



# Detecting Flood-based DoS Attacks with SNMP/RMON\*

William Streilein,  
David Fried, Robert Cunningham

MIT Lincoln Laboratory

\*This work is sponsored by the U.S. Air Force under Air Force Contract F19628-00-C-0002.  
Opinions, interpretations, conclusions and recommendations are those of the author and are  
not necessarily endorsed by the United State Government.

MIT Lincoln Laboratory



# Outline

- ➔ • **Motivation and Goals**
- **Background: DoS and RMON**
- **Data collection and analysis**
  - **Using RMON to Detect DoS**
- **Test bed**
  - **Prototype Results**
- **Summary**



# Motivation and Goals

- **Motivation:**
  - **Denial of Service attacks continue to be a threat to networks and computers connected to the Internet: >4,000 attacks per week (src: CAIDA)**
  - **Commercial detection systems may not be best**
    - Proprietary, expensive**
    - Require network re-design**
    - Often use simple statistical models (i.e. threshold comparison)**
- **Goal:**
  - **Detect flow-based DoS using SNMP/RMON**
    - Utilize existing devices w/no changes to network, non-proprietary solution**
    - RMON1 commonly implemented on network devices**
  - **Develop improved detection models**
    - Compare uni-variate statistical to machine learning approach**



# Denial-of-Service Background

- **Denial-of-service: “the prevention of authorized access to a system resource or the delaying of system operations and functions” – CERT/CC, 2001**
- **Logic-based attacks exploit peculiarities or programming vulnerabilities in computer application or system software**
  - Usually targeted at individual machines
  - Vulnerability in popular software can lead to widespread impact
  - Examples: IIS DoS, Ping-of-death
- **Flow-based attacks exhaust resources available for normal use**
  - Network-based attacks which exhaust bandwidth on connected links
  - Examples: SMURF, Fraggle, Trinoo, Stacheldraht



# Flow-based DoS In the News

- **DoS a constant threat for Internet users:**
  - **1996, March: Panix Attack**  
TCP Stack peculiarity: SYN Flood causes service outage
  - **2000, February: '.com' attacks**  
Ebay, Zdnet, Amazon, etc. shutdown for hours: eCommerce is threatened
  - **2001, June: [www.grc.com](http://www.grc.com)**  
Script kiddies flood Gibson Research's T1s with ICMP, UDP traffic and bring site down
  - **2002, November: UltraDNS under DoS attack**  
Fills up two T1 pipes during peak
  - **2003, March: Uecomms AU link under DoS attack**
  - **4,000 DoS attacks per week (CAIDA, 2001)**
- **Denial-of-Service attacks are evolving**
  - **DDoS: Distributed sources, zombies**
  - **Automation (t0rnkit, ramen)**
  - **Amplification techniques in widespread use**
  - **Encrypted control channels, IRC (botnets)**
  - **New targets: Infrastructure devices (e.g. routers, hubs)**



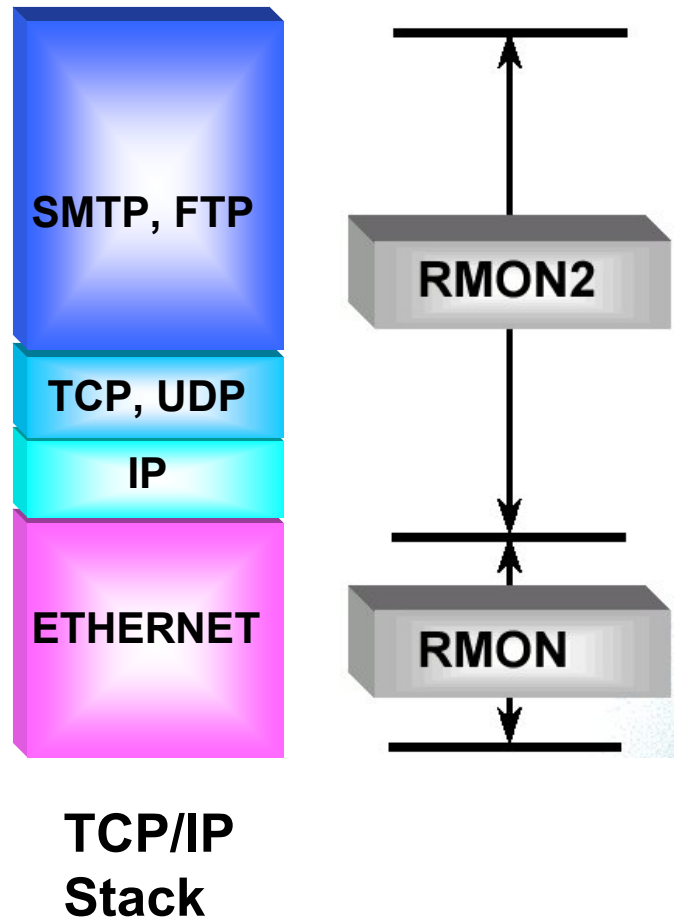
# State of the Art in DoS Detection

- **Passive:**
  - **Traffic analysis assistance**  
Asta Networks, Arbor Networks
  - **Rate threshold comparison, simple “statistical” approach**  
Many commercial offerings
  - **‘Backscatter’ algorithm**  
Detect response to spoofed packets w/ random src addrs
  - **Ramp-up signature, spectral analysis of arrival times,  $x(t)$**   
Discriminate between DoS and DDoS
- **Active:**
  - **RMON variables**  
IP-level RMON stats (i.e., above RMON1) were used to detect DoS
  - **Agent detection proactively scans network for (D)DoS agents**  
RID – Trinoo, TFN, stacheldraht  
Dds, gag, etc.  
Nessus



# RMON Management Information Base

- Remote Monitoring (RMON) is a standard monitoring specification that allows agents to report network info to managers via SNMP (Simple Network Management Protocol)



- RMON 2 monitors network and application layer
  - Monitor existing connections by address and protocol
  - Filter and capture specific traffic for later analysis
- RMON 1 monitors link layer
  - Count bytes, pkts, errors on segment
  - Set up proactive alarms to detect problems



# Approach

- **Collect SNMP/RMON data from live production network**
- **Analyze data, develop models of normal traffic**
- **Develop detection capability for simulated DoS attack**
  - **Superimpose simulated DoS traffic on collected data**
- **Create testbed to replicate traffic and stage DoS attack with real traffic**
- **Develop prototype DoS detection tool and test on testbed traffic**



# Outline

- **Motivation and Goals**
- **Background: DoS and RMON**
- **Data collection and analysis**
  - Using RMON to Detect DoS
- **Test bed**
  - Prototype experimental Results
- **Summary**

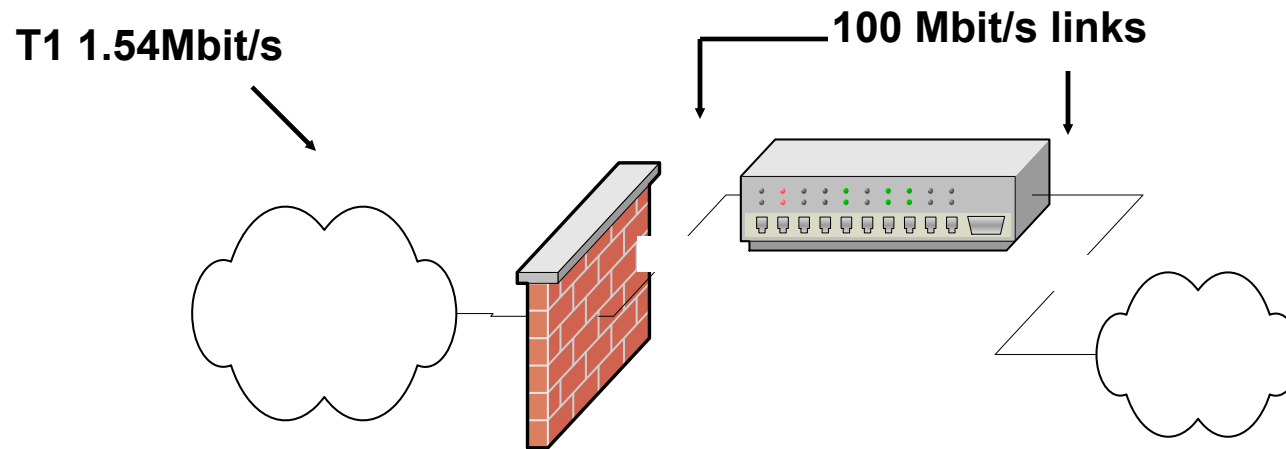


# Data Collection

- **Goal: use existing network infrastructure to detect *flow-based* DoS attacks**
  - Collect data from CISCO 2900 switch in production network
  - Gain familiarity with RMON1 statistics
  - Develop models of day-to-day traffic for purpose of recognizing anomalous behavior
  - Use data as guideline for prototype construction
- **RMON 1 Statistics automatically collected from switch**
  - Data sent daily for analysis



# Network Diagram (Simplified)



- **Internal network behind switch, behind firewall**
- **Switch controls traffic to/from Internet**
- **All traffic to/from Internet limited by T1 link**



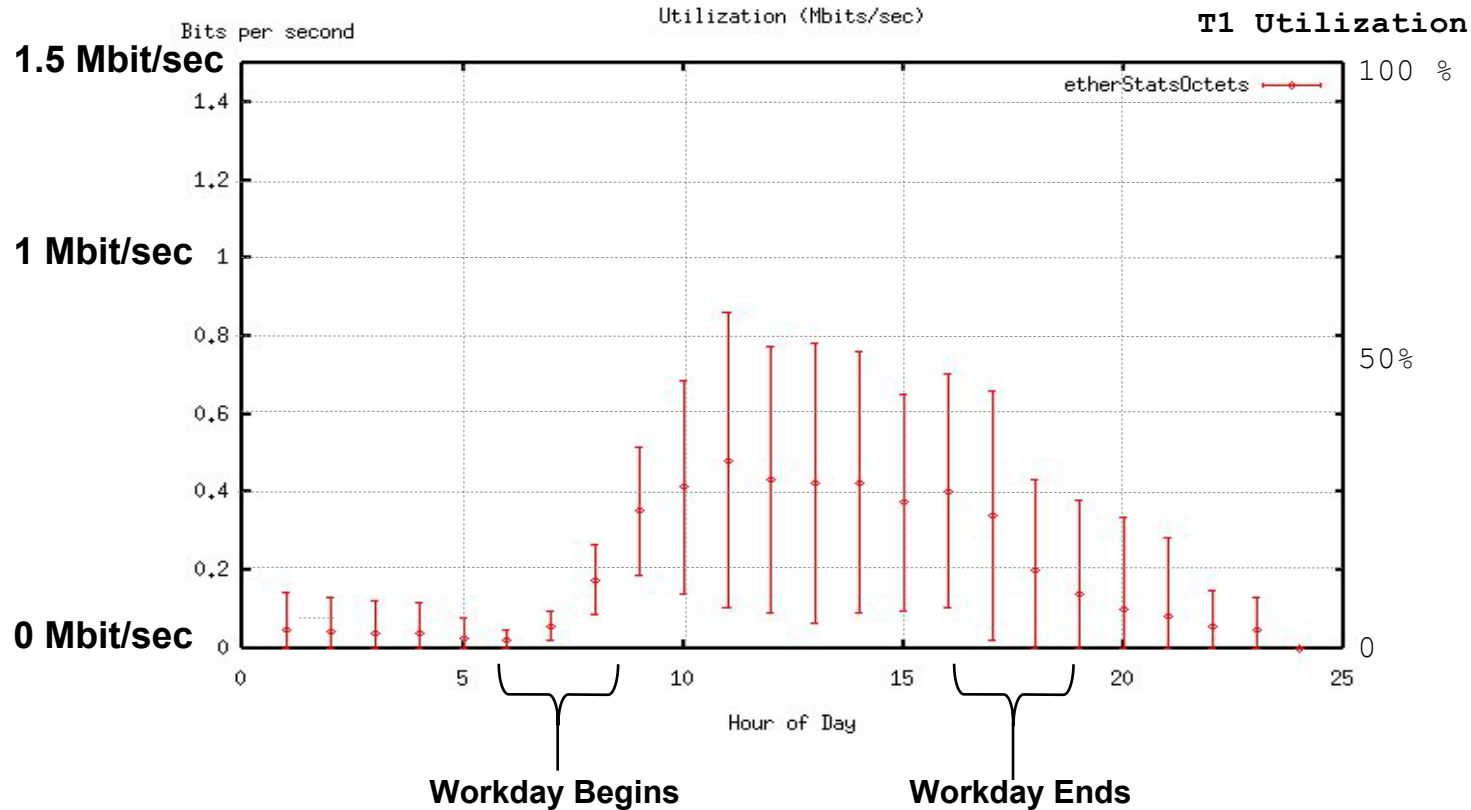
# Data Analysis

- **Data represents snapshots of RMON 1 variables**
    - Statistics group samples every 5 minutes from Oct, 2001 - Jan, 2002
    - CISCO 2900, 24 100 Mbit/sec ports
    - Ports under observation: 3 (to Internet), 24 (internal)
  - **Observations**
    - Daily traffic pattern clearly evident data (weekends and holidays excluded)
    - Packet ratios reflect typical communication with web-based servers
    - Network utilization on key ports < 1% of 100M capacity
- **Analysis of 52 days of switch data suggest DoS attacks can be detected as anomalous when compared to models of typical traffic**



# Measured Daily Traffic Pattern and Network Utilization

Packet counts seen on Port 3 (Workday Average)

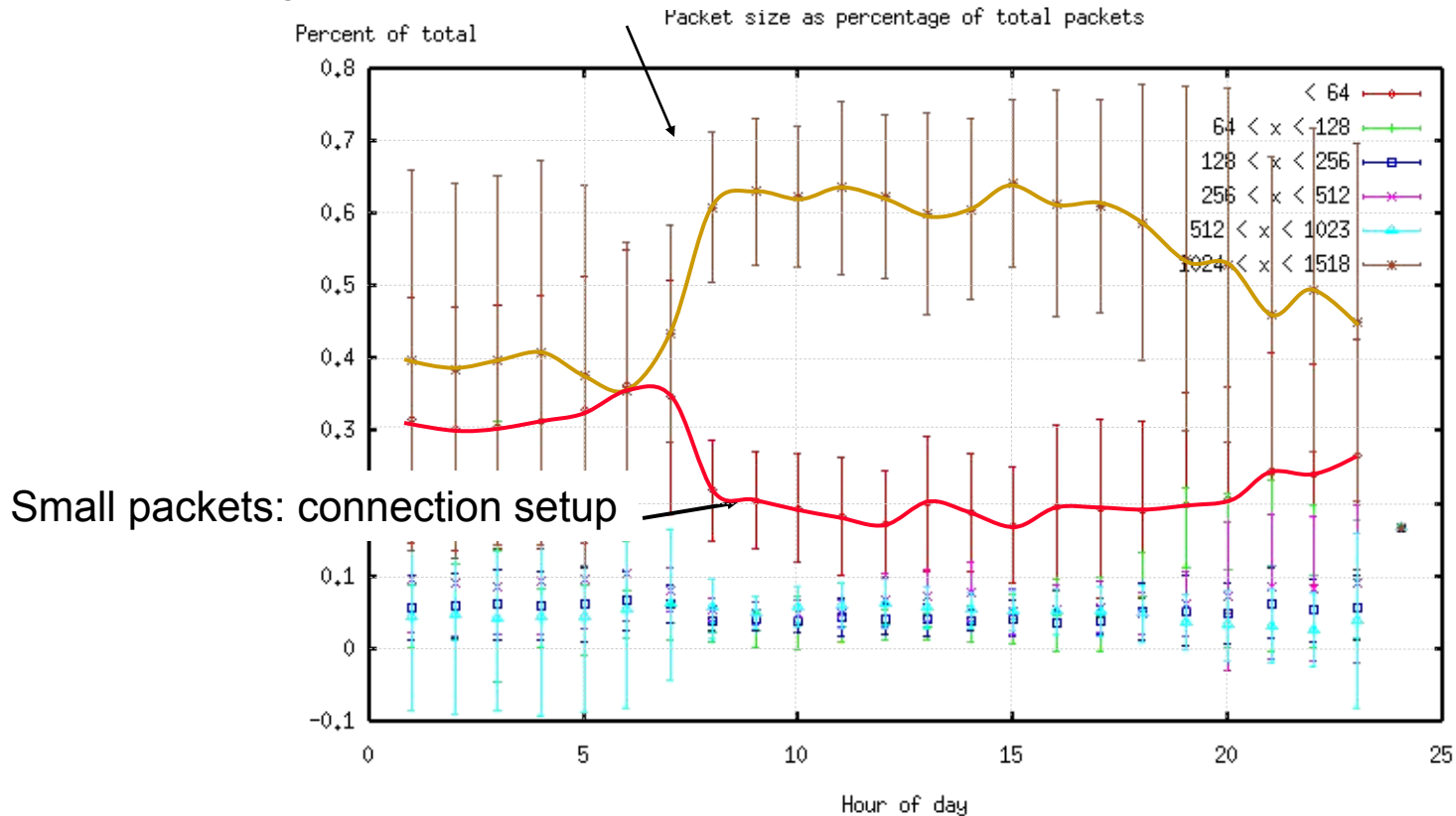


- Weekends and holidays excluded from analysis



# Measured Daily Traffic Pattern: Packet Size Ratios

## Large packets represent data packets      Port 3 – To the Internet



- Connection setup accounts for 20% of traffic
- Data packets from server account for 60% of traffic
- Together they account for 80% of traffic



# Detecting DoS with RMON Variables

- **Octet count**
  - Indicates network utilization
  - Intuition: Deviation from model of network utilization implies DoS
- **Packet size ratios**
  - Small packets represent client data requests
  - Large packets represent server replies
  - Intuition: Deviation from expected ratio of large/small pkts to total is indicative of something amiss (e.g. DoS)
- **Error variables**
  - Alignment error
  - Collisions
  - Fragments
  - Undersized packets (runts)
  - Intuition: Increase in error counts indicates communication problem, hence, DoS
  - **NOTE: no traffic errors were seen by RMON in collected data**

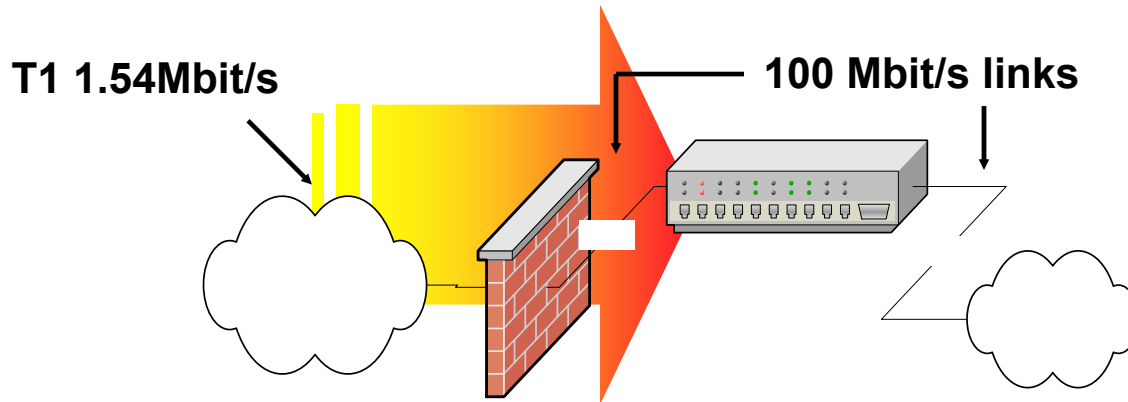


# Two Detection Models

- **Simple Statistical Model**
  - Commonly used by commercial systems
  - Univariate Gaussian assumption described by  $\mu$ ,  $\sigma$
  - Input feature: network utilization as % of available bandwidth
  - 288 separate models (every 5 minute period during day)
  - Anomalous traffic is greater than 3  $\sigma$ 's from  $\mu$
- **Machine learning model**
  - Multilayer perceptron : 8 input, 5 hidden, 2 output (normal, anomalous)
  - Enhanced feature vector
    - Packet bin ratios: (e.g, # of 64 byte packets / total packets, etc.)
    - Utilization, as % of available bandwidth
    - Time, as fraction of full day



# Experiment #1: Detect Superimposed DoS-like Traffic

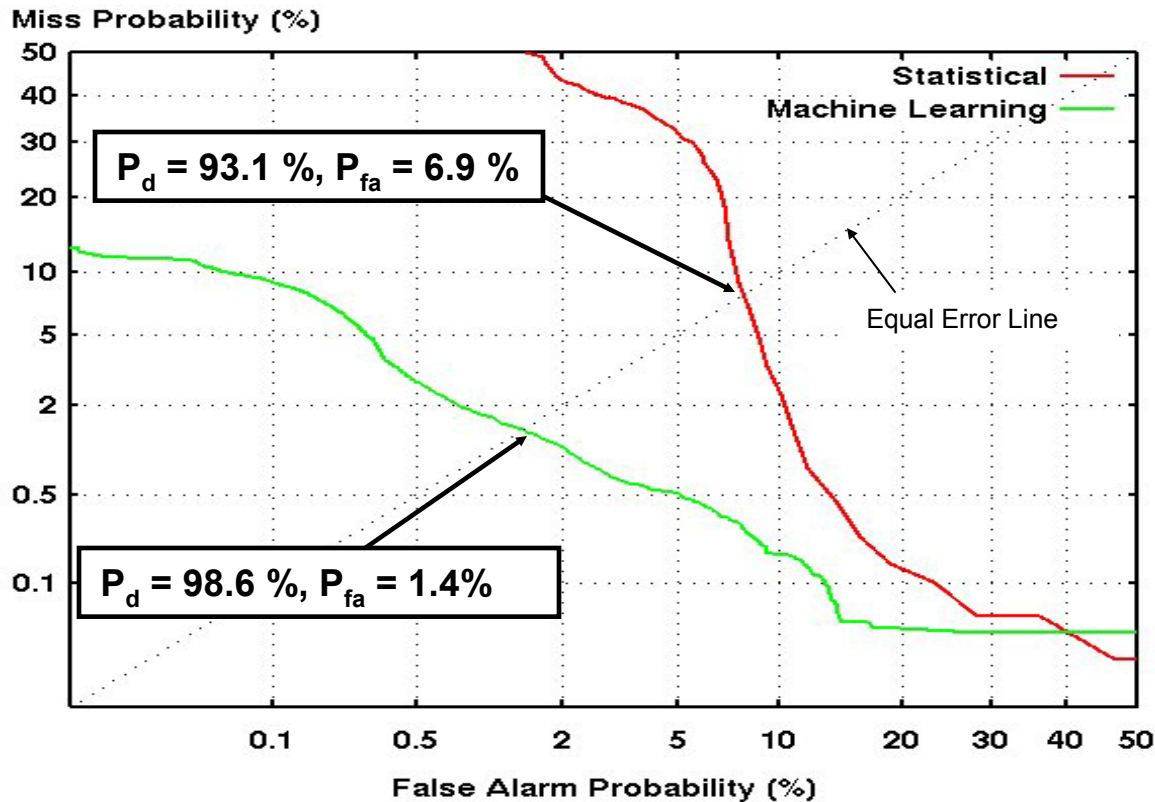


- **Impose DoS-like traffic utilization on data from Port 3**
  - **Attack Plan: Flood Internet T1 link, degrade Internet service**  
Add-in  $\frac{1}{2}$  T1 bandwidth (750kbits/sec), vary packet sizes
  - **Detection Plan: detect anomalous behavior when RMON variable counts (utilization, pkt count, etc.) exceed expected values**



# Statistical vs. Machine Learning

## Port 3 – To the Internet



- Assume DoS of 50% of T1 bandwidth added randomly to each bin
- MLP achieves better detection with lower false alarm rate (4 FA/day)



# Survey of Dos, DDoS Pkt Sizes

DoS	Pkt Size	Mechanism
Smurf	84	ICMP Echo Storm
Fraggle	84	UDP Echo Storm
Neptune	60+	TCP SYN Flood
★ Bzs	1460	Stream zeros in packets

DDoS	Pkt Size	Mechanism
Trinoo	1000	UDP Packet Flooding
TFN	1000+	UDP Packet Flooding
Stacheldraht	1000+	ICMP/UDP/TCP Flooding
TFN2K	1000+	ICMP/UDP/TCP Flooding
★ Mstream	60+	TCP Packet Flooding

- D/DoS pkts appear to be either large ( $> 1000$ ) or small ( $< 100$ )
- Caveat: Can be changed by attacker
- ★ - Simulated for detection

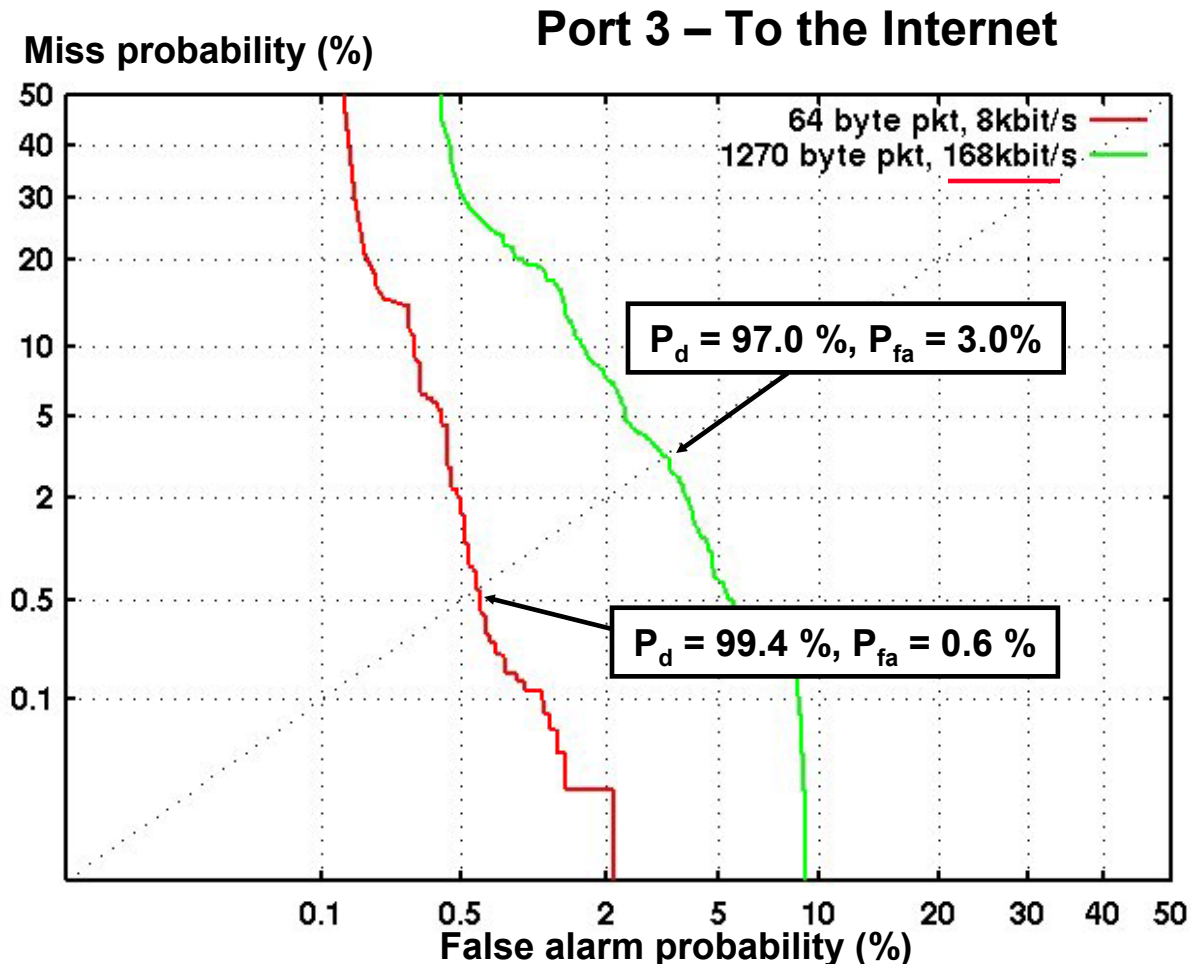


# Experiment #2: Detect DoS with Different Packet Sizes

- **Given DoS attack with a specific packet size which RMON packet bin will the DoS traffic be counted in?**
  - Use machine learning detector, MLP
- **Simulate two representative DoS attacks:**
  - Binary Zero has 1460 byte packets and will get counted in 1023 – 1518 packet bin
  - Mstream has 60 byte packets and will go in 64 packet bin
- **How well does detection algorithm work when DoS is in these bins alone?**
  - Simulate 64-byte packet DoS
  - Simulate 1270-byte packet DoS



# Machine Learning Detection Results for 64 and 1270 byte packets



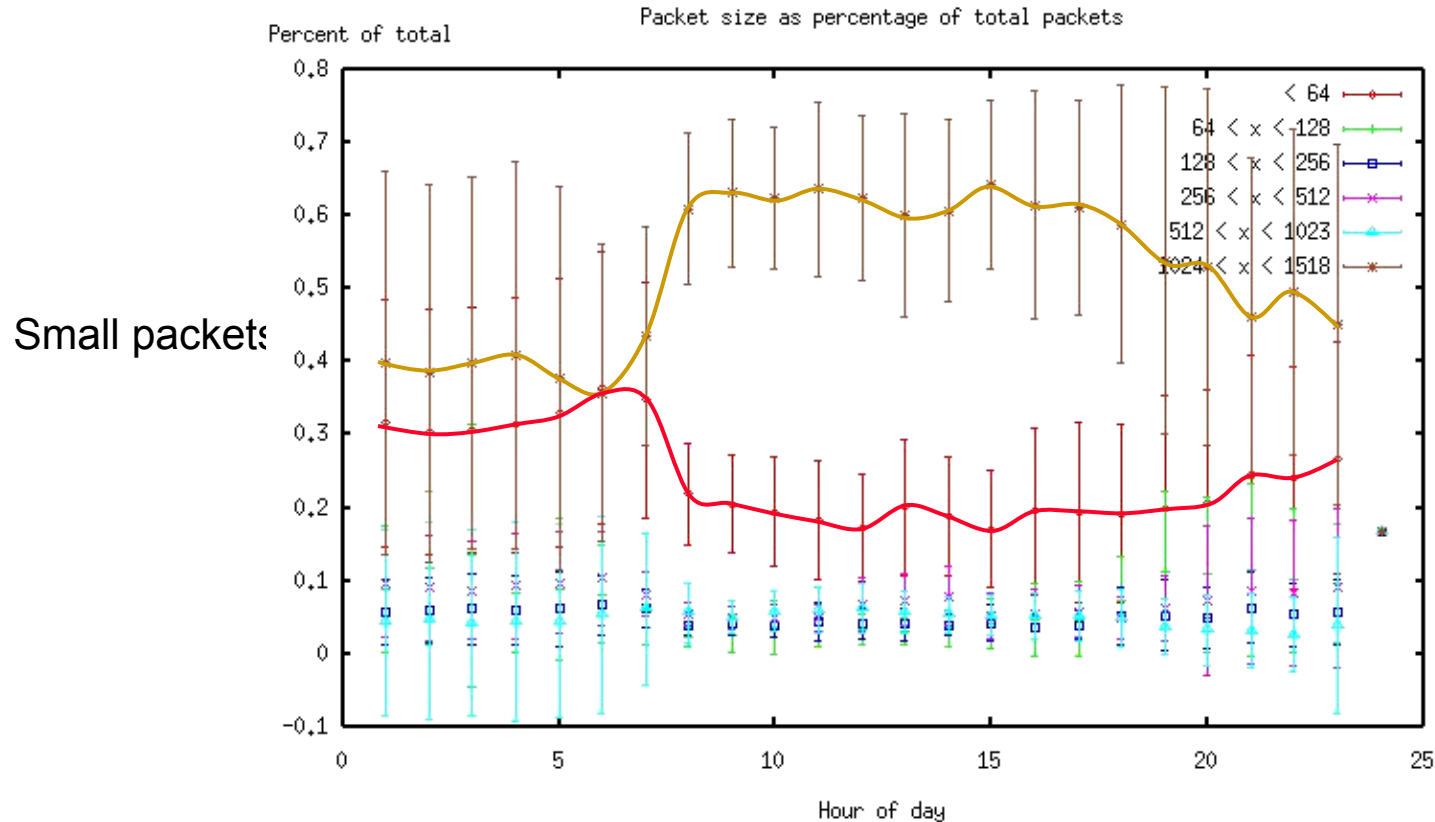
- For 64 byte pkts, DoS detected well at very low data rates
- Much more DoS traffic needed if 1270 byte packets used



# Explaining Detection Difficulty w/ Large Packet DoS

Large packets represent data packets(?)

Port 3 – To the Internet



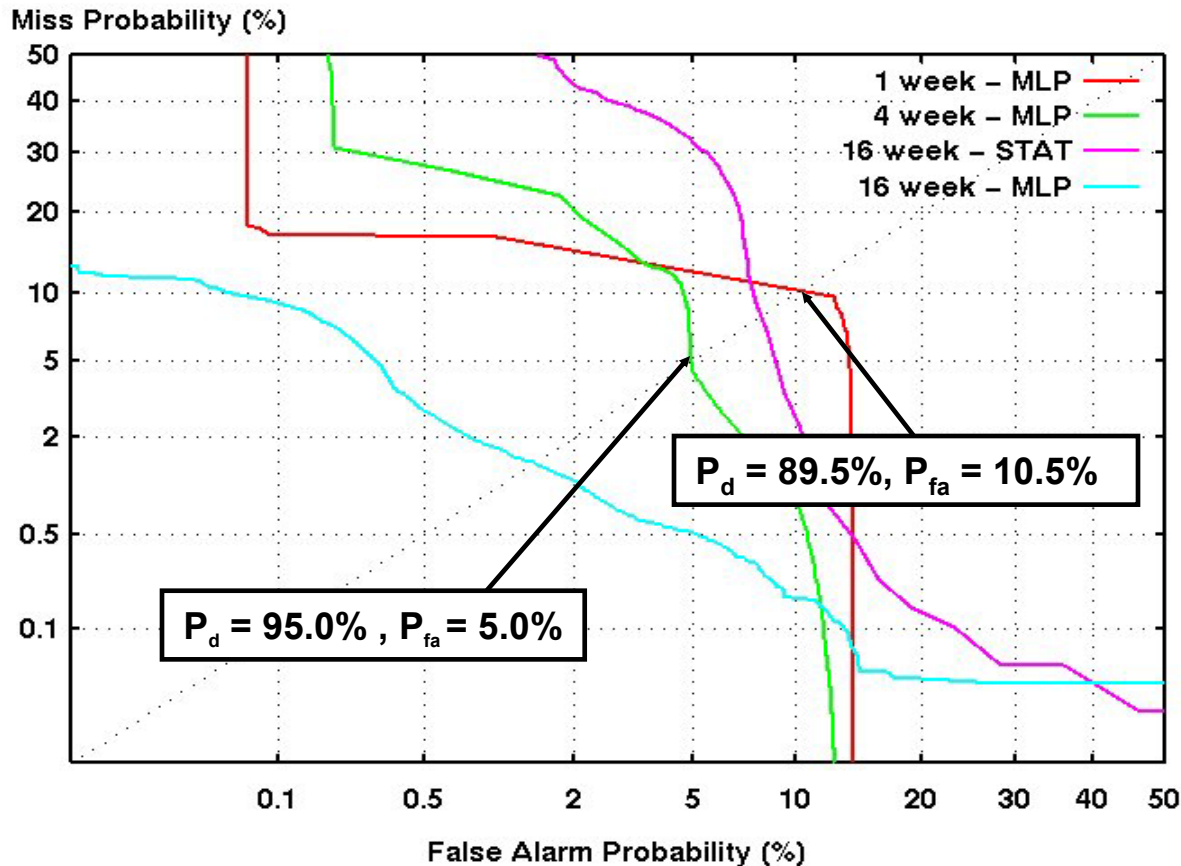
- When most of traffic is 1023-1518 packets, added DoS traffic doesn't change ratio of large packets to total count
- Utilization becomes important detection feature



# Experiment #3

## How much training data is necessary?

### Port 3 – To the Internet



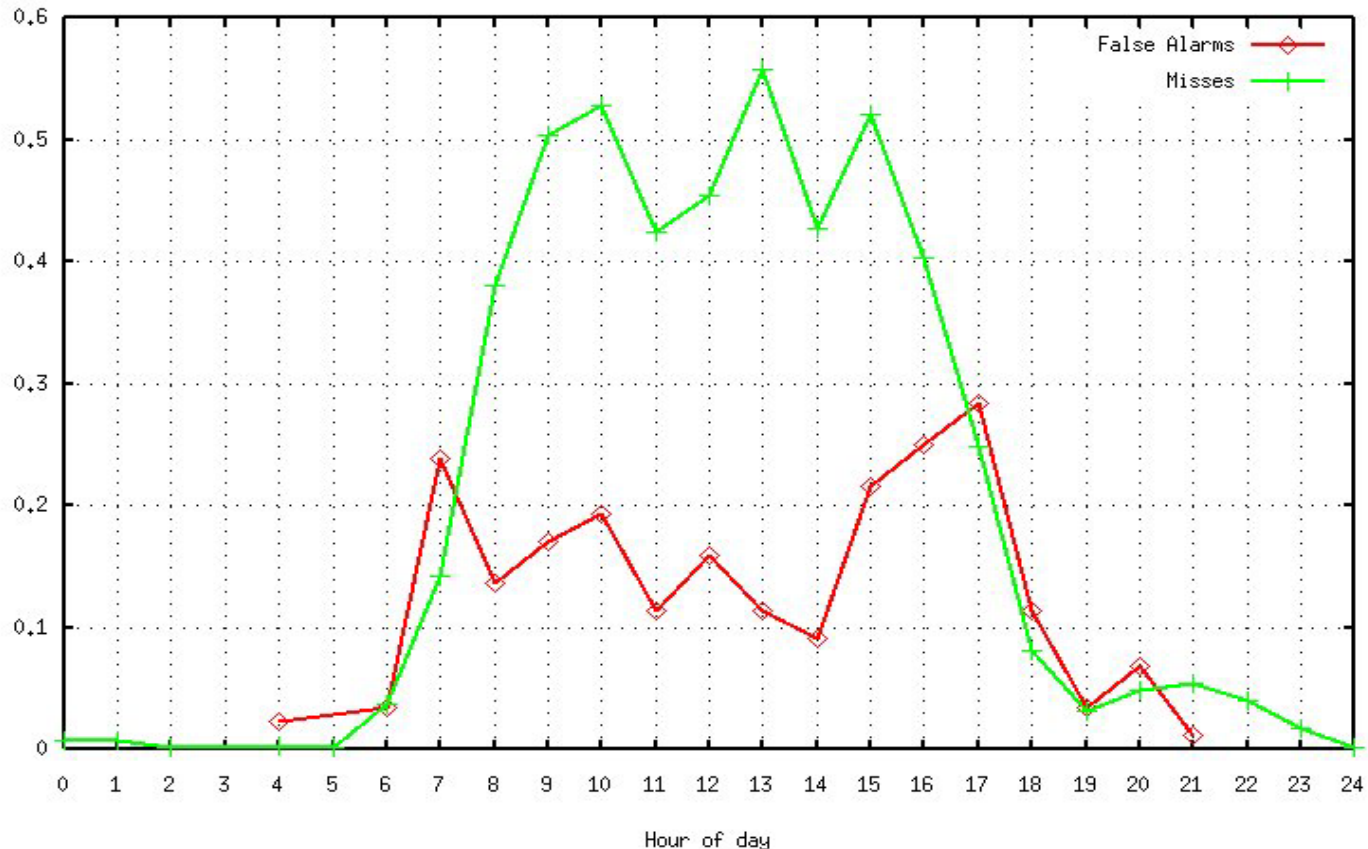
- With 1 month's training data, detection is better than statistical model
- Could imply short training time before deployment



# Experiment #4

## When do misses/false alarms occur?

Expected number of occurrences per hour



- False alarms and misses occur during heavy traffic periods of the day
- Misses can occur throughout day



# Simulation Comments

- **Statistical model detection results 'ok'**
  - Detection rate is good, but FA Rate is still too high (24/day)
- **Machine learning method yields superior results**
  - Multi-layer perceptron detector achieves very high detection rate with very low false alarm rate (4 /day)
  - DoS of small pkts (64 byte) more easily detected than DoS of large packets (1270 byte)
  - On-line learning facilitates deployment, simpler training
- **DoS response can be mediated by detection mechanism**
  - Localizing the source of attack via network map guides router/switch control to stem attack upstream
  - This information contained in RMON2



# Outline

- **Motivation and Goals**
- **Background: DoS and RMON**
- **Data collection and analysis**
  - Using RMON to Detect DoS
- **Test bed**
  - Prototype Results
- **Summary**





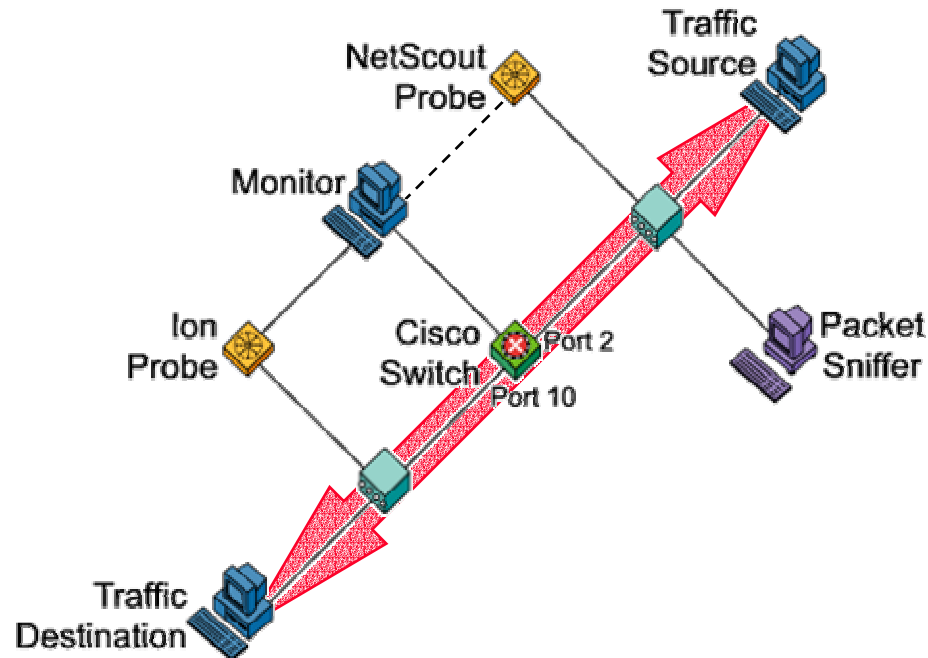
# Test DoS Detection Prototype on Live Traffic

- **Assembled testbed**
  - Utilize CISCO 2900 switch, RMON probes (NetScout, ION)
  - Create background traffic to mimic collection environment
  - Network speed limited to T1 rates
- **Collected SNMP data from RMON probes**
  - Poll every minute
- **Staged two DoS attacks**
  - Binary Zero (Windows Attack)
  - Mstream (DDoS)
  - Approach %50 of network bandwidth

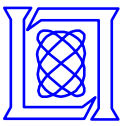


# Testbed Configuration for Prototype Testing

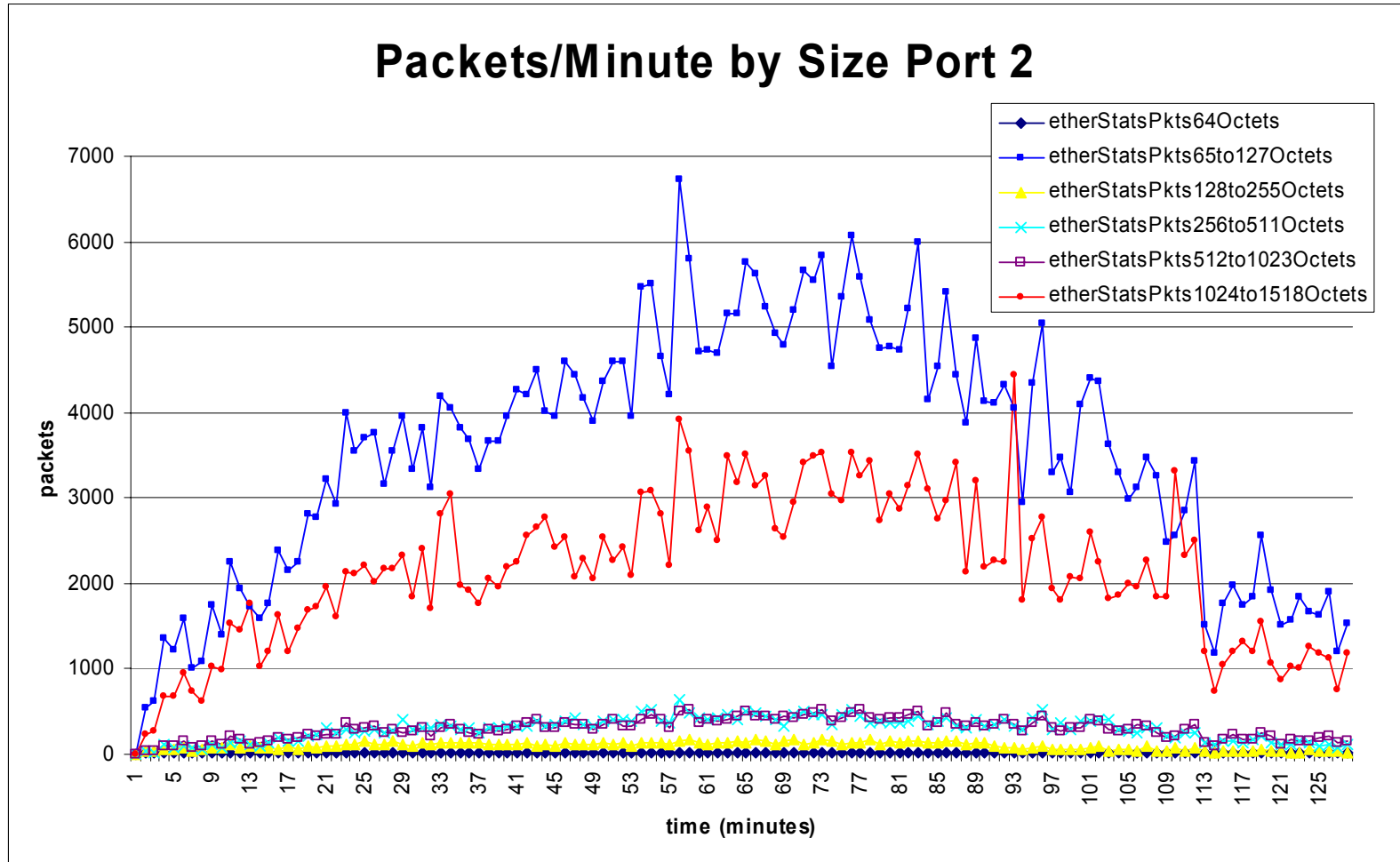
- **Binary Streaming Zero (BSZ) attack** generated using netcat
  - **Command:** `netcat host_address port < /dev/zero` (Port 80)
  - **Packet size:** 1514 bytes
  - **Network Utilization:**  $\approx 40.3\%$



- **mstream Attack** generated using mstream agent "server.c"
  - **Command:** `echo "mstream/192.168.0.40:192.168.0.40/5"  
|.netcat -n -u -p 9325 192.168.0.10 7983`
  - **Packet size:** 60 bytes
  - **Network Utilization:**  $\approx 47\%$



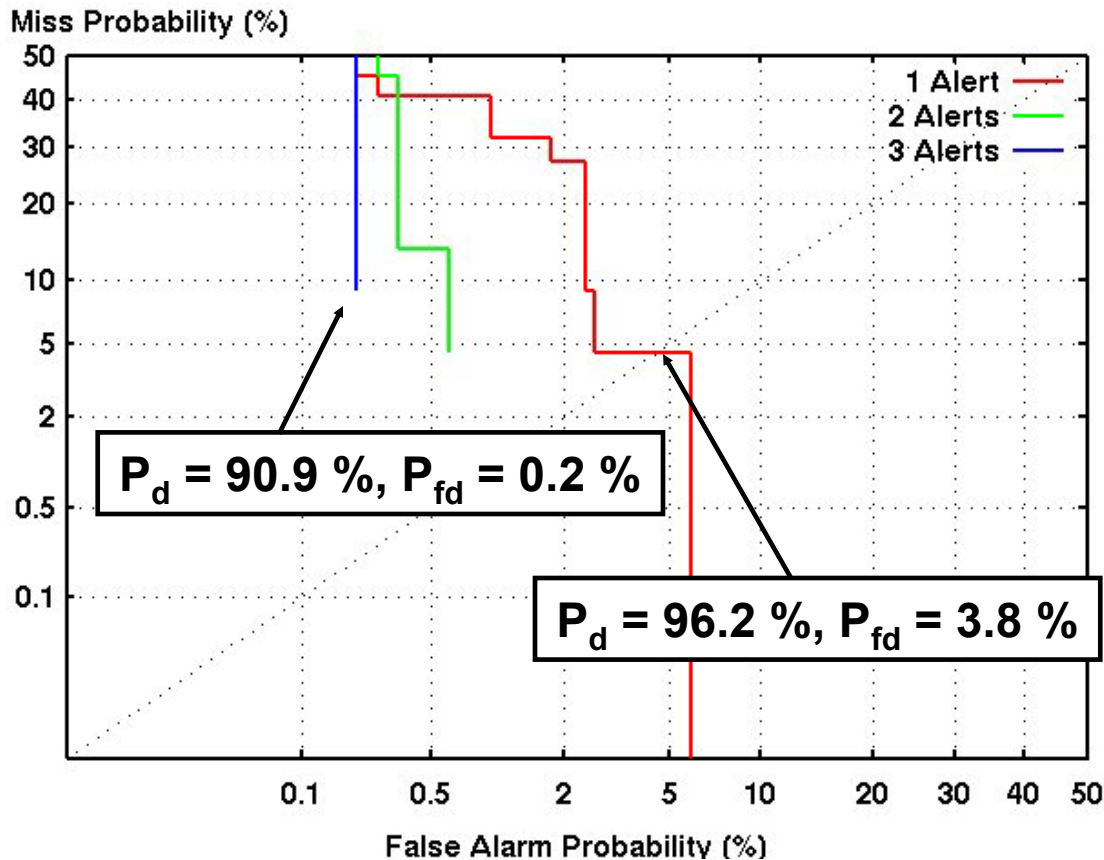
# LL Testbed Background Traffic



- Short-term emulation of data patterns
- Background traffic displays diurnal behavior over 2 hour period



# Detection Prototype (MLP) Results Testbed



- Relatively good performance on testbed staged DoS attacks
- Performance improves when consecutive alerts are required
- Requirement of consecutive alerts puts upper limit on  $P_d$



# RMON2 Contains Useful Information

- **Network matrix**
  - Tracks connections by IP address
  - Indicates source/target of DoS attack
  - Helps build network map
  - Assists localizing of DoS for directed response
- **Application Matrix**
  - Identifies specific application under attack, possibly source of DoS
  - Recognize unusual application (port) usage
- **Protocol Distribution**
  - Tracks packet distribution across transport ports
  - Can be used to recognize scan or DoS hitting multiple services



# Outline

- **Motivation and Goals**
- **Background: DoS and RMON**
- **Data collection and analysis**
  - Using RMON to Detect DoS
- **Test bed**
  - Prototype testing, deployment
- **Summary**





# Summary

- **Traffic model built using actual SNMP traffic**
    - Collected from production network switch
    - RMON1 Statistics used to model traffic
    - Machine learning model superior to simple statistical model
  - **Prototype detection tool**
    - Implemented on test bed
    - Good detection rate and low false alarm rate at emulated DoS utilization levels
- **DoS detection via SNMP/RMON is possible**
    - Given appropriate detection features and sufficient training



# Future Work

- **Network Map Development**
  - Dynamically update model network extent
  - Track normal traffic flows
  - Reduce FA rate by recognizing foreign flows
- **Investigate use of RMON2 variables, other MIBS**
  - Transport, application stats can be used to model network traffic
  - Track traffic by IP address, applications vs. utilization
  - Reduce FA rate by recognizing foreign addrs/utilization
  - Other mibs can be used to trace throughput: CISCO, etc.
  - Investigate dedicated RMON probes
- **DoS Traceback**
  - Using network map, RMON2 locate source of DoS
  - Important research, commercial model
  - Based upon detection issue ACLs, control commands to network devices