USENIX SRUTI '05

Reducing Unwanted Traffic in a Backbone Network

Kuai Xu, Zhi-Li Zhang, and Supratik Bhattacharyya

July 7, 2005

The Unwanted Traffic Problem

- Unwanted traffic proliferates on the Internet
  - pose security threats, e.g., worms, scans, DOS
  - waste resources, e.g., bandwidth, space on SMTP servers

- Challenges for a transit backbone
  - large volumes of traffic, diverse hosts and applications
  - little (or no) knowledge about customer networks
  - customer satisfaction is paramount
    - minimize false positives, can not block vulnerable ports, etc.
  - need concise representation of filtering policies
    - Core routers support less than 10K ACLs

Filtering traffic in the backbone

- Why in the backbone?
  - A better vantage point for detecting “maltraffic”
  - Early filtering minimizes potential for harm, resource wastage
  - A value-added service for additional revenues or competitive edge

- Existing mechanisms
  - Customer premise solutions, e.g., IDS/IPS, firewalls
  - Unicast reverse path forwarding (uRPF) checks on ingress routers
  - Regional “scrubbing” centers for DDOS
  - Hand-crafted filters in response to specific events

Our focus

- Questions
  - How to identify unwanted traffic?
  - What are efficient and practical blocking strategies?

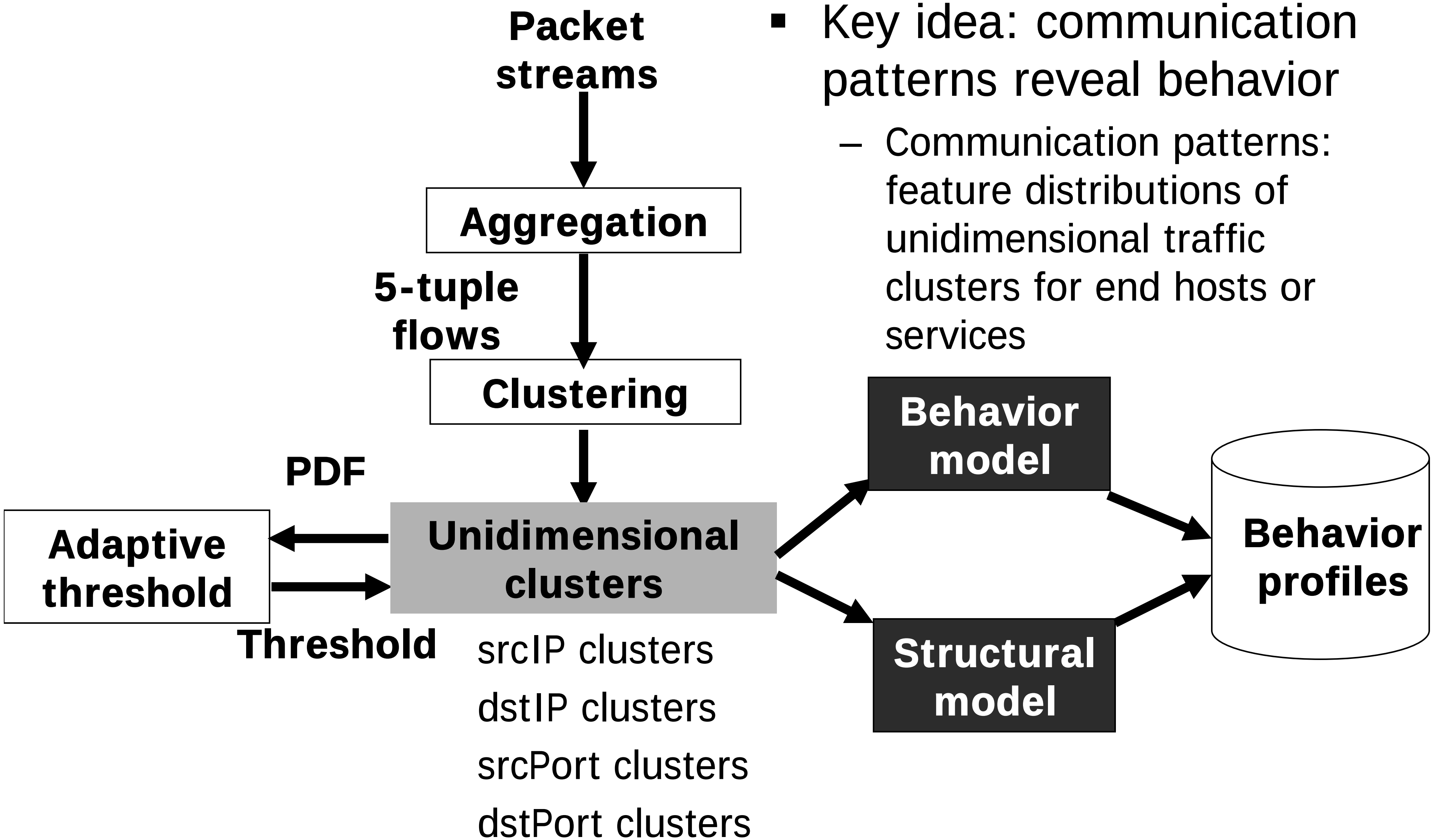
- Approach
  - Use backbone traffic profiling to identify sources of unwanted traffic
  - Devise simple blocking strategies based on the characteristics of unwanted traffic
  - Evaluate the cost/benefit tradeoffs of these strategies

Outline of this talk

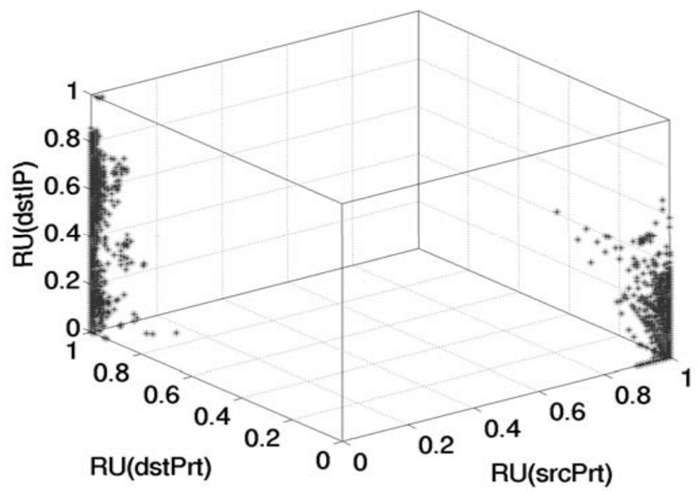
- Traffic profiling framework
- Simple blocking strategies
- Ongoing and future work

Traffic profiling framework

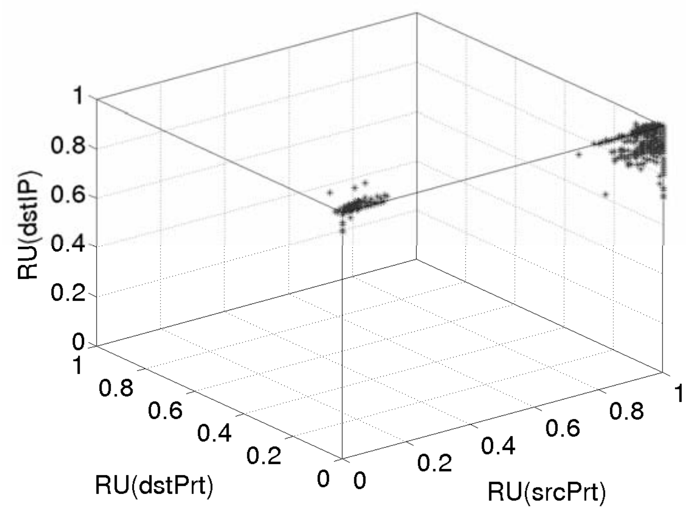
- Key idea: communication patterns reveal behavior
  - Communication patterns: feature distributions of unidimensional traffic clusters for end hosts or services



Canonical behavior profile



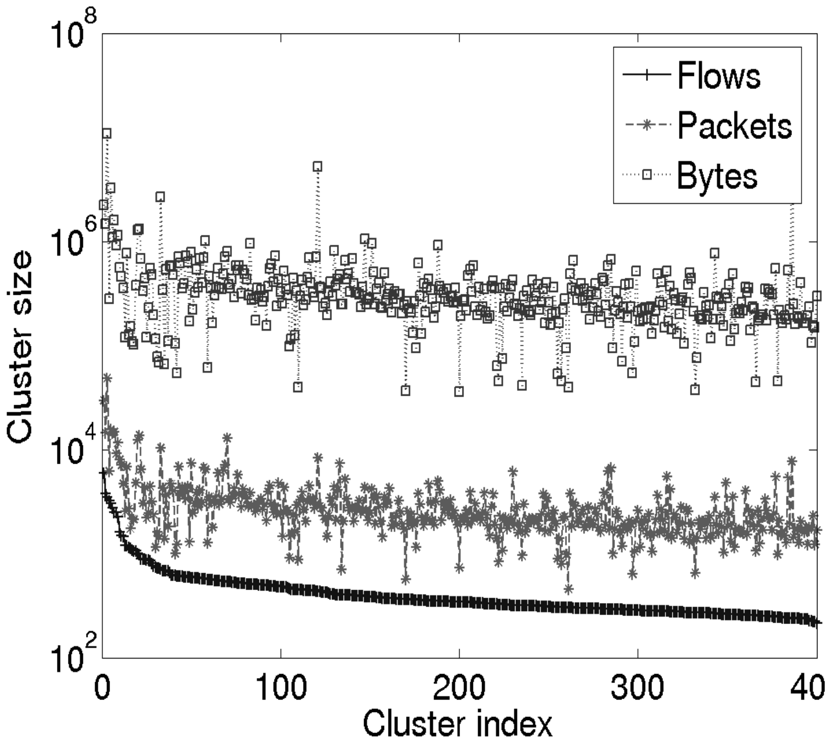
Server/client behavior

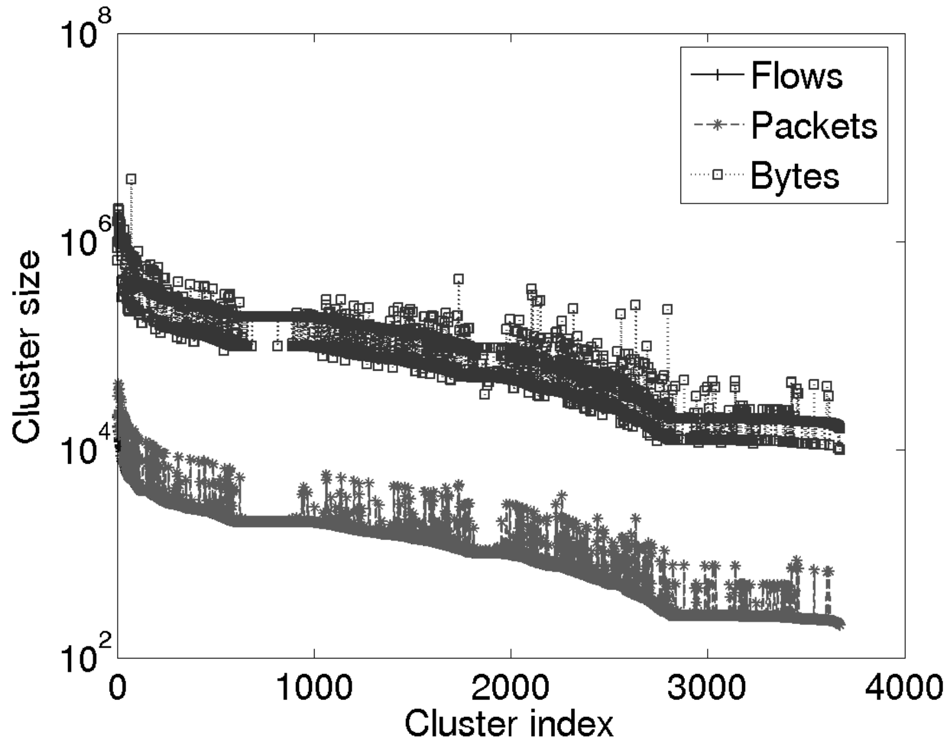


Exploit behavior

- Server/service behavior (low uncertainty on srcPort, high uncertainty on dstPort)
- heavy hitter client behavior profile (low uncertainty on dstPort, high uncertainty on srcPort)
- Scan/exploit behavior profile (low uncertainty on dstPort, high uncertainty on dstIP)

Additional flow features





srcIPs with server behavior profiles

srcIPs with exploit behavior profiles

Dataset

- Validate the framework using a diverse set of links from Sprint backbone network

- One link (L_1) as an example
  - Duration: 24 hours
  - Profiling done every 5-minute time slot
  - Total time slots: 288

- Identify sources with an exploit profile
  - 3728 (significant) srcIPs with exploit profile

Devising blocking strategies

- Objective
  - Reduce exploit traffic
  - Reduce threats and damage

- What factors to consider in a strategy?
  - Policies
    - whom to block: all or a subset of sources with exploit profile
    - what to block: all traffic or only traffic to exploit port
  - Mechanism
    - Route all srcIPs to null0/discard
    - ACL entries: <srcIP, dstPort>
  - Performance tradeoff

Performance Tradeoff

- Benefits of reducing unwanted traffic
  - Reduce potential threats of exploit traffic (hard to quantify)
  - Exploit traffic (flows, packets, bytes) reduction
- Cost: number of ACL entries created
  - An estimate of the actual cost incurred in ingress routers
- Wastage: ACL entries that are never invoked

Base rule

- Rule
  - Identify *srcIP* with an exploit behavior on *dstPort*
  - Create an ACL entry <*srcIP, dstPort*>
  - Apply the ACL entry for all future time slots
- Performance on the link L_1
  - Benefits: reduce 76% (exploit) flows, 71% packets, and 67% bytes from sources with exploit profile
  - Cost: 3756 ACL entries
  - Wastage: 1310 ACL entries (35%)
- ACL entries increase as the number of links monitored
  - Reduce the cost/wastage via selectively blocking
  - Can we learn from characteristics of unwanted traffic?

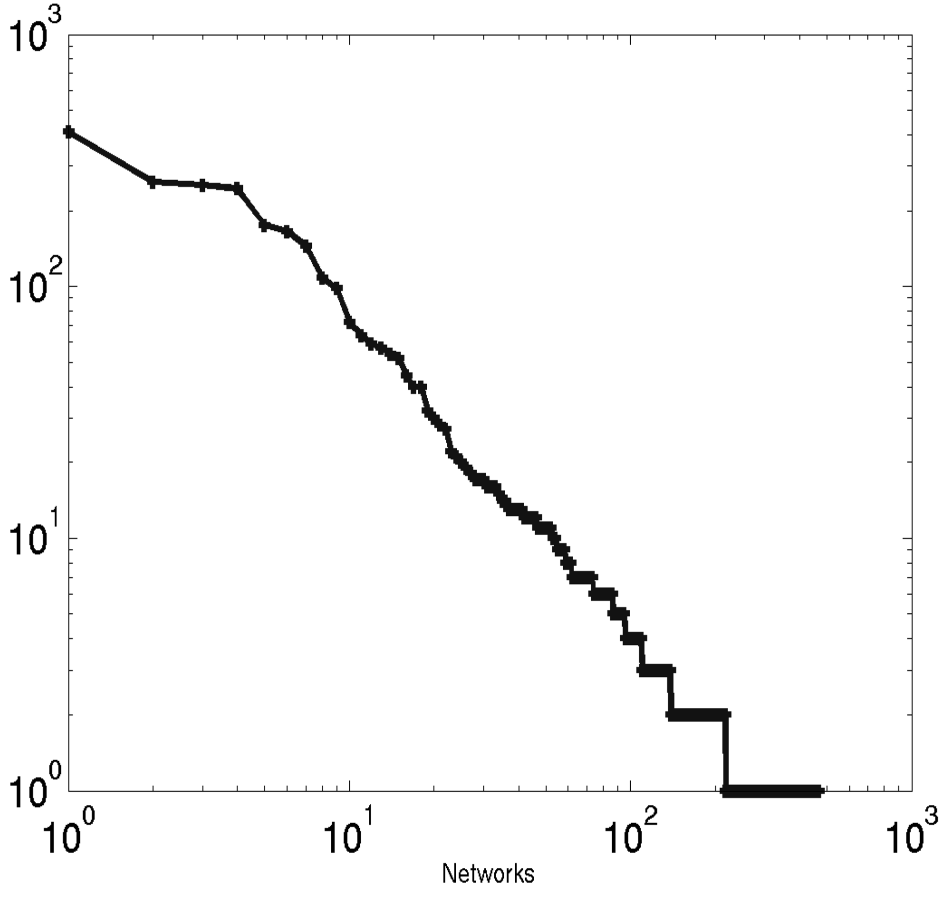
Characteristics of exploit traffic

- Source of exploit traffic
  – where are they from?

- Port of exploit traffic
  – What ports are exploited?

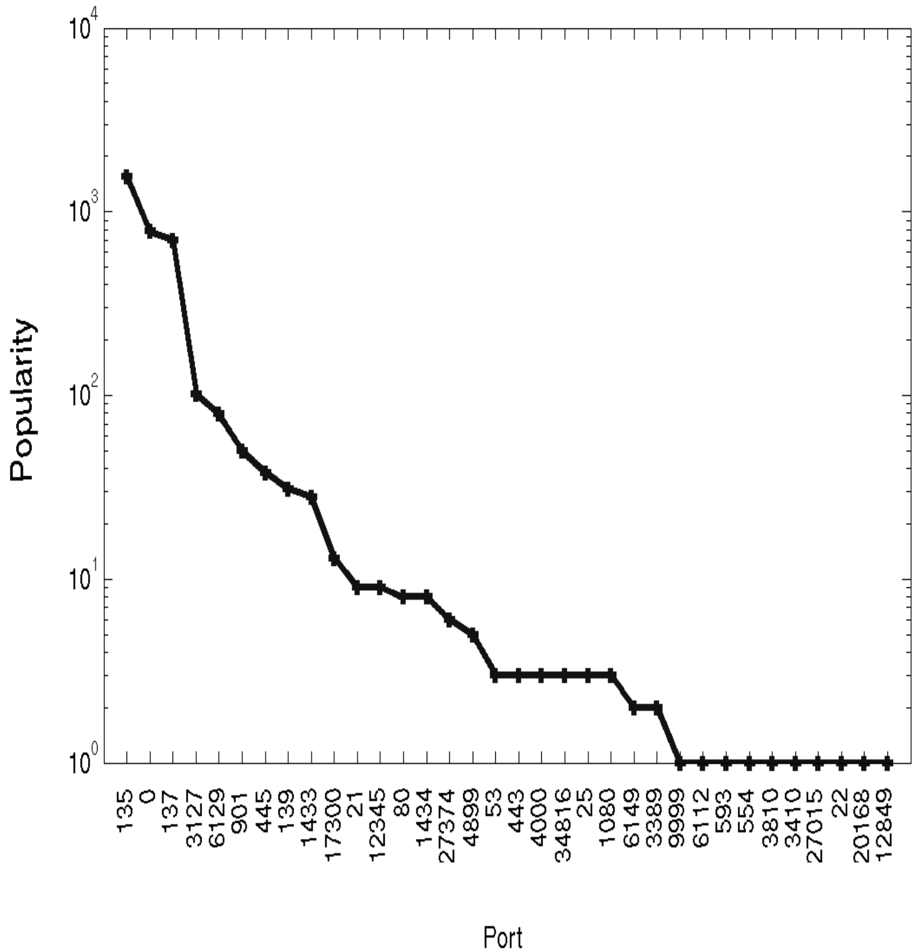
- Severity of exploit traffic
  – frequency: # of time slots of each source observed
  – persistency: # of consecutive slots (frequency > 1)
  – intensity: (average) # of targets touched per minute

Original ASes



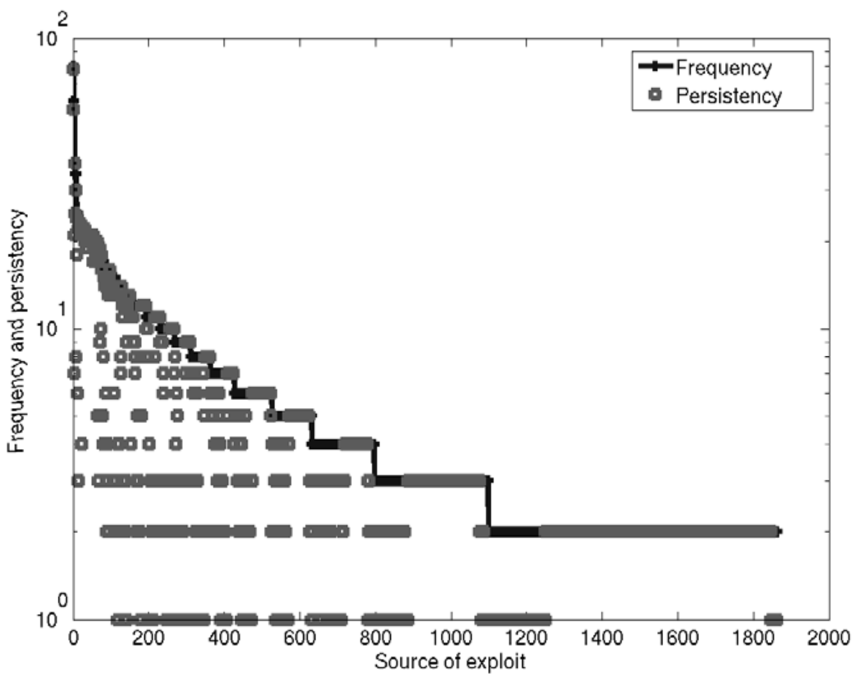
- Rule 1: Block srcIPs only from the top x ASes
- Performance (x = 10)
  - Benefits: 22% flows, 19% packets, 17% bytes
  - Cost: 1942 ACL entries
  - Wastage: 1071 (55%) ACL entries

Popular exploit port



- Rule 2: Block srcIPs only targeting the top k popular ports
- Performance (k = 5)
  - Benefits: 67% flows, 56% packets, 52% bytes
  - Cost: 3471 ACL entries
  - Wastage: 1216 (35%) ACL entries

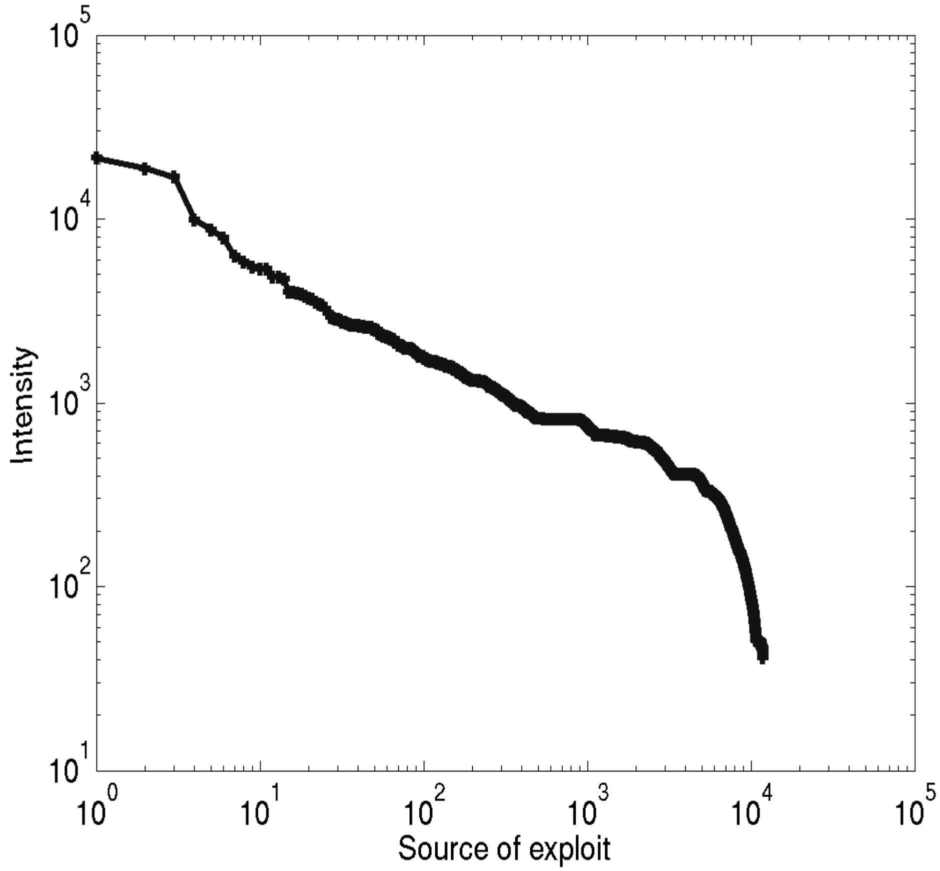
Frequency and persistency



- Rule 3: Blocking srcIPs with an exploit profile for at least n consecutive time periods
- Performance (n = 2)
  - Benefits: 48% flows, 43% packets, 37% bytes
  - Cost: 1586 ACL entries
  - Wastage: 505 (32%) ACL entries

- 1918/3728 srcIPs are profiled with the same exploit behavior more than once.
- 1370/1918 srcIPs are profiled for at least two consecutive time slots.

Intensity of exploit traffic



- Rule 4: Block srcIPs with at least m targets per minute
- Performance (m = 300)
  - Benefits: 64% flows, 57% packets, 48% bytes
  - Cost: 1789 ACL entries
  - Wastage: 302 (17%) ACL entries

Summary of blocking rules

Rule	**Heuristic**
Base rule	block every source with an exploit profile
Rule 1	block sources from the top *x origin ASes*
Rule 2	block source have an exploit profile with one of the *top k popular ports*
Rule 3	block sources have an exploit profile for at least *n consecutive periods*
Rule 4	block source have an intensity of at least *m targets per minute*

Summary of performance evaluations

Rule	Cost	Flow reduction	Packet reduction	Byte reduction	Wastage (%)
Base rule	3756	76.8%	71.1%	67.2%	1310 (34.8%)
Rule 1 (top 10 ASes)	1942	22.7%	19.5%	17.9%	1071 (55.1%)
Rule 2 (top 5 ports)	3471	67.1%	56.3%	52.1%	1216 (35.0%)
Rule 3 (2 consecutive time slots)	1586	48.4%	43.5%	37.9%	505 (31.8%)
Rule 4 (300 targets per minute)	1789	64.7%	57.2%	48.8%	302 (16.9%)

Ongoing/Future Work

- More concise filters
  - To what extent can we aggregate exploits sources with common prefixes?
  - Timing out ACL entries that are never or less used
  - Quantify threat reductions
- Develop a network-wide view across multiple links
  - Can we identify exploit activities not visible at any single link?
  - How does the number of exploit sources grow?
- Sequential behavior analysis
  - What is the communication patterns of a source before and after an exploit?
  - What is the collateral damage caused by blocking it?