PeerRush

Mining for Unwanted P2P Traffic

Babak Rahbarinia[a], Roberto Perdisci[a,b], Andrea Lanzi[c], Kang Li[a]

[a]University of Georgia

[b]Georgia Tech [c]EURECOM

University of Georgia
Dept. of Computer Science

Network Security Intelligence

Introduction

- P2P traffic represents a significant fraction of all Internet traffic
  - Apps: File Sharing, VoIP, P2P Botnets, …
- Net admins need to categorize traffic that crosses their network's perimeter
  - Detect malware infections related to P2P botnets
  - Identify/block some types of P2P traffic
- Identifying P2P traffic can aid Net-based IDSes

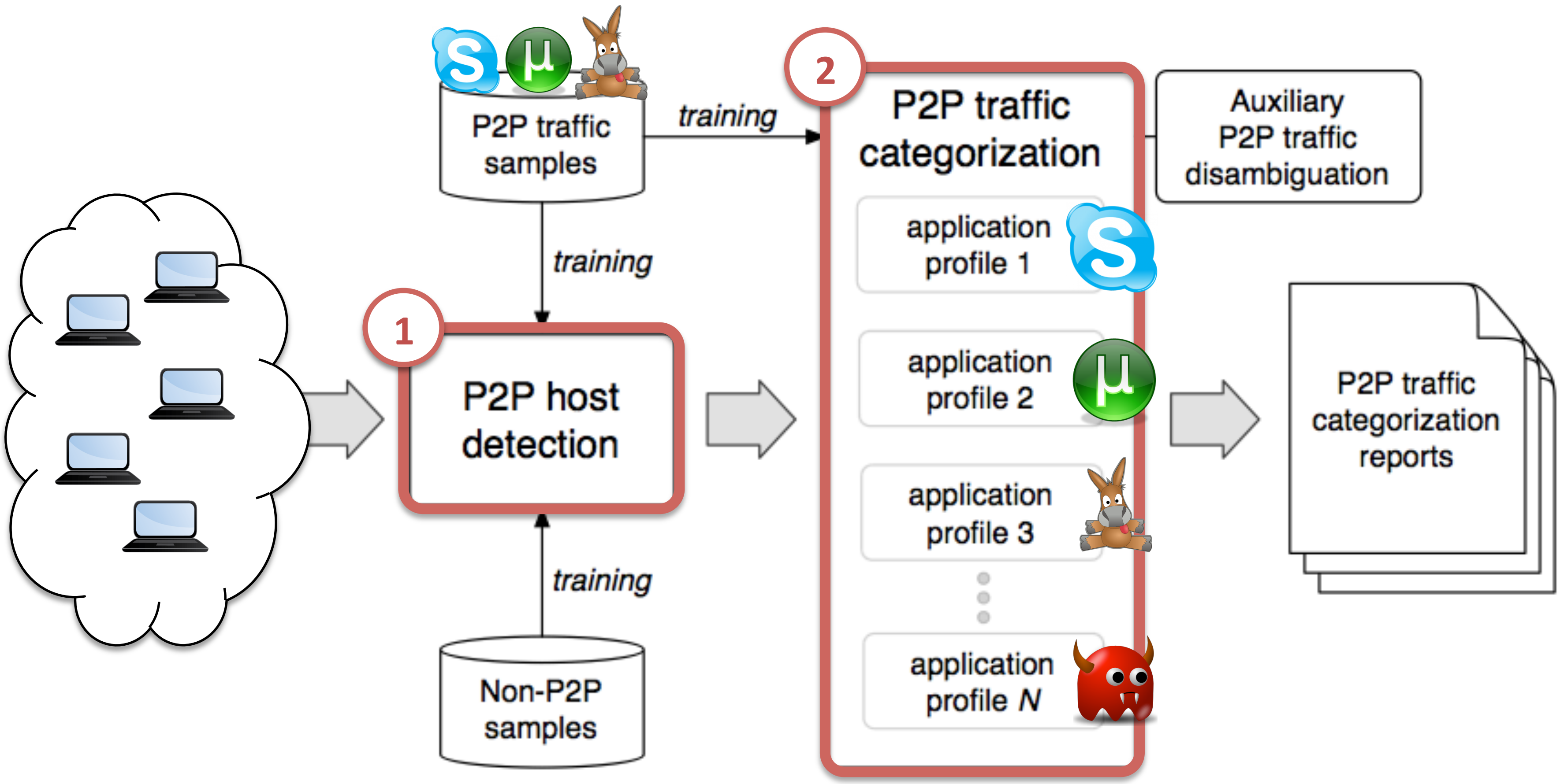
Previous Work

- Several papers on P2P traffic *detection*
  - Port numbers, Sig-based, DPI, statistical traffic analysis
- Very little research on non-sig-based P2P traffic *categorization*
  - Profiling P2P traffic (Hu et al., *Computer Networks'09*)
  - only applied to non-encrypted traffic, very few apps
- Some work on P2P botnet *detection*
  - BotMiner (Gu et al.), Statistical traffic fingerprints (Zhang et al.), Traders or Plotters? (Yen et al.), …
  - Cannot distinguish between different P2P botnets

PeerRush Goals

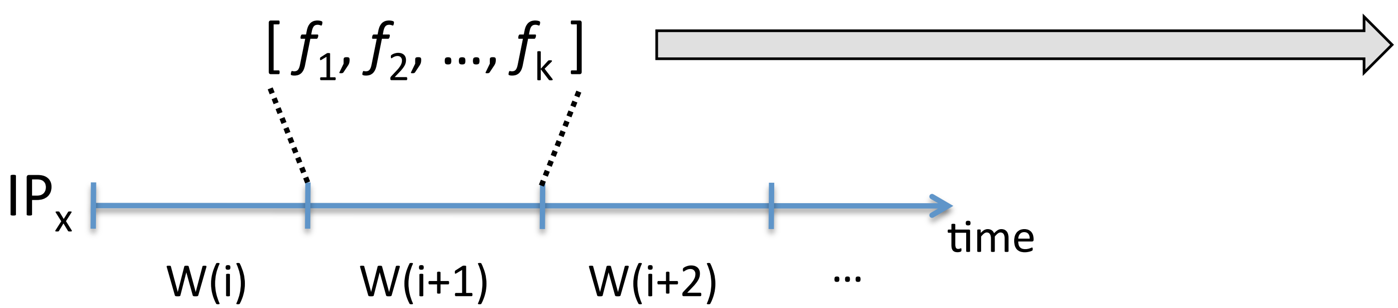
- Detect and ***categorize P2P traffic***
  - Generic/flexible traffic categorization approach
  - Statistical traffic features
  - Agnostic to payload encryption
- Identify ***unwanted*** P2P traffic
  - "unwanted" depends on network management and security policies
  - Includes malicious traffic, such as P2P botnets
  - May include other legit but unwanted apps, such as file sharing (eMule, BitTorrent, etc.)

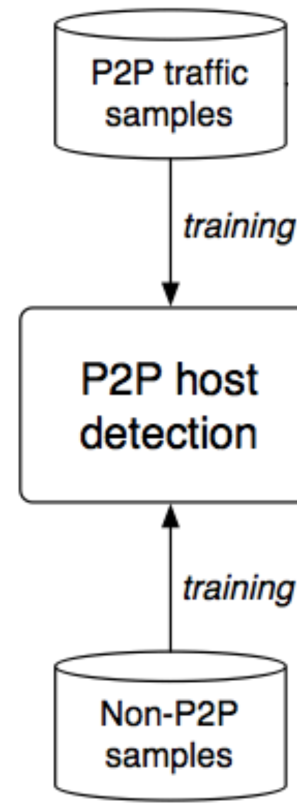
PeerRush: System Overview



P2P Host Detection - Overview

- Input: *live* network traffic
- Approach: statistical two-class classifier
- Output: IPs that generate P2P traffic





P2P Host Detection - Features

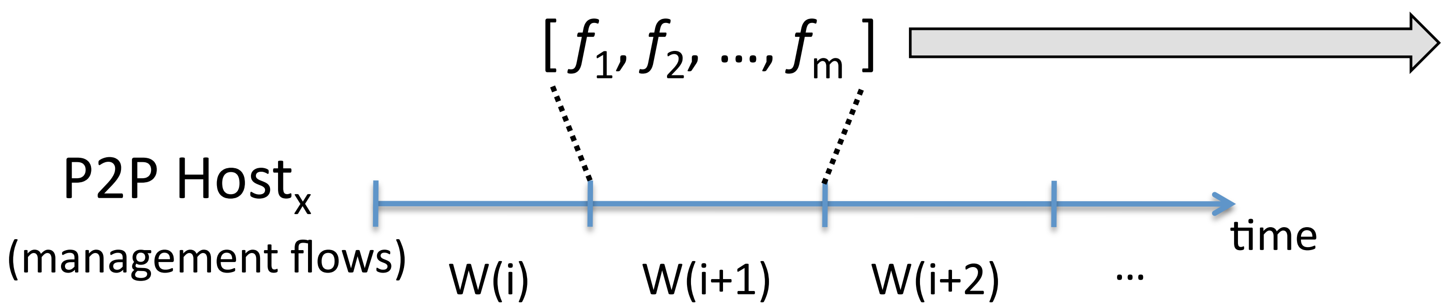
$[\, f_1, f_2, \ldots, f_k \,]$

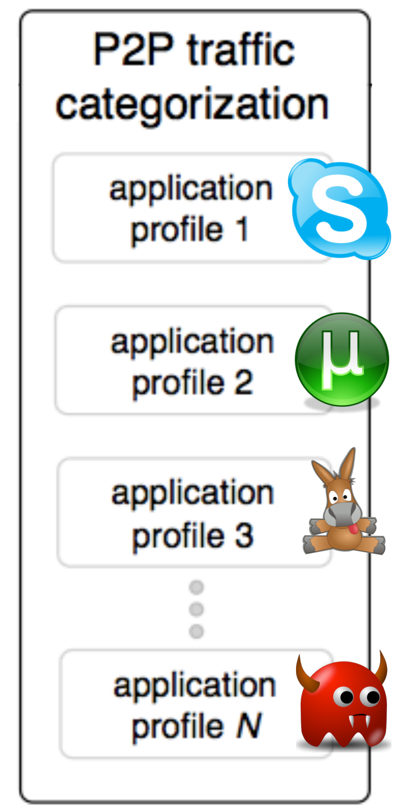
- Statistical features
  - # TCP/UDP “connections” with no DNS query
  - # failed connections (peer churn effect)
  - Non-DNS dst IPs scattered in many different networks
    - successful, failed, and all connections

- Non-P2P traffic has low feature values
  - e.g., web traffic
  - Most non-P2P connections “start” with DNS query
  - Only few failed connections

P2P Traffic Categorization - Overview

- Input:
  - traffic from each P2P host
  - *P2P management flows*
- Approach:
  - Application profiles modeled by one-class classifiers
- Output:
  - P2P traffic profile matches



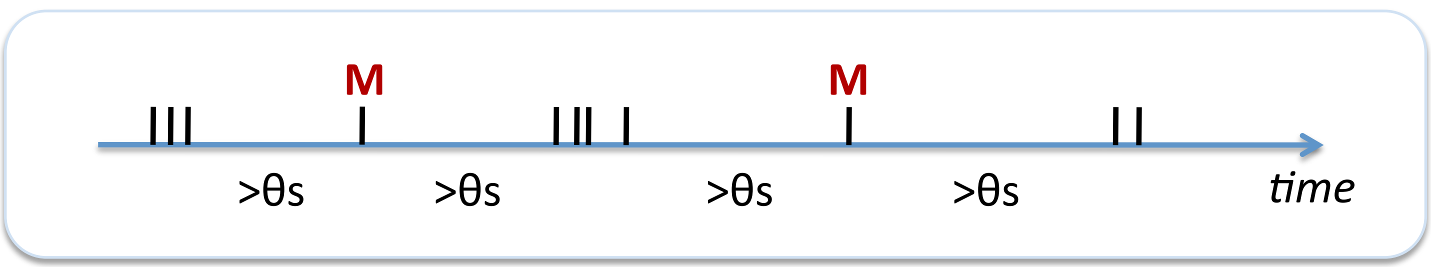


P2P Traffic Categorization - Features

- Different P2P apps generate different traffic
  - Use different P2P protocols
  - Connect to different network of peers
- P2P management (or control) flows
  - P2P traffic overall depends on user activities
  - need to find user-independent features!
  - better to focus on P2P control traffic
    - e.g., periodic "keep alive" messages
    - protocol-specific, more user-independent
- 1st goal
  - separate management flows from data flows

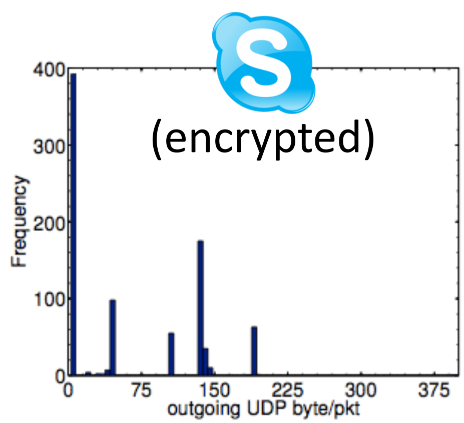
Finding Management Flows

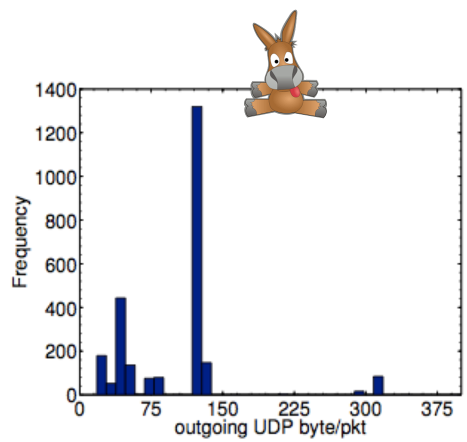
- Heuristics-based approach
  1. Consider only non-DNS flows
  2. Consider long-lived (TCP/UDP) flows
     - packet exchange for a significant portion of analysis window
  3. Leverage inter-packet delays
     - Data transfers typically involve bursts of packets
     - Management messages are exchanged periodically

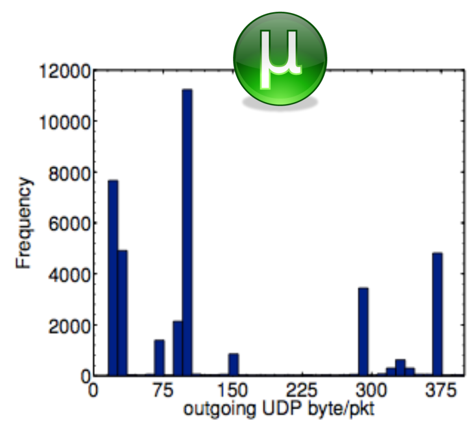


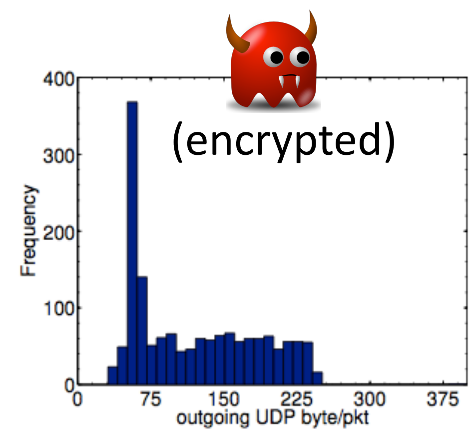
Management Flow Features

- Distribution of bytes per packet (BPP)
- Distribution of inter-packed delays (IPD)
  - Find top *n* BPP and IPD peaks
  - Measure peak location and relative height



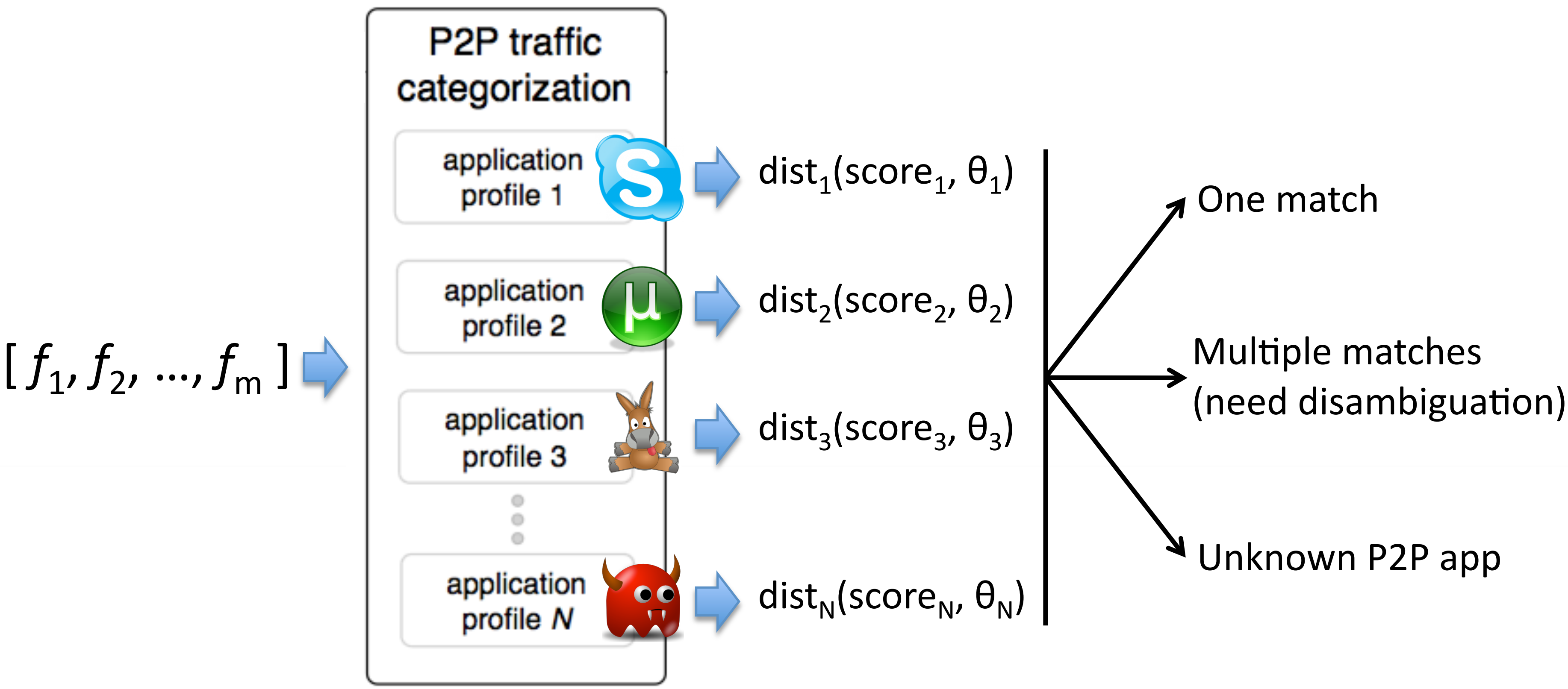






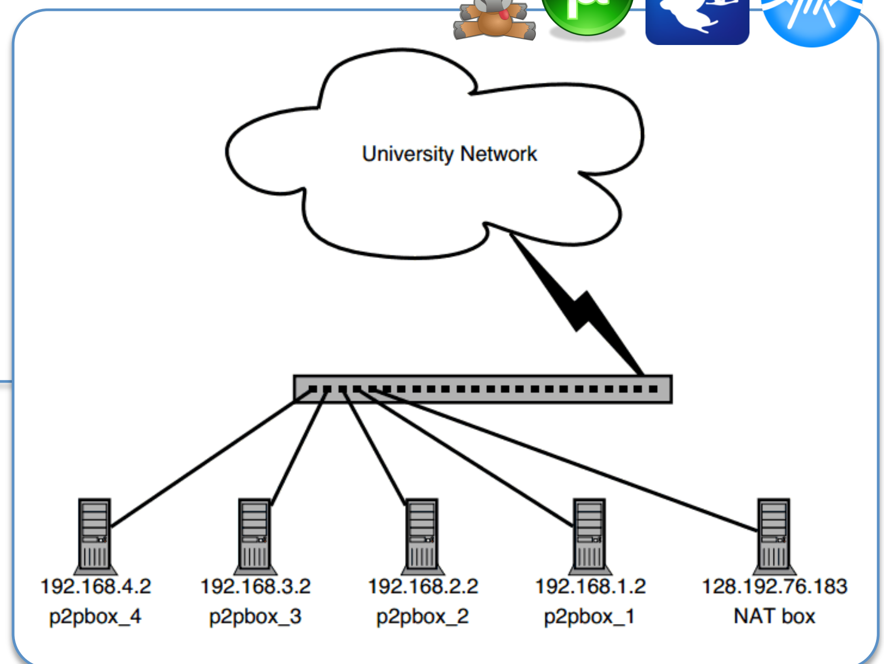
P2P App Profiles

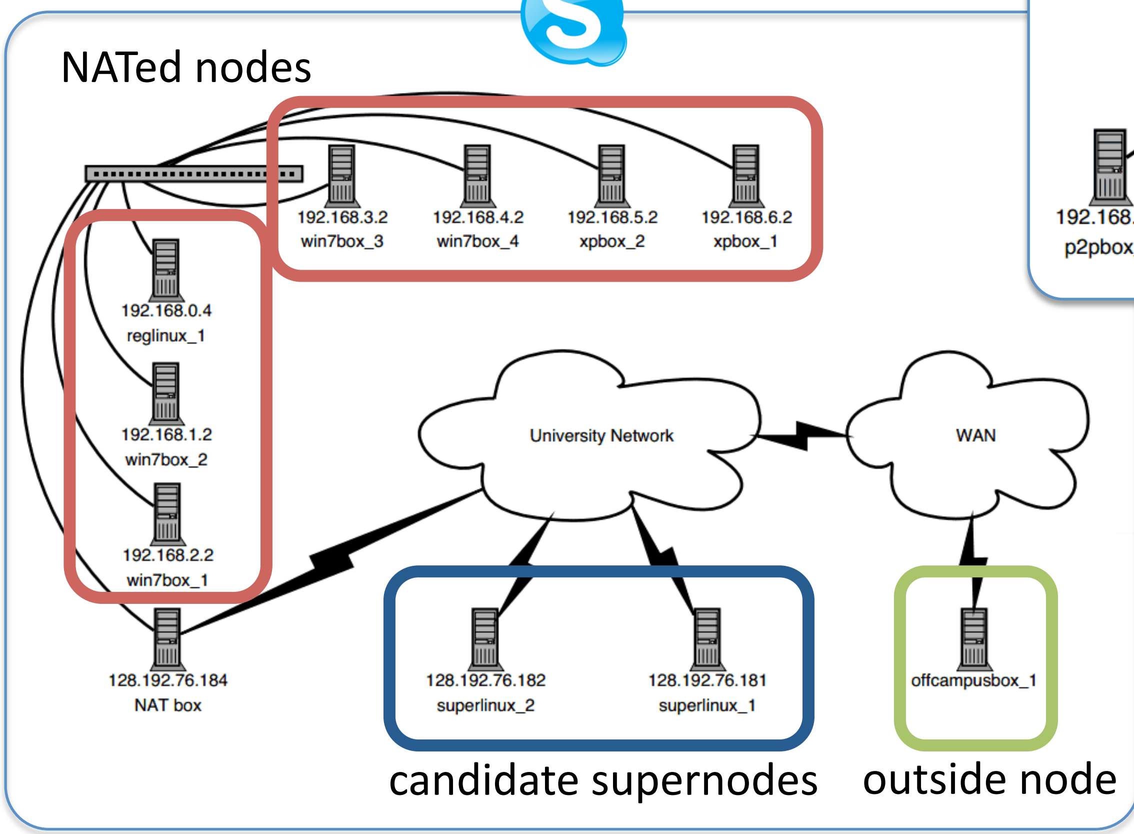
- One-class classification approach
  - Each traffic profile trained using only examples of traffic from target app
  - Flexibility: different decision function and threshold per each app



Evaluation Datasets

- 5 ordinary (non-malicious) apps
  - Several days per app
  - Hundreds of GB of traffic





Automated UI input

Evaluation Datasets

- Traffic from 3 real-world P2P botnets
  - Storm, Waledac, Zeus P2P (encrypted)
- Non-P2P traffic
  - about 5 days of CS dept. network
  - custom sniffing, anonymizes packets “on the fly”
  - pruned all src IPs that are suspected P2P hosts
    - any query to *.skype.com, any match of Snort P2P rules
    - 21 out of 931 hosts pruned overall

Eval of P2P Host Detection

- Cross-validation on non-malicious apps
  - Datasets: ordinary P2P traffic + non-P2P traffic
  - Classifier: Boosted Decision Trees

time window	TP	FP	AUC
60 min	99.5%	0.1%	1
40 min	99.1%	0.8%	0.999
20 min	98.4%	1.1%	0.999
10 min	97.9%	1.2%	0.997

- Separate "hold-out" test on P2P Botnets

Time Win.	Botnet	Instances	TPs	IPs detected
60 min	Storm	306	100%	13 out of 13
	Zeus	825	92.48%	1 out 1
	Waledac	75	100%	3 out 3
10 min	Storm	1,834	100%	13 out of 13
	Zeus	4,877	33.46%	1 out of 1
	Waledac	444	100%	3 out of 3

Eval of P2P Categorization

- App profile = one-class classifier
  - Different "optimum" classifier configuration per app
  - Cross-validation results

App.	#Inst.	Configuration	TP	FP	AUC
Skype	526	60min; KNN; 32 feat.; PCA	96.54%	0.74%	0.998
	579	10min; Parzen; 16 feat.; -	91.27%	1.00%	0.978
eMule	387	60min; Parzen; 16 feat; Scal.	90.64%	0.92%	0.989
	483	10min; KNN; 8 feat.; PCA	88.40%	1.16%	0.961
Frostwire	382	60min; KNN; 12 feat.; PCA	85.58%	0.96%	0.966
	467	10min; KNN; 8 feat.; PCA	92.68%	1.25%	0.989
μTorrent	370	60min; KNN; 8 feat.; -	92.94%	1.30%	0.948
	609	10min; Parzen; 4 feat.; Scal.	94.55%	1.24%	0.992
Vuze	376	60min; KNN; 8 feat.; -	91.92%	0.95%	0.979
	514	10min; KNN; 8 feat.; PCA	84.18%	1.17%	0.964
Storm	162	60min; Parzen; 16 feat.; -	100%	0%	1.000
	391	10min; Parzen; 12 feat.; PCA	100%	0%	1.000
Zeus	375	60min; KNN; 4 feat.; -	97.29%	0.99%	0.996
	188	10min; KNN;12 feat.; -	94.53%	0.79%	0.976
Waledac	37	60min; Gaussian; 12 feat.; PCA	99.99%	0.90%	0.998

Overall Eval of P2P Categorization

ordinary P2P

80% training

20% testing

Botnets

80% training

20% testing

time window: 60 minutes			
Application	**TP**	**FP**	**AUC**
Skype	100%	0.86%	1
eMule	93.59%	1.44%	0.9968
Frostwire	88.31%	0.97%	0.9873
μTorrent	96.97%	1%	0.9789
Vuze	93.1%	0.7%	0.9938
Storm	100%	0%	1
Zeus	96.69%	1.26%	0.9964
Waledac	57.14%	0.83%	0.9420

Classified as "unknown": 3.96% (29 out of 732)
Misclassified as other P2P: 0% (0 out of 732)
Disambiguation needed: 4.64% (34 out of 732)
· Correctly disambiguated: 33, Incorrectly disambiguated: 1
Total misclassified as other P2P: 0.14% (1 out of 732)

Conclusion

- PeerRush allows for flexible and accurate P2P traffic detection and categorization
- Enables detection of unwanted P2P traffic
  - different types of modern P2P botnets
  - unwanted "ordinary" P2P apps
  - agnostic to traffic encryption
- Extensive evaluation
  - 5 ordinary P2P apps + 3 modern P2P botnets
  - High accuracy of different system components
  - Promising results on robustness against traffic noise (results in the paper)

perdisci@cs.uga.edu

Overall Eval of P2P Categorization

ordinary P2P
80% training
20% testing

Botnets
80% training
20% testing

artificial noise
non-P2P traffic mixed to flows
+

time window: 60 minutes			
Application	**TP**	**FP**	**AUC**
Skype	90.4%	1.29%	0.9891
eMule	94.87%	2.39%	0.9935
Frostwire	94.73%	0.48%	0.9927
μTorrent	98.99%	0.66%	0.9997
Vuze	93.22%	3.02%	0.9873
Storm	45.45%	0%	0.7273
Zeus	97.32%	0.72%	0.9991
Waledac	40%	0.8%	0.8610

Classified as "unknown": 6.15% (45 out of 732)
Misclassified as other P2P: 0.68% (5 out of 732)
Disambiguation needed: 4.37% (32 out of 732)
· Correctly disambiguated: 32, Incorrectly disambiguated: 0
Total misclassified as other P2P: 0.68% (5 out of 732)

Evaluation Datasets

- 5 non-malicious P2P apps
  - Skype
  - µTorrent
  - eMule
  - Vuze
  - Frostwire
- 3 P2P botnets
  - Storm
  - Waledac
  - Zeus P2P