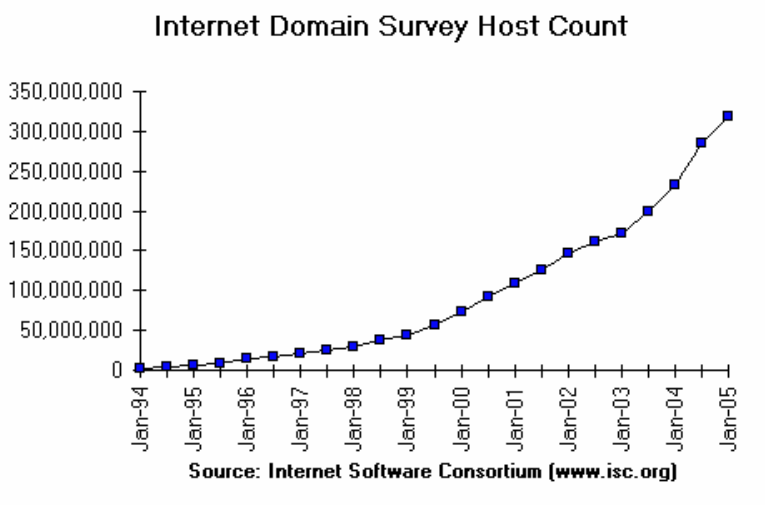
Measuring a Malicious Internet

David Moore

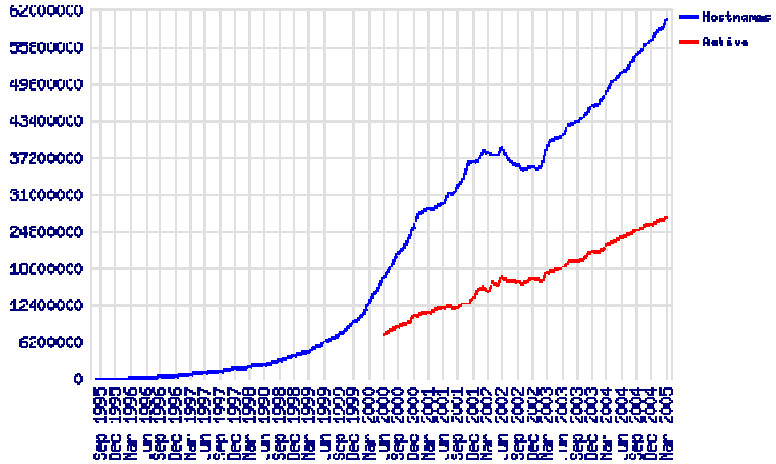
University of California, San Diego

Thesis Proposal – March 18, 2005

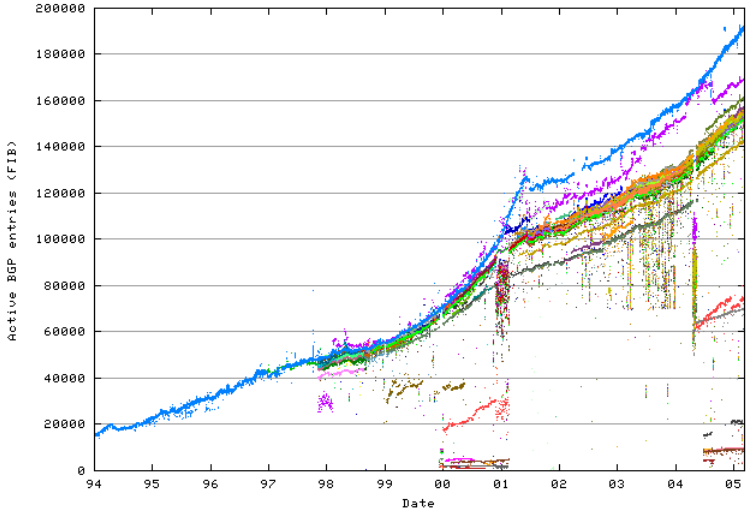
Growth of the Internet



Webserver Count (source: netcraft.com)



Global BGP Routing Prefixes (source: Geoff Huston)



Growth of Malicious Traffic

- DDoS (Distributed Denial-of-Service) attacks
- Spam/Spim/Spit
- Phishing
- Worms/Viruses
- Spyware

- Botnets
  - Collections of 10s – 100,000s of compromised hosts
  - “Backdoor” software installed, allowing coordinated remote control by some entity
  - Typically the source of DDoS, spam, phishing sites

Thesis Motivation (1)

- Need to measure the Internet
  - Traffic engineering, billing, QoS, improving protocols, …
- But it is increasingly hard to measure
  - More traffic, higher speed links, malicious activity
- We have had initial success at measuring (or ignoring) certain non-subtle malicious activity
  - DDoS, worms
  - "Easy" since these are so blatant
    - large scale, global, lots of traffic, lots of hosts
    - often a distinguishing feature (single victim, single service, etc)

Thesis Motivation (2)

- However, some critical malicious activity is much harder to detect
  - E.g., command and control communication of botnets
  - Localized, point-to-point communication
  - Small amount of traffic hidden in the background noise of legitimate traffic
  - Traffic content may closely (or exactly) mimic legitimate

$\Rightarrow$ We need specialized measurement techniques tuned for low-volume malicious traffic

Outline

- **Background**
  - Traditional flow measurement
  - Difficulties introduced by blatant malicious activity: DDoS, worms, scanning
- Scaling and hardening measurement of “normal” traffic
  - Adaptive NetFlow (SIGCOMM 2004)
  - Flow Counting Extension (SIGCOMM 2004)
  - Traffic Summaries (SIGMETRICS 2005)
- The missing piece: lower volume malicious traffic
- Resources, Plans, Timeline

Typical Operational Measurement Questions

- What is the application breakdown in packets & bytes?
- How much traffic came from or went to a particular subnet?
- What are the best ISPs to peer with to decrease my costs based on the actual traffic of my customers?
- Where is the best place to deploy a new web cache?
- Which of my web servers has the most unique clients?
- Which of my hosts seem to be spam servers?

Flow Measurement

- How do we answer these questions?

- Current operational traffic measurement:
  - Typically collected on routers
  - Packet sampling employed on high-speed links
  - Flow-based (next slide)

Background: What are flows?

Src. IP Addr.	Dest. IP Addr	Proto	Src. Port	Dest. Port	Packet Count	Byte Count
6.1.1.93	1.82.0.1	UDP	53	53	2	497
6.1.0.14	4.44.0.1	TCP	80	2223	4	646
6.3.0.27	1.95.0.1	TCP	1214	62772	125	187008
6.1.1.93	4.71.0.6	TCP	49200	80	3	565
6.1.0.28	1.82.0.1	TCP	49199	80	5	647
6.1.1.93	1.95.0.1	TCP	49198	80	5	647
6.1.1.93	4.71.0.6	TCP	49196	80	6	708
6.1.2.59	7.88.0.1	TCP	51643	80	6	817

- These flow reports can be very large
- So operators aggregate into smaller, meaningful classes to summarize the data.

Background: Flow aggregation (by source ip)

Src. IP Addr.	Dest. IP Addr	Proto	Src. Port	Dest. Port	Packet Count	Byte Count
6.1.1.93	1.82.0.1	UDP	53	53	2	497
6.1.0.14	4.44.0.1	TCP	80	2223	4	646
6.3.0.27	1.95.0.1	TCP	1214	62772	125	187008
6.1.1.93	4.71.0.6	TCP	49200	80	3	565
6.1.0.28	1.82.0.1	TCP	49199	80	5	647
6.1.1.93	1.95.0.1	TCP	49198	80	5	647
6.1.1.93	4.71.0.6	TCP	49196	80	6	708
6.1.2.59	7.88.0.1	TCP	51643	80	6	817

Background: Flow aggregation *(by source ip)*

Src. IP Addr.	Dest. IP Addr	Proto	Src. Port	Dest. Port	Packet Count	Byte Count
6.1.1.93	1.82.0.1	UDP	53	53	2	497
6.1.0.14	4.44.0.1	TCP	80	2223	4	646
6.3.0.27	1.95.0.1	TCP	1214	62772	125	187008
6.1.1.93	4.71.0.6	TCP	49200	80	3	565
6.1.0.28	1.82.0.1	TCP	49199	80	5	647
6.1.1.93	1.95.0.1	TCP	49198	80	5	647
6.1.1.93	4.71.0.6	TCP	49196	80	6	708
6.1.2.59	7.88.0.1	TCP	51643	80	6	817

6.1.1.93	16	2417

Background: Flow aggregation (by source ip)

Src. IP Addr.	Dest. IP Addr	Proto	Src. Port	Dest. Port	Packet Count	Byte Count	Flow Count
6.1.1.93	1.82.0.1	UDP	53	53	2	497	
6.1.0.14	4.44.0.1	TCP	80	2223	4	646	
6.3.0.27	1.95.0.1	TCP	1214	62772	125	187008	
6.1.1.93	4.71.0.6	TCP	49200	80	3	565	
6.1.0.28	1.82.0.1	TCP	49199	80	5	647	
6.1.1.93	1.95.0.1	TCP	49198	80	5	647	
6.1.1.93	4.71.0.6	TCP	49196	80	6	708	
6.1.2.59	7.88.0.1	TCP	51643	80	6	817	

6.1.1.93					16	2417	**4**

Flow reporting

- Need to transfer the flow data from the router to a data collection machine for further analysis

- Fixed size time bins (commonly 10 sec. - 5 min.)
  - Terminate all flow records at the end of the bin
  - Spread reporting throughout next bin

  - For example, every 5 minutes the router would export all of the flow records for the previous 5 minutes of traffic

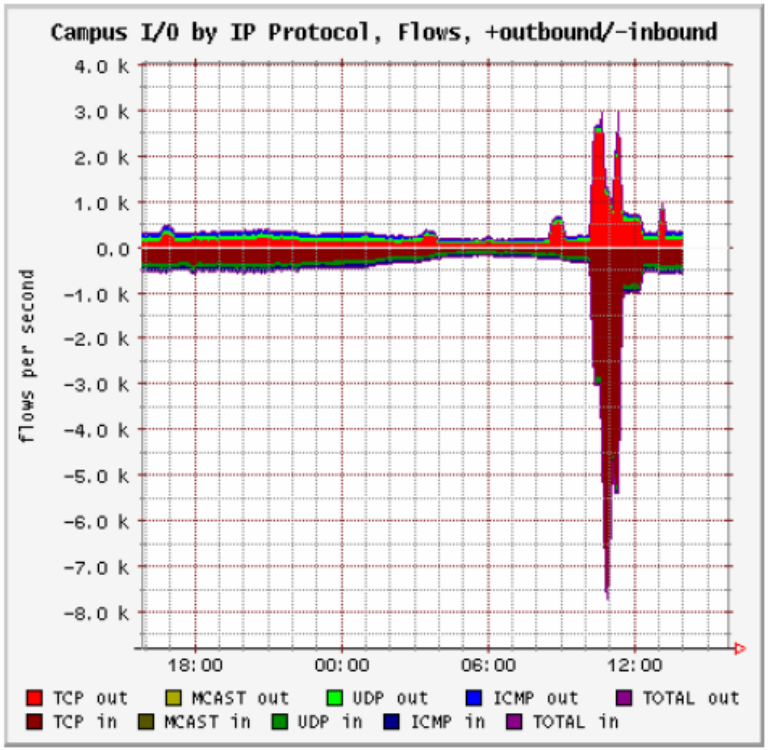
Sample Reports

IPMON

Application Breakdown			
Category	Packets (%)	Bytes (%)	Flows (%)
Web	54.35	61.48	47.33
File Sharing	3.35	2.43	3.74
FTP	0.52	0.54	0.07
Email	4.67	4.06	3.24
Streaming	7.26	13.07	1.60
DNS	6.13	1.16	27.26
Games	0.06	0.01	0.03
Other TCP	21.03	15.86	6.05
Other UDP	0.78	0.48	0.84
Not TCP/UDP	1.86	0.90	9.84

Site: San Jose (sj-20)
Date: February 5th, 2004

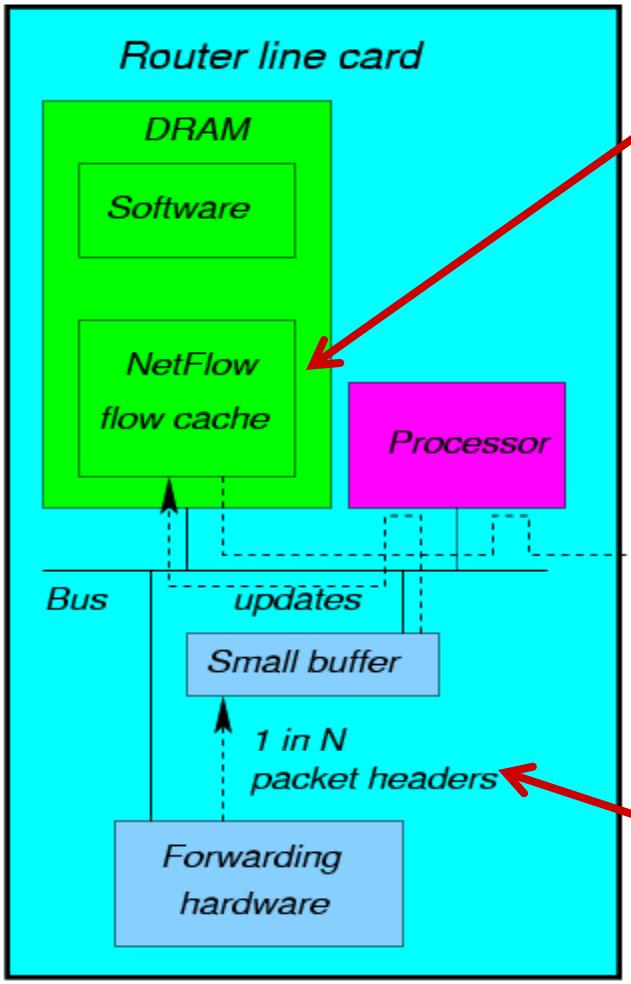
FlowScan



Background: Large-Scale Malicious Traffic

- Denial-of-service attacks, worm spread and port scanning can overwhelm flow measurement systems
- Fields in the flow key take on a much larger range than in normal traffic:
  - Spoofed source DoS = random source IP address
  - Typical Internet worm = random destination IP address
  - Port scanning = walk of large # of ports and addresses
- In these situations every single packet may result in a separate flow

Background: Flow Collection System Diagram



Large numbers of flow records can overwhelm the memory.

Exporting many records can overwhelm the collection path.

terminated
flow records

Data
collection
and
analysis
server

Sampling rate set statically, but optimal sampling rate depends on traffic mix.

Outline

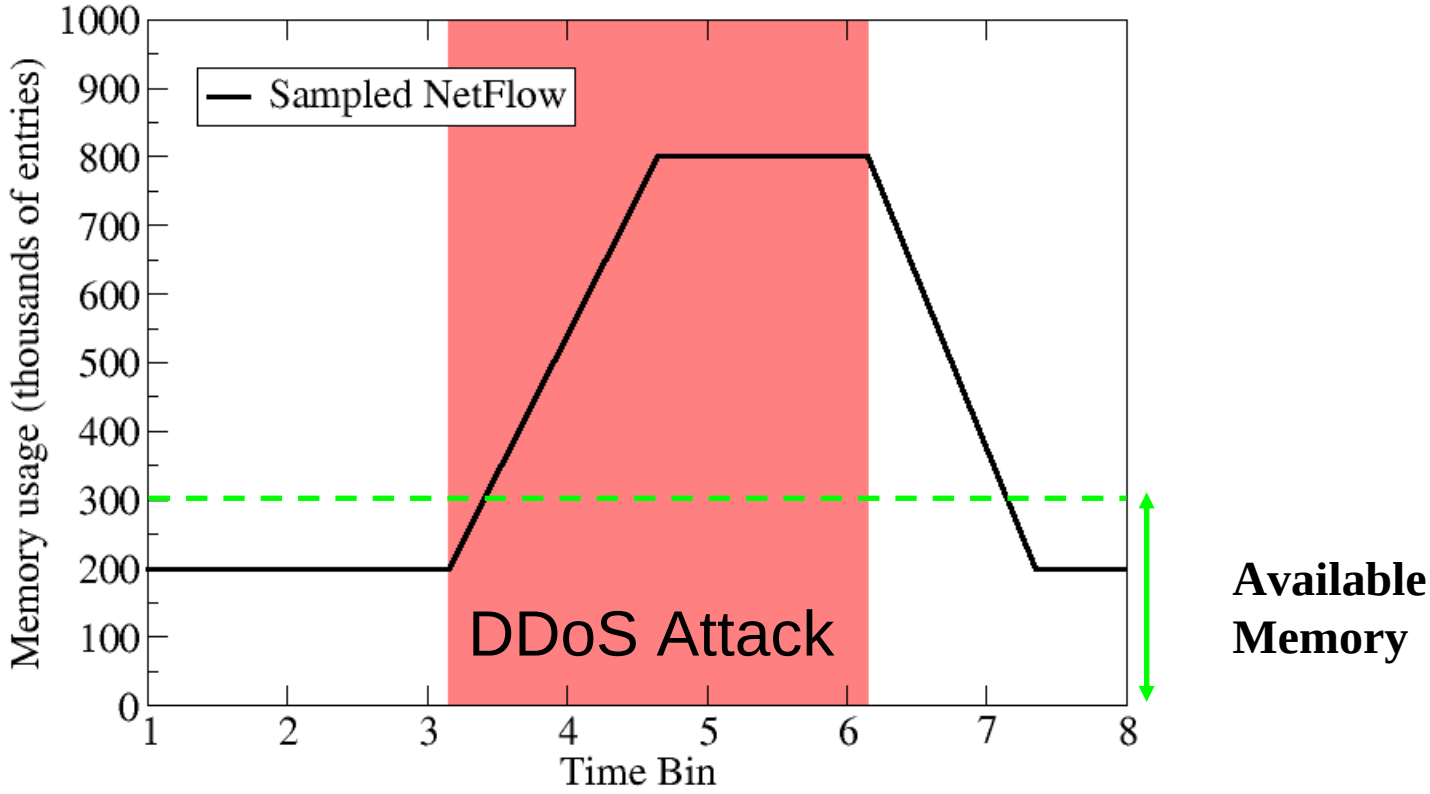
- Background
  - Traditional netflow measurement
  - Difficulties introduced by blatant malicious activity: DDoS, worms, scanning
- Scaling and hardening measurement of “normal” traffic
  - **Adaptive NetFlow (SIGCOMM 2004)**
  - Flow Counting Extension (SIGCOMM 2004)
  - Traffic Summaries (SIGMETRICS 2005)
- The missing piece: lower volume malicious traffic
- Plans, Timeline, Conclusions

Adaptive NetFlow
(SIGCOMM 2004)

Problem	Solution
Memory and bandwidth usage strongly depend on traffic mix	**Adapting sampling rate**
Admin must set sampling rate	

Flow Measurement: Simulated memory usage under DDoS

- Traditional flow measurement with fixed sampling rate



Adaptive NetFlow

- Goals:
  - Guaranteed accuracy under any traffic mix
  - Graceful response to adverse traffic
  - Meaningful tuning knob:
    - # of desired records, not static sampling rate

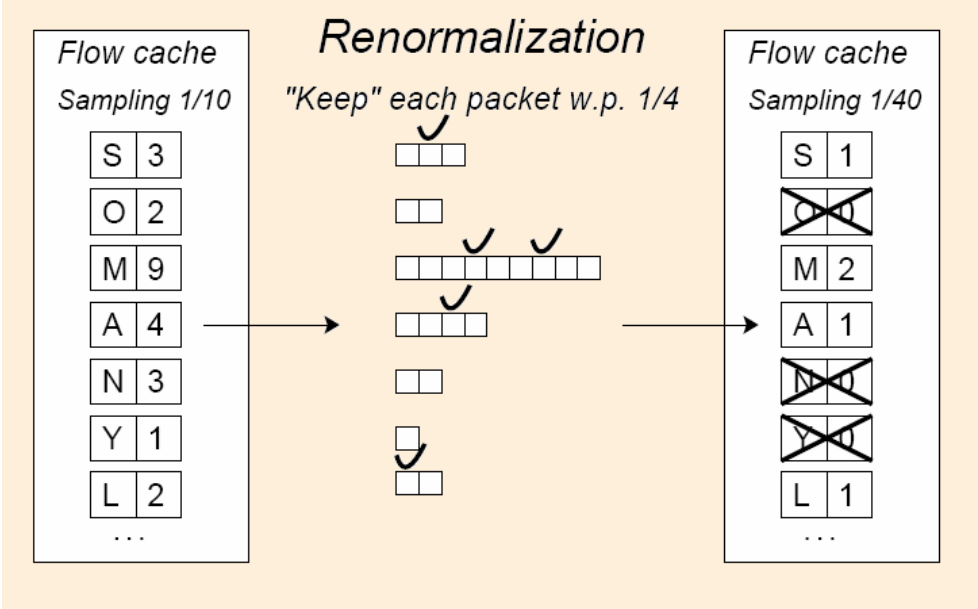
- Choose the sampling rate based on traffic:
  - Use a high sampling rate when traffic allows
  - Within each time bin, reduce the rate when necessary:
    - Ensure we never overload CPU
    - Ensure we never run out of memory
  - Keep counters meaningful as sampling rate varies

Adaptive NetFlow: Main tuning knob: # of records M

- User configures number of records to be exported for each measurement bin
  - Memory in router, resources for data collector
  - Accuracy of results
  - Independent of traffic mix
- Relative error in estimating an aggregate that is a certain fraction of the traffic depends on ***M***
- Dropping random records worse than generating fewer records by using lower sampling rate [DL03]

Adaptive NetFlow: Renormalizing counters

- Decreasing sampling rate
  - pretend to throw away previously observed packets
- Increasing sampling rate
  - information has already been discarded
  - would increase error
- Start each measurement bin with optimistically aggressive sampling



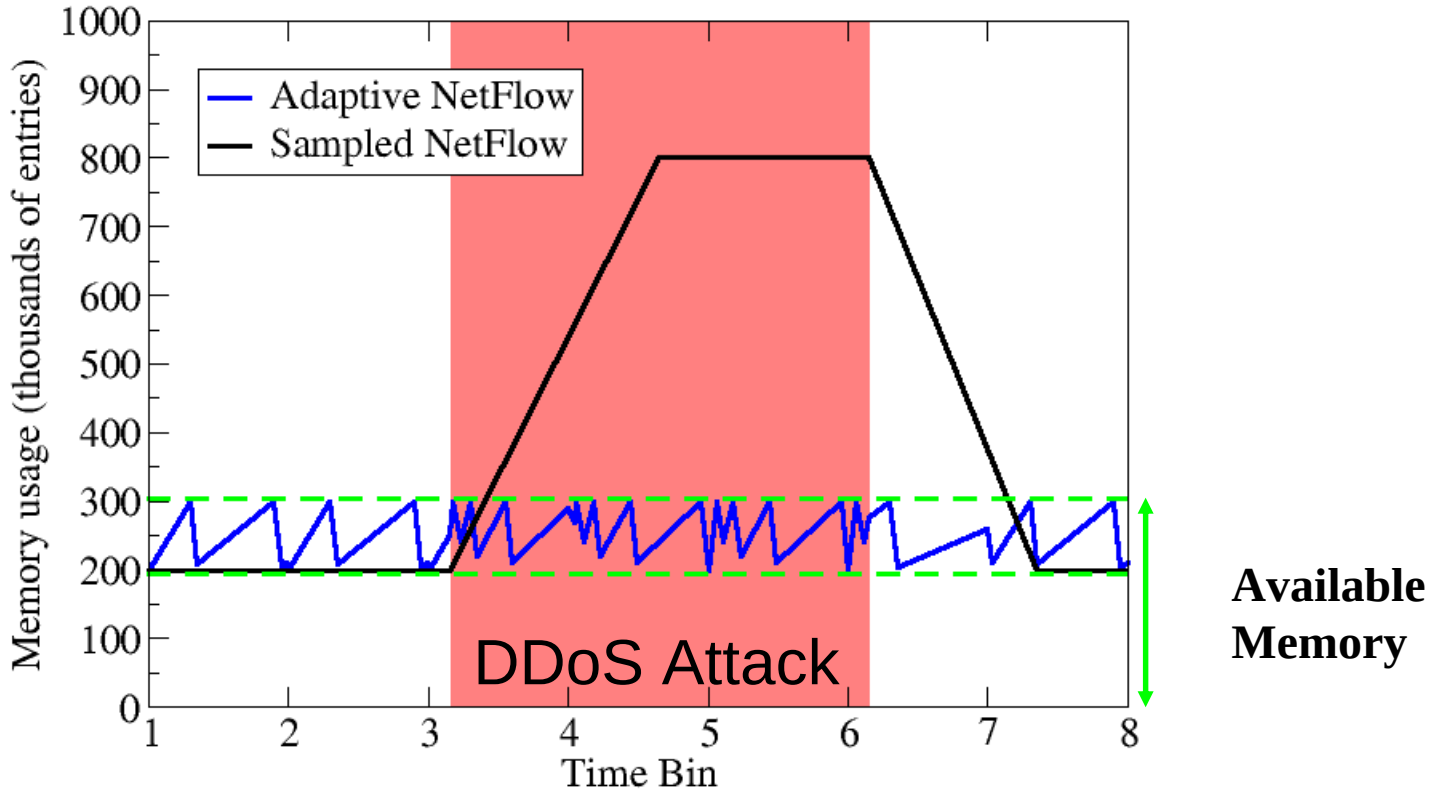
Adaptive NetFlow: CPU usage

- Renormalization in parallel with operation
- Efficient renormalization – for most records only simple integer arithmetic, no random numbers
  - Updating 1 entry 3.4 μs
  - Renormalizing 1 entry 1.5 μs
- Initial sampling rate chosen to allow update and renormalization with worst-case traffic mix

Adaptive NetFlow: Picking the sampling rate

- Chose new sampling rate to leave M flow records after renormalization
- If traffic slows, the sampling rate is not too low
  - The M flow records accurately describe the traffic
- If traffic increases, the sampling rate is not too high
  - Renormalization frees space faster than new entries appear
  - Each time that the sampling rate is reduced, the worst case rate of new entries decreases

Adaptive NetFlow: Simulated memory usage under DDoS



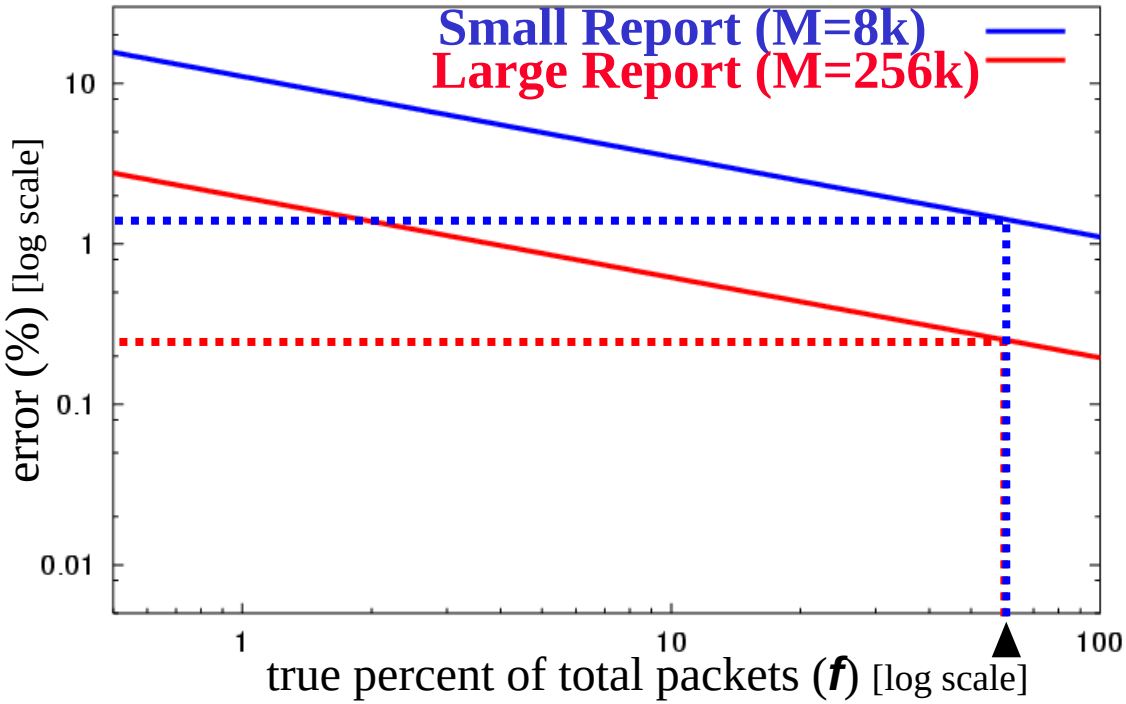
M=200,000 records, 1 minute time bins

Adaptive NetFlow guarantees

If ANF generates ***M*** entries, the relative standard deviation for an aggregate that is fraction ***f*** of the traffic is at most $\sqrt{1/(\boldsymbol{Mf})}$ in packets and $\sqrt{s_{max}/(s_{avg}\boldsymbol{Mf})}$ in bytes (for any the traffic mix).

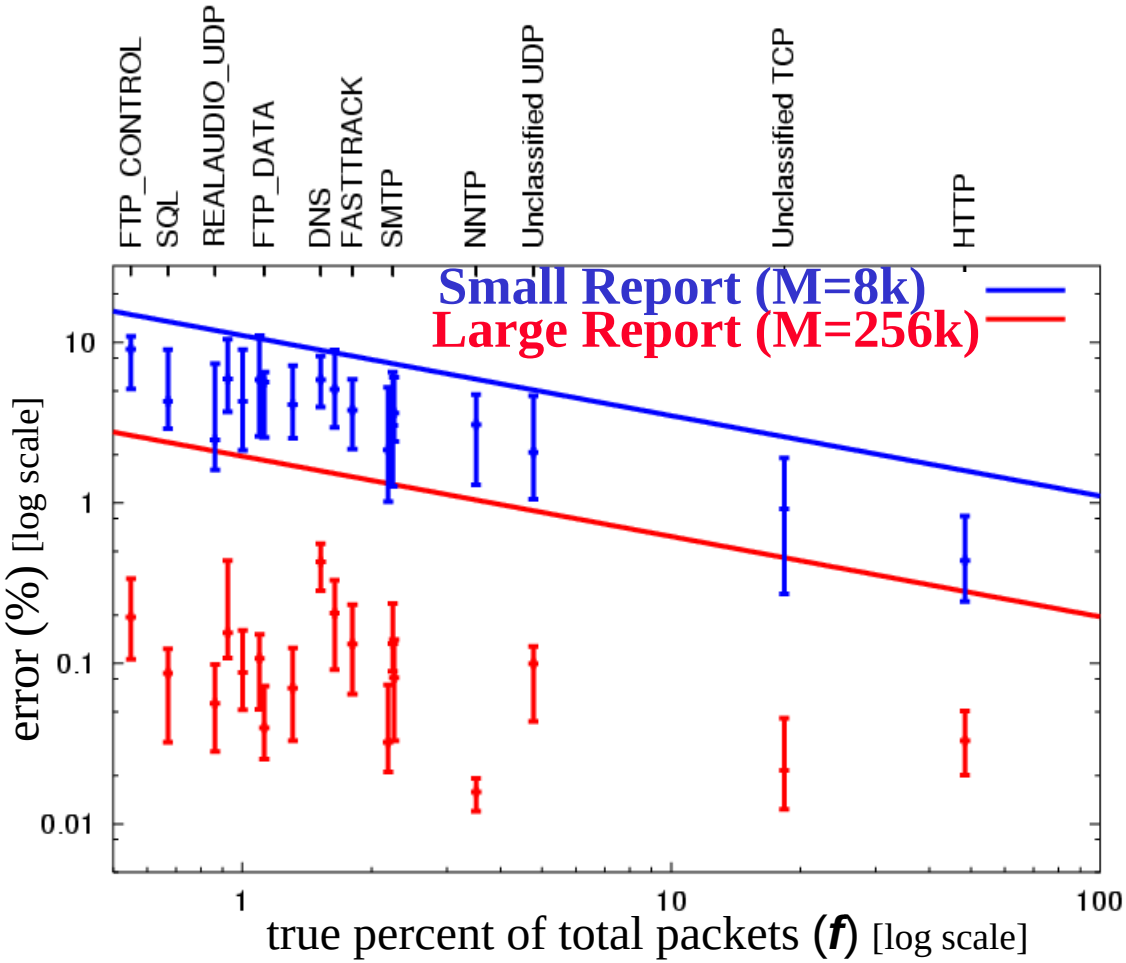
Example:

If HTTP traffic represents 60% of the actual total packets,
then the expected error is less than 2% for M=8k,
and less than 0.3% for M=256k.



Adaptive NetFlow results

Measured error is significantly better than theoretical bounds for normal traffic mixes.



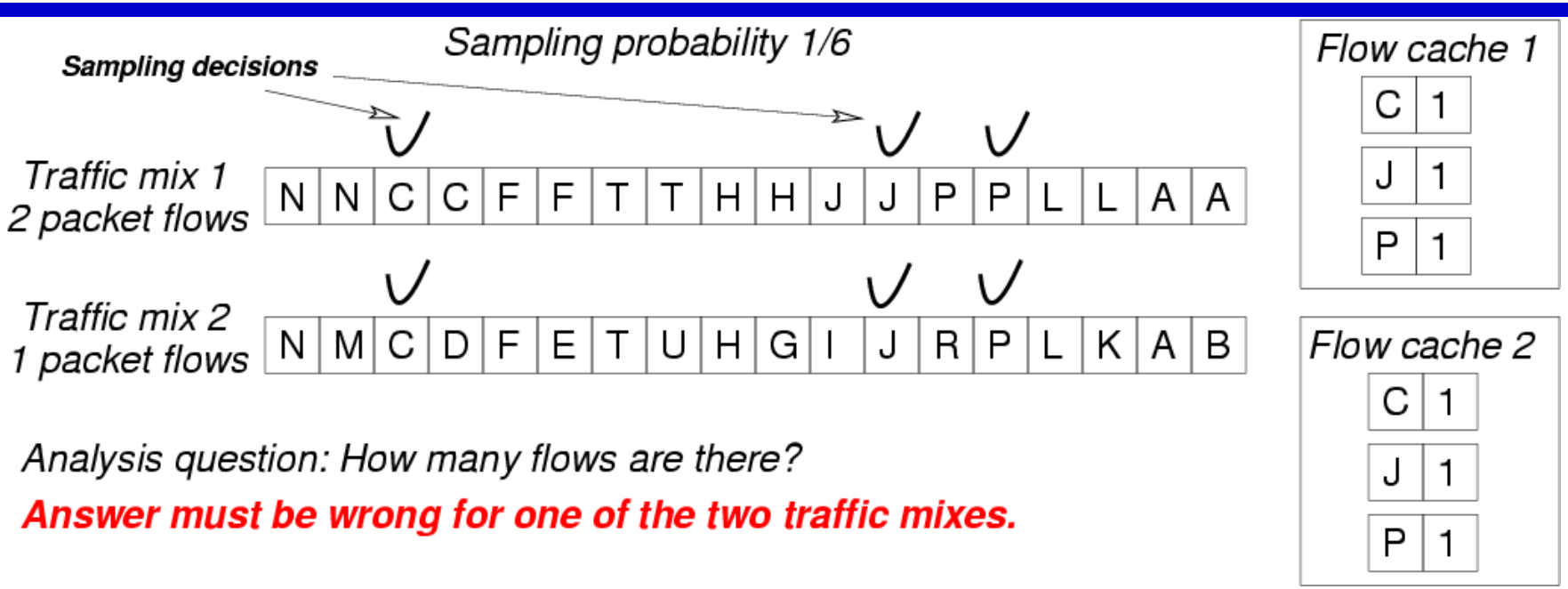
Adaptive Netflow Summary

- Replacement for existing, widely deployed flow-based measurement system
  - Guaranteed accuracy under any traffic mix
  - Meaningful tuning knob: # of desired records
  - Graceful response to large-scale, high-volume malicious traffic
- However, all approaches using packet sampling are unable to answer “flow counting” questions
  - Which of my web servers has the most unique clients?
  - Which of my hosts seem to be spam servers?

Outline

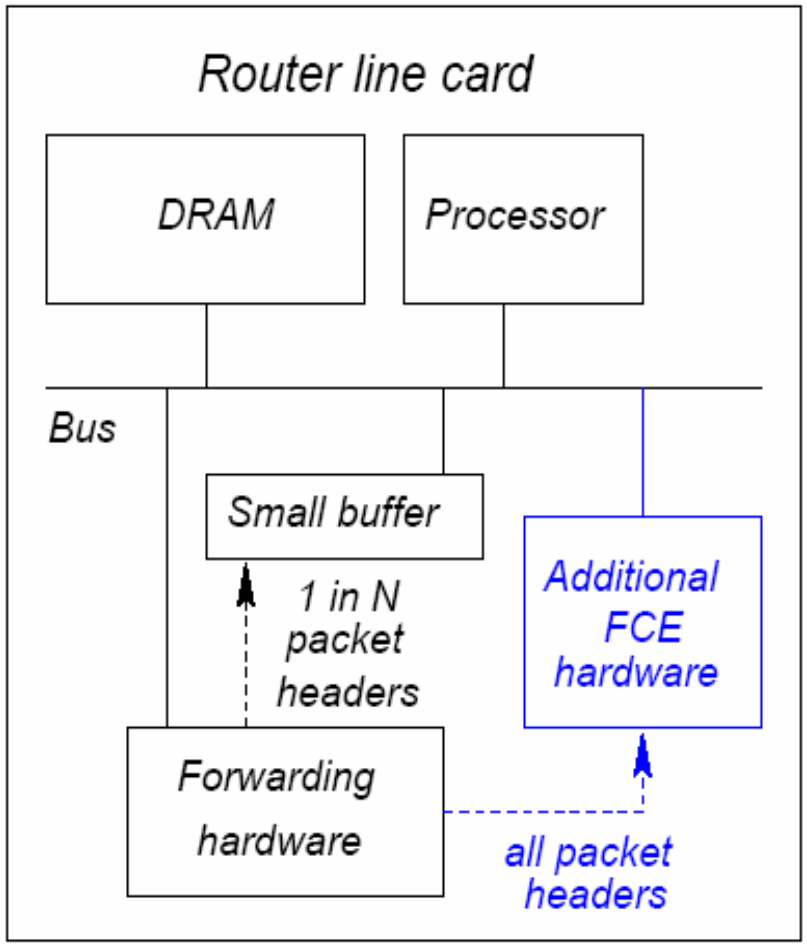
- Background
  - Traditional netflow measurement
  - Difficulties introduced by blatant malicious activity: DDoS, worms, scanning
- Scaling and hardening measurement of “normal” traffic
  - Adaptive NetFlow (SIGCOMM 2004)
  - **Flow Counting Extension (SIGCOMM 2004)**
  - Traffic Summaries (SIGMETRICS 2005)
- The missing piece: lower volume malicious traffic
- Plans, Timeline, Conclusions

Counting flows



- Goal: Unbiased, accurate flow counts for arbitrary post aggregation of the flows.
- Solution: Statistical sampling of flows via hash function. (Requires hardware support.)

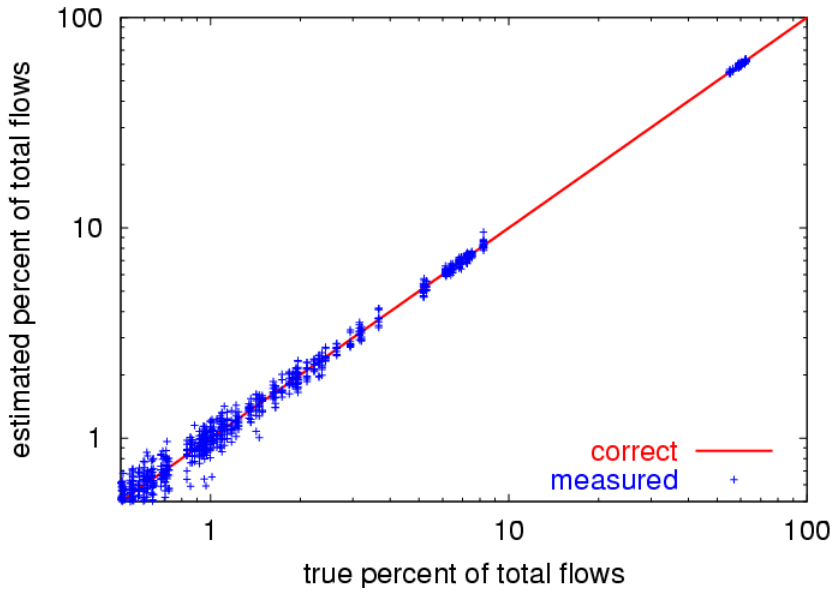
Flow Counting Extension



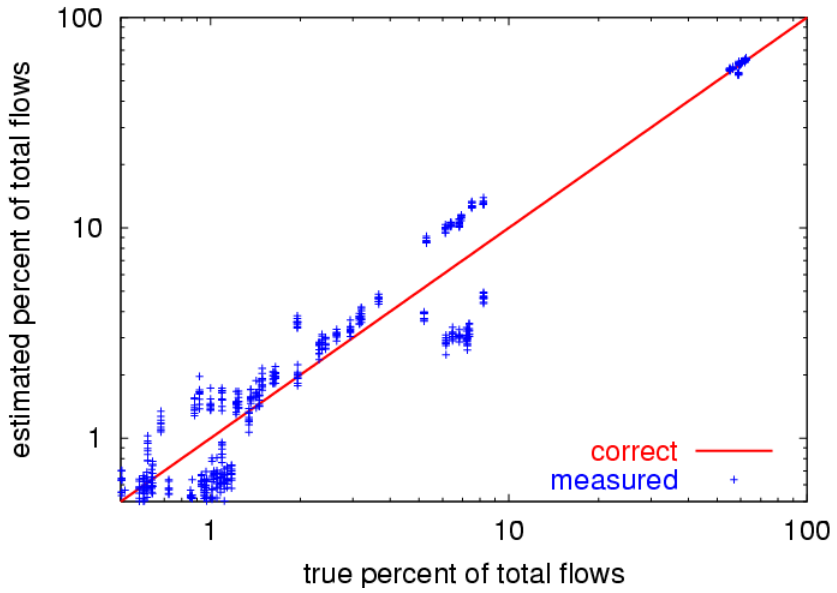
- Use “adaptive sampling” by Wegman and Flajolet
- Keep a table of all flow identifiers with hash(flowID)<1/2^{depth}
- At analysis scale flow counts by 2^{depth}
- Implement with CAM
- To fit memory, increase depth dynamically

Flow Counting Extension results

- SYN counting technique estimates number of TCP flows.
- SYN counting cannot estimate UDP, ICMP flows (~20%).
- FCE can count UDP and improves accuracy for TCP



FCE with 8k entries



SYN with M=64k ANF

Flow Counting Extension Summary

- Able to accurately count flows
  - Guaranteed bounds on accuracy for *any* traffic mix
  - Correctly samples flow data even in the face of large-scale, high-volume malicious traffic

- However, while ANF and FCE both provide flow data which accurately approximates the entire traffic mix during high-traffic malicious events, fidelity may be lost on legitimate traffic

Outline

- Background
  - Traditional netflow measurement
  - Difficulties introduced by blatant malicious activity: DDoS, worms, scanning
- Scaling and hardening measurement of “normal” traffic
  - Adaptive NetFlow (SIGCOMM 2004)
  - Flow Counting Extension (SIGCOMM 2004)
  - **Traffic Summaries (SIGMETRICS 2005)**
- The missing piece: lower volume malicious traffic
- Resources, Plans, Timeline

Traffic Summaries

- Most users have a well-defined set of reports and aggregations they normally want.
- Can we do better than Adaptive NetFlow and the Flow Counting Extension when the user specifies the desired aggregations in advance?
- Yes!
  - Smaller, more specific reports.
  - More precise estimates, including tight lower-bounds.
  - Isolation of damage from DoS, worms and scanning.

Traffic Summaries

- Provided reports are of the “heavy-hitters”
  - All aggregates contributing significant numbers of packets, bytes or flows are reported
- Operator configures desired aggregations
- For example:
  - Source IP addresses – top sources by pkts, bytes or flows
  - Protocol/Ports – for determining top applications

Traffic Summary Isolation

- We would prefer that the separate aggregation reports were independent and isolated:
  - Traffic which causes one table to rapidly fill should not interfere with the accuracy of the other tables
- To solve this, we:
  - Adjust the sampling rates independently for each report
  - Dynamically adapt memory consumption for each separate table to ensure high fidelity for all

Outline

- Background
  - Traditional netflow measurement
  - Difficulties introduced by blatant malicious activity: DDoS, worms, scanning
- Scaling and hardening measurement of “normal” traffic
  - Adaptive NetFlow (SIGCOMM 2004)
  - Flow Counting Extension (SIGCOMM 2004)
  - Traffic Summaries (SIGMETRICS 2005)
- **The missing piece: lower volume malicious traffic**

  (New work I’m proposing as remainder of my thesis)
- Resources, Plans, Timeline

Lower Volume Malicious Traffic

- Most DoS, worm, scanning traffic is not subtle.
- In fact, this non-subtlety is why measurement systems need improvements to be robust.

- However, some malicious traffic is different:
  - Low volume, 100s of packets an hour
  - All addresses legitimate
  - Overlay on existing legitimate protocols and services
  - May make use of unwitting 3rd parties to communicate
    - These 3rd parties may be heavily used, legitimate services

An example: botnets

- Botnets:
  - Collections of 10s – 100,000s of compromised hosts
  - "Backdoor" software installed, allowing coordinated remote control by some entity
  - Typically the source of DDoS, spam, phishing sites
- "Command" host(s)
  - Machine(s) from which the controlling entity issues commands to the botnet
- "Coordination" host(s)
  - Often, rather than the command host connecting to each participant machine, the participants check-in with a coordination host
  - Forwards commands and status information among participants

Desired Goals

- Detect all communicating participants of a botnet
- Determine what actions the botnet is engaged in
- Identify the coordination host(s)
- Identify the command host(s)
- Work on botnets of any size
- Find how participants locate the coordination host

Why potentially feasible?

- Typically for a given botnet codebase there will be many variants created
  - Similar to viruses/worms, where other miscreants take the code and modify it for their own purposes
  - Different from viruses/worms, since automated polymorphism for each infectee does not apply
  - Difficult to replace major components of design
    - General software engineering problem
    - Requires significant programming effort
  - Variants mostly change:
    - Ports
    - Hostname/IP address used for coordination
    - Spelling of commands

Basic Approach

- Detection of novel botnet code is very difficult
  - Continue using honeypots and investigative techniques to obtain samples of botnet code
- From sample code, extract “signatures” which are unlikely to change in variants
  - These signatures describe certain behavioral characteristics rather than just being an exact string match
- Examining traffic on a link, match against the signature with the goal of reconstructing as much information about the entire botnet as possible

Outline

- Background
  - Traditional netflow measurement
  - Difficulties introduced by blatant malicious activity: DDoS, worms, scanning
- Scaling and hardening measurement of “normal” traffic
  - Adaptive NetFlow (SIGCOMM 2004)
  - Traffic Summaries (SIGMETRICS 2005)
- The missing piece: lower volume malicious traffic
- **Resources, Plans, Timeline**

Resources*, Plans, Timeline*

- Contacts with ISP operational security personnel
  - Both personally and via CAIDA
- Access to existing efforts in tracking botnets and intrusion detection
  - Personally, via CAIDA and via CIED
- Access to high-speed packet monitors on useful network links
  - CAIDA has deployed OC-48 and GigE monitors
  - Permission details for this project not yet finalized, but should be achievable

*Resources, **Plans,** Timeline*

- Build proof-of-concept measurement system for botnets based on IRC communication (in progress)
- Develop abstractions for generic measurement of similar lower volume malicious traffic
- Develop algorithms and techniques for implementation on routers/measurement platforms

Questions?