Network Telescope Data Analysis: IBR Monitoring

telescope/darknets/darkspace, 22 Mar 11

Nevil Brownlee

Background, Earlier Work

- Moore, Shannon et al, CAIDA/UCSD, 2000..2006
  - telescope gives a whole-world (1/250) view, mathematical model of that
  - used it to track the rise of fast-spreading worms
- Pang, Yegneswaran, Barford, Paxson and Peterson *Characteristics of Internet Background Radiation,* SIGCOMM 2004
  - passive analysis showing activity over time for and different ports & telescopes (darkspaces)
  - active responders to investigate what sources were trying to do, e.g. Code Red, Agobot, Welchia, etc
- Wustrow, Karir, Bailey, Jahanian and Huston *Internet Background Radiation Revisited,* IMC, 2010
  - evolution of IBR since 2004 steady increase in Mb/s each year
  - address pollution (looking at newly-allocated /8 prefixes; traffic to /8 prefixes within 1.0.0.0/8

Telescope Monitoring – what do we *want?*

- A set of web pages we can look at each day that tells us "something interesting is happening"
- Would like to classify the unsolicited traffic sources into groups somehow, so that we could look for
  - changes in levels of each groups
  - new groups appearing
  - old groups disappearing
- This is the same problem as that of managing a network
  - Network Managers want a display that shows them "what's happening in the network now"
  - and the ability to 'drill down' (by clicking on the display) to find more detail

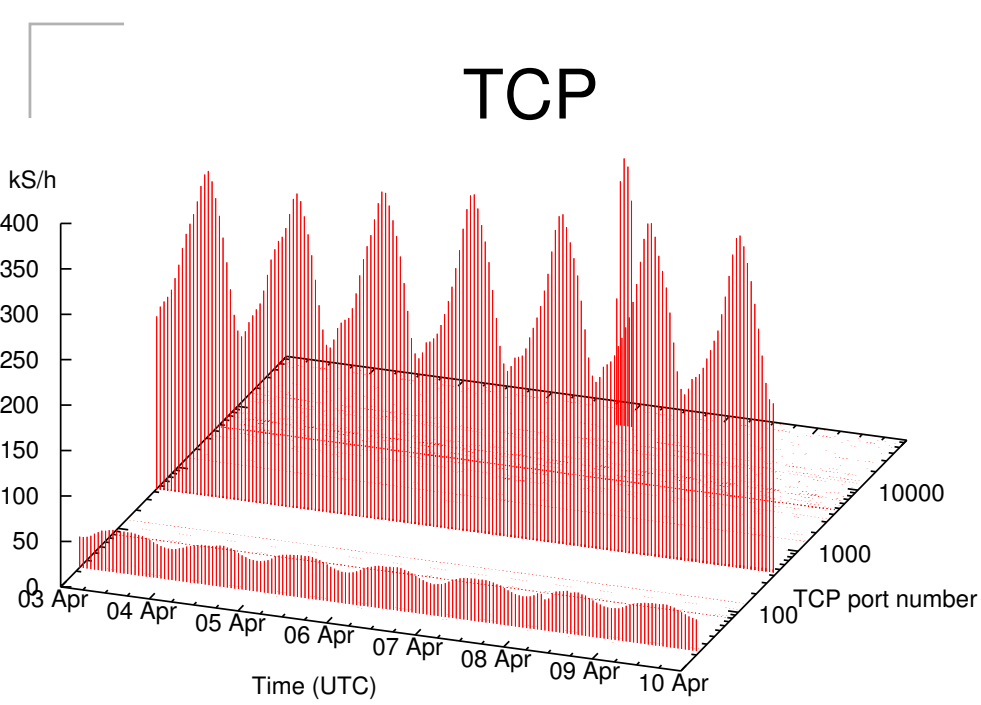
Constraints, Approaches

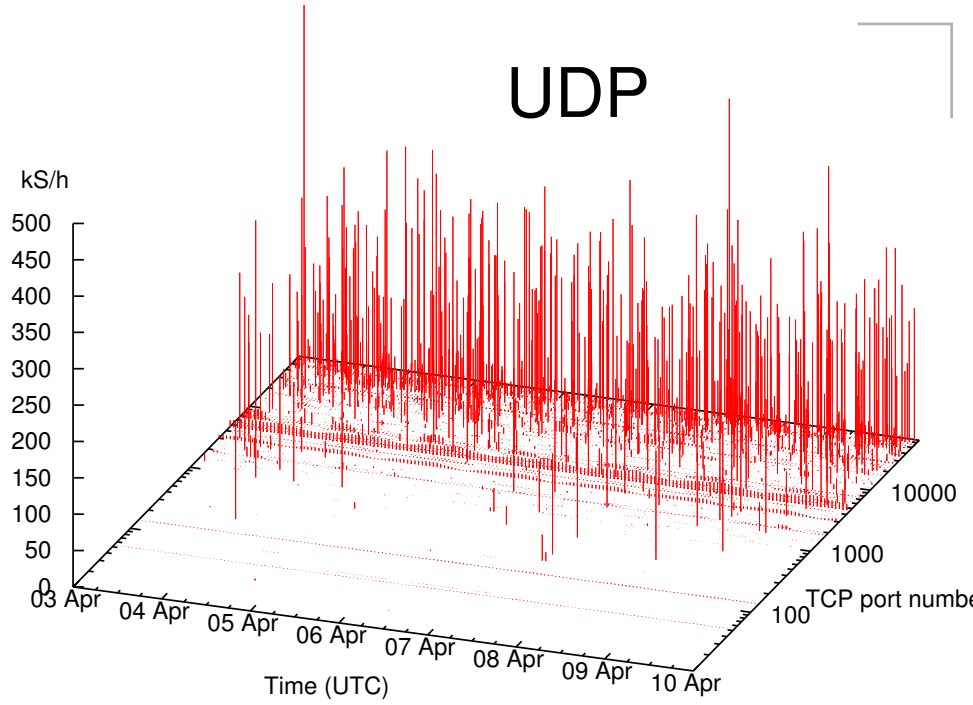
- Problems:
  - data volume: UCSD telescope trace files are big, about $4 \sim 10$ GiB every hour
  - we we only do passive monitoring
  - we need to do the monitoring in near-real time so as to see changes as they appear
  - we'd like to save 'interesting' trace files for later fine-detail analysis
- *Many opinions about what's 'interesting!'*
  - for long-term monitoring (per-hour plots) we need to decide what we want to plot
  - e.g. (simple example) TCP/UDP/ICMP source/packet/byte volumes
- Two approaches
  - (1) Use fixed groups
  - (2) Automated grouping (clustering)

Approach 1: Pre-determined Groups

- Nevil's work Mar 2010 – Feb 2011
- Define taxonomy of 'interesting' source groups
  - TCP: port probe, vertical & horizontal scans, other
  - UDP: port probe, vertical & horizontal scans, other
  - Backscatter: (TCP ACK+SYN & TCP ACK+RST, ICMP TTL exceeded & destination unreachable)
  - Others: Conficker C, ICMP only, . . .
- Analysis methodology
  - build table of sources, count number of TCP/UDP/ICMP/other packets, and ports used by TCP/UDP
  - at end of trace, use those counts to classify sources into above groups. Write summary file for the trace
  - summary has counts & distributions of various packet metrics for each group, e.g. source lifetime, number of packets sent by source, . . .

Approach 1, example plots (a) probe ports

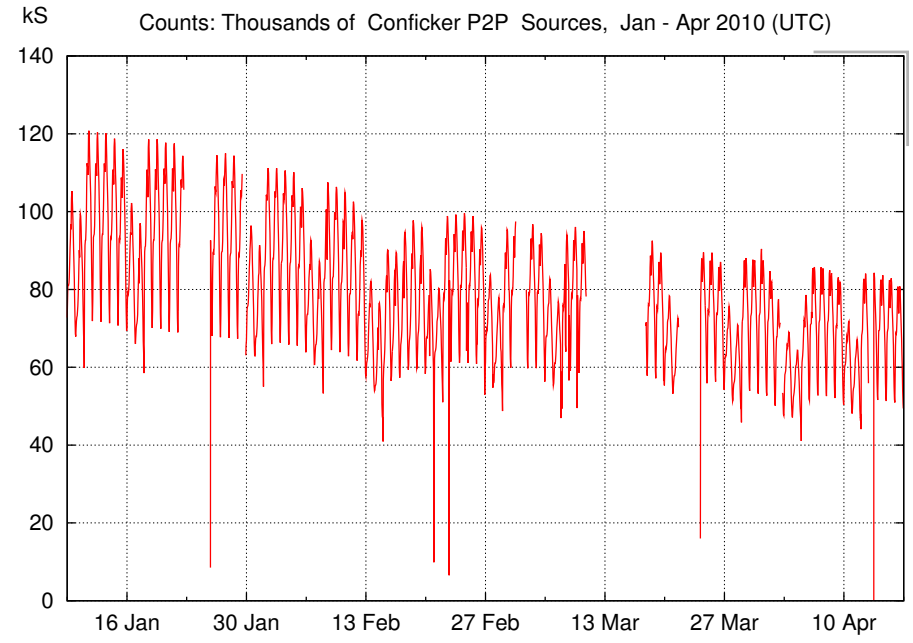


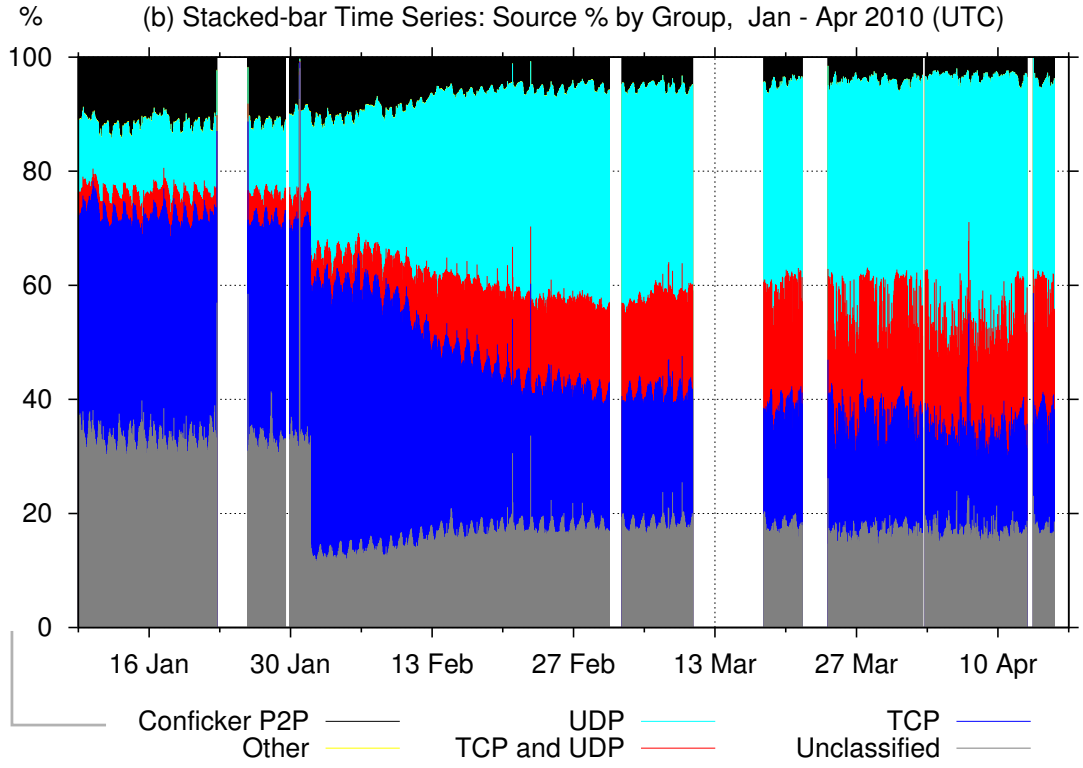


- Number of probe sources sending to top 100 destination ports each hour for the week 03-10 Apr 2010
  - only two popular ports for TCP probe sources
  - UDP probe sources used a wide range of ephemeral high-numbered ports

Approach 1, example plots (b)

Conficker C (p2p) sources seen each hour, showing a steady decrease from early January





Percentage of source groups seen each hour:
– 30 Jan 1.8 MS/h
– UDP sources growth early Feb

Approach 2: Automated grouping of traffic sources

- Classify into groups using a 'volume' metric (bytes/packets/flows)
- Split the groups into smaller groups using a 'classifier' metric
- Example analysis systems
  - aguri: volume = byte/s
    - classifier = source address / prefix length
    - simple system, no GUI (produces lists of prefix hierarchy)
  - nethadict: volume = bytes
    - classifier = n-grams (p, n)
      p = byte position in pkt, n = value of byte(s)
    - Automatically determines n-gram used to split a group, find some bytes common to 50% of group
    - picks arbitrary n-grams

Clustering metrics

- Volume metrics:
  - sources seen / s
  - packets seen / s
  - sources seen / s
  - . . .
- Classifier metrics:
  - source address / length (/ means 'split on')
  - source port / port number $\rightarrow$ p% in group
  - IP protocol (really only see TCP, UDP, ICMP)
  - average packet length (not useful for TCP)
  - packets/bytes (big $\Rightarrow$ DOS attack, small $\Rightarrow$ vulnerability probe)
  - packet inter-arrival distribution *(Nevil's current project)*

Comments on clustering

- When we observe a group, we don't know what application is generating its packets
  - to find that out we need to select out packets for sources in the group, and examine them so as to determine their protocol and (perhaps) generating application
  - that's hard to do automatically!

- Groups found by automatic classifiers are not stable.
  If we use clustering techniques to make groups $0..n$
  - n will vary over time
  - a group with the same characteristics may change group numbers with each sample
  - Such variability makes automatic grouping difficult to use for long-term trend monitoring

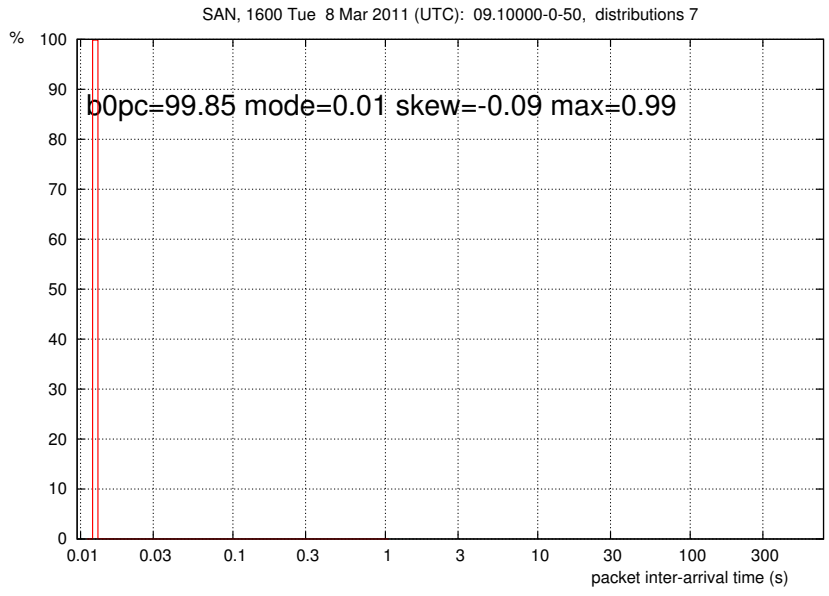
Approach 2: Clustering, using *kmeans*

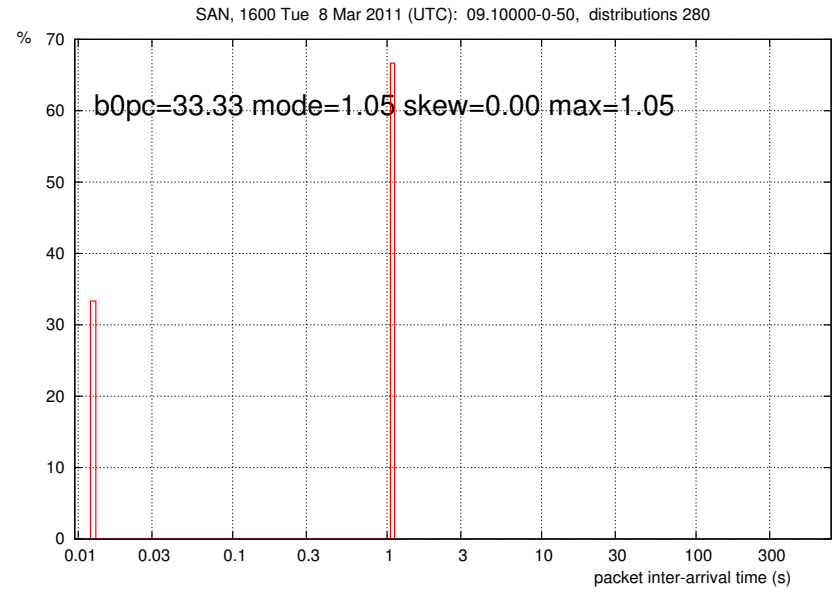
- Nevil's work 7-18 Mar 2011
- Look at packet interarrival time (IAT) distributions for each source. Can we use IAT statistics to identify source applications?
- Collect IAT distributions (180 bins) for every source in an hour. Hour ending 1600 pm 8 March had 1.5 M sources.
- Find metrics we can use to represent an IAT distribution
  - use log-scale bins, 0.012 to 600 s
  - two metrics: *median* and *skewness*
- Tried clustering using using Using Dan Pelleg's *kmeans* program (Dan is at the Auton lab, CMU).
  - k-means clustering finds clusters in n–dimensional space, given that you know n. Dan has extended this idea so that the system determines how many clusters it can reliably find.
- This idea simply did not work well for IBR IATs

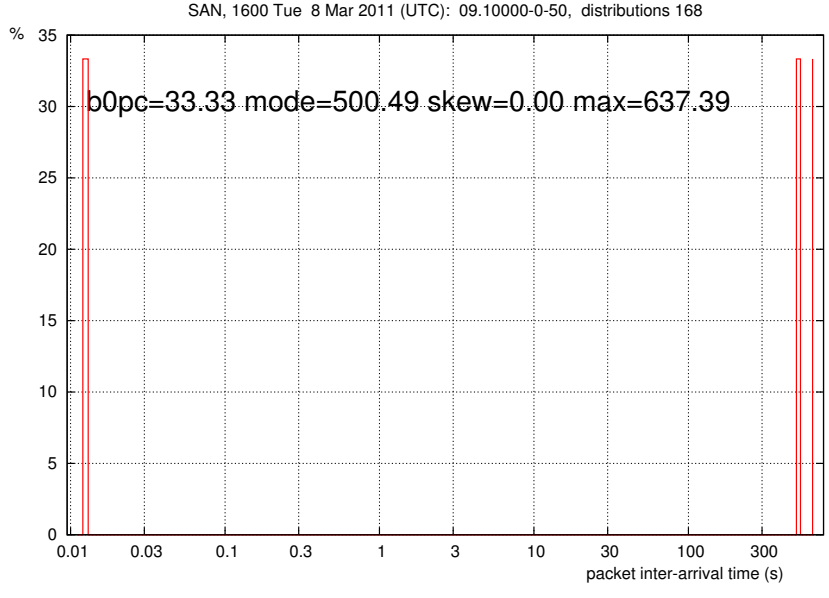
IATs using pre-determined groups

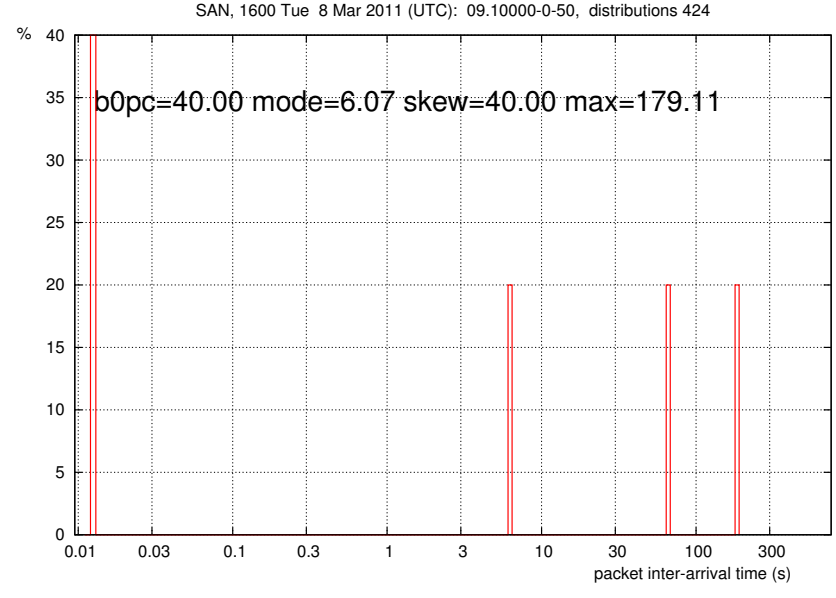
- Nevil's work from 19 Mar 2011 (!)
- Simpler pragmatic approach:
  - make 'postage-stamp' sheets showing individual IAT distributions
  - find metrics for the distributions, print them on the sheets
  - look for recurring patterns, i.e. source groups; find metric ranges that could be used to determine each distribution's group
  - print new postage-stamp sheets, one for each group
  - iterate as more groups become apparent
- IAT metrics
  - bin-zero %: $> 95\% \rightarrow$ DOS source
  - mode IAT: $2.5..3.5s \rightarrow$ Windows XP's TCP retry
  - skewness: $\rightarrow$ left, right or evenly balanced
  - maximum IAT: high values $\rightarrow$ 'stealth' probe sources

Group 0: DOS sources

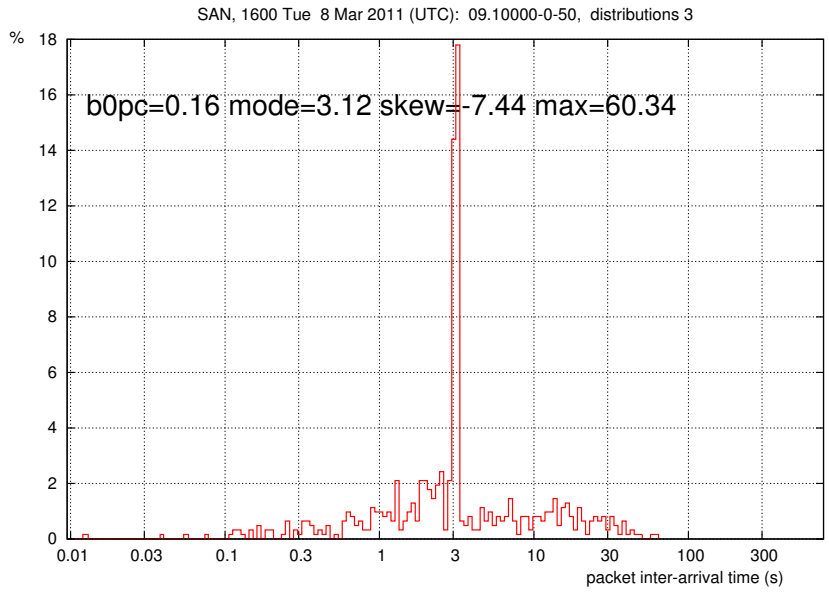


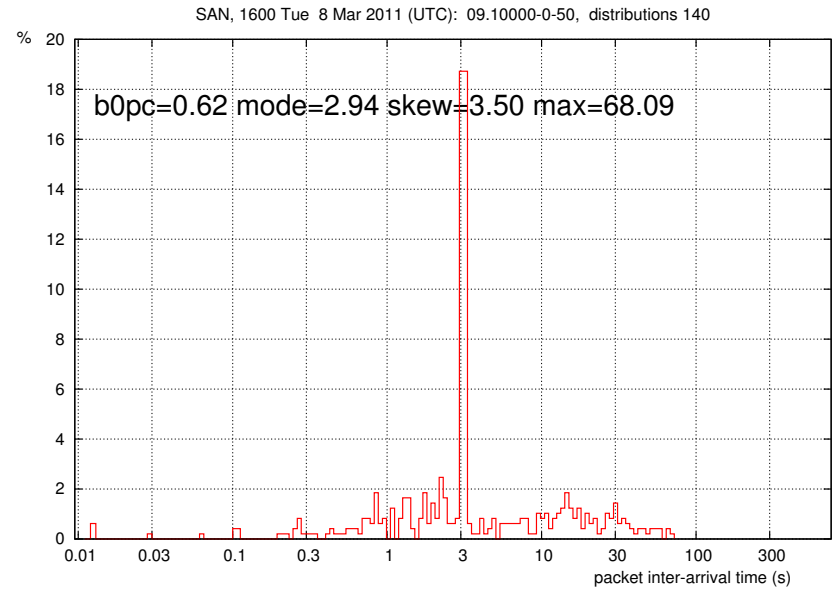


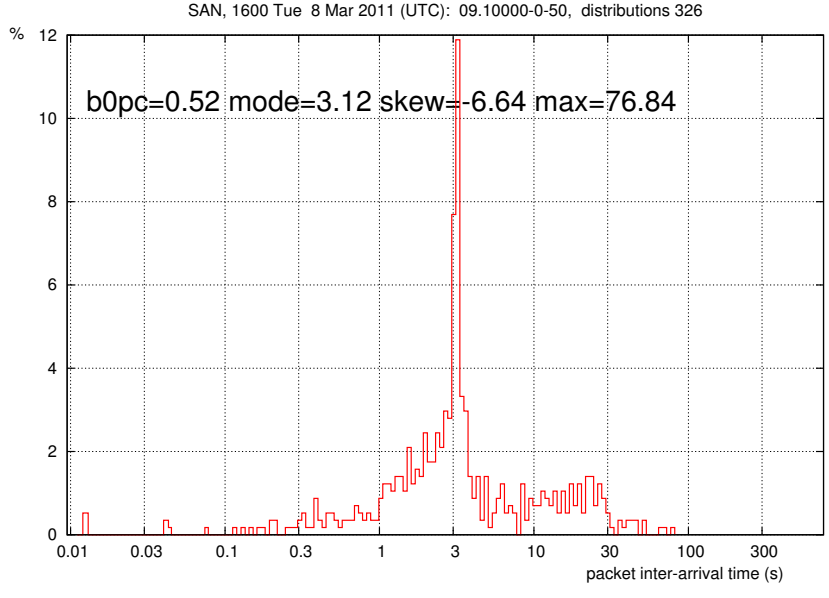


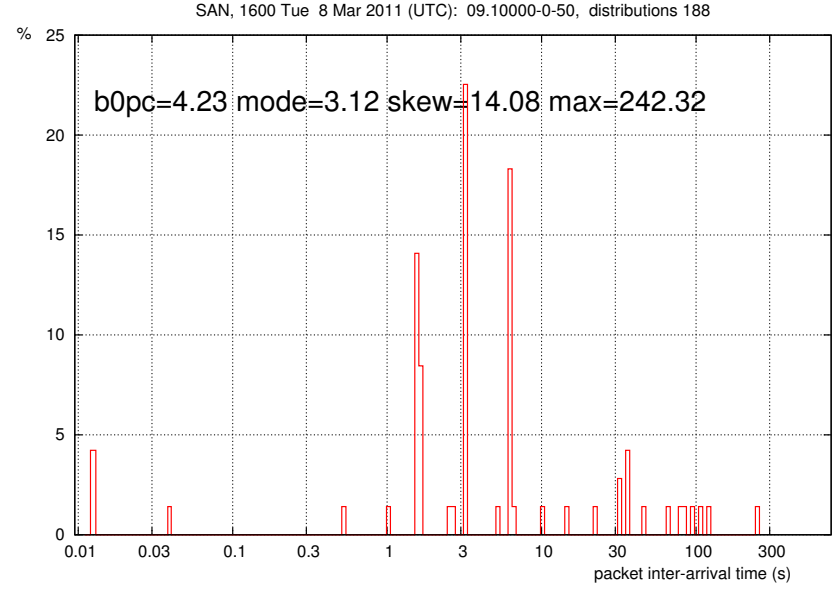


Group 1: XP_even sources

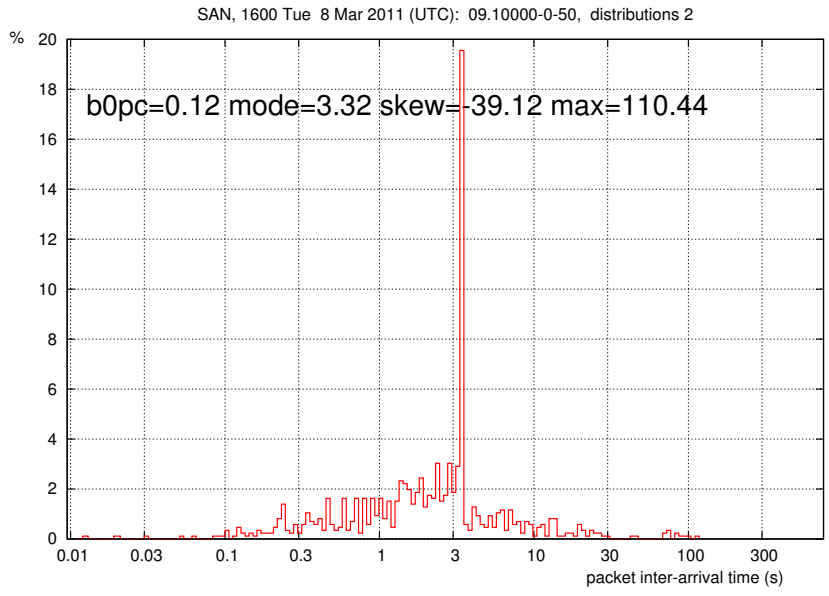


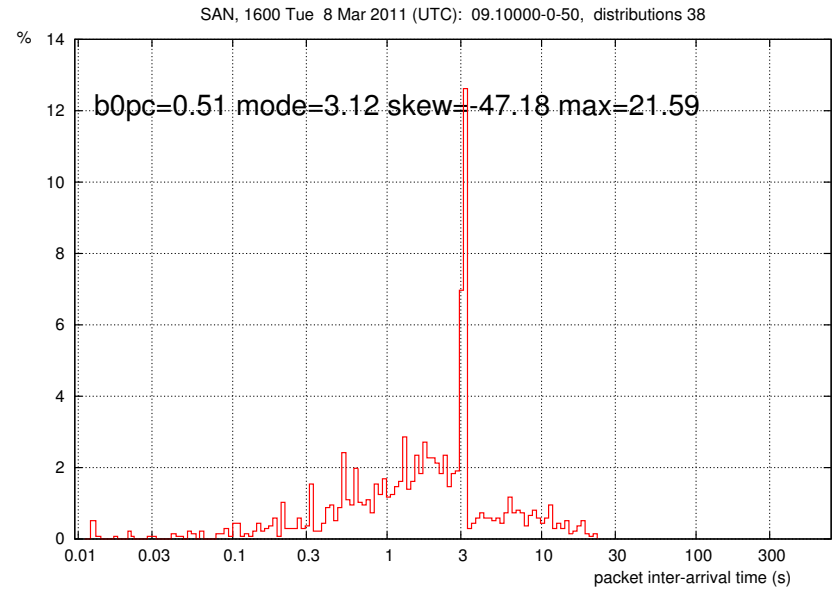


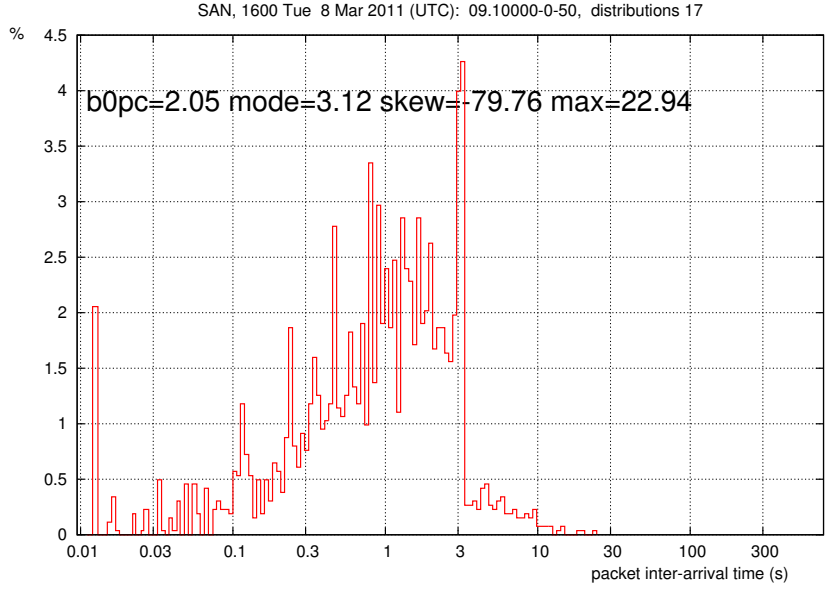


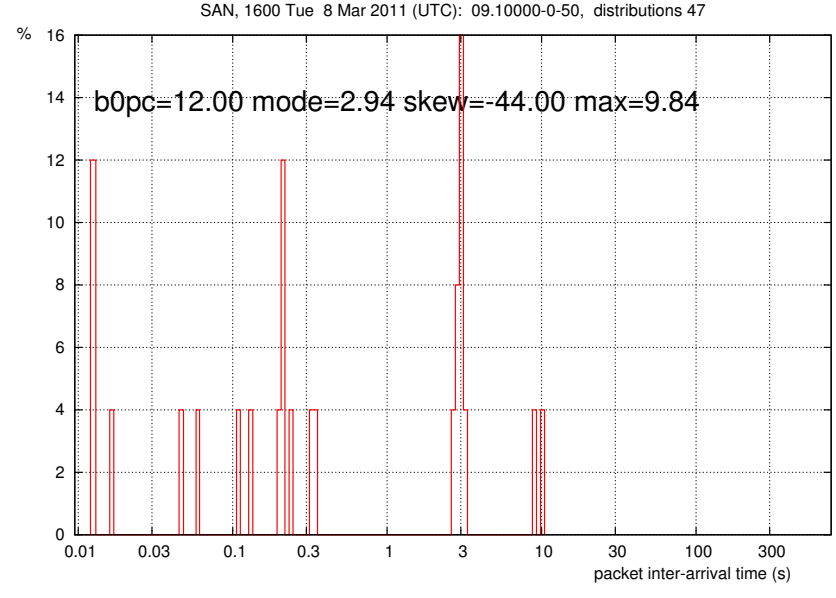


Group 2 XP_left sources

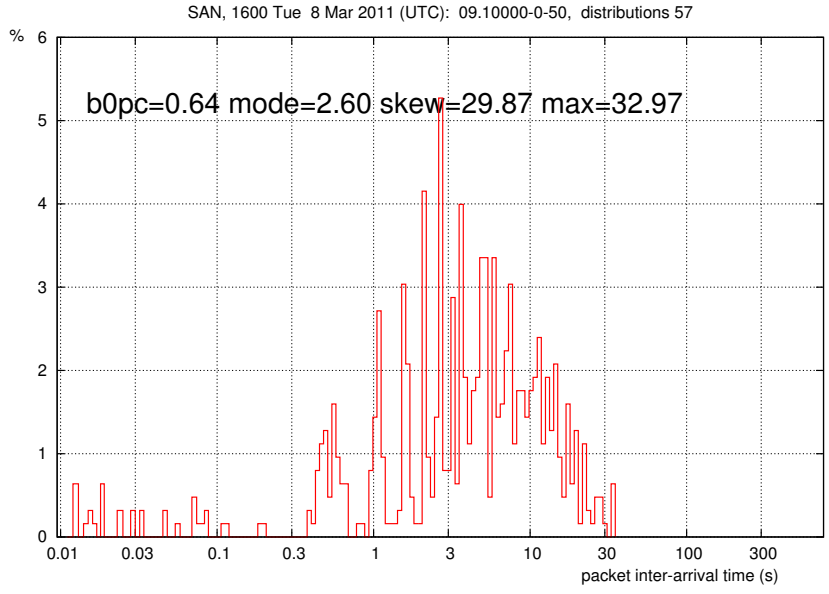


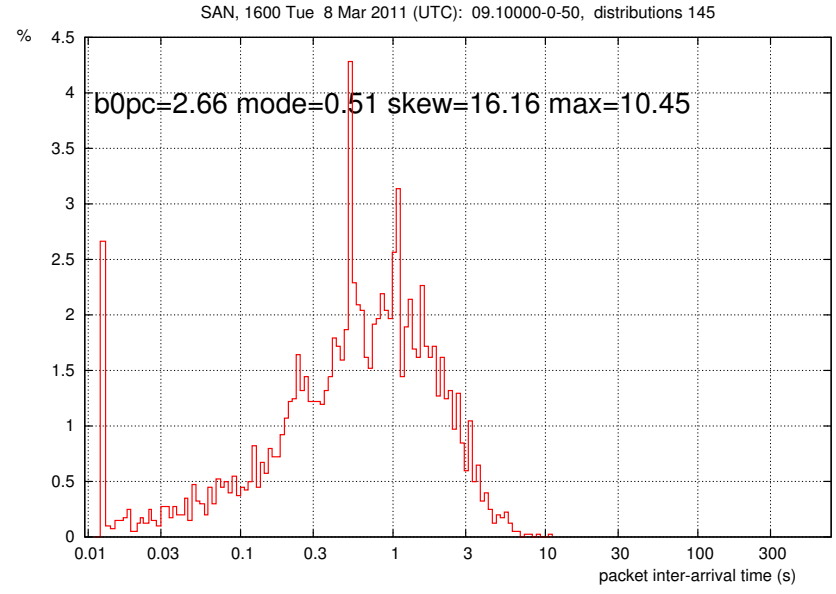


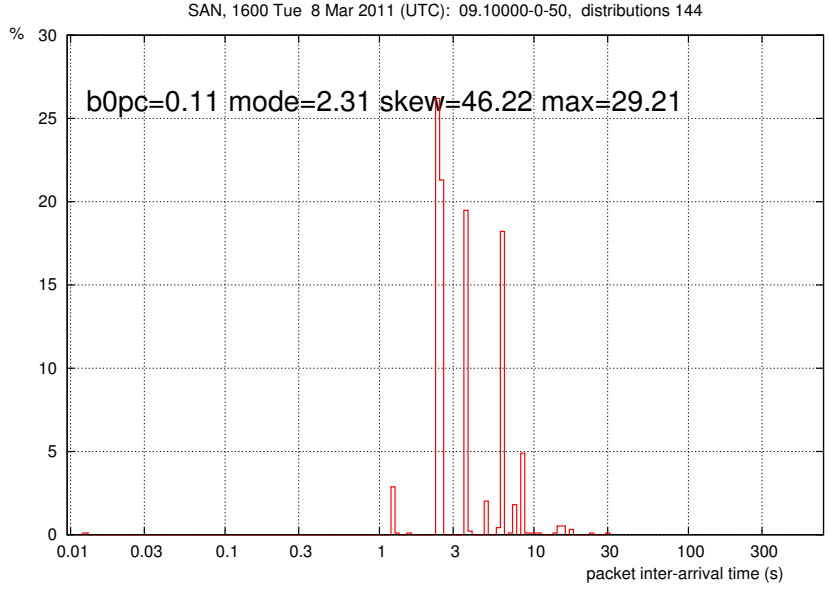


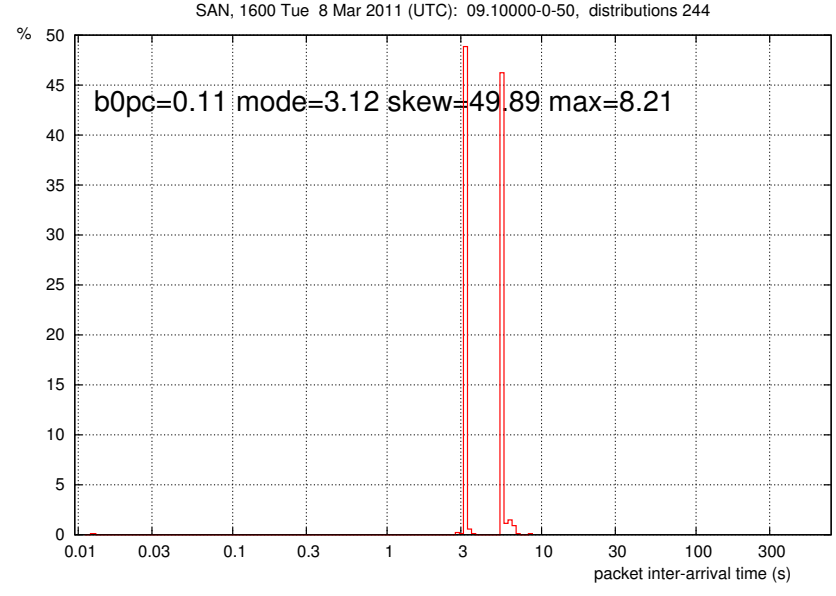


Group 3: Other sources









Conclusion, Future Work

- Work on *understanding* the IAT distributions
- Look at their counts
  - how many are there in each group over an hour?
  - how does time of first packet within the hour influence the distribution?
- How many IAT groups do we think are common?
- How do the IAT groups relate to the “Nevil’s taxonomy” groups?