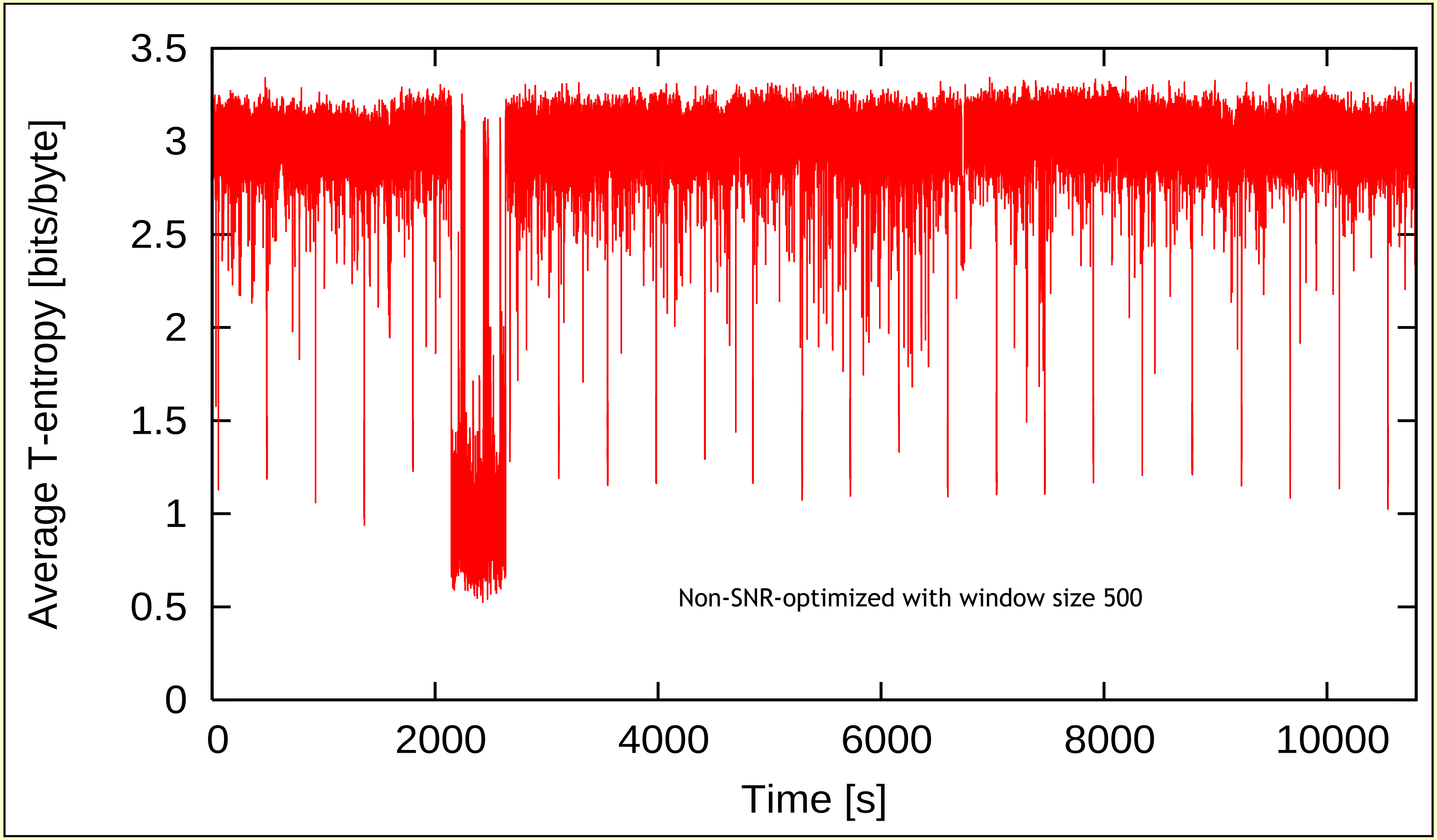
- Observation 1: Data is noisy!
- Observation 2: Depth of entropy drops depends on size of window – the longer the window, the shallower the drops
- Observation 3: The longer the window, the less noise we get
- Question: Can we define a kind of SNR (signal-to-noise ratio) and try to optimize the window size?



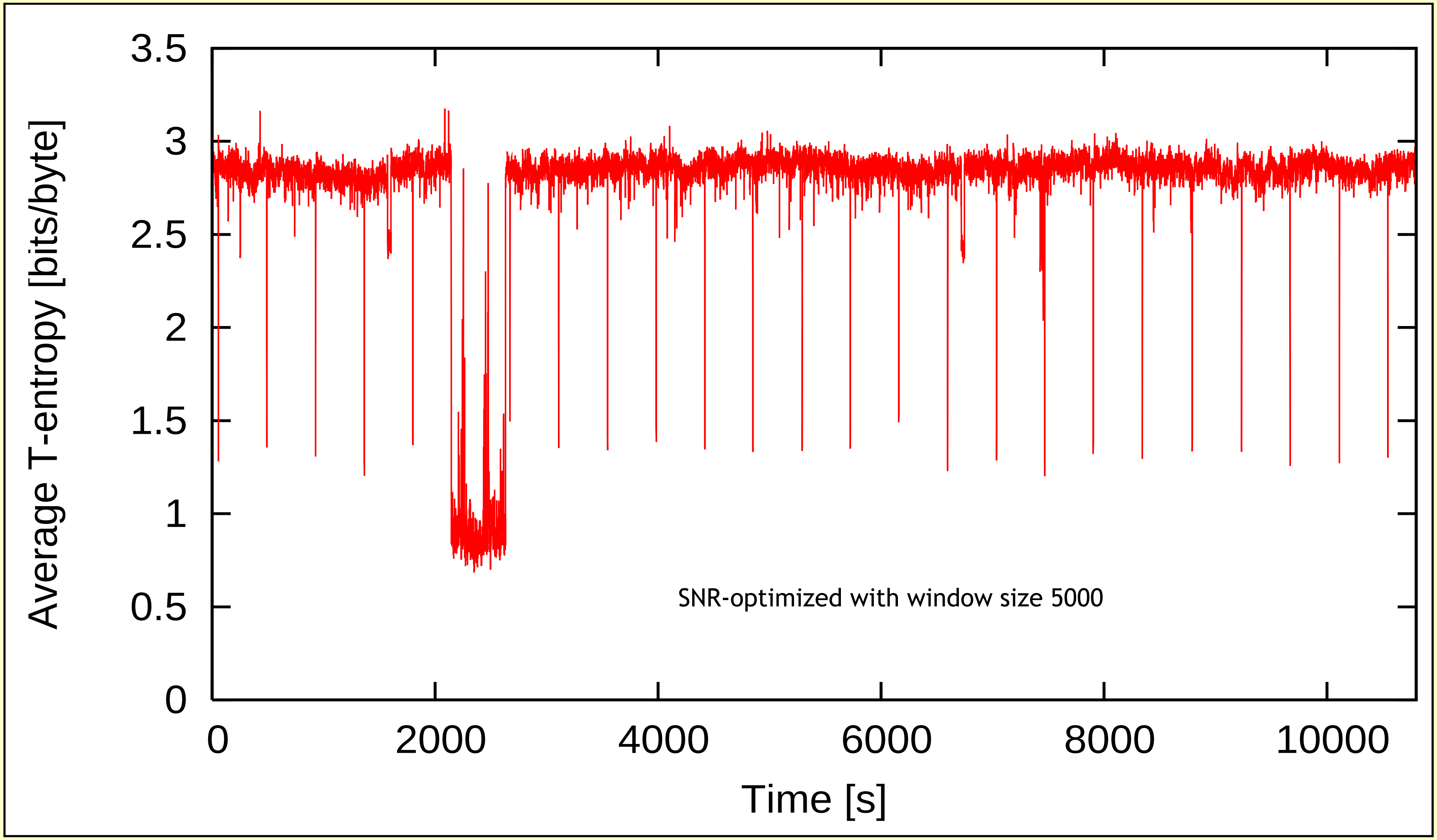
Findings:

- Window size of approx. 5000 maximizes SNR in this sample
- Optimal window size is event duration dependent

Before SNR optimization



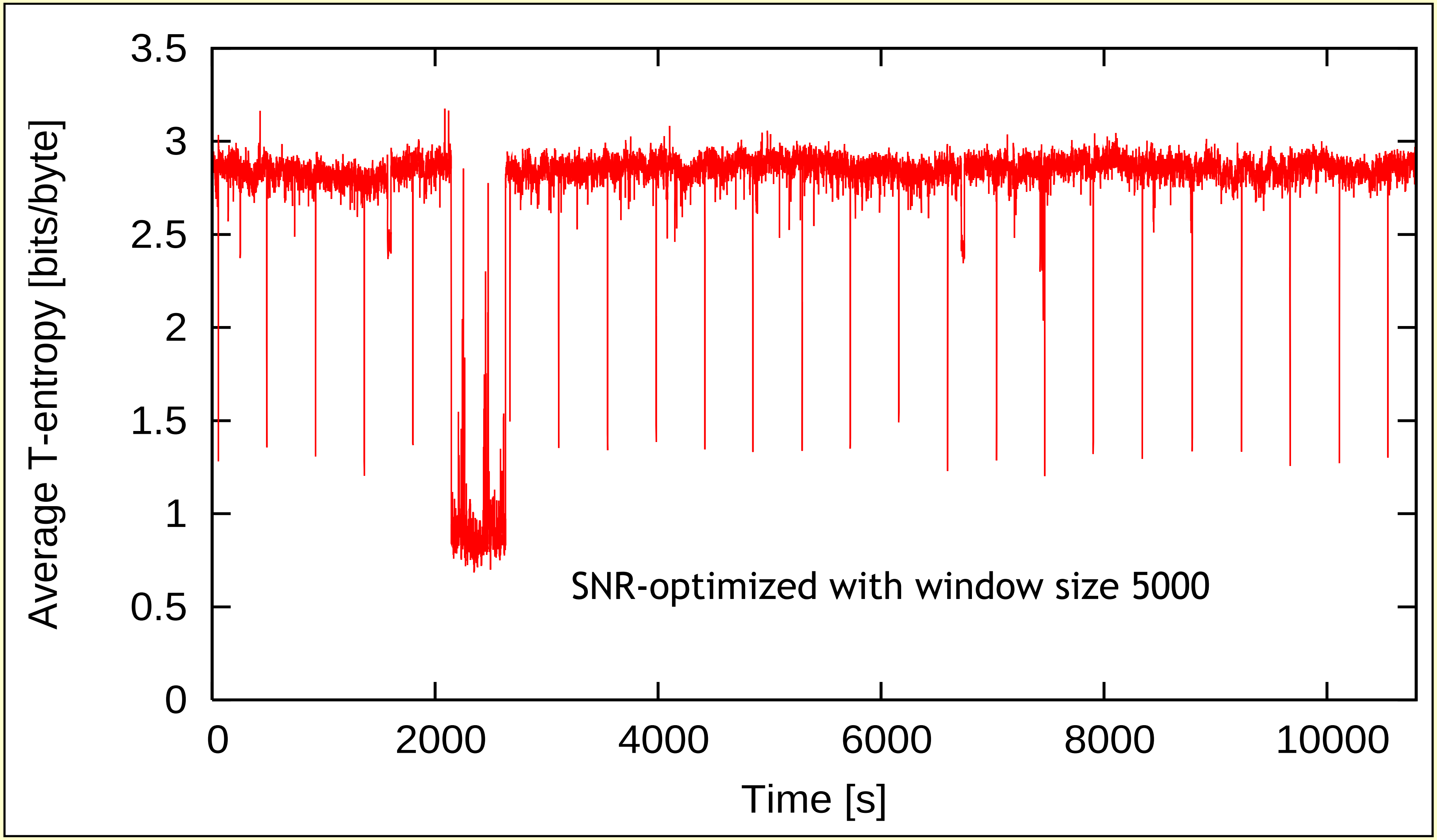
After SNR optimization



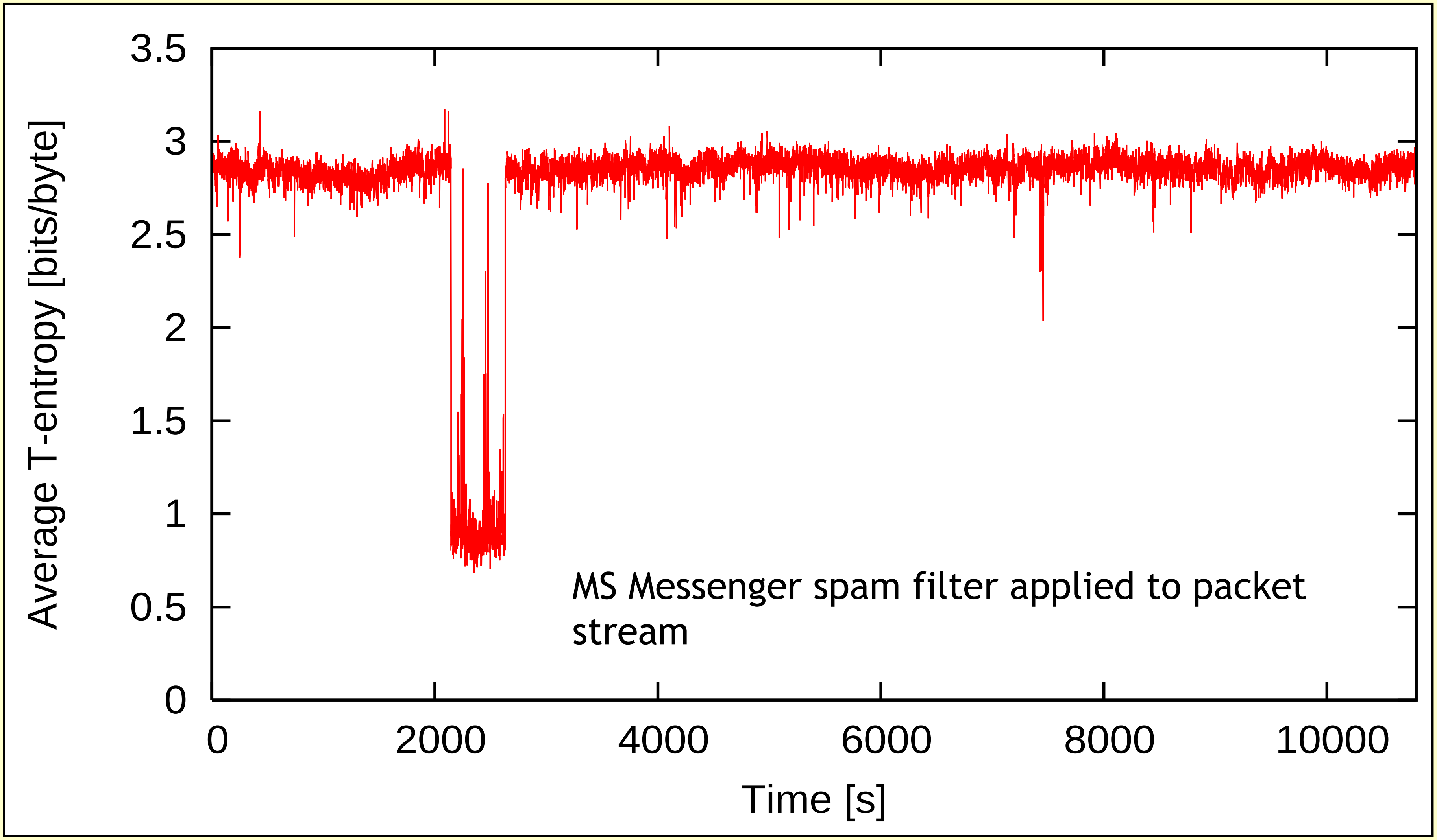
How do we know that the drops are caused by the events?

- Need to show that the events are both necessary and sufficient to cause entropy drops
- Can show necessity by removing event-related packets
- Can show sufficiency by artificially inserting synthetic events into the traces (simulation)

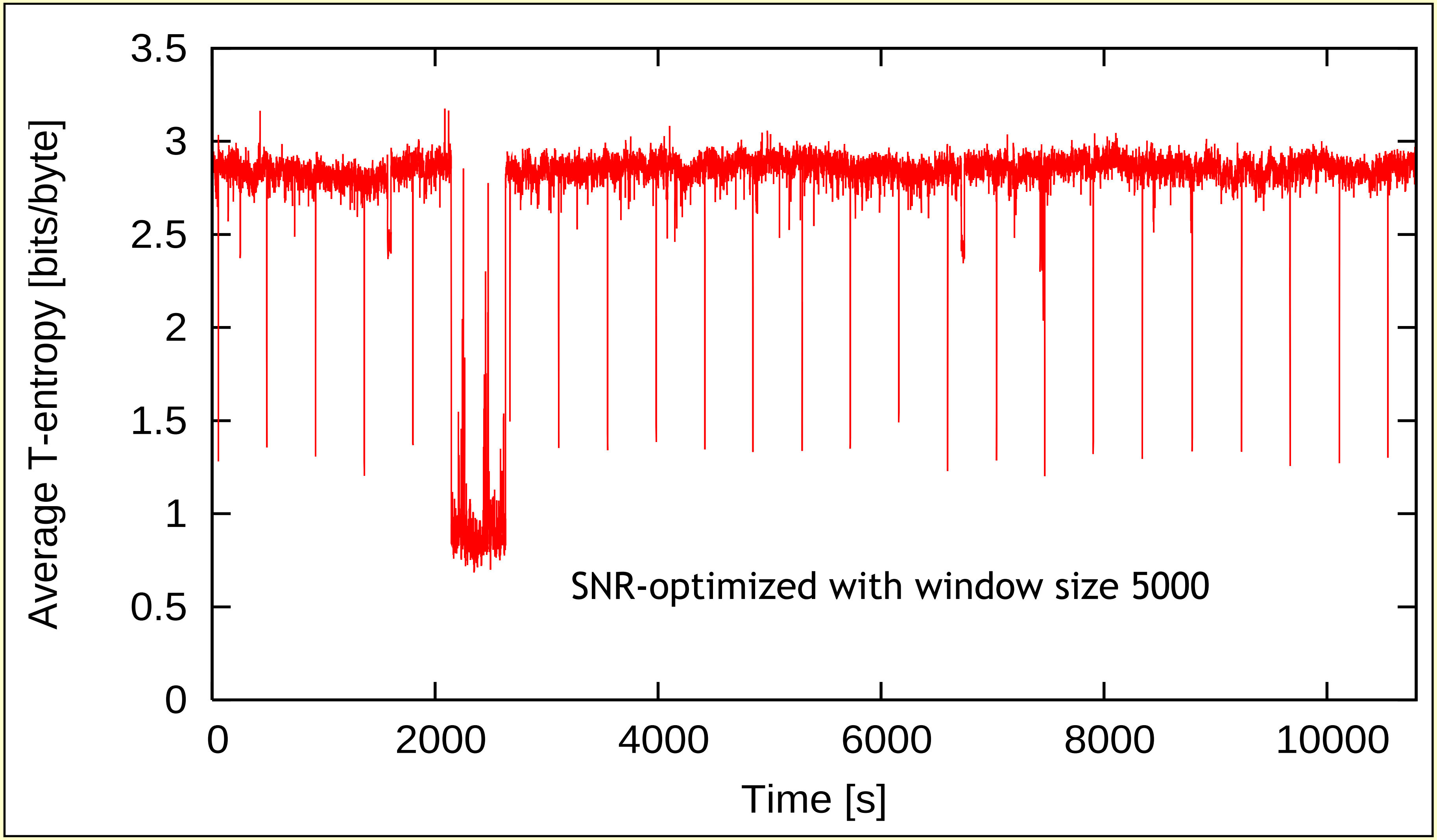
Entropy - unfiltered



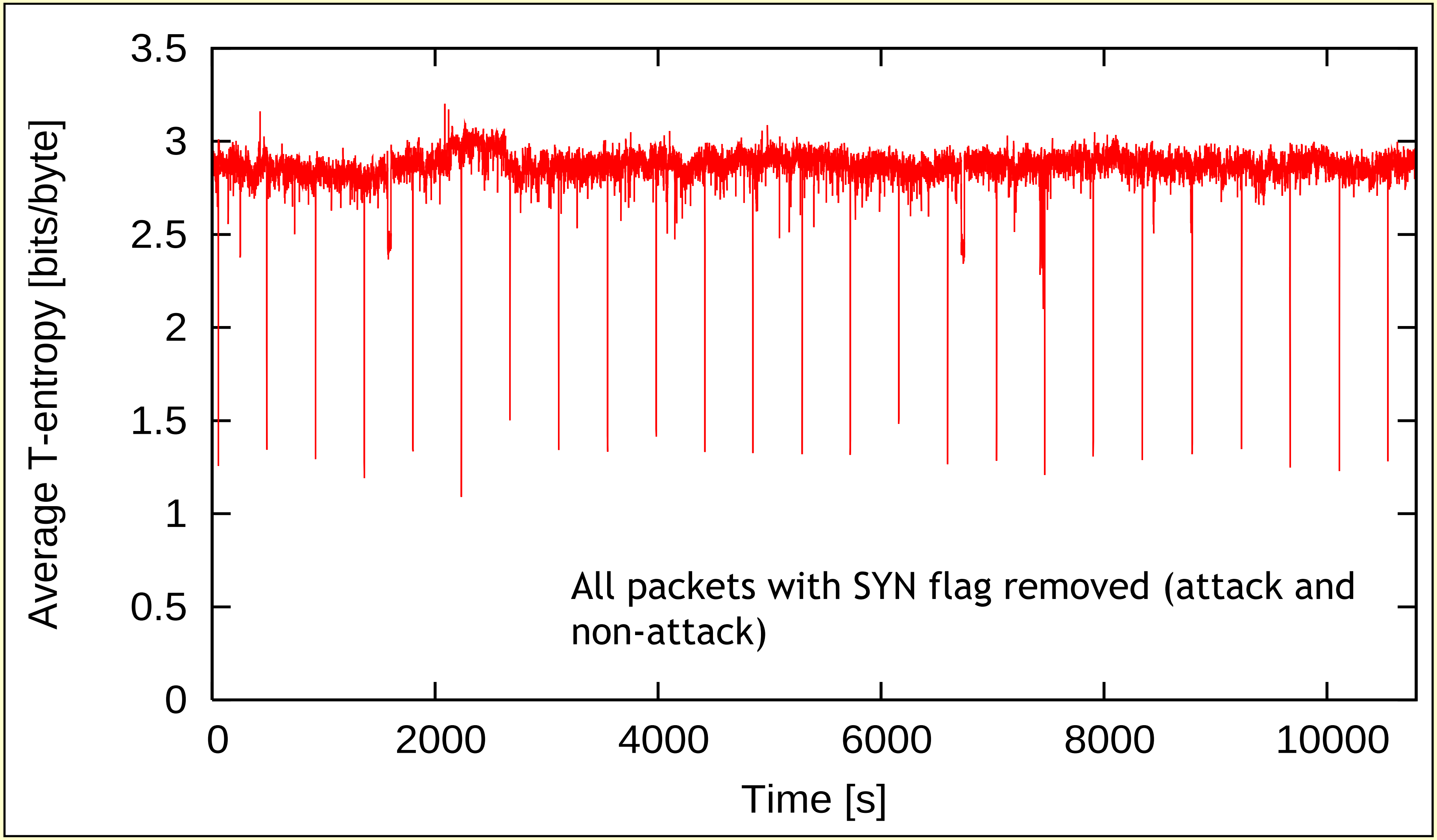
Entropy - filtered



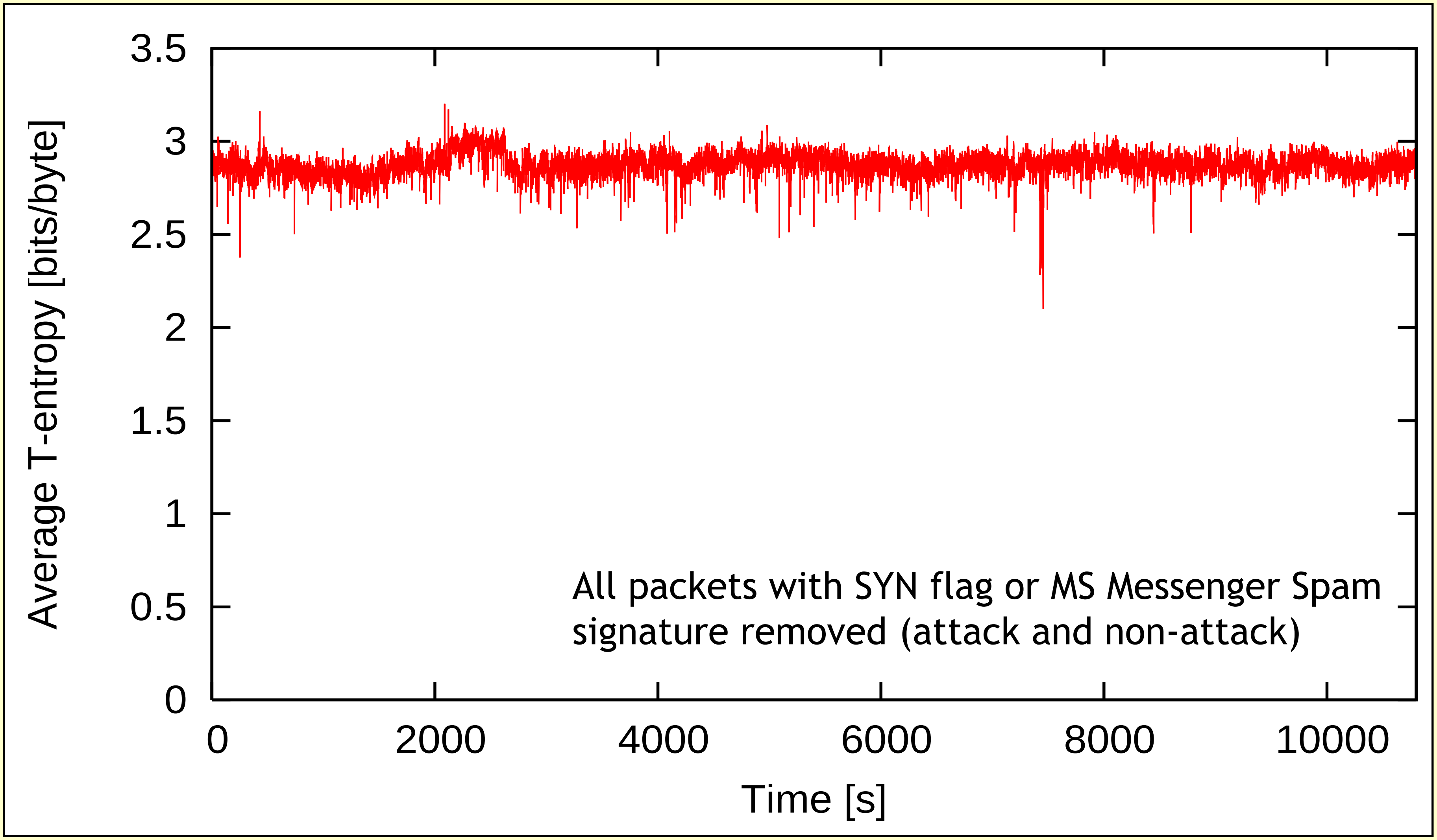
Entropy - unfiltered



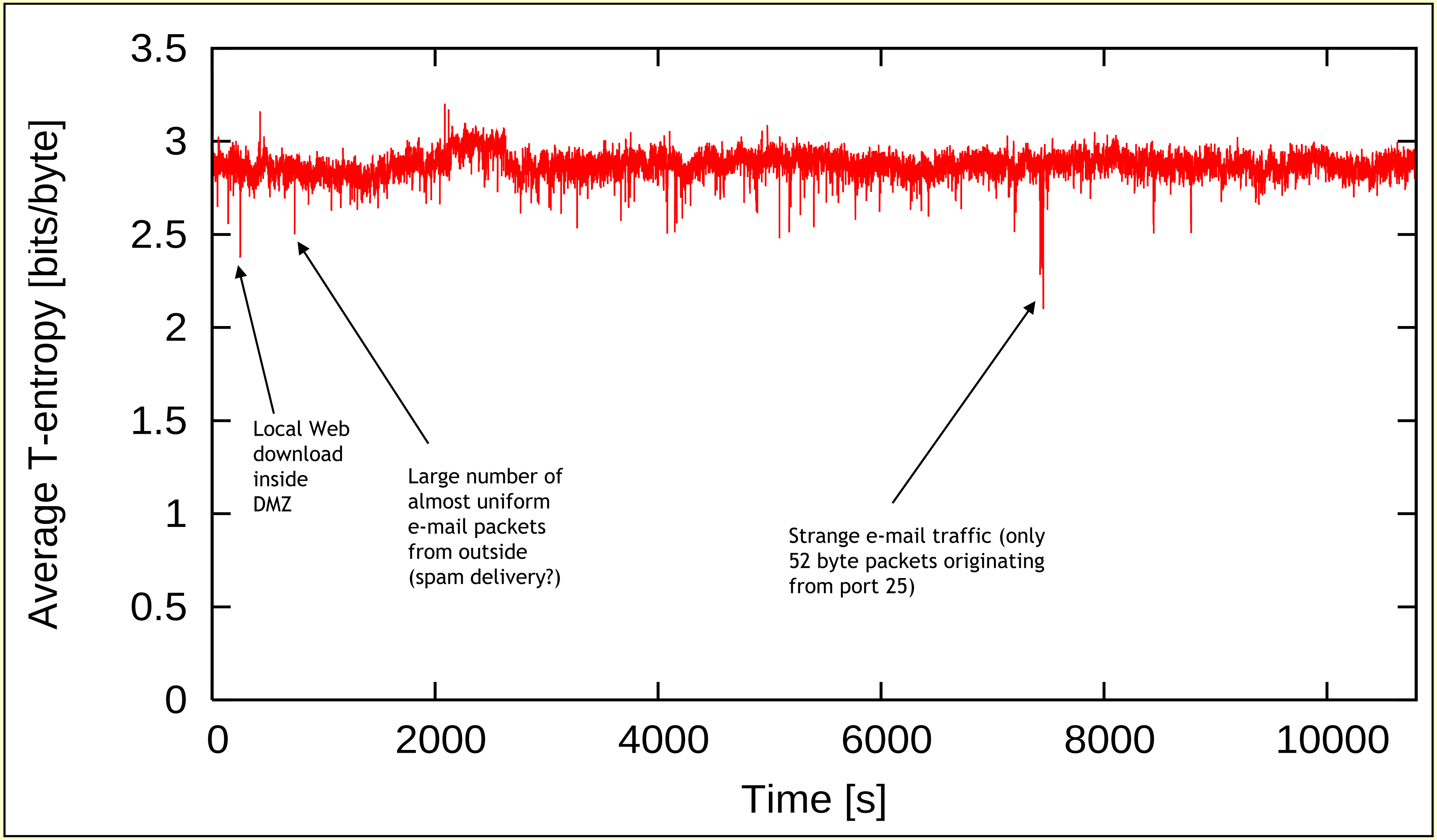
Entropy - filtered



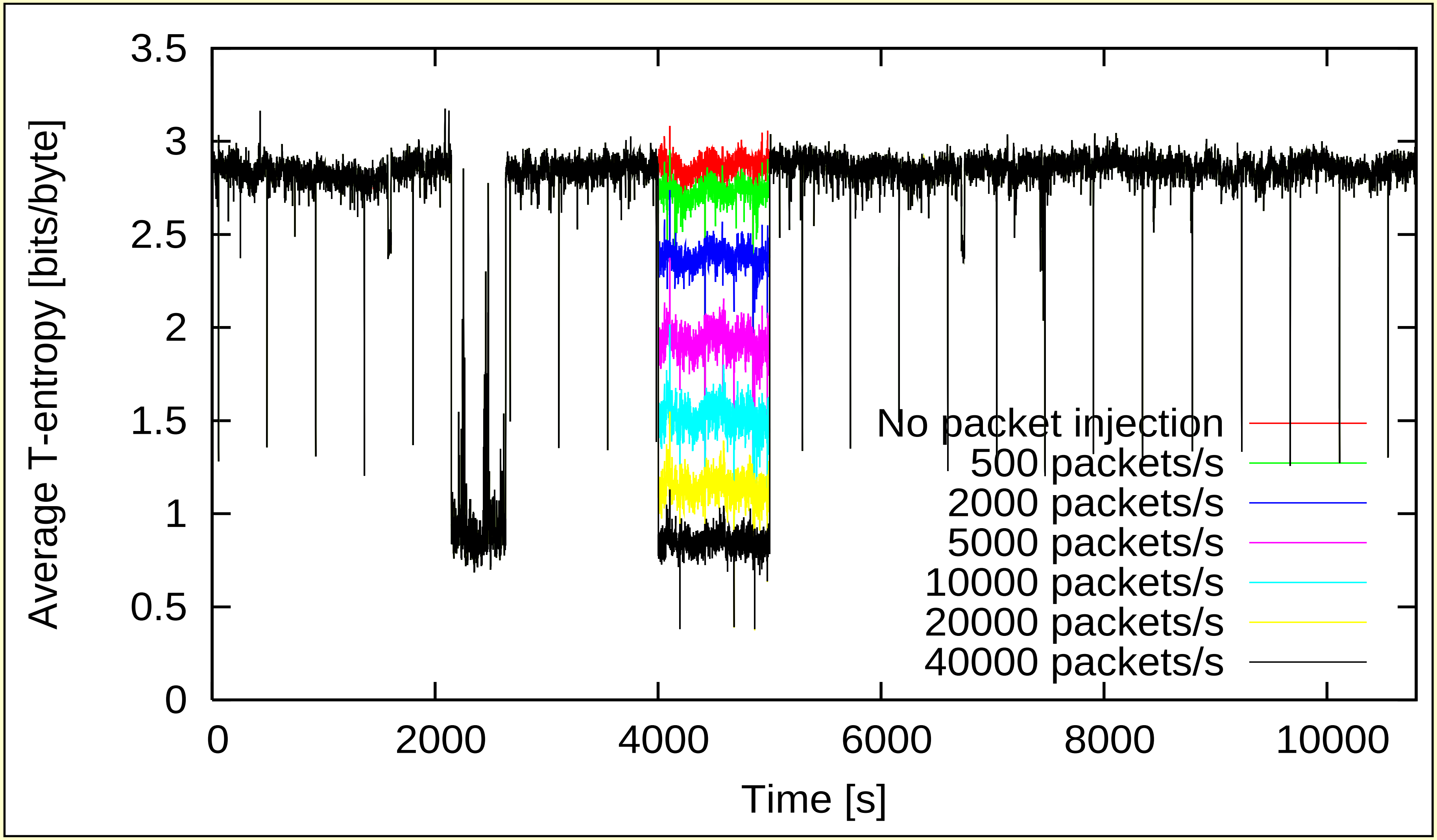
Entropy - filtered



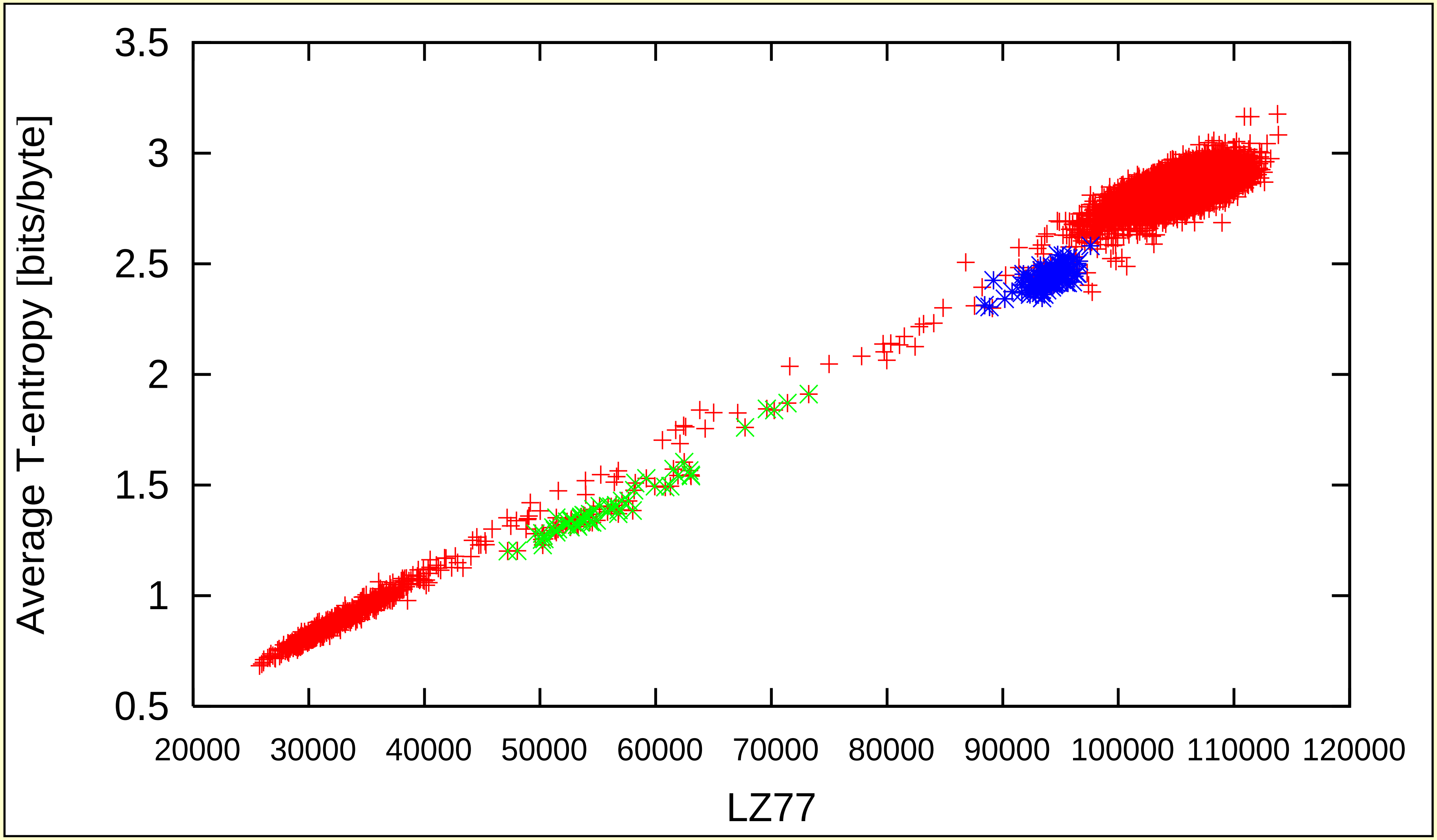
Some of the other stories



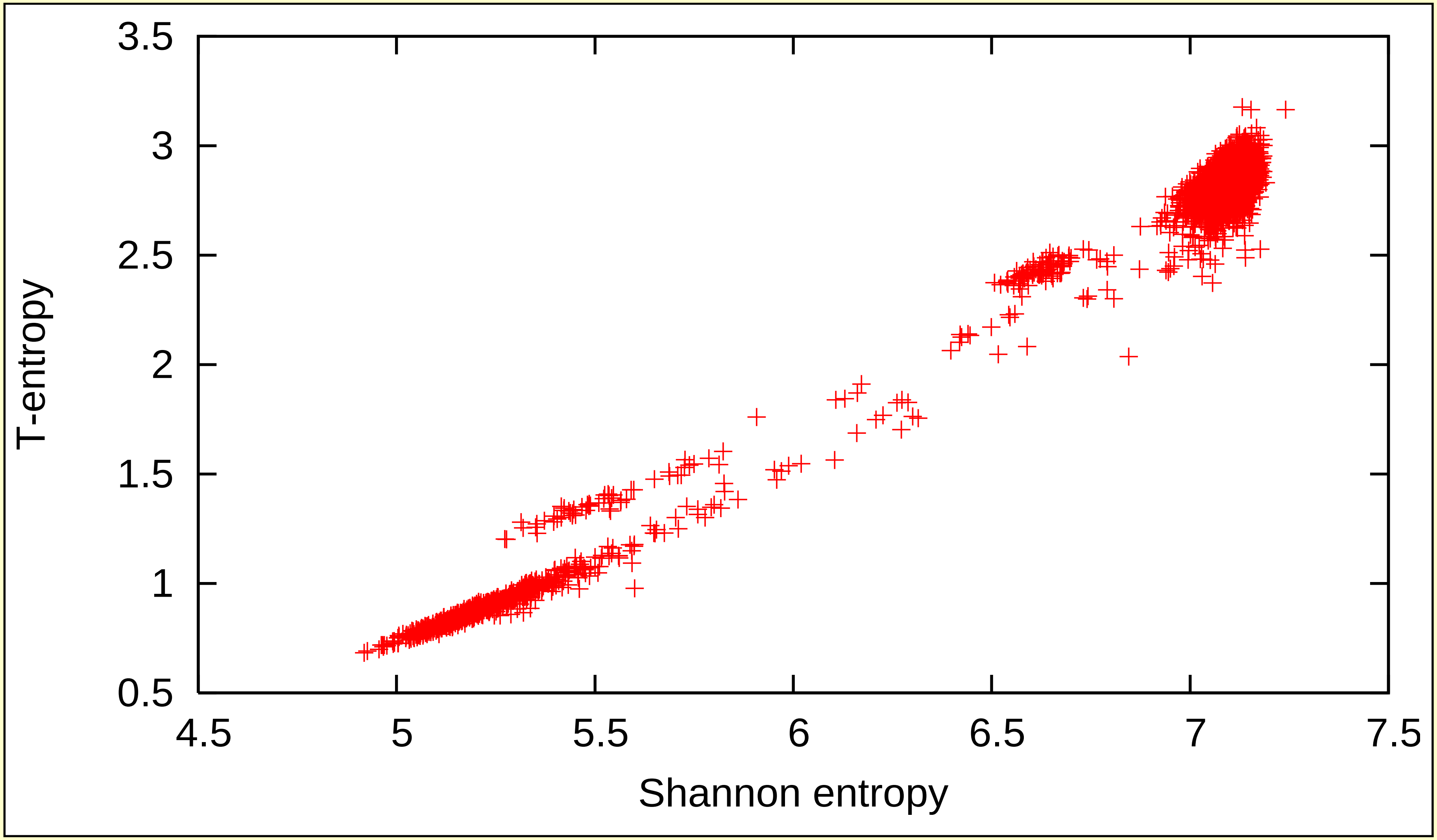
T-entropy sensitivity



Comparison – T-entropy vs. LZ77



Comparison – T-entropy vs. Shannon



Observations

- T-entropy suffers least from overestimation and has a denser range
- LZ production complexity does slightly better than T-entropy but practical algorithms are slow
- Combination of different measures may be useful in event classification

Observations

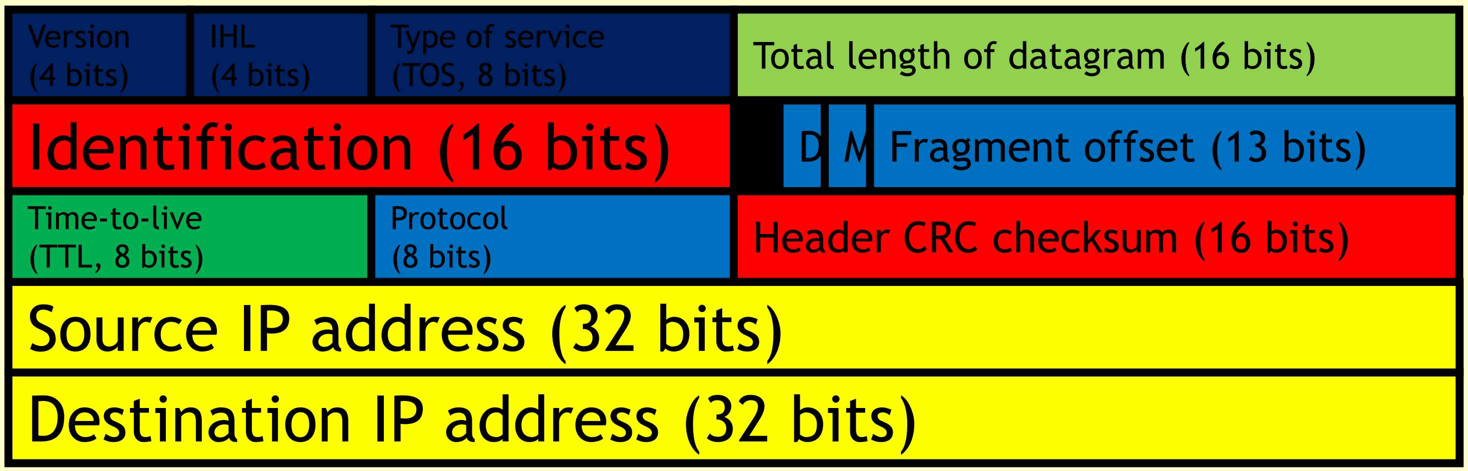
- Monitoring just a subset of data from IP headers can mean high or low “normal” entropy
- May monitor for drops OR rises depending on “normal” entropy
- “Normal” entropy seems to be site-dependent

IP datagrams and entropy

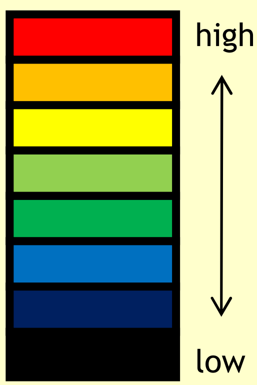
- Entropy in TCP/IP traffic is contributed by several sources:
- IP header, usually 20 bytes
- TCP/UDP header, usually 20 bytes TCP, 4 bytes UDP
- Payload (packet content), first N bytes captured by **`tcpdump`** utility

- May want to use all or just part of the header and packet information

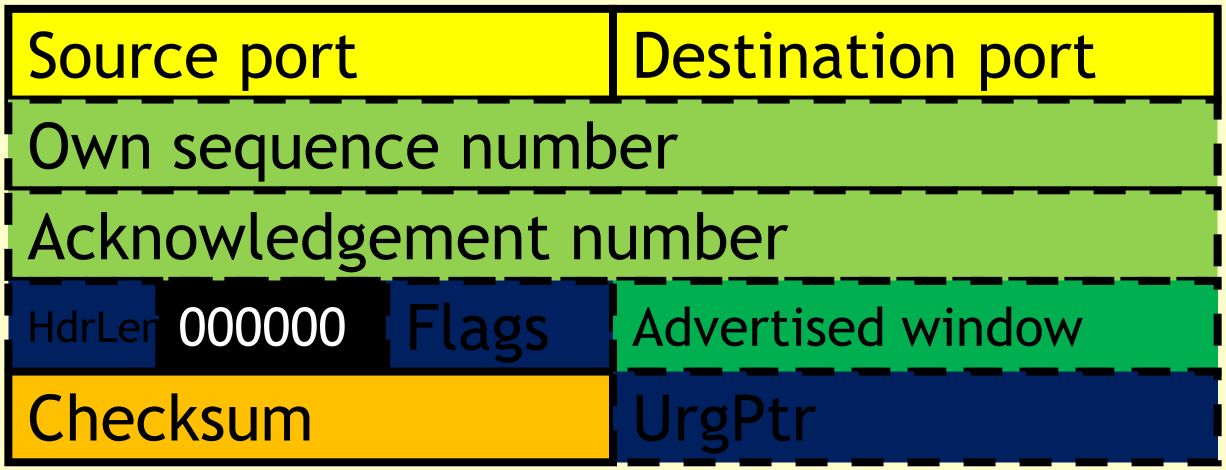
Entropy sources in IP headers



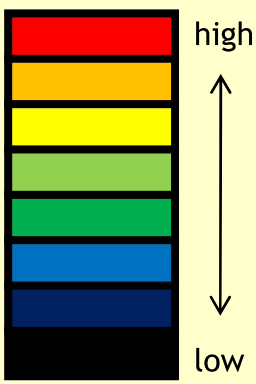
Entropies typically observed at a gateway router in “normal” traffic:



Entropy sources in TCP/UDP headers



Entropies typically observed at a gateway router in “normal” traffic:



- - - -TCP only

UDP has a 16 bit length field before the checksum &
UDP checksum covers header only

Future work

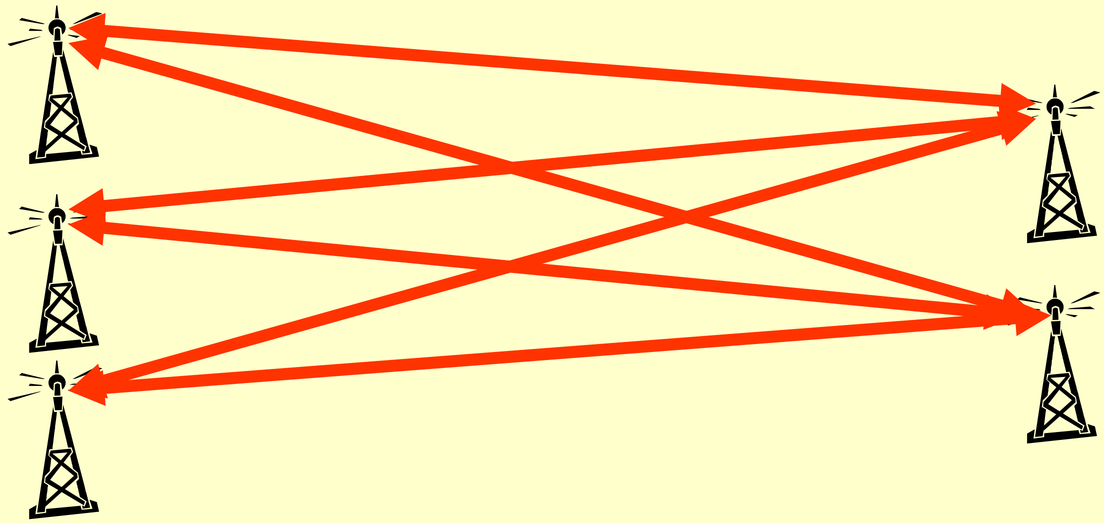
- At some observation sites, sampling all packets is not feasible
- Need to look at flow records and sampled packets rather than full records
- This means throwing information away that may be useful

Other applications

- Technique isn't restricted to networks
- Observation: most time-varying observables of complex systems have a pretty stable entropy as long as the system itself is stable
- Can use entropy changes as indicators for events

Example

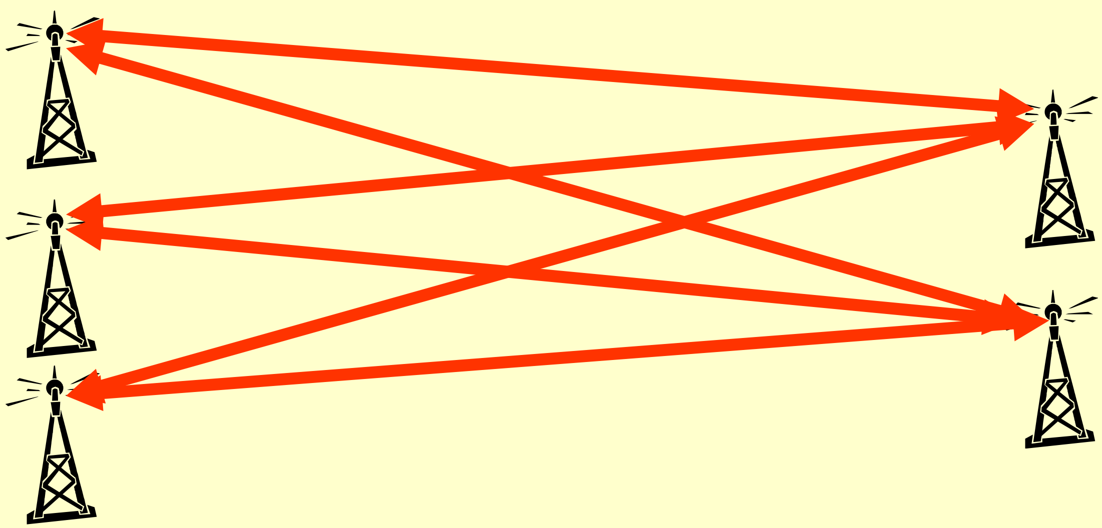
- MIMO communication system



- Multi-user, mobile, etc. - channel conditions fluctuate naturally in complex way

Example

- How do we notice permanent changes that may indicate deterioration in equipment performance?

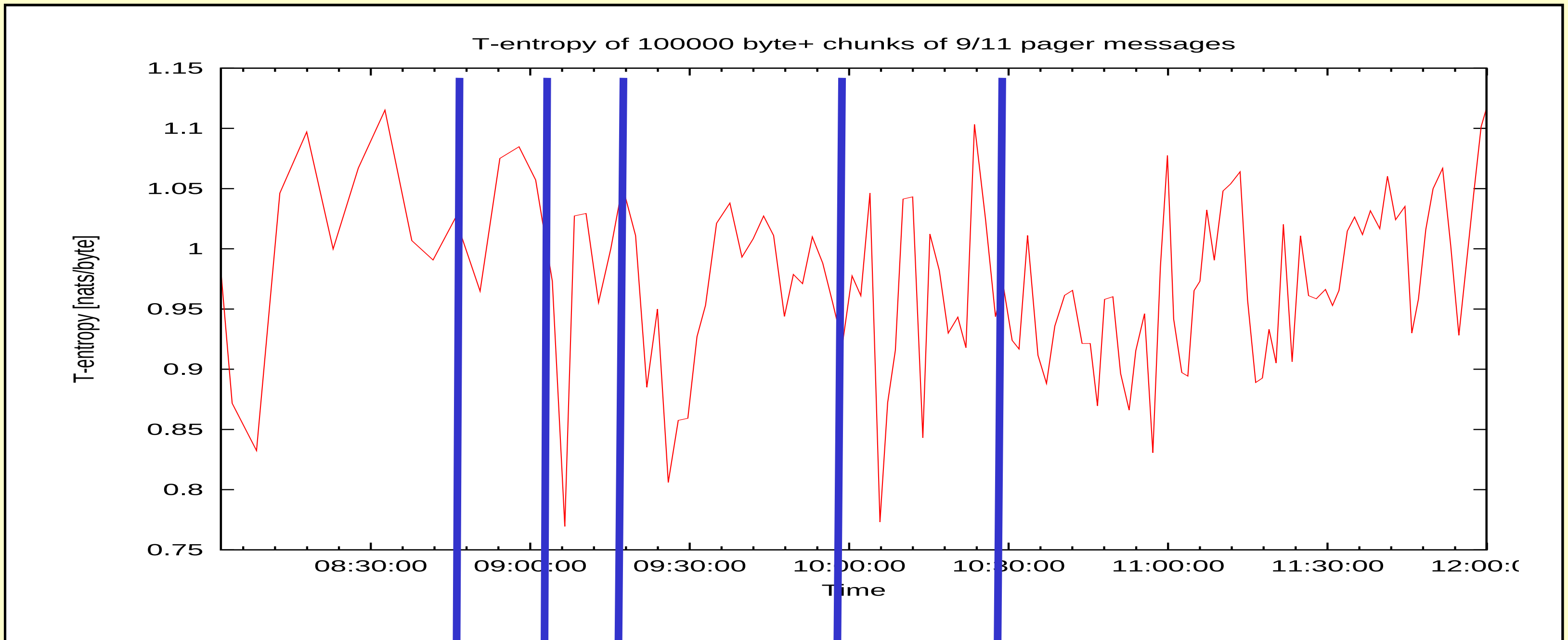


Possible answer

- Monitor entropy of channel quality data from feedback
- If entropy remains near-constant compared to reference sample, it is usually reasonable to assume that all is OK
- If entropy rises or falls significantly - something is afoot!

For the conspiracy theorists...

- Entropy of pager messages of 9/11...



First plane hits

Second plane hits

All NY airports shut

South Tower collapses

North Tower collapses

Message source: wikileaks.org

Conclusions

- Duality between T-codes and strings opens up a number of areas of application – network event monitoring is one of them
- Seems to be pretty useful in network event detection!
- Theoretical results are slowly catching up