TippingPoint

# Protocol Identification via Statistical Analysis (PISA)

BlackHat 2007

Rohit Dhamankar and Rob King

# Agenda

TippingPoint

IPS-SECURED NETWORKS

- Why PISA?
- Generalized Traffic Identification Axes
- Case Study: Skype
- Ongoing Work

TippingPoint

# Why PISA?

# The Problem



- **Encrypted Traffic is becoming common**
  - Bots are using encrypted traffic for communication
- **Next generation Peer-to-Peer protocols are encrypted**
  - First Generation P2P protocols HTTP-like or proprietary
    - Examples: KaZaa, eDonkey, Gnutella etc.
    - Protocol can be reverse-engineered
    - Easily detectable and stoppable via network monitoring systems
  - Next Generation P2P protocols are proprietary
    - Skype binary difficult to reverse engineer
    - Skype protocol cannot be easily detected via network monitoring systems

# The Problem

TippingPoint

IPS-SECURED NETWORKS

- **P2P protocols tend to hog lot of bandwidth and increasing bandwidth is not a solution – detection is!**

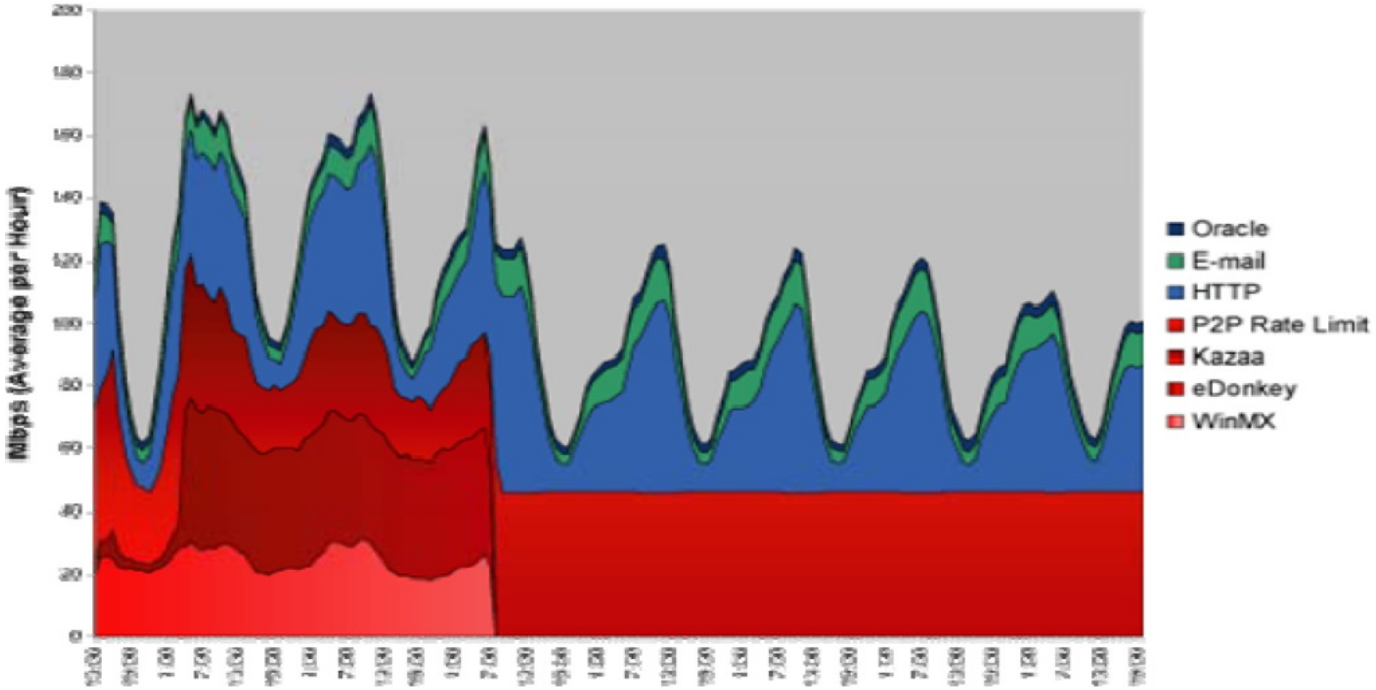



*Figure 7: This graph details an eight-day period. Each peak represents the peak traffic during that day. All of the red data represents P2P traffic, which is rate-limited to 45Mbps on day three. The data in blue and green represents mission critical traffic: Oracle, E-mail, and HTTP. It is not rate limited receiving the full bandwidth advantage of the pipe.*

# Example: KaZaa Traffic



- **172.16.5.20:1277 -> 24.141.247.100:2785**
- **HTTP Request**
  - GET /.hash=1b48a19af2dab74f73990f6336ad16dac40ecffe HTTP/1.1
- **HTTP Headers**
  - Host: 24.141.247.100:218:3090
  - X-Kazaa-Network: KaZaA
  - X-Kazaa-Username: geaiez
  - Range: bytes=2097152-2359295

# Example: eDonkey Traffic



- **File upload from**
  - 24.153.164.134:4662 -> 217.230.32.179:3939
- **Packet Content**

  E3 36 00 00 00 59 56 EA 7F 9B 8D B9 D7 0A EF 91
  B3 90 C3 F5 13 A8 23 00 54 65 72 72 79 20 43 6C
  61 72 6B 20 2D 20 41 20 4C 69 74 74 6C 65 20 47
  61 73 6F 6C 69 6E 65 2E 6D 70 33
- **The file upload command “e3”**
- **Run length data encoding with length 0x36 = 54**
- **Filename in clear text: A Little Gasoline.mp3**

# Example: Skype Traffic

TippingPoint

IPS-SECURED NETWORKS

- **Interleaved UDP and TCP traffic**
  - Size UDP port numbers
  - 995419 Mar 17 11:21 pcap.skype.filtered.41329.7593
  - 1958896 Mar 17 11:21 pcap.skype.filtered1.41329.31020
  - 3573717 Mar 17 11:21 pcap.skype.filtered2.41329.2126
- **Packet content is encrypted**
  - 192.168.0.101.41329 > 74-92-88-202 Philadelphia.hfc.comcastbusiness.net.2126: [udp sum ok] UDP, length: 22
    - Packet data

```
0x0000:  4500 0032 0cbd 0000 8011 c9ca c0a8 0065  E..2.........e
0x0010:  4a5c 58ca a171 084e 001e c431 a357 0256  J\X..q.N...1.W.V
0x0020:  9430 3e9c ed3a 7477 697b 4921 0c08 b8a1  .0>..:twi{I!...
0x0030:  dc19                                     ..
```

# Solution – Paradigm Shift



- **From: Content-based detection**
  - Most network monitoring systems use content in packets i.e. signatures to detect traffic

- **To: Statistics-based detection**
  - Is a framework possible to guess the most likely protocol just based on observed statistics on the flow?

Statistics is like a bikini that reveals what is interesting and hides what is vital

# PISA 10-dimensional Traffic Space



- **The axes of the PISA space decided by a couple of "beer-gut-feelings"**

# PISA Co-ordinates: 10-dimensional Traffic Space



- Average Packet Size to client
- Average Packet Size to server
- Average Time for client responses
- Average Time for server responses
- Standard Deviation of Packet Size to client
- Standard Deviation of Packet Size to server
- Standard Deviation of Time for client responses
- Standard Deviation of Time for server responses
- Traffic difference between server and client

Standard deviation measures how far the majority of data set lies from the average

# PISA Co-ordinates: 10-dimensional Traffic Space



- **These co-ordinates help us differentiate between protocols that are:**
  - Chatty (Microsoft Exchange)
  - Sending traffic mostly in one direction (scp, https)
  - Traffic is balanced in both directions. Voice traffic tends to be
    - Unless you are turning a deaf ear to the boss on other side of the line without muttering a word!

# The 10th PISA Co-ordinate: Shannon Entropy



- **Shannon Entropy is a measure of data randomness**
  - $-\sum p(x_i)\log_2 p(x_i)$
  - $p(x_i)$ is the probability of occurrence of element $x_i$
- **Example**
  - Data: “aaaaaaaa”
  - Shannon Entropy: 0 since p(a) = 1
  - Data: “aaaabbbb”
  - Shannon Entropy: -2*1/2*log(1/2) = 1
  - If all characters from 0x00 and 0xff are present with equal frequency, the Shannon Entropy is maximum for the flow.
  - Max Entropy possible: 8

# Experimental Data (Ongoing to collect more traffic)



- **45-50 Gigabytes of:**
  - Skype Voice data
  - Skype Video data
  - Gizmo Voice data
  - UDP DNS Traffic
  - NFS Traffic
  - NTP Traffic
  - NetBIOS Traffic
  - Other UDP Traffic
- **Traffic collected mostly in broadband environment – corporate and university LANs and home broadband**

# Experimental Data

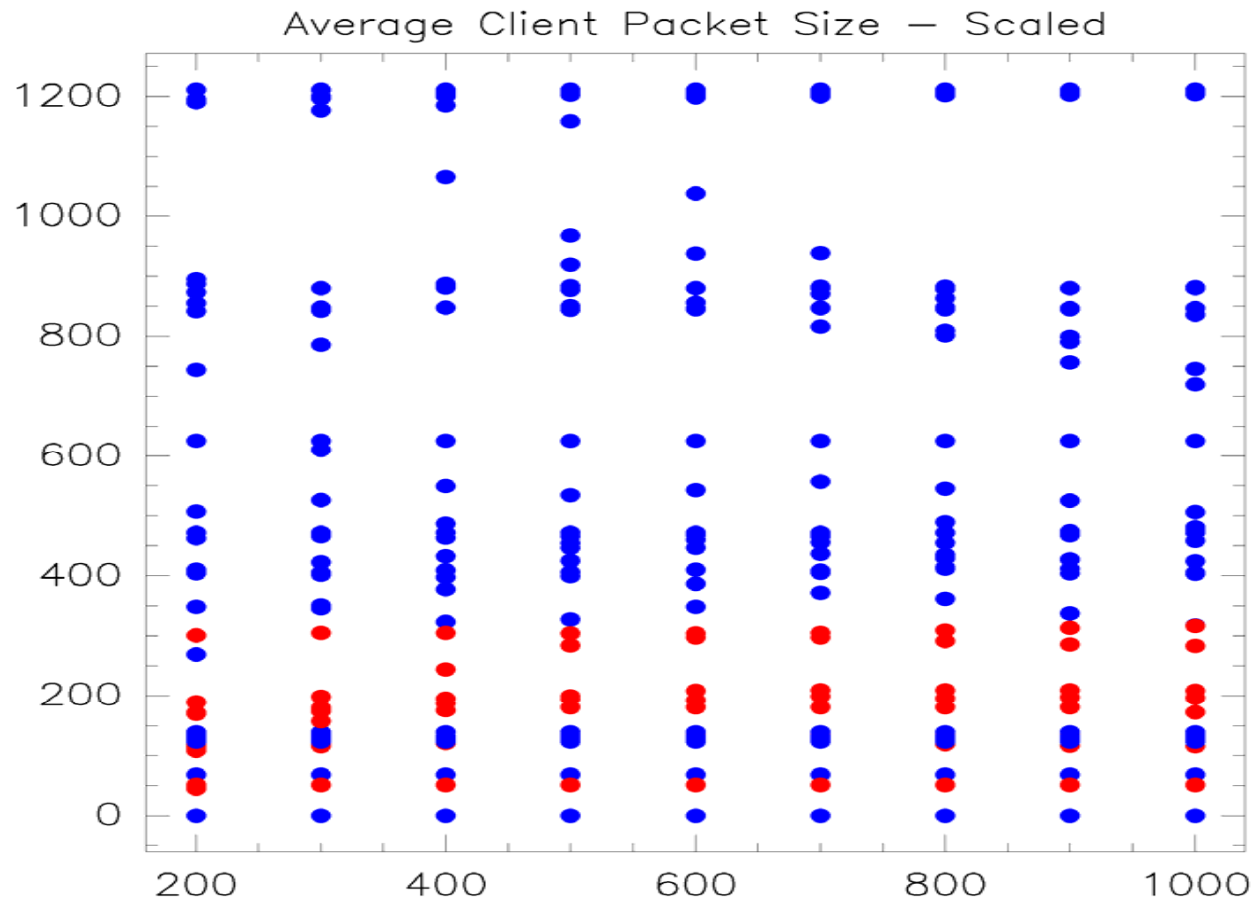


- **As our first distinguishing experiment, we wanted to separate Skype from the rest of the UDP traffic**
- **Calculate the co-ordinates as a function of Skype packets**
- **The next set of slides are graphs of scaled Skype co-ordinates**
- **Scaled == All variables on a equal footing to remove the inherent scale difference.**
  - Time delay is in milliseconds whereas packets size is in thousands of bytes

# Graph 1: Average Client Packet Size

TippingPoint

IPS-SECURED NETWORKS




# Graph 2: Average Server Packet Size

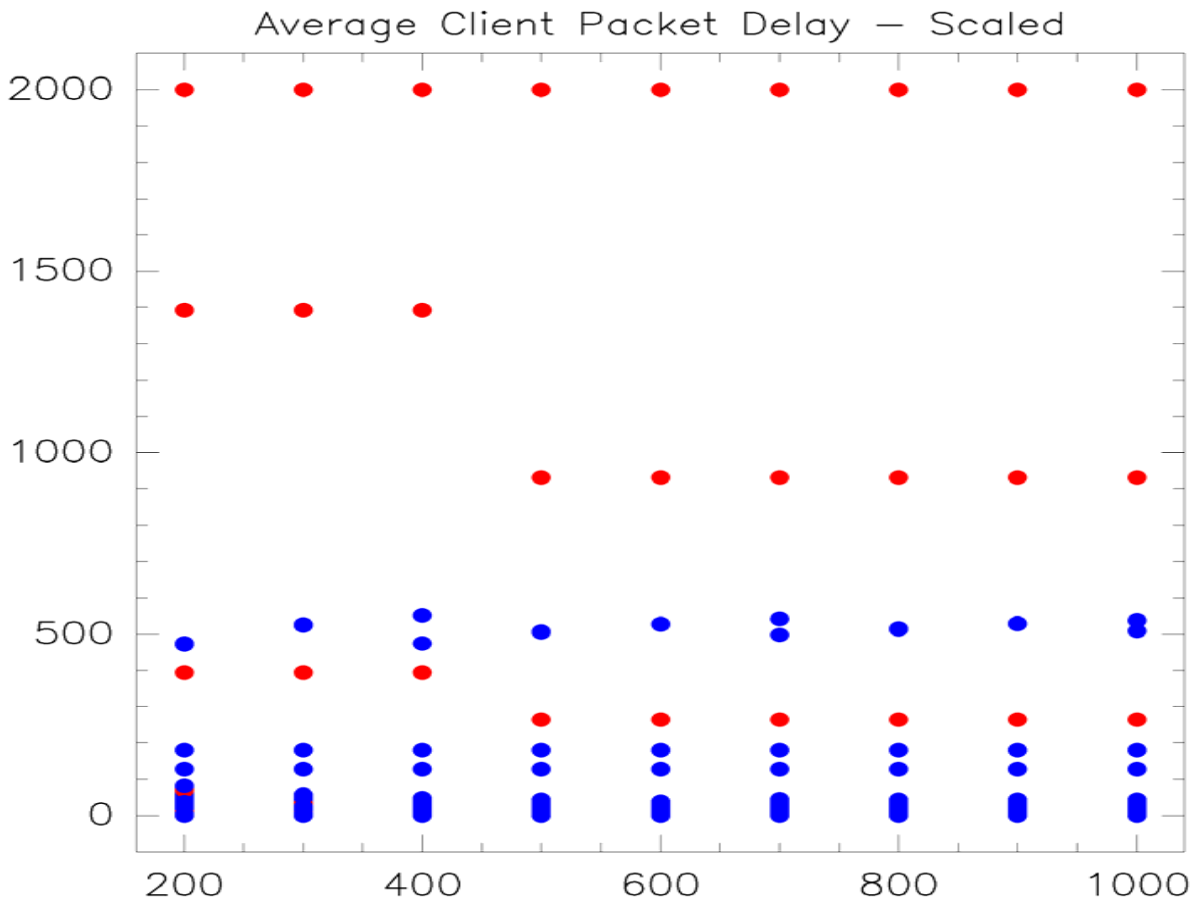
TippingPoint

IPS-SECURED NETWORKS


# Graph 3: Average Client Response Delay

TippingPoint

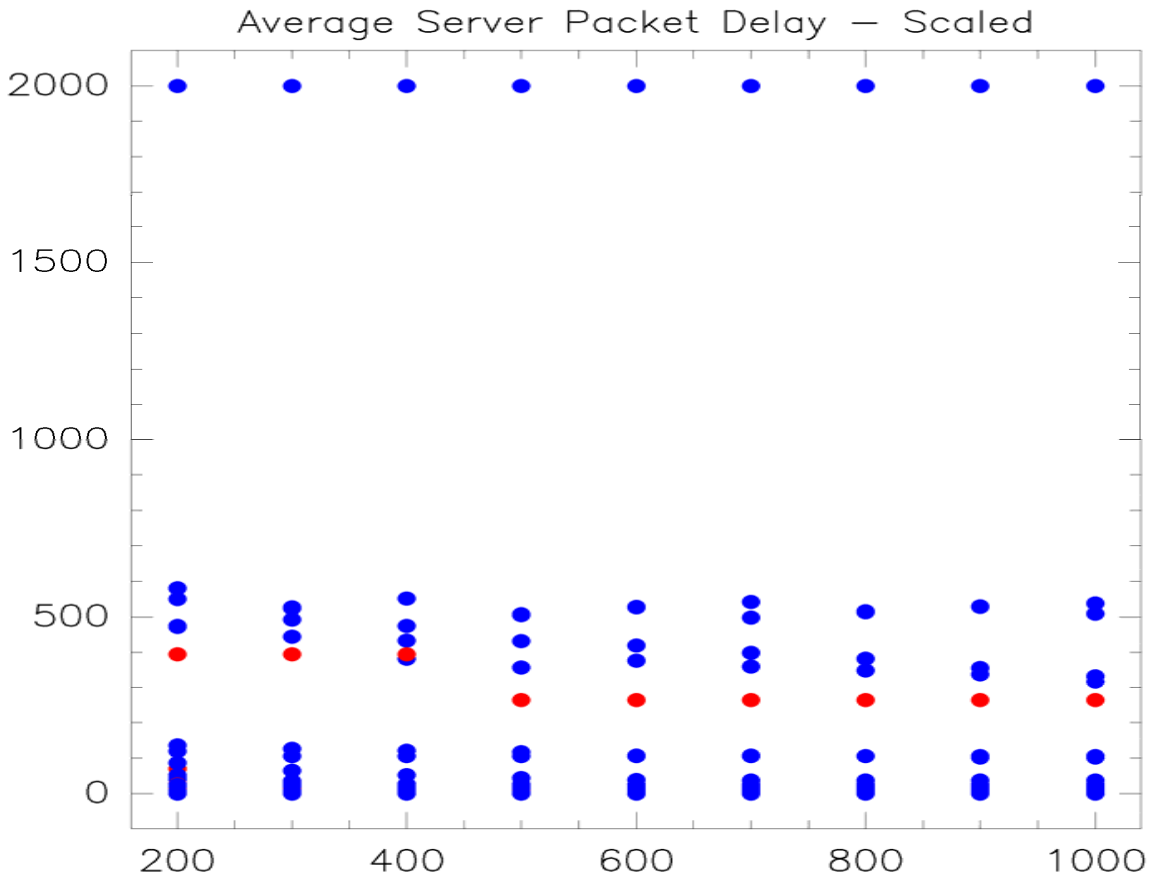
IPS-SECURED NETWORKS




# Graph 4: Average Server Packet Delay

TippingPoint

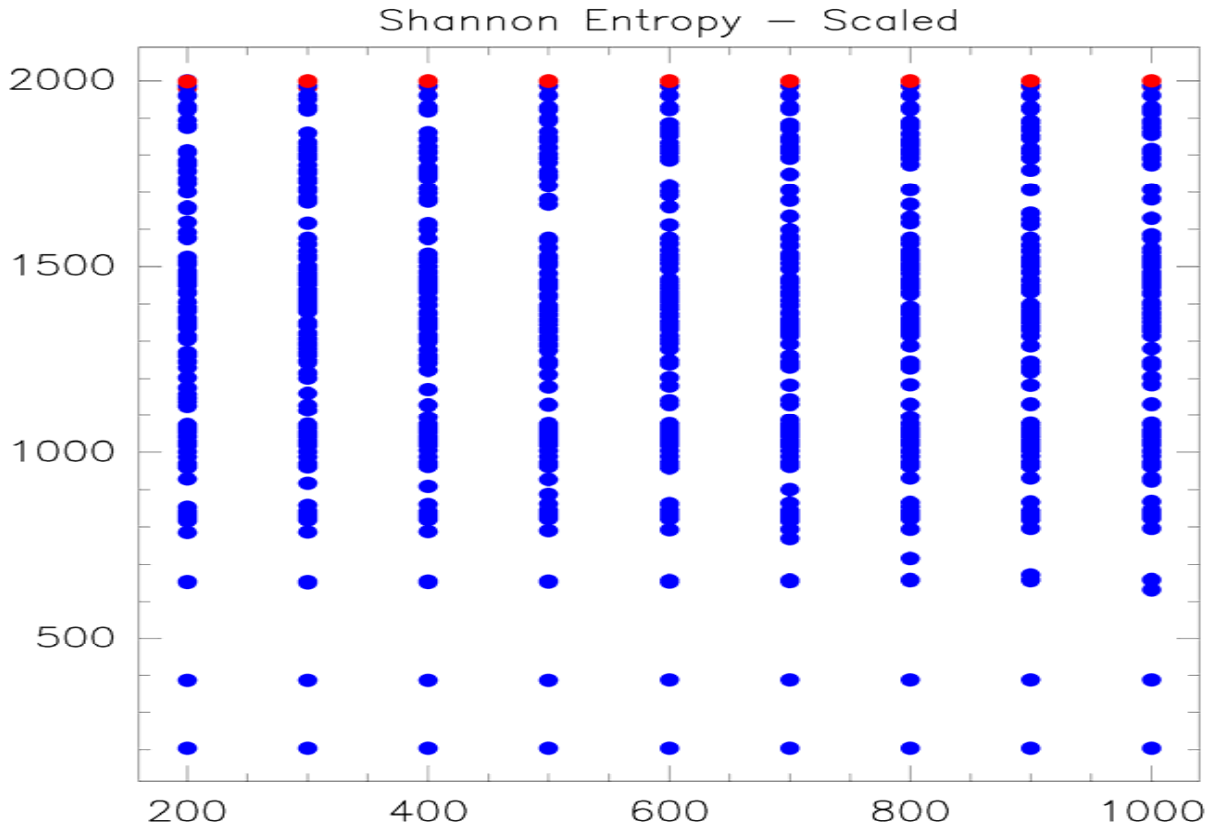
IPS-SECURED NETWORKS




# Graph 5: Shannon Entropy

TippingPoint

IPS-SECURED NETWORKS



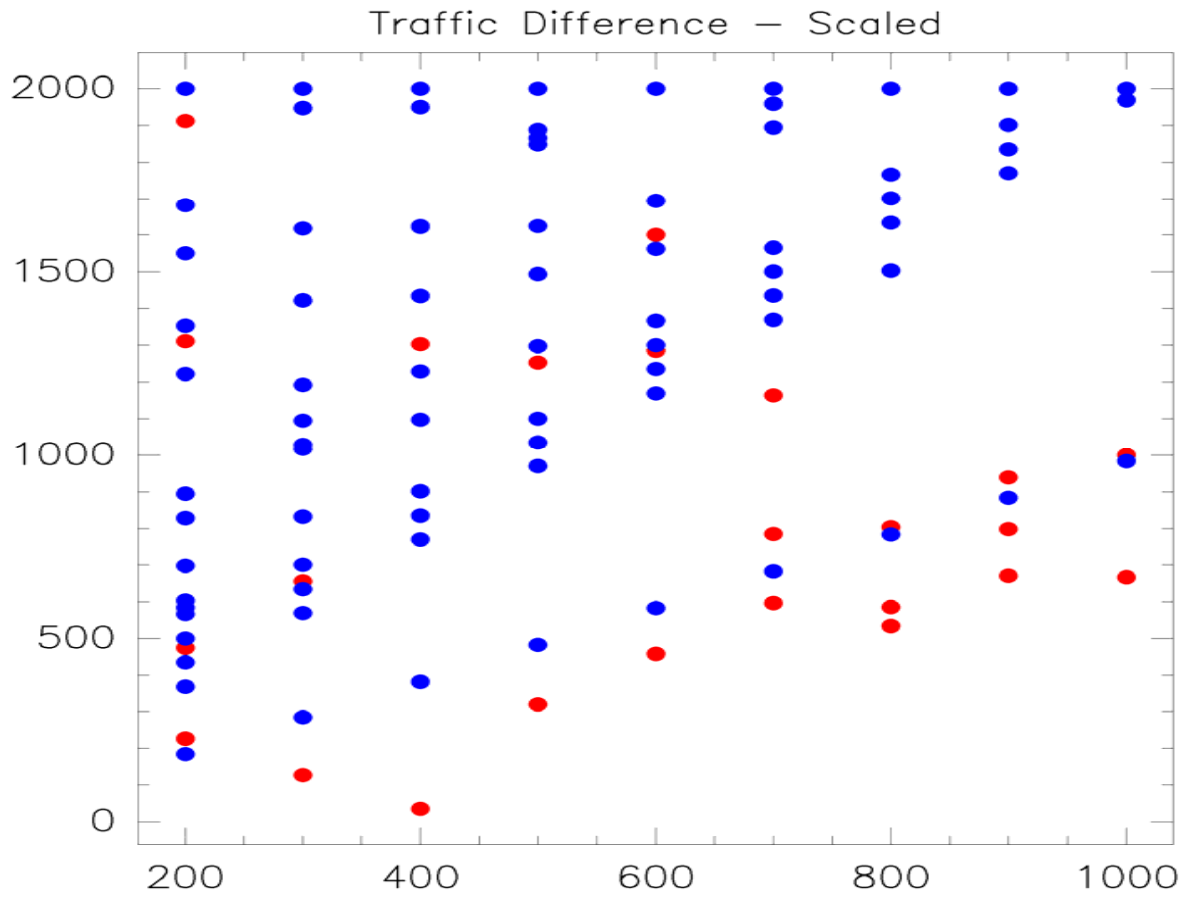

Skype

Other

# Graph 6: Traffic Difference

TippingPoint

IPS-SECURED NETWORKS




# Skype Data Observations



- **By about 600th packet, Skype statistics are stable**
  - ➔Detection possible within one and half seconds of Skype call

- **Different types of traffic fall in different bands**
  - Note: “Blue” is all other traffic

# Euclidean Distance in 10-d Space

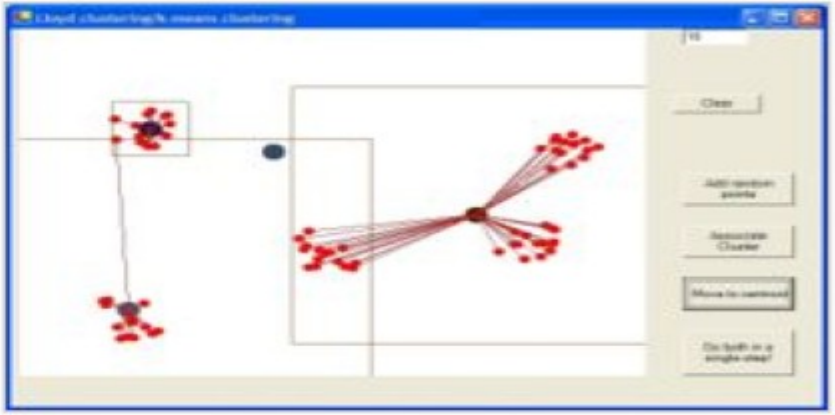


- **Scaled Co-ordinates for distance computation**
  - $\sqrt{\sum d_i * d_i}$ (i varies from 1 – 10)
- **Average distance for Skype computed at 600$^{th}$ packet as the values for distance start converging**
- **The mean and standard deviation of distance computed for each sample Skype flow**
- **The samples lie close to each other – Hurray --**

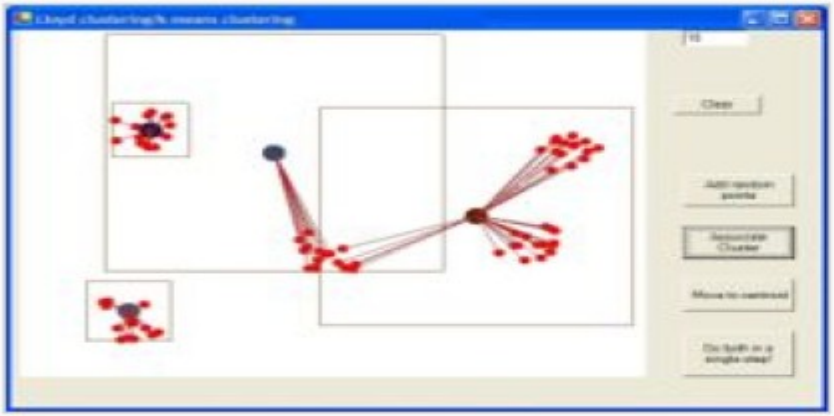
# K-Means Algorithm and Clustering

TippingPoint

Shows the initial randomized centers and a number of points (actual size here)



Centers have been associated with the points and have been moved to the respective centroids (actual size here)



Now, the association is shown in more detail, once the centroids have been moved. (actual size here)

Again, the centers are moved to the centroids of the corresponding associated points. (actual size here)

# Live Demo



- **Point-by-point plotting and visualization of data rela-time**

# Results: NetBIOS protocol

TippingPoint

IPS-SECURED NETWORKS

- pcap: 192.168.61.25:137-192.168.61.255:137
- expected protocol: netbios-ns
- output:
- 1780.30860264 = ntp
- 1936.35599254 = route
- 2764.66914234 = snmp
- 1832.0630088 = netbios-dgm
- 1818.12445314 = skype
- 2199.13745758 = nfs
- 676.334483051 = netbios-ns
- 3244.52297705 = bootpc

- best guess: 676.334483051 = netbios-ns
- second best guess: 1780.30860264 = ntp
- distance between guesses: 1103.974119589

# Results: Skype Protocol



- pcap: pcap.skype.nana.2126.41329
- expected protocol: skype
- output:
- 1960.45561284 = ntp
- 2522.05029833 = route
- 2689.22193848 = snmp
- 2549.95681014 = netbios-dgm
- 737.228693256 = skype
- 1837.09071885 = nfs
- 1710.04898741 = netbios-ns
- 3296.3372724 = bootpc

- best guess: 737.228693256 = skype
- second best guess: 1710.04898741 = netbios-ns
- distance between guesses: 972.820294154

# Results: RTP With Steganography



- **Real-time Transfer Protocol (RTP) is used by Voice over IP technologies to provide an audio channel for calls.**
  - Allows for creation of a covert communications channels
- **RTP Data Analyzed From Corporate SIP calls:**
  - Shannon Entropy: 4.3
- **RTP Data Analyzed Via SteganRTP Tool**
  - Shannon Entropy: 5.8 (35% increase over normal calls)
    - The character set used in RTP traffic was “visually” different with and without the steganography data

# Conclusion



- **PISA can be used to accurately identify protocols with some error margin**
- **PISA can be used to identify the same protocols being used in an anomalous fashion such as covert channels**
- **Code will be posted at:**
  - http://dvlabs.tippingpoint.com/projects/pisa

TippingPoint

# Thank you!

rohitd@tippingpoint.com
rking@tippingpoint.com