# Jäljed võrgus - I: Netflow ??

Toomas Lepik

Natuke minust

Auväärne Draakon
Ex CERT-EE
TTÜ nooremteadur/infoturbe ekspert

Esitlusest

Enamus  ingliskeelsed slaidid !

- Natuke meelde tuletust
- Mis on Flow
- Kust ja kuidas leida netflowd
- Mis sealt välja lugeda.

… võtan järjestust natuke vabalt ..

Application

Presentation

Session

Transport

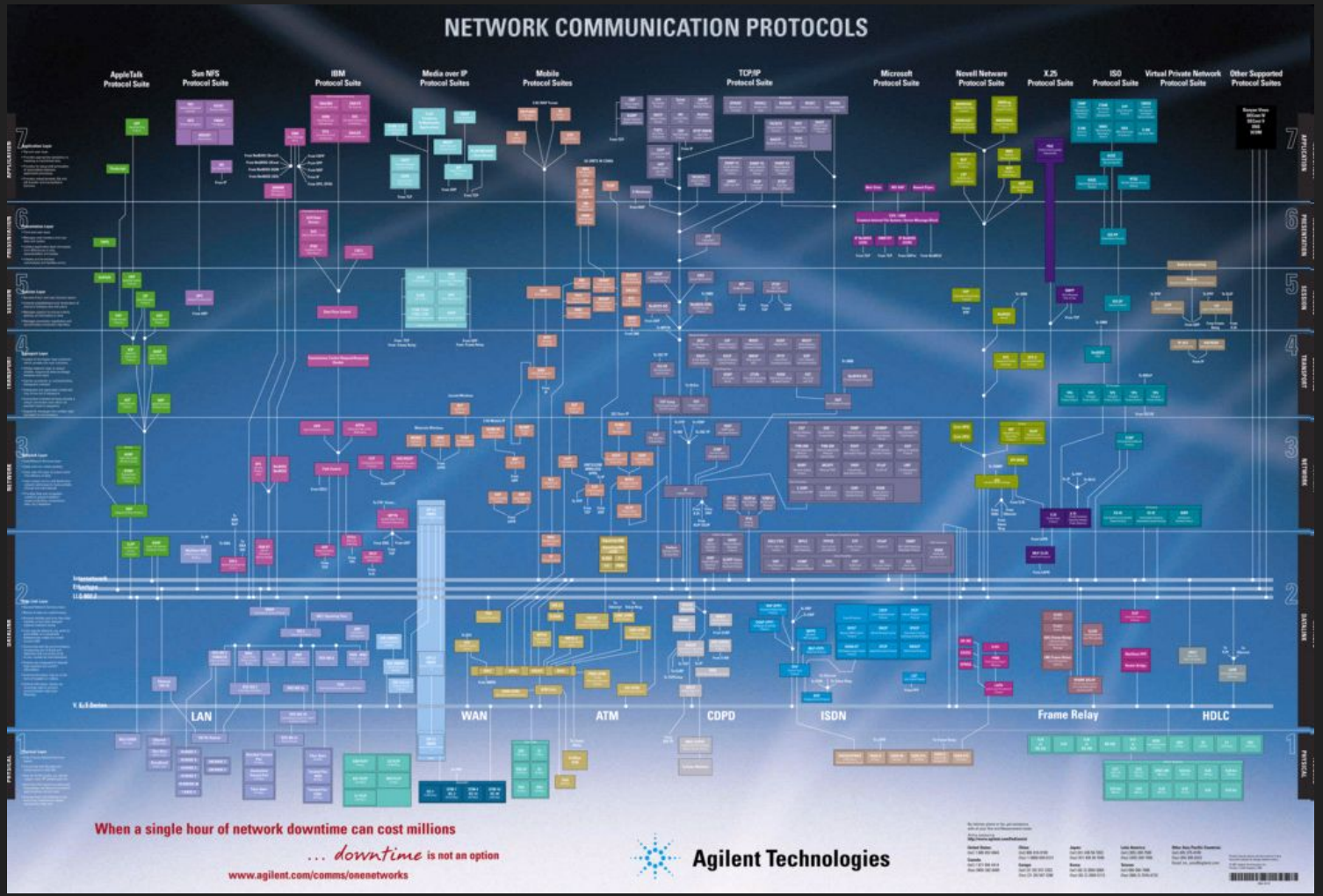
Network

Data Link

Physical

| | | | |
|---|---|---|---|
| | | | TSL + HTTP++ /HTTP/FTP/ |
| | | TCP | TSL + HTTP++ /HTTP/FTP/ |
| | IP | TCP/ UDP | TSL + HTTP++ /HTTP/FTP/ |
| ETH | IP | TCP/ UDP | TSL + HTTP++ /HTTP/FTP/ |

Protokollid

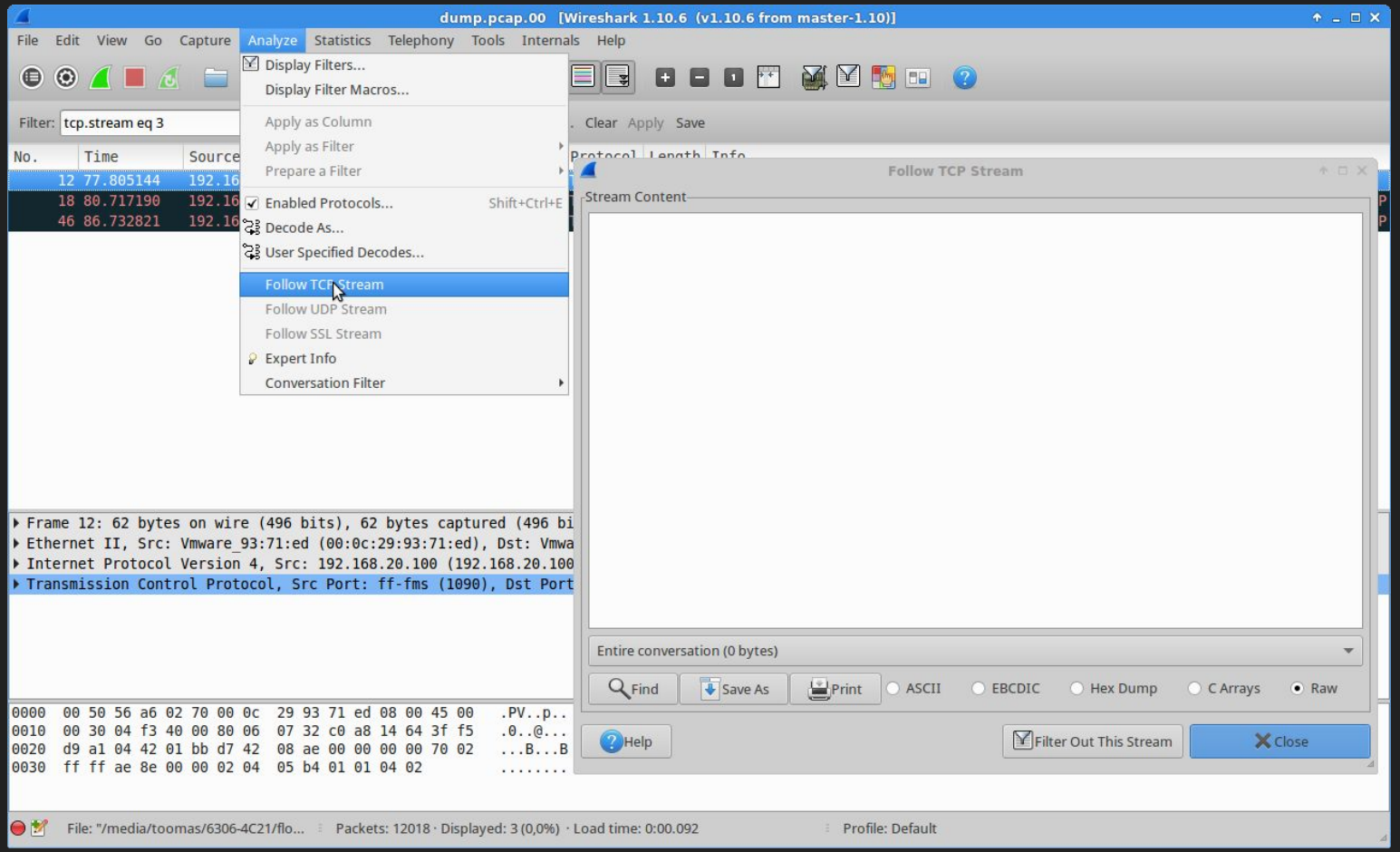


## FLow ? Voog

1 Järjestatud pakettide voog

2 Netflow/IPFIX

# Flow - 1


dump.pcap.00 [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Display Filters...
Display Filter Macros...
Apply as Column
Apply as Filter
Prepare a Filter
Enabled Protocols... Shift+Ctrl+E
Decode As...
User Specified Decodes...
Follow TCP Stream
Follow UDP Stream
Follow SSL Stream
Expert Info
Conversation Filter
Filter: tcp.stream eq 3
Clear Apply Save
No. Time Source
12 77.805144 192.16
18 80.717190 192.16
46 86.732821 192.16
Follow TCP Stream
Stream Content
Entire conversation (0 bytes)
Find Save As Print ASCII EBCDIC Hex Dump C Arrays Raw
Help Filter Out This Stream Close
Frame 12: 62 bytes on wire (496 bits), 62 bytes captured (496 bi
Ethernet II, Src: Vmware_93:71:ed (00:0c:29:93:71:ed), Dst: Vmwa
Internet Protocol Version 4, Src: 192.168.20.100 (192.168.20.100
Transmission Control Protocol, Src Port: ff-fms (1090), Dst Port
0000 00 50 56 a6 02 70 00 0c 29 93 71 ed 08 00 45 00 .PV..p..
0010 00 30 04 f3 40 00 80 06 07 32 c0 a8 14 64 3f f5 .0..@...
0020 d9 a1 04 42 01 bb d7 42 08 ae 00 00 00 00 70 02 ...B...B
0030 ff ff ae 8e 00 00 02 04 05 b4 01 01 04 02 ........
File: "/media/toomas/6306-4C21/flo... Packets: 12018 · Displayed: 3 (0,0%) · Load time: 0:00.092 Profile: Default

# Flow analüüsi tehnikad

Lists Conversations and Flows

Flow Eksport

File and Data Carving

# Flow - 2

In RFC 70011, a Flow is defined as a set of packets or frames passing an Observation Point in the network during a certain time interval. All packets belonging to a particular Flow have a set of common  properties

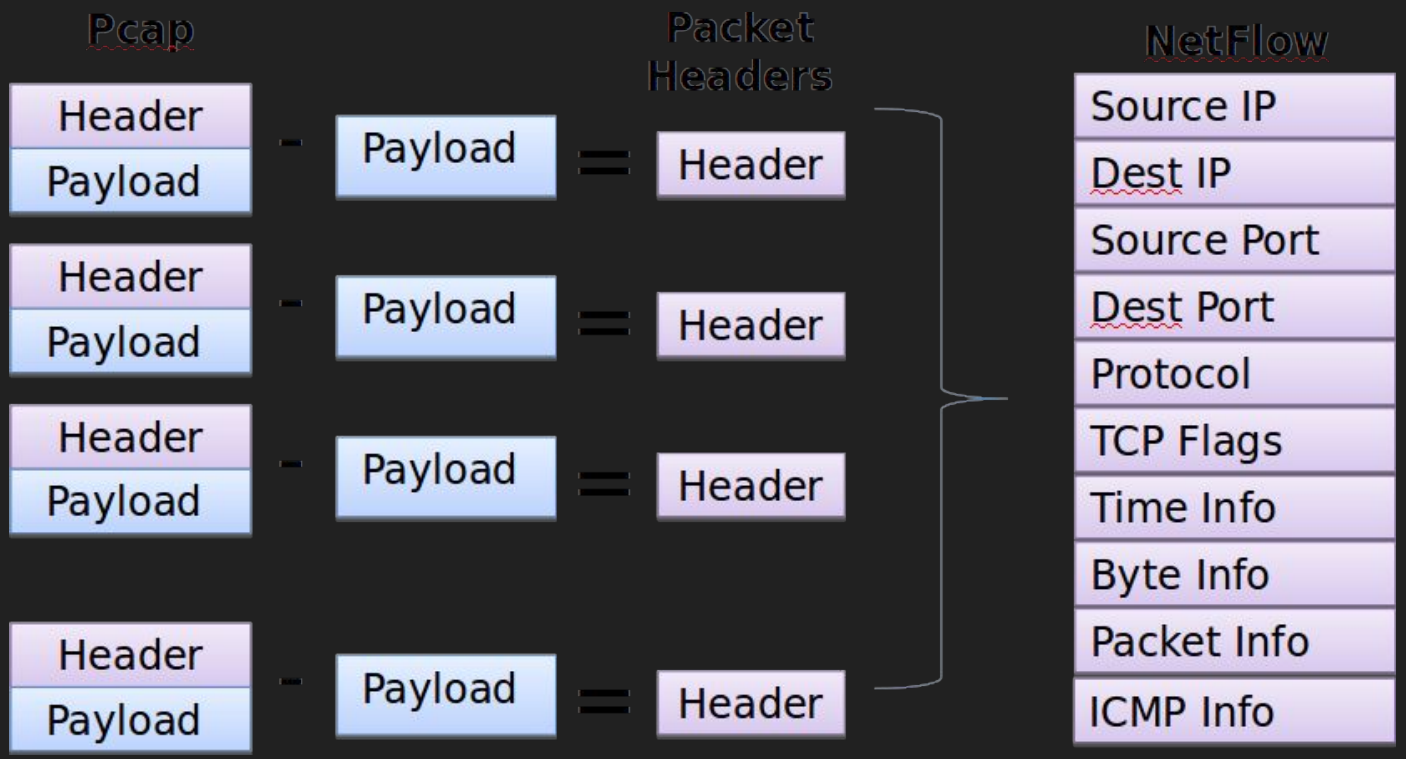
. A flow could consist of all packets in a specific transport connection or a media stream. However, a flow is not necessarily 1:1 mapped to a transport connection.

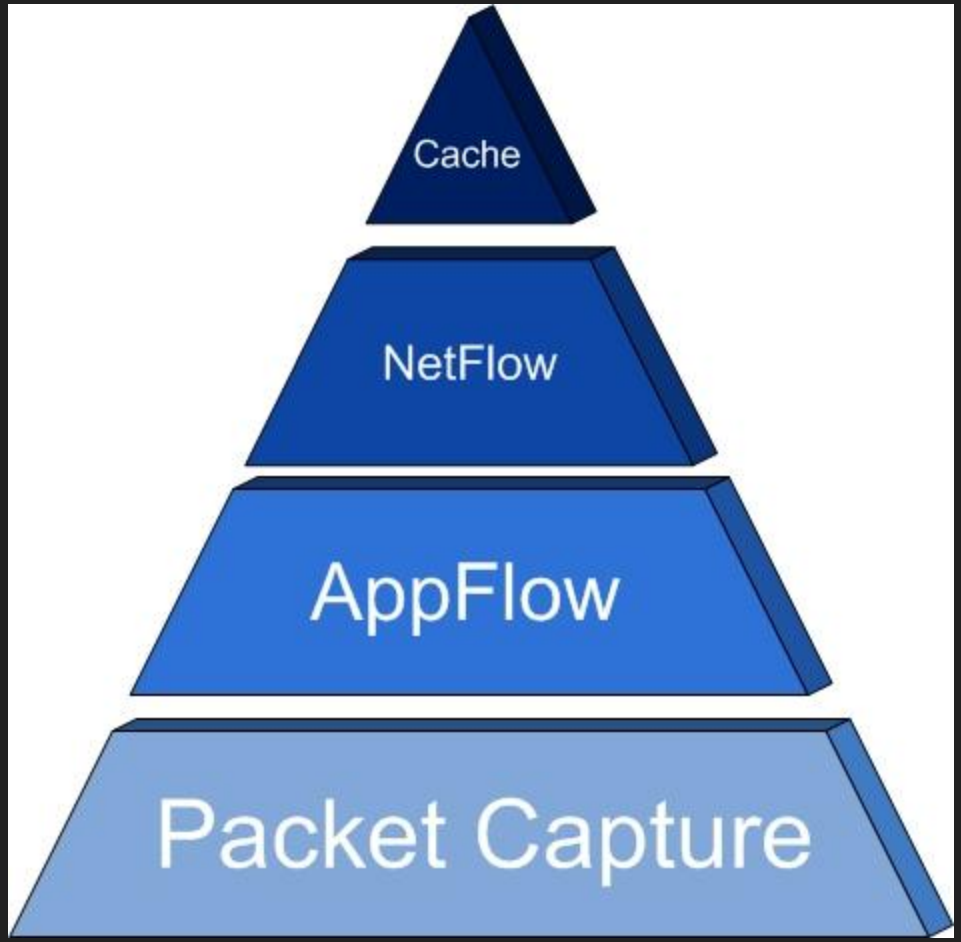
eg . netflow / IPFIX

## Netflow

- Developed - 1996 by CISCO ™
- Became a standard for collecting IP traffic information
- Widely used in monitoring and traffic analysis systems

# Flow simplified



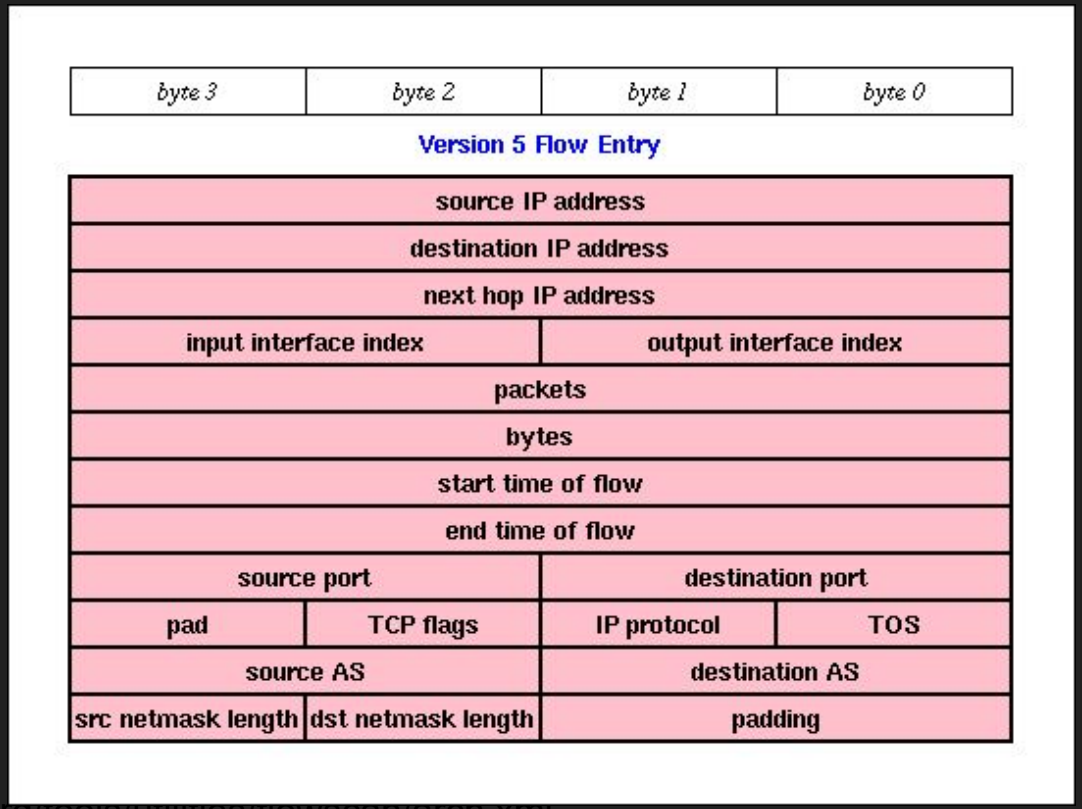

Cache
NetFlow
AppFlow
Packet Capture

# Netflow Versions

| Version | |
|---|---|
| v1 | Initial version - rarely used today |
| v2-v4 | Internal, never released |
| v5 | Most common and very popular version.<br>Fixed format and restricted to IPv4 |
| v7 | Catalyst switches: v5 + source router field |
| v8 | v5 + aggregation. |
| v9 | Template based format. Includes lots of new elements such as IPv6, MPLS labels, BGP next hop etc. |
| IPFIX (v10) | IETF standardised. Similar netflow v9. |

We really only care about v5, v9 and v10 (IPFIX)

# Netflow v5



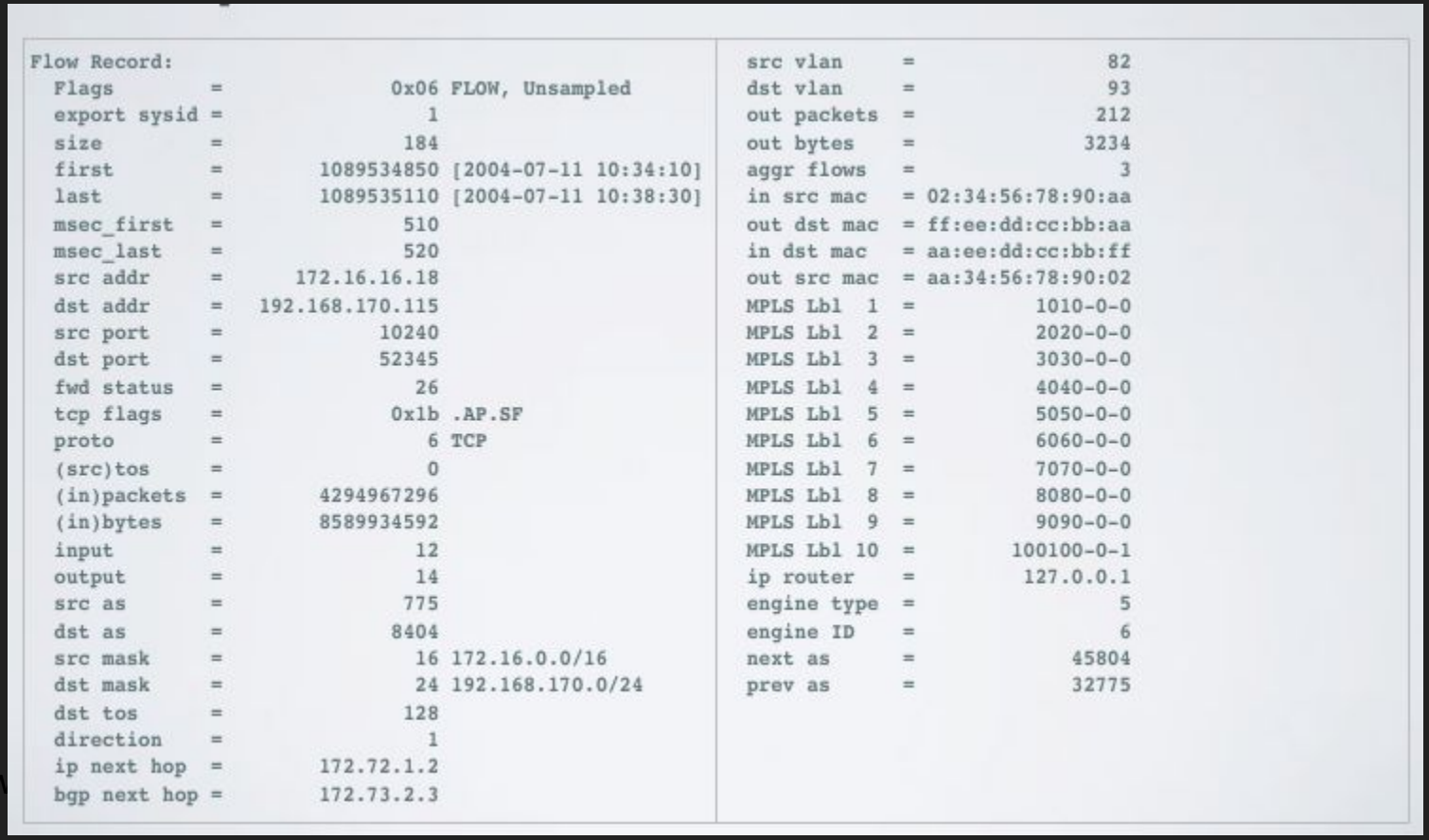
| byte 3 | byte 2 | byte 1 | byte 0 |
|---|---|---|---|

**Version 5 Flow Entry**

| | | | |
|---|---|---|---|
| source IP address | | | |
| destination IP address | | | |
| next hop IP address | | | |
| input interface index | | output interface index | |
| packets | | | |
| bytes | | | |
| start time of flow | | | |
| end time of flow | | | |
| source port | | destination port | |
| pad | TCP flags | IP protocol | TOS |
| source AS | | destination AS | |
| src netmask length | dst netmask length | padding | |

http://www.caida.org/tools/utilities/flowscan/arch.xml

# Netflow v9



```
Flow Record:
  Flags         =               0x06 FLOW, Unsampled
  export sysid  =                  1
  size          =                184
  first         =         1089534850 [2004-07-11 10:34:10]
  last          =         1089535110 [2004-07-11 10:38:30]
  msec_first    =                510
  msec_last     =                520
  src addr      =       172.16.16.18
  dst addr      =    192.168.170.115
  src port      =              10240
  dst port      =              52345
  fwd status    =                 26
  tcp flags     =               0x1b .AP.SF
  proto         =                  6 TCP
  (src)tos      =                  0
  (in)packets   =         4294967296
  (in)bytes     =         8589934592
  input         =                 12
  output        =                 14
  src as        =                775
  dst as        =               8404
  src mask      =                 16 172.16.0.0/16
  dst mask      =                 24 192.168.170.0/24
  dst tos       =                128
  direction     =                  1
  ip next hop   =         172.72.1.2
  bgp next hop  =         172.73.2.3
  src vlan      =                 82
  dst vlan      =                 93
  out packets   =                212
  out bytes     =               3234
  aggr flows    =                  3
  in src mac    = 02:34:56:78:90:aa
  out dst mac   = ff:ee:dd:cc:bb:aa
  in dst mac    = aa:ee:dd:cc:bb:ff
  out src mac   = aa:34:56:78:90:02
  MPLS Lbl  1   =           1010-0-0
  MPLS Lbl  2   =           2020-0-0
  MPLS Lbl  3   =           3030-0-0
  MPLS Lbl  4   =           4040-0-0
  MPLS Lbl  5   =           5050-0-0
  MPLS Lbl  6   =           6060-0-0
  MPLS Lbl  7   =           7070-0-0
  MPLS Lbl  8   =           8080-0-0
  MPLS Lbl  9   =           9090-0-0
  MPLS Lbl 10   =         100100-0-1
  ip router     =          127.0.0.1
  engine type   =                  5
  engine ID     =                  6
  next as       =              45804
  prev as       =              32775
```

https://www

# J-Flow / cflow

- J-Flow and cflowd is essentially NetFlow
- NetFlow collectors will support J-Flow/cflow output
- J-Flow and cflow are terms used by Juniper(v5, v8,v9)
- cflow term used by Alcatel (v5, v8, v9)
- Implementations do not support flexible templates

# NetStream

- NetStream comes in three formats: v5, v8 and v9

  Essentially mirroring NetFlow v5, v8 and v9

- Generally easily supported by NetFlow collectors

  However differences exist between NetStream v9 and NFv9

  eg: NetFlow represents interfaces using ifIndex

  (standard MIB), NetStream represents using proprietary

  interface MIB

# NetStream

- Collector needs to be explicitly told record is NetStream
- NetStream is supported Huawei , HP
- Implementations do not support flexible templates

# IETF: IP Flow Information Export  WG (IPFIX)

- IPFIX protocol specifications

  Changes in terminology but same NetFlow Version 9 principles (IPFIX  version field says ‘10’)

  - Improvements vs. NetFlow v9: SCTP-PR, security, variable
  - length information element, IANA registration, etc.
  - Generic streaming protocol, not flow-centric anymore
  - Security:
    - Threat: confidentiality, integrity, authorization

      Solution: DTLS on SCTP-PR  Anonymization draft

# IETF: IP Flow Information Export  WG (IPFIX)

- IPFIX information model
  - Most NetFlow v9 information elements ID are kept
  - Proprietary information element specification

# IETF: IP Flow Information Export  WG (IPFIX)

- RFC7011  Specification of the IPFIX Protocol for the Exchange of IP Traffic Flow Information
- RFC 7012 Information Model for IP Flow Information Export
- RFC5103 “Bidirectional Flow Export using IP Flow Information Export (IPFIX)”
- RFC 6183 - IPFIX Mediation: Framework - IETF Tools
- RFC 7013 Guidelines for Authors and Reviewers of IP Flow Information Export (IPFIX) Information Elements
- RFC8158 -IP Flow Information Export (IPFIX) Information Elements for Logging NAT Events
- 
- ...

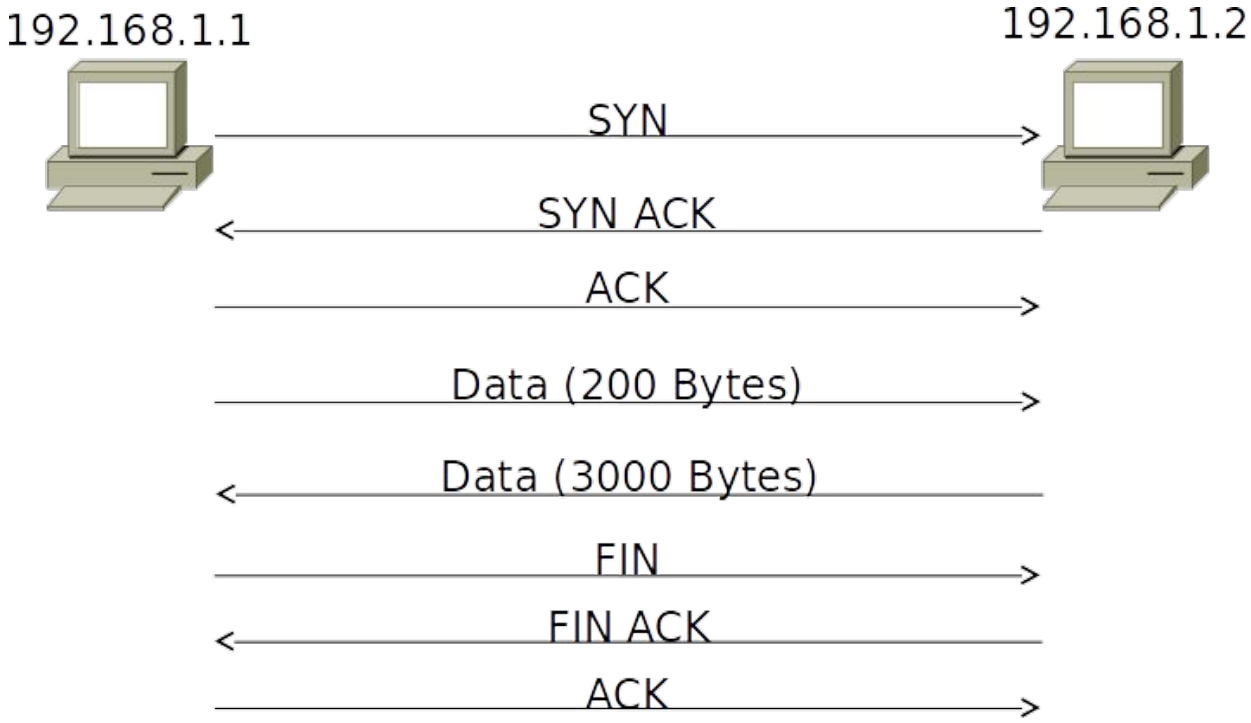
# S-flow

# Sflow

| Version | Comment |
|---|---|
| **v1** | Initial version |
| **v2** | (Unknown) |
| **v3** | Adds support for *extended_url* information.[12] |
| **v4** | Adds support BGP communities.[12] |
| **v5** | Several protocol enhancements.[13] This is the current version, which is globally supported. |

# Random Kitten

# TCP Connection Example

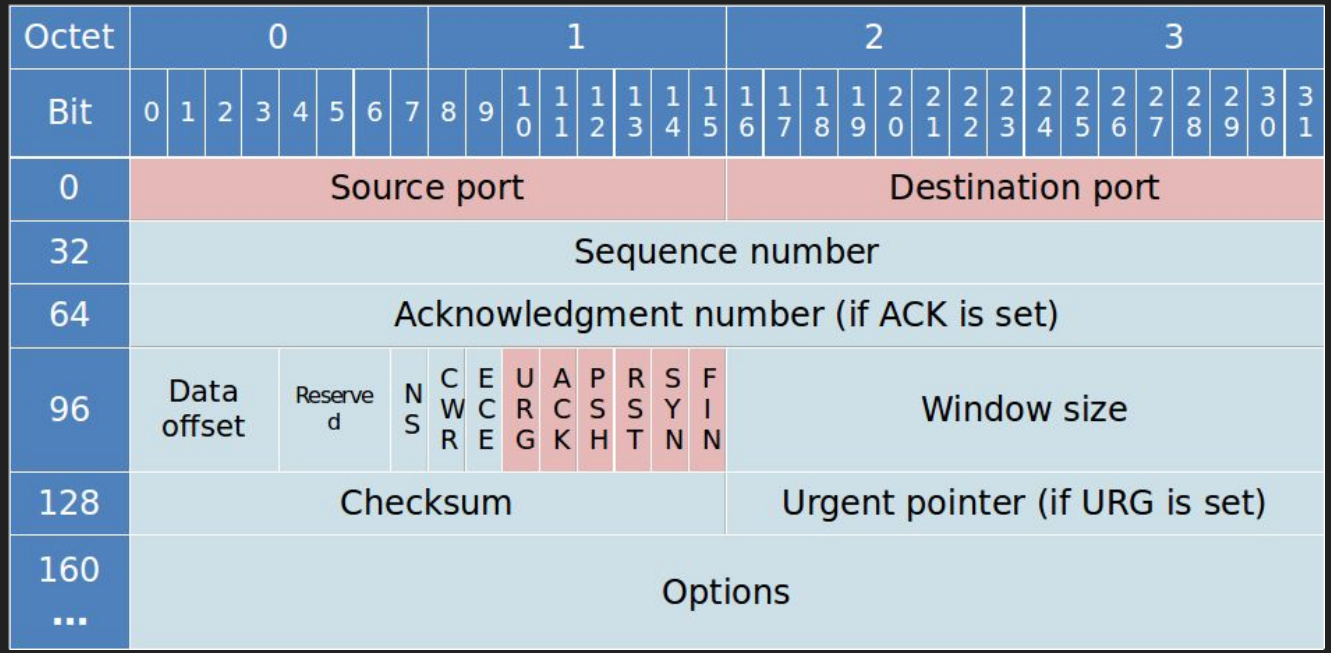


# IPv4

| Bit offset | 0-3 | 4-7 | 8-13 | 14-15 | 16-18 | 19-31 |
|---|---|---|---|---|---|---|
| 0 | Version | Internet Header Length | Differentiated Services Code Point | Explicit Congestion Notification | Total Length | |
| 32 | Identification | | | | Flags | Fragment Offset |
| 64 | Time to Live | | Protocol | | Header Checksum | |
| 96 | Source IP Address | | | | | |
| 128 | Destination IP address | | | | | |
| 160 | Options | | | | | |
| 160 or 192+ | Data | | | | | |

Adapted from: http://en.wikipedia.org/wiki/IPv4#Packet_structure

# TCP


Octet
0
1
2
3
Bit
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0
Source port
Destination port
32
Sequence number
64
Acknowledgment number (if ACK is set)
96
Data offset
Reserved
NS
CWR
ECE
URG
ACK
PSH
RST
SYN
FIN
Window size
128
Checksum
Urgent pointer (if URG is set)
160 ...
Options

# UDP

| Bit Offset | 0-15 | 16-31 |
|---|---|---|
| 0 | Source port | Destination port |
| 32 | Length | Checksum |
| 64 … | Data | |

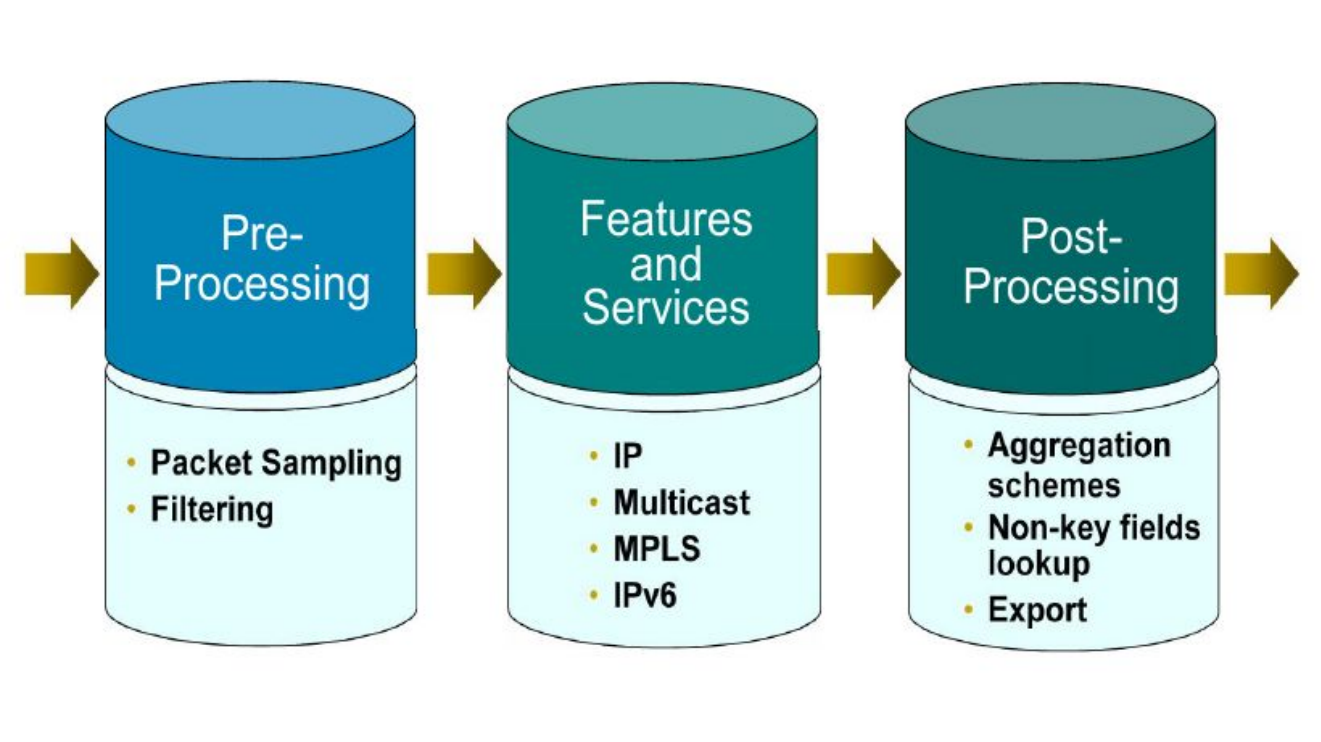
# Uniflow vs. Biflow

| | | | | | |
|---|---|---|---|---|---|
| 192.168.1.1 | 56981 | 192.168.1.2 | 80 | 1/17/12 15:23:30 | 300 bytes |

| | | | | | |
|---|---|---|---|---|---|
| 192.168.1.2 | 80 | 192.168.1.1 | 56981 | 1/17/12 15:23:30 | 3060 bytes |

| | | | | | | |
|---|---|---|---|---|---|---|
| 192.168.1.1 | 56981 | -> | 192.168.1.2 | 80 | 1/17/12 15:23:30 | 3360 bytes |

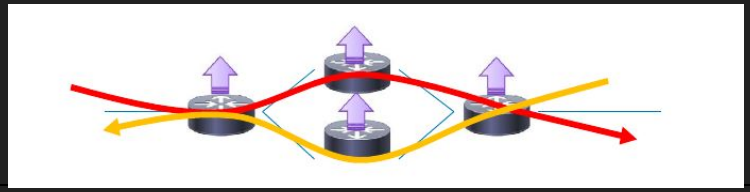
# Netflow processing order in router



# Cost of flow

- How is the measurement done (ASIC, CPU, etc)
  - Might have TCAM impact (number of flow entries)
  - Might have CPU impact (reducing forwarding performance)
- Does every packet need to be measured?
  - Various sampling methodologies (random time, random packet, 1 in X, etc)
  - Drastic reduction in measurement hit
- Where is the export done?
  - Distributed across line cards vs. centralized
- Considerations and tests laid out in:
  - draft-novak-bmwg-ipflow-meth-*

Key concepts



- Network traffic is often asymmetrical, even on small networks
- Probes typically require engineered symmetry
- This means that with *Flow, there's no need to engineer the network around the instrumentation
  - We can follow traffic through the network over its natural path
  - We can see pps, bps, packet-size, QoS markings, TCP flags,etc. for specific apps/services at each point in the network
  - We can validate traffic engineering, policy enforcement, etc. at any point in the topology, as long as *Flow is enabled

# Sensor deployment

Where are sensors deployed logically?

Sensor hardware limitations

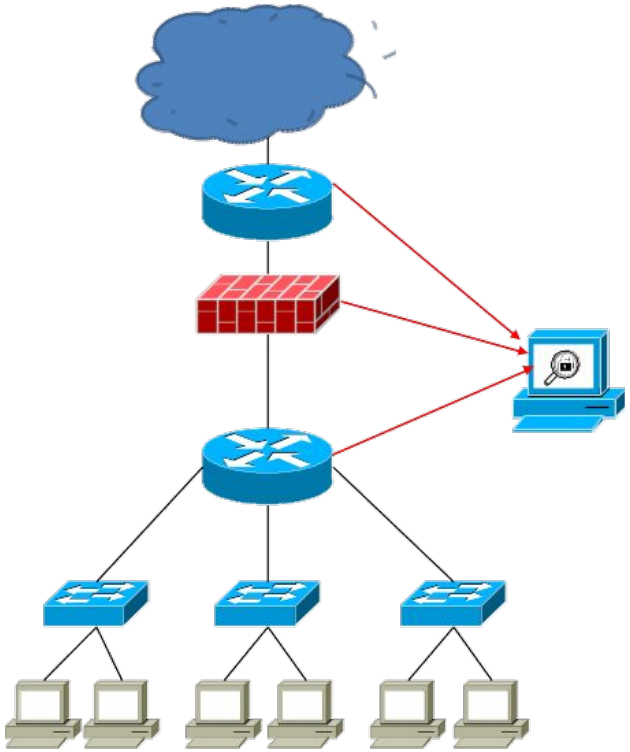
Relative to the bandwidth of the link being monitored

Straight collection vs. pushing analytics forward

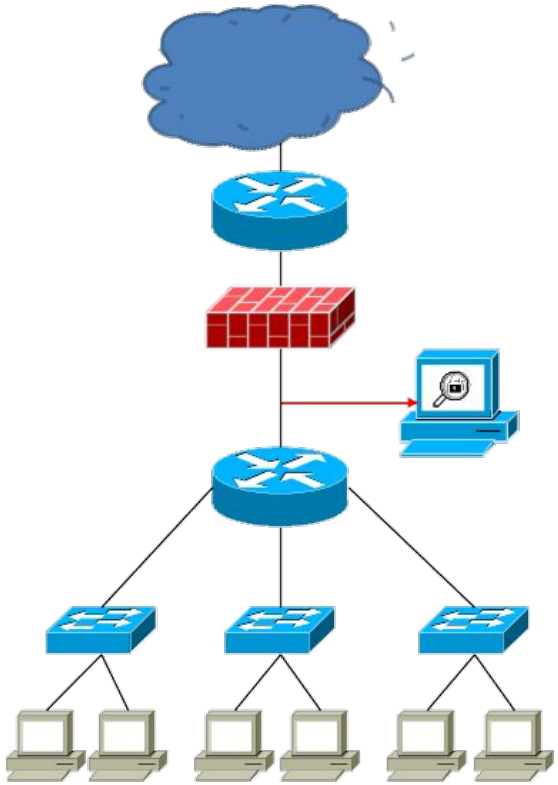
Bandwidth back to centralized processing and storage

Passive stand-alone sensor vs. getting netflow off of routers

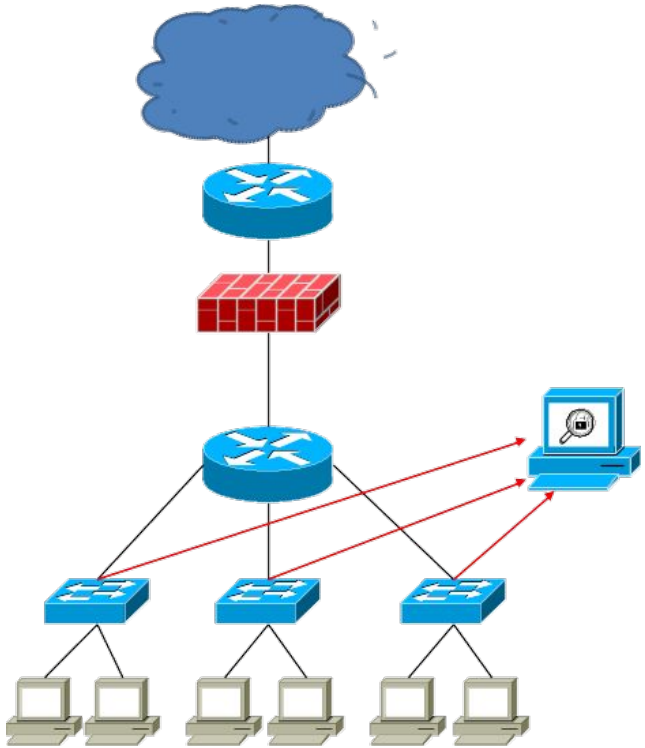
# Perimeter visibility



# Perimeter visibility



# Enclave -Level



# Hostflow

MITRE developed tool

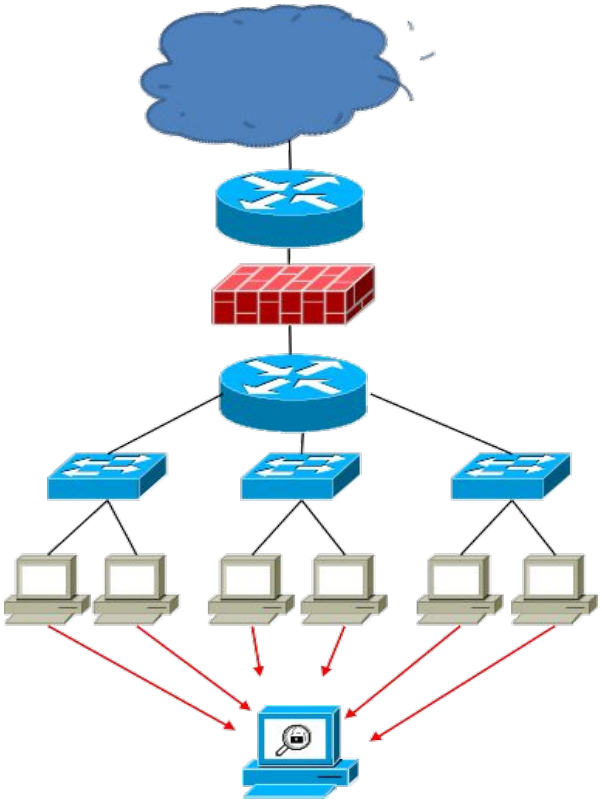
Collects netflow-like data from the host

Requires deploying an agent on the device

Bridges the gap between network and host

Allows visibility when deploying network sensors at the access layer is cost prohibitive

# Host Level visibility



# Flow sampling

- Routers do typically  flow sampling
  - Selecting one out of every n packets
  - Great for net ops, hard to use for security
- Enclave-level sensors produce a lot of flows
- Depending on your analytics you can do some intelligent sampling and aggregation
- Particularly important if your enterprise is geographically distributed

# Kittens

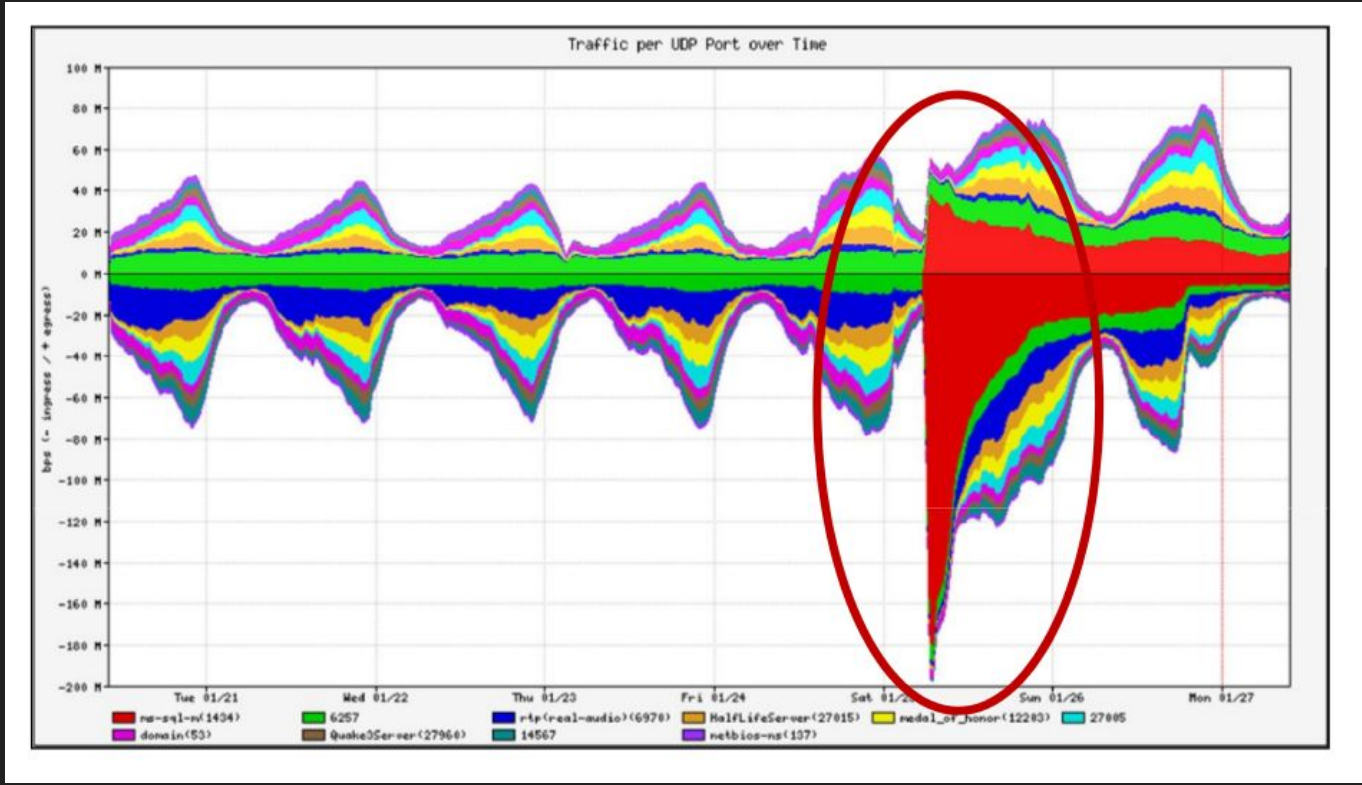
# Principal Flow uses

| Service Provider | Enterprise |
| --- | --- |
| Peering Arrangements | Internet access monitoring (protocol distribution, where traffic is going/coming) |
| Network Planning | User Monitoring |
| Traffic Engineering | Application Monitoring |
| Accounting and Billing | Chargeback billing for departments |
| Security Monitoring | Security Monitoring |
| Performance Monitoring | Performance Monitoring |

# Examples of use

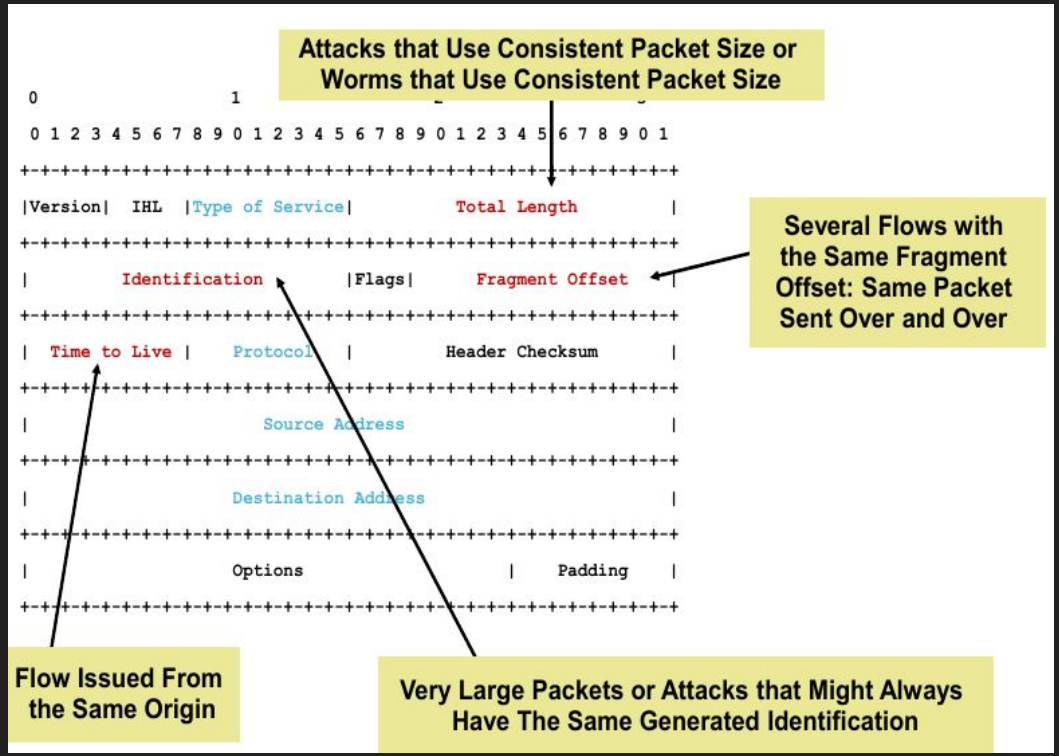
| Customer Challenge | Description | Problem Situation | NetFlow Resolution |
|---|---|---|---|
| Security | Detect SQL Slammer on day one | Detrimental incapacity of servers | NetFlow day-zero anomaly detection |
| Traffic Analysis | Bandwidth Hog | - Sluggish network performance<br>- Single user application monopolizing network | Cost savings of $7k in labor costs |
| Traffic Analysis | Full Circuit | Circuit 100% utilized | Quickly tracked problem and saved 300 hours = $34k in labor costs |
| Capacity Planning | Slow network performance | - More servers and bandwidth added<br>- Users still complained<br>- Rented RMON probes - didn't work | Cost savings of $126k in probe costs |
| Capacity Planning | Poor network performance – low bandwidth | We need more bandwidth | Tracked point of slowdown – saved $36k per yr. circuits |

# Example SQL slammer



# Random Kittens

# Useful fields for security monitoring



# What Does Dos Attack Look Like

```
Router# show ip cache flow
…
SrcIf   SrcIPaddress SrcP   SrcAS   DstIf DstIPaddress   DstP   DstAS    Pr Pkts B/Pk
29      192.1.6.69     77     aaa      49   194.20.2.2   1308    bbb     6    1   40
29      192.1.6.222  1243     aaa      49   194.20.2.2   1774    bbb     6    1   40
29      192.1.6.108  1076     aaa      49   194.20.2.2   1869    bbb     6    1   40
29      192.1.6.159   903     aaa      49   194.20.2.2   1050    bbb     6    1   40
29      192.1.6.54    730     aaa      49   194.20.2.2   2018    bbb     6    1   40
29      192.1.6.136   559     aaa      49   194.20.2.2   1821    bbb     6    1   40
29      192.1.6.216   383     aaa      49   194.20.2.2   1516    bbb     6    1   40
29      192.1.6.111    45     aaa      49   194.20.2.2   1894    bbb     6    1   40
29      192.1.6.29   1209     aaa      49   194.20.2.2   1600    bbb     6    1   40
```

- Typical DoS attacks have the same (or similar) entries:

  Input interface, destination IP, one packet per flow, constant bytes per packet (B/Pk)

- Don't forget show ip cache verbose flow | include …

# Flow record techniques

- Goals and resources
  - This should shape your analysis
  - Access available time, staff, equipment and tools
- Starting indicators – triggering event
  - Example evidence:
    - IP address of compromised or malicious system
    - Time frame of suspect activity
    - Known ports of suspect activity
    - Specific flows which indicate abnormal or unexplained activity

# Flow record techniques

- Analysis techniques
  - Filtering
  - Baselining
  - “Dirty Values”
  - Activity pattern matching

# Filtering

- Important to narrow down a large pool of evidence
- Remove extraneous data
  - Start by isolating activity relating to specific IP address/es
  - Filter for known patterns of behavior
- Use small percentages of data for detailed analysis

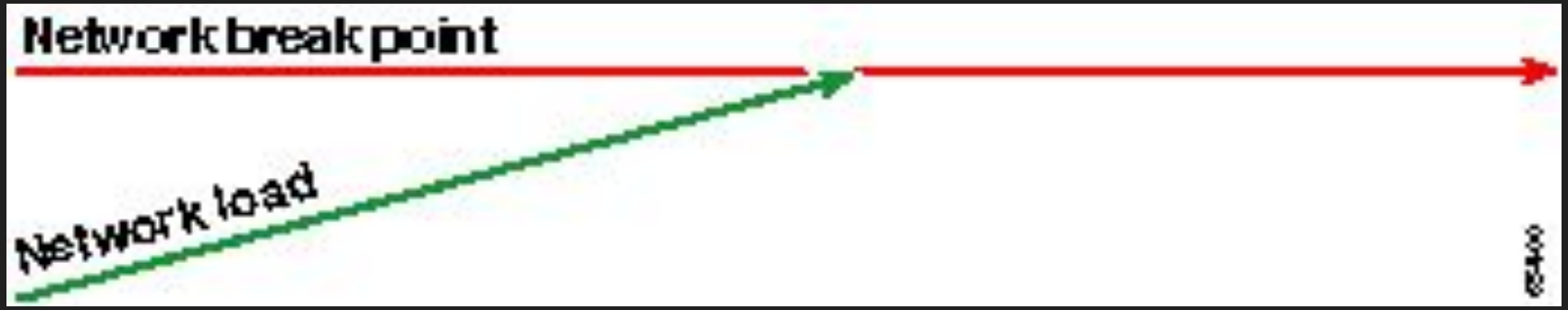
# Random kitten

# Baselining

- Advantage of flow record data vs full traffic capture
  - Dramatically smaller allowing for longer retention
- Build a profile of “normal” network activity
- Network baseline
  - General trends over a period of time
- Host baseline
  - Historical baseline can identify anomalous behavior
  - Most flow patterns will change dramatically if host is compromised or under attack

# Baselining

http://www.cisco.com/c/en/us/support/docs/availability/high-availability/15112-HAS-baseline.html



# “Dirty Values”

- IP addresses
- Ports
- Protocols

# Activity pattern matching

- Elements
  - IP address
    - Internal network or Internet-exposed network
    - Country of origin
    - Who are they registered too?
- Ports
  - Assigned / well-known ports link to specific applications
  - Is system scanning or being scanned?

## Activity pattern matching

- Protocols and Flags
  - Layer 3 and 4 are often tracked in flow record data
    - Connection attempts
    - Successful port scans
    - Data transfers

## Activity pattern matching

- Directionality
  - Data coming in (something downloaded) or going out (something uploaded)
- Volume of data transferred
  - Lots of small packets can indicate port scanning
  - Large amounts of data usually cause for concern

Simple patterns

- Many-to-one IP addresses
- One-to-many IP addresses
- Many-to-many IP addresses
- One-to-one IP addresses

# Simple patterns

- Many-to-one IP addresses
  - DOS attack
  - Syslog server
  - “Drop box”  data repository on destination IP
  - Email server (at destination)
  - Web Server (at destination)

so on ...

## Simple patterns

- One-to-many IP addresses
  - Web server
  - Email server (at source)
  - SPAM bot
  - Warez server
  - Network port scanning
  - …

# Kittens

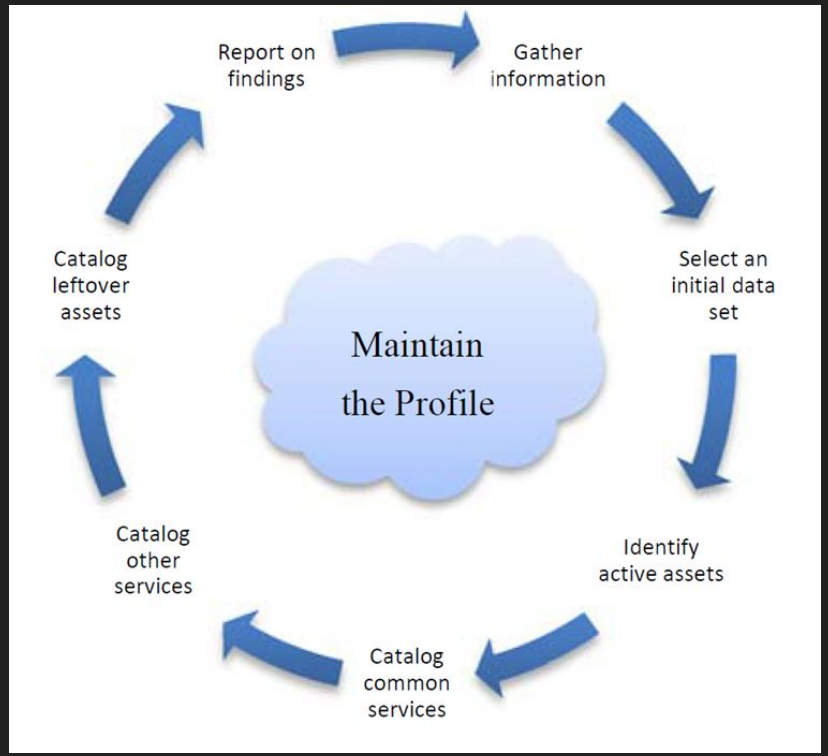
## Simple patterns

- Many-to-many IP addresses
  - Peer-to-peer file sharing
  - Widespread port scanning
- One-to-one IP addresses
  - Targeted attack
  - Routine Server communication

# Complex patterns

- Fingerprinting
  - Matching complex flow record patterns to specific activities
  - Example:
    - TCP SYN port scan
    - One source IP address
    - One or more destination IP addresses
    - Destination port numbers increase incrementally
    - Volume of packets surpass a specified value within a given period of time
    - TCP protocol
    - Outbound protocol flags set to “SYN”

# Profiling (Part of asset management/fingerprinting / Baselining)

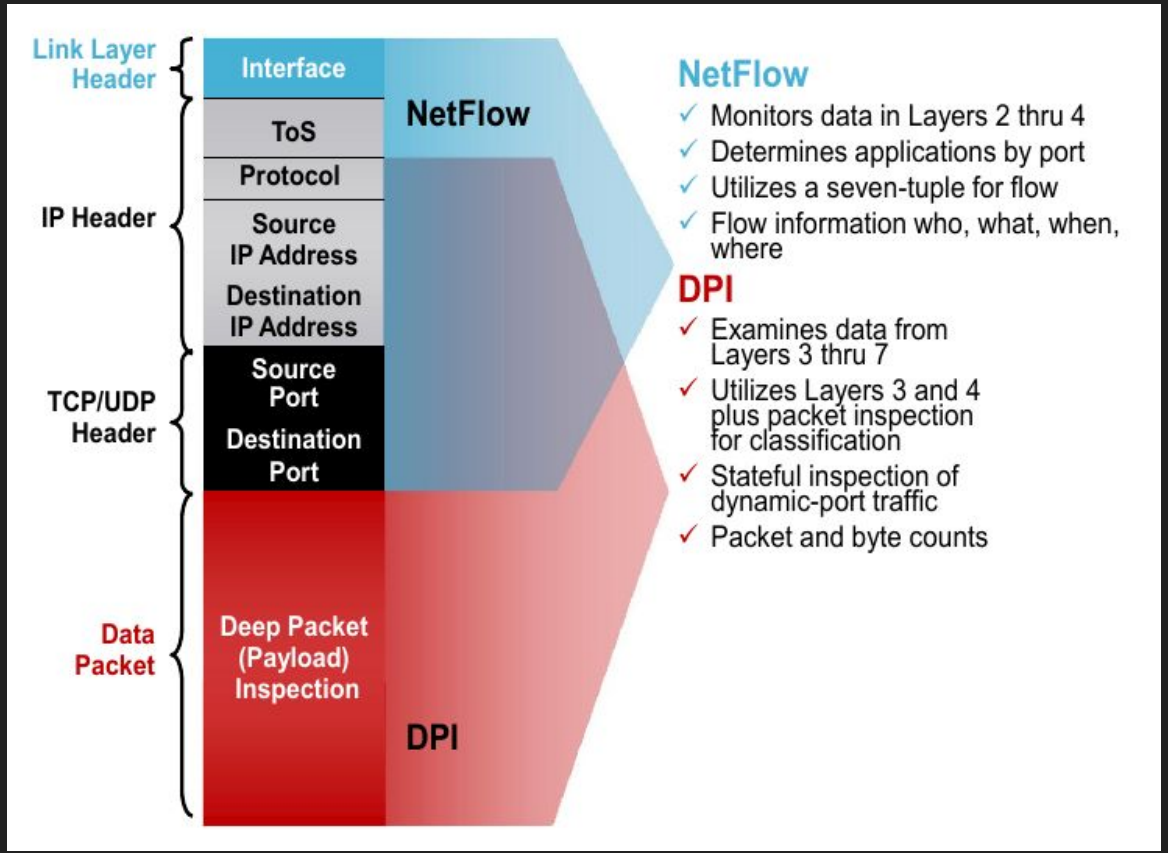


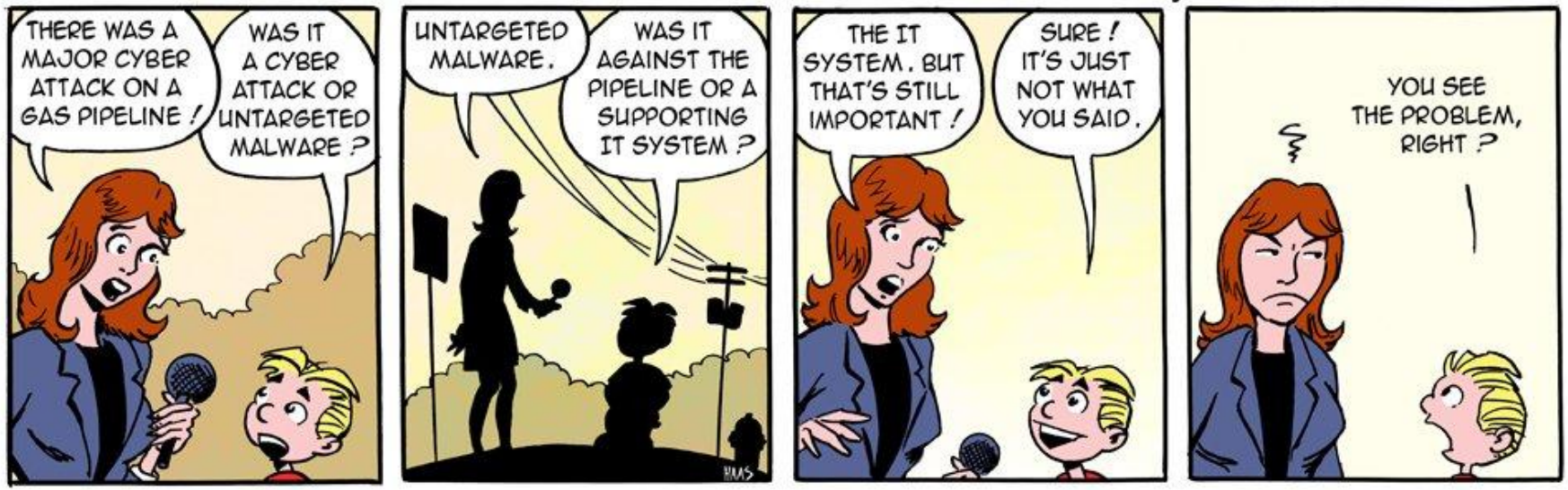
# Darkspace Monitoring

- If you've got darkspace on your network , keep an eye on it
- Entropy in IP Darkspace Data by Tanja Zseby, FloCon 2012
  - Work tracks distribution of entropy in {sip, dip, sport, dport} and characterizes common events

# Random kitten

# Network Based Application Recognition (fingerprinting)




LITTLE BOBBY
by Robert M. Lee and Jeff Haas
THERE WAS A MAJOR CYBER ATTACK ON A GAS PIPELINE!
WAS IT A CYBER ATTACK OR UNTARGETED MALWARE?
UNTARGETED MALWARE.
WAS IT AGAINST THE PIPELINE OR A SUPPORTING IT SYSTEM?
THE IT SYSTEM. BUT THAT'S STILL IMPORTANT!
SURE! IT'S JUST NOT WHAT YOU SAID.
YOU SEE THE PROBLEM, RIGHT?
HAAS