Detecting Automatic Flows

Jeffrey Dean, PhD
United States Air Force

My Job & Background

- Air Force civil service, Electrical Engineer
- We design, build and support IDS/IPS platforms for the Air Force
  - Extensible, scalable system of systems for network defense
- PhD in Computer Science, Naval Postgraduate School
  - Information Assurance Scholarship Program (IASP)
    - Program geared to increase DoD military/civilian personnel with advanced cyber defense related degrees (a good deal!)
- The information presented here reflects work I did for my PhD research
  - It does not reflect any Air Force projects or positions

Overview of My Talk

- Rationale for Analysis
- Initial Efforts
- Experimental Setup
- Observations
- Filtering Methods
- Effectiveness
- Conclusions

Rationale for Analysis

- Legitimate network users can be biggest threat
  - Have access to network resources
  - Can do great harm
- Network flow based monitoring can provide insight into users activities
  - Many flows not user initiated
  - OS and applications can spawn flows automatically
- We need methods to “cut the chaff”
  - Focus on user generated flows

Rationale for Analysis (cont.)

- Problem needed solving to support research
  - Testing assumption that users with same roles exhibited similar network behaviors
  - Was evaluating five weeks of traffic from /21 network router
    - 1.162 x 10^9 flow records
    - Various operating systems & system configurations
    - Traffic from 1374 different users
- Needed solution that was platform independent

Initial Efforts

- Initially we looked at port usage
  - We removed flows not related to user activity
    - Ports 67/68 (DHCP), 123 (NTP), 5223 (Apple Push Notification)
- For other ports, identifying automatic flows not so easy
  - Ports 80 & 443 used by many applications
  - E-mail clients sometimes get new mail, sometimes just checking. Same for many applications looking for updates

Experimental Setup

- We created two virtual machines (Windows 7 and Ubuntu)
  - Each system had a version of tcpdump installed
  - Traffic was captured while performing scripted activities

Action	Windows 7 Application	Ubuntu Application
Connect to Windows share drive, load/save files	Windows Explorer	Nautilus
Sent/received emails	Outlook	Thunderbird
Opened SSH link	Not tested	Command line, SSH
Browsed www.cnn.com	Chrome and Internet Explorer	Chrome and Firefox
Browsed www.foxnews.com	Chrome and Internet Explorer	Chrome and Firefox
Browsed www.usaa.com	Chrome and Internet Explorer	Chrome and Firefox
Browsed www.nps.edu	Chrome and Internet Explorer	Chrome and Firefox

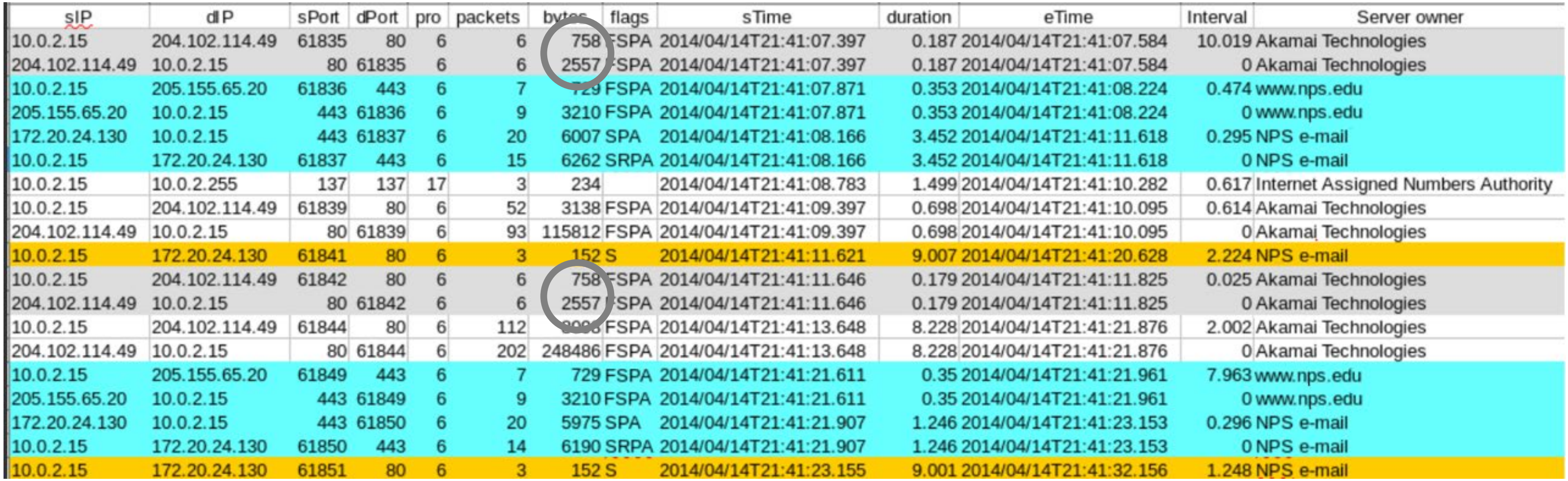
Dean, Jeffrey S., Systematic Assessment of the Impact of User Roles on Network Flow Patterns, PhD Dissertation, 2017

Experimental Setup (cont.)

- Activities were separated by 3-5 minute intervals
  - Enabled related flows to complete
  - Start times of each action recorded
- Also captured traffic while system was idle overnight
  - Applications (e.g. mail client and/or web browser) left open
  - Capture of flow activity with NO user actions
- PCAP files were converted to Netflow v5 using SiLK
  - All flows hand labeled: user initiated or automatic

Observations

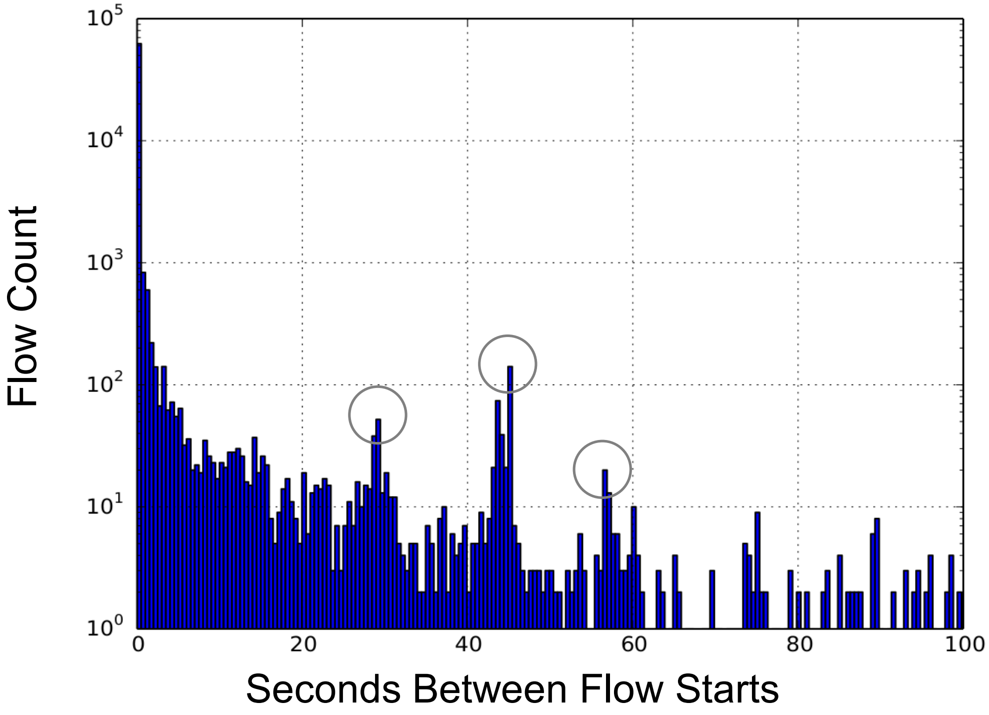
- Flows generated overnight were most useful in identifying non-user generated flows. We saw:
  - Repeated exchanges between the VM and servers



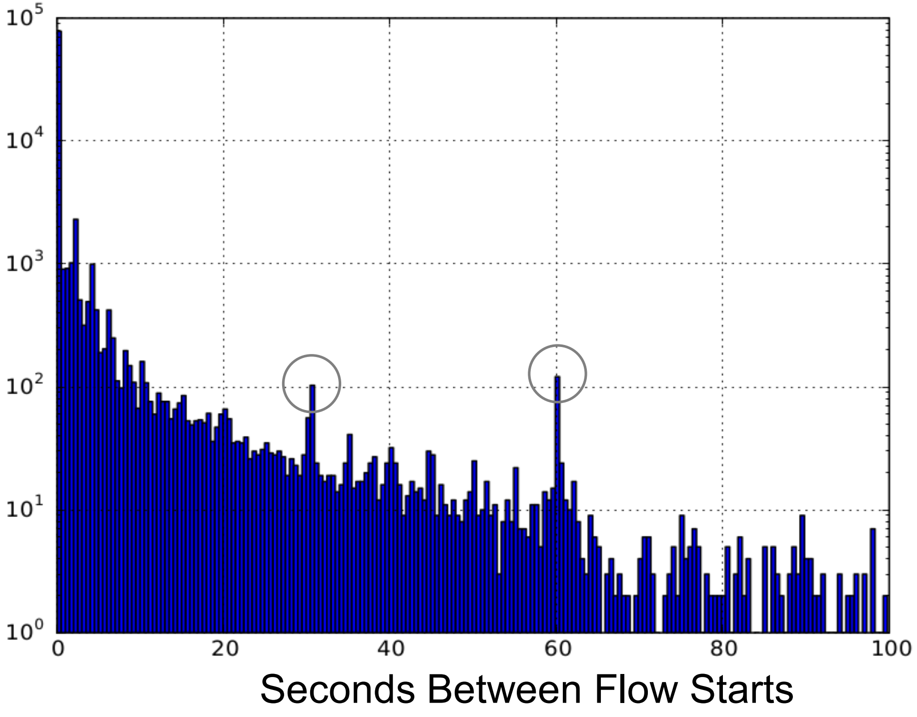
sIP	dIP	sPort	dPort	pro	packets	bytes	flags	sTime	duration	eTime	Interval	Server owner
10.0.2.15	204.102.114.49	61835	80	6	6	758	FSPA	2014/04/14T21:41:07.397	0.187	2014/04/14T21:41:07.584	10.019	Akamai Technologies
204.102.114.49	10.0.2.15	80	61835	6	6	2557	FSPA	2014/04/14T21:41:07.397	0.187	2014/04/14T21:41:07.584	0	Akamai Technologies
10.0.2.15	205.155.65.20	61836	443	6	7	729	FSPA	2014/04/14T21:41:07.871	0.353	2014/04/14T21:41:08.224	0.474	www.nps.edu
205.155.65.20	10.0.2.15	443	61836	6	9	3210	FSPA	2014/04/14T21:41:07.871	0.353	2014/04/14T21:41:08.224	0	www.nps.edu
172.20.24.130	10.0.2.15	443	61837	6	20	6007	SPA	2014/04/14T21:41:08.166	3.452	2014/04/14T21:41:11.618	0.295	NPS e-mail
10.0.2.15	172.20.24.130	61837	443	6	15	6262	SRPA	2014/04/14T21:41:08.166	3.452	2014/04/14T21:41:11.618	0	NPS e-mail
10.0.2.15	10.0.2.255	137	137	17	3	234		2014/04/14T21:41:08.783	1.499	2014/04/14T21:41:10.282	0.617	Internet Assigned Numbers Authority
10.0.2.15	204.102.114.49	61839	80	6	52	3138	FSPA	2014/04/14T21:41:09.397	0.698	2014/04/14T21:41:10.095	0.614	Akamai Technologies
204.102.114.49	10.0.2.15	80	61839	6	93	115812	FSPA	2014/04/14T21:41:09.397	0.698	2014/04/14T21:41:10.095	0	Akamai Technologies
10.0.2.15	172.20.24.130	61841	80	6	3	152	S	2014/04/14T21:41:11.621	9.007	2014/04/14T21:41:20.628	2.224	NPS e-mail
10.0.2.15	204.102.114.49	61842	80	6	6	758	FSPA	2014/04/14T21:41:11.646	0.179	2014/04/14T21:41:11.825	0.025	Akamai Technologies
204.102.114.49	10.0.2.15	80	61842	6	6	2557	FSPA	2014/04/14T21:41:11.646	0.179	2014/04/14T21:41:11.825	0	Akamai Technologies
10.0.2.15	204.102.114.49	61844	80	6	112	[illegible]	FSPA	2014/04/14T21:41:13.648	8.228	2014/04/14T21:41:21.876	2.002	Akamai Technologies
204.102.114.49	10.0.2.15	80	61844	6	202	248486	FSPA	2014/04/14T21:41:13.648	8.228	2014/04/14T21:41:21.876	0	Akamai Technologies
10.0.2.15	205.155.65.20	61849	443	6	7	729	FSPA	2014/04/14T21:41:21.611	0.35	2014/04/14T21:41:21.961	7.963	www.nps.edu
205.155.65.20	10.0.2.15	443	61849	6	9	3210	FSPA	2014/04/14T21:41:21.611	0.35	2014/04/14T21:41:21.961	0	www.nps.edu
172.20.24.130	10.0.2.15	443	61850	6	20	5975	SPA	2014/04/14T21:41:21.907	1.246	2014/04/14T21:41:23.153	0.296	NPS e-mail
10.0.2.15	172.20.24.130	61850	443	6	14	6190	SRPA	2014/04/14T21:41:21.907	1.246	2014/04/14T21:41:23.153	0	NPS e-mail
10.0.2.15	172.20.24.130	61851	80	6	3	152	S	2014/04/14T21:41:23.155	9.001	2014/04/14T21:41:32.156	1.248	NPS e-mail

Observations (cont.)

- Some inter-flow intervals were more common



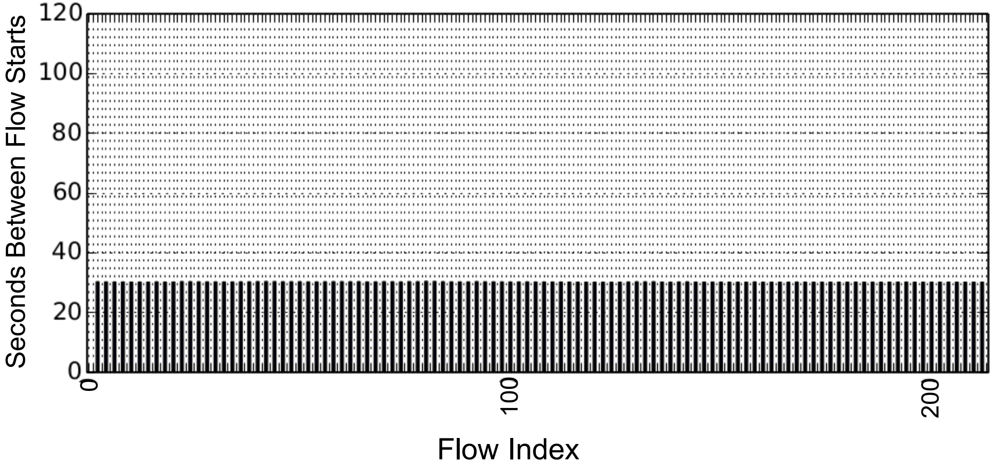
Ubuntu



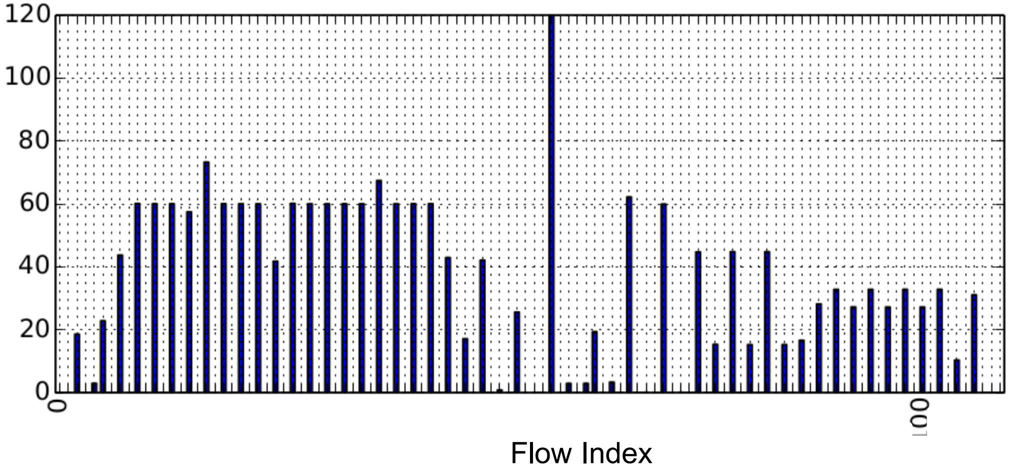
Windows 7

Observations (cont.)

- Repeated intervals more visible when we focused on a single distant IP address, server port and protocol



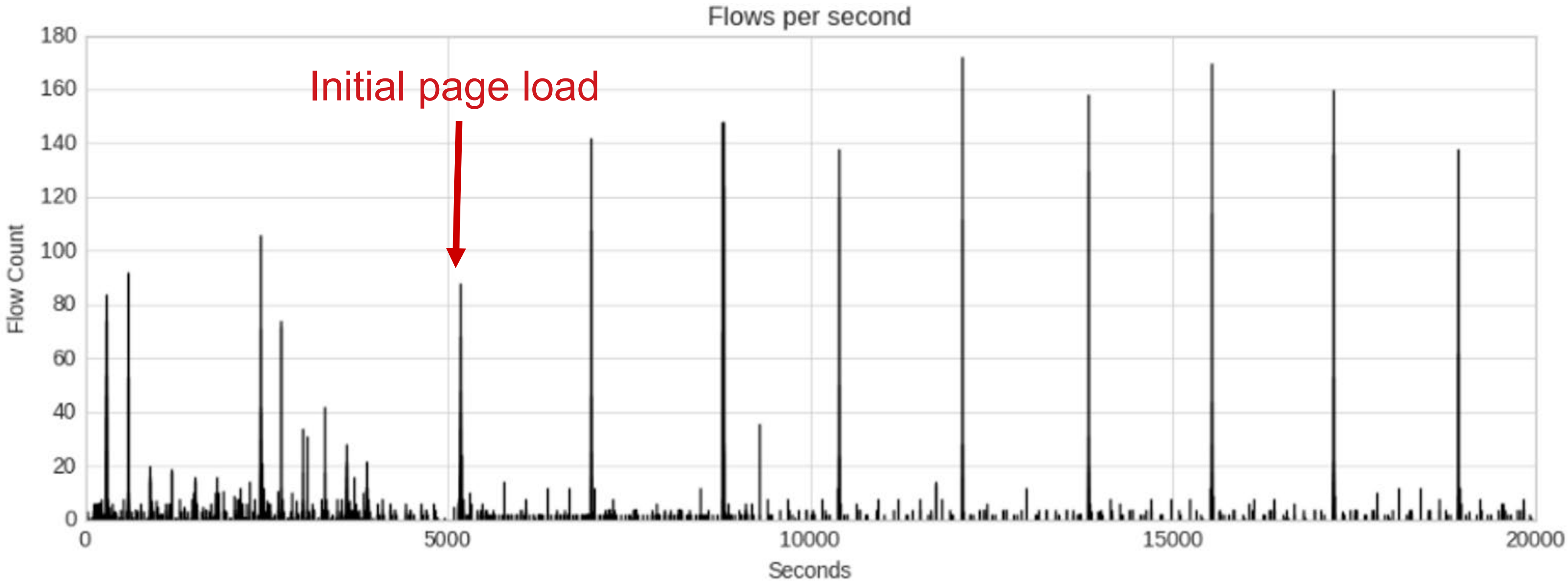
Dropbox LANsync, port 17500



Windows Exchange, Port 60000

Observations (cont.)

- Repeated web-page loads were observed for some web pages (e.g. CNN and Fox News)

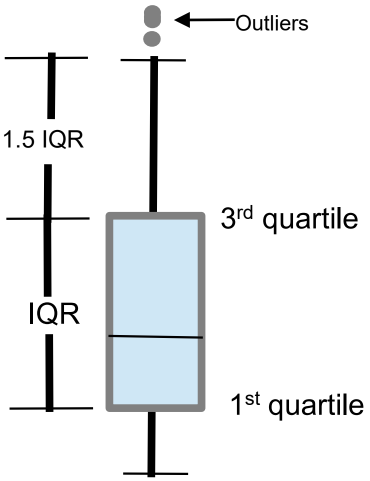


Observations (cont.)

- Labeling automatic flows in data not always straightforward
  - Most inferred without examining payload data
  - Browsers talk to web pages long after initial load
    - A number of “keep-alive” connections continue
    - Often no payload data
  - Often see sequences of flows with “close” byte values
  - Most defining characteristic is an increasing average interval between flow starts

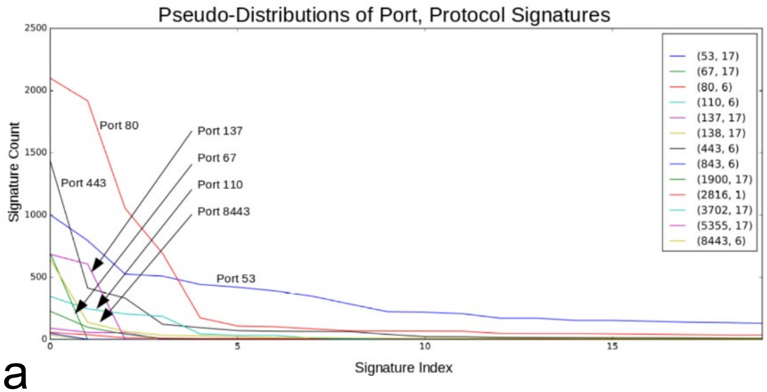
Filtering Methods

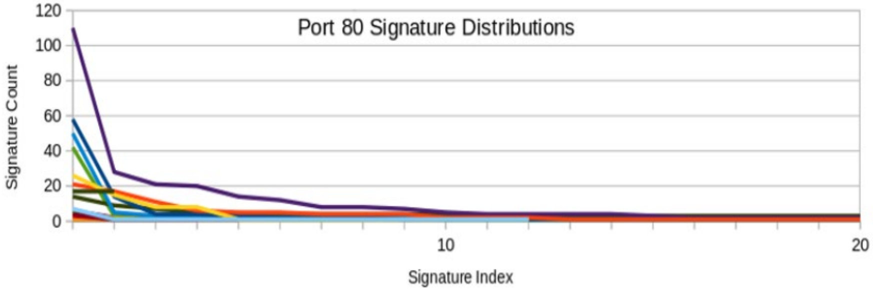
- To identify repeated behaviors, we had to identify outlier counts
  - We found that the definition used by boxplots worked well
  - High value outliers
    - > 3rd quartile + 1.5 x IQR
- Exceptions
  - Less than 10 flows
    - Too few to identify outliers
  - Less than 10 count values
    - List of counts padded to reach 10 values
    - Padded values: min(min(counts)*0.1, 10)
    - Captured instances of a few high count values



Filtering Methods:
Repeated Exchanges

- Tried grouping VM flow records by shared “signatures”
  - Hash of server port, protocol, outgoing packets, bytes, flags and incoming packets, bytes, flags
  - Counts for traffic to/from all distant addresses
  - Outlier counts were mostly TCP handshakes
- We then added distant server address to grouping criteria
  - Counted bidirectional flows to/from single servers
  - Repeated exchanges (bi-directional flows) lined up well with flows labeled as automatic





Filering Methods:
Repeated Intervals

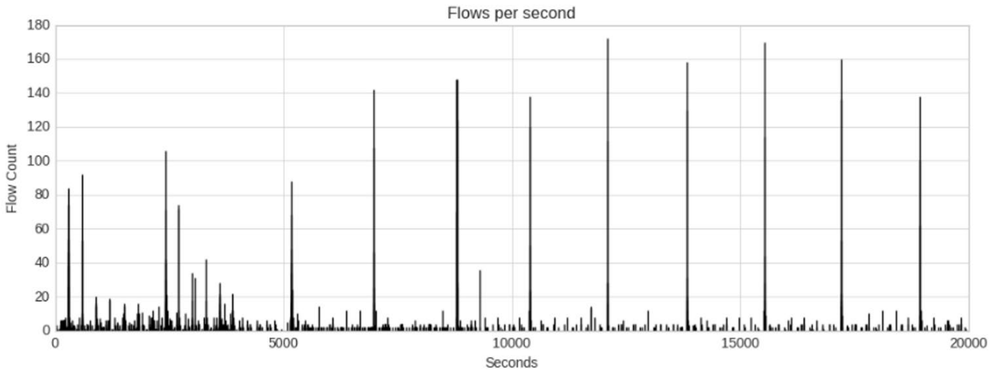
- Flows grouped based on shared distant IP address, server port, protocol, flow direction
  - Intervals between flow start times rounded to nearest second
  - Counted intervals > 2 seconds
  - For outlier interval counts, the flows following the identified interval were counted as automatic
  - CAUTION: Long flow records end at specified (active-timeout) intervals
    - Usually 30 minutes

Filtering Methods:
Web-Page Reloads

- Identifying automatic web-page reloads required:
  - Identifying web-page loads
  - Determine if the page loads were to the same site
    - Not simple, if multiple third-party connections
  - Identify loading time intervals that were “close”
    - Intervals were not precise, especially when long

Filering Methods:
Web-Page Reloads

- Identifying web-page loads
  - Flow bursts: intervals between flow starts < 4s
  - Fraction of HTTP & HTTPS (80 | 443) flows in burst ≥ 0.9
  - Burst size ≥ 20 flows (with packet payloads)



Filtering Methods:
Web-Page Reloads

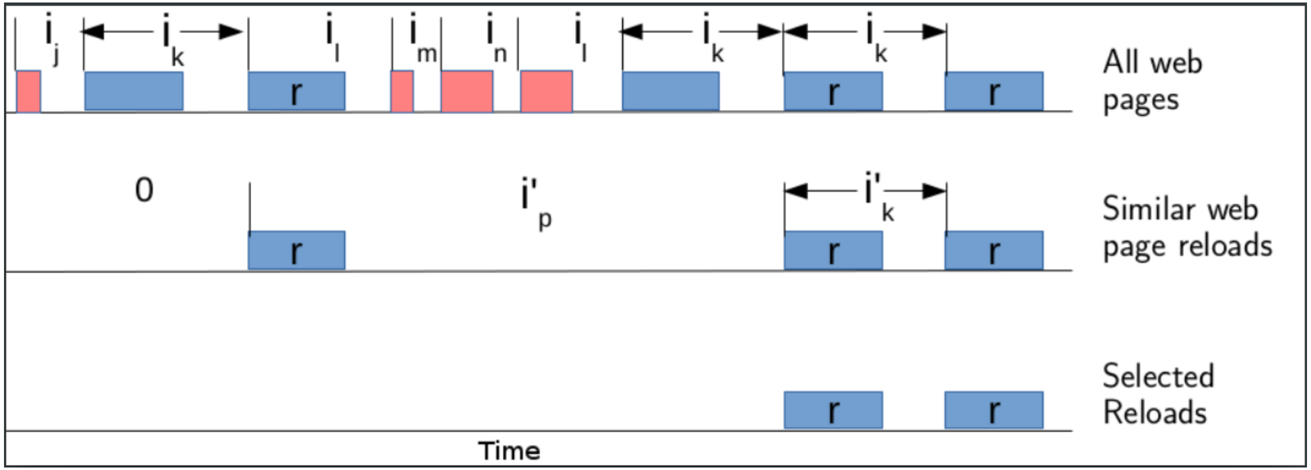
- Page loads are similar, if:
  - Flow count difference ≤ 25%
  - Distance between flow sets F_1 and F_2
    - Let $b(F_1[a_i])$ = bytes to/from distant IP address a_i, flow set F_1
    - Let $b(F_1[p_j])$ = bytes to/from distant server port p_j, flow set F_1
    - Let $m_{ip} = \max(b(F_1[a_i]), b(F_2[a_i]))$, $m_p = \max(b(F_1[p_j]), b(F_2[p_j]))$
    - IP distance $d_{ip} = ((\sum_{i=1}^{m} (\frac{b(F_1[a_i])}{m_{ip}} - \frac{b(F_2[a_i])}{m_{ip}})^2)^{1/2})/m$
    - Port distance $d_p = ((\sum_{j=1}^{n} (\frac{b(F_1[p_j])}{m_p} - \frac{b(F_2[p_j])}{m_p})^2)^{1/2})/n$
    - D = $\frac{d_{ip}+d_p}{2}$ ≤ 0.9

Filtering Methods:
Web-Page Reloads

- Close time intervals
  - Intervals were rounded
    - Rounding value proportional to duration
    - I = interval between web loads
      - Rounding value $d = I\delta\ (0 \leq \delta \leq 1.0)$
      - d rounded to nearest multiple of 10 seconds
    - $I' = d \lfloor ((I + 0.5d)/d) \rfloor$

Filering Methods:
Web-Page Reloads

- Identified sequences of two or more page reloads
  - Outlier count intervals (rounded) between load starts
  - Page reloads after original load identified as automatic



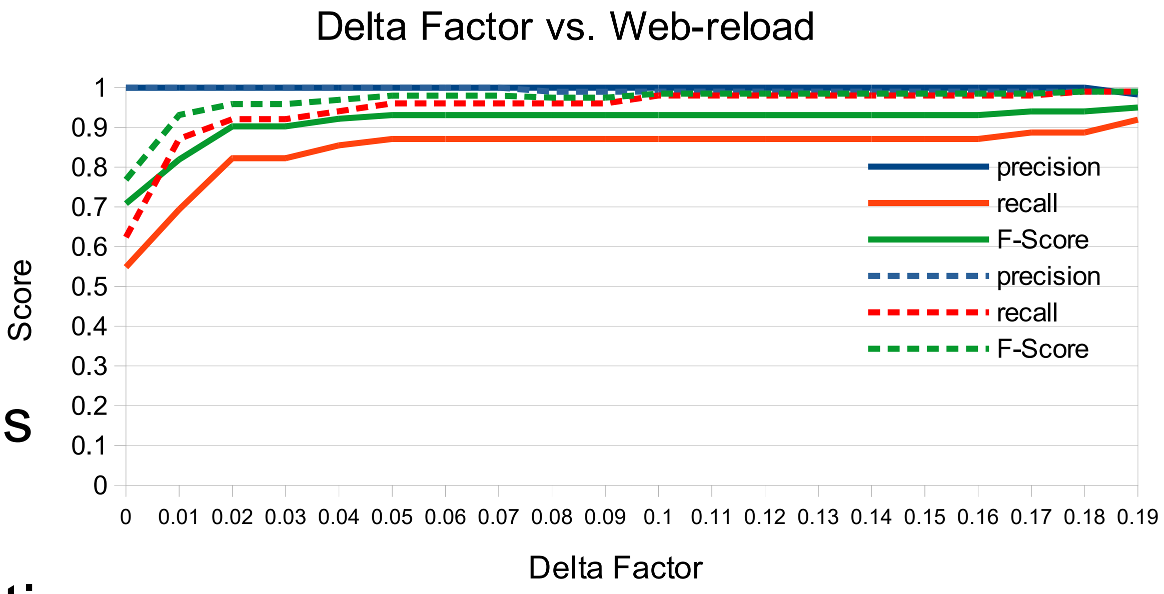
Results

- The signature and interval detection algorithms showed fairly good precision
  - Didn’t detect all flows labeled as automatic

Virtual Machine	Algorithm	Precision	Recall	F-Score
Ubuntu	Signatures	0.89	0.59	0.71
	Timing	0.96	0.21	0.34
Windows	Signatures	0.93	0.50	0.65
	Timing	0.99	0.13	0.23

Results (cont)
Web Reload Detection

- Combination of criteria:
  - Timing
  - Similarity
  - Web page load
  - String of 3 or more loads



- Enabled accurate detection

Conclusions

- The algorithms did fairly well, but didn’t detect all flows labeled as automatic
  - Could be labeling issue (in part), due to classification criteria and some ambiguity in whether flows were truly automatic
- Detection needs to be performed below proxies/NAT’ing
- Approach could be leveraged to carve out flow sets
  - Malware generated traffic could be considered automatic