Covert channel detection using flow-data

Guido Pineda Reyes

MSc. Systems and Networking Engineering
University of Amsterdam

July 3, 2014

Outline

Covert Channels

Definition

Lampson, 1973

"... A communication channel that is used for information transmission, but that is not intended for communications..."

National Computer Security Centre Maryland Meade, 1985

"Communication channel that can be exploited ... to transfer information in a manner that violates the system's security policy"

Malicious usage

- Data exfiltration
- Intrusion maintenance
- Botnet control
- Malware updates
- Gathering of sensitive information
- ...

Chosen techniques

- ICMP tunnel
- ICMP reverse shell
- DNS tunnel
- HTTP reverse shell

Flow-data

Overview

- Netflow is a monitoring tool
- Describes the method for a collector to export statistics about IP packets passing an observation point.
- Netflow v10 aka IPFIX (RFC 5101)
- Payload is not included

Flow

Packets with a set of common properties:

- source address and port number
- ingress interface
- destination address and port number
- network layer protocol
- type of service (TOS)

Research questions

- Is it possible to detect network-based covert channel malicious activity by using flow-data?
  - How do the selected covert channel techniques work?
  - What is the difference between normal traffic and covert channel traffic behaviour using the chosen techniques?
  - What algorithms can be used to detect network-based covert channel traffic?
  - How can this results be validated?

Regular traffic

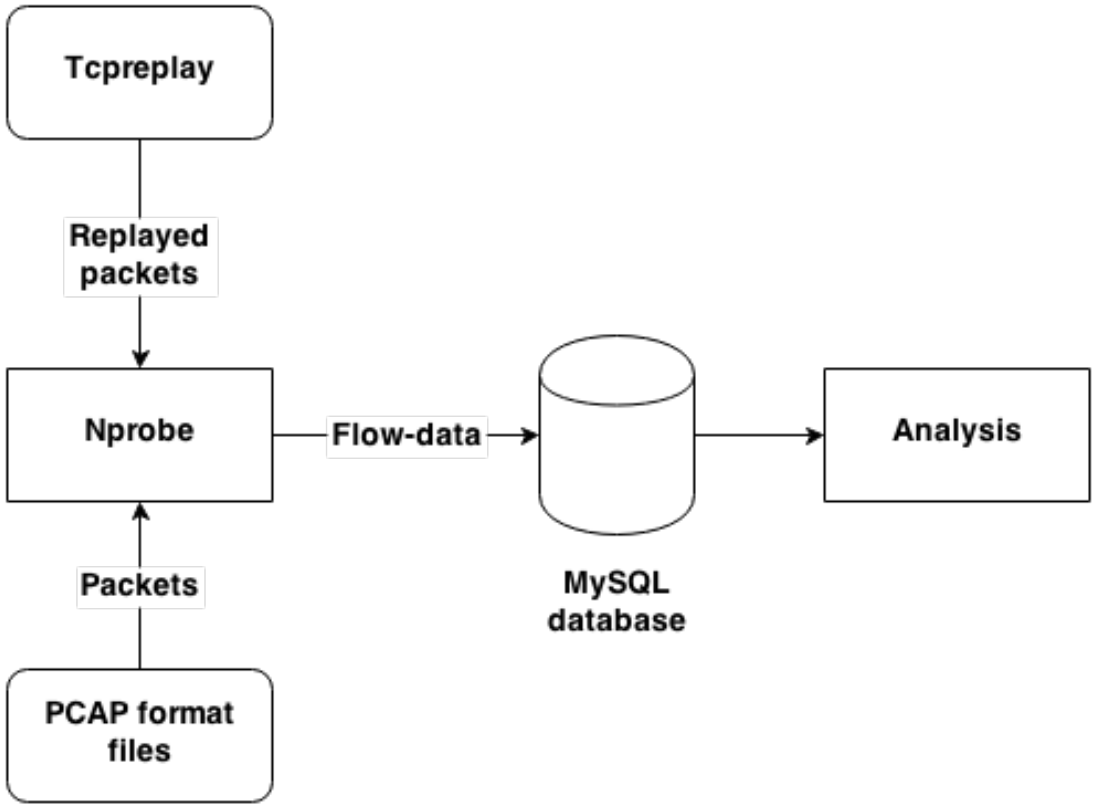
Protocol	Total bytes (MB)	Total packets	Total bidirectional flows
ICMP	698.5	3445152	169
DNS	1638.6	3981600	53490
HTTP	1956.27	1818293	40107

Malicious traffic

Technique	Total bytes (MB)	Total packets	Total bidirectional flows
ICMP tunneling	3957.08	4491868	30
ICMP reverse shell	196.2	3481308	75
DNS tunneling	2746.7	3376230	172
HTTP reverse shell	311.39	470985	166

Approach

Experimental environment



IPFIX templates

Export template: ICMP

Field	Description
IPV4_SRC_ADDR	IPv4 source address
IPV4_DST_ADDR	IPv4 destination address
PROTOCOL	IP protocol byte
IN_BYTES	Incoming flow bytes (src ->dst)
IN_PKTS	Incoming flow packets (src ->dst)
OUT_BYTES	Outgoing flow bytes (dst ->src)
OUT_PKTS	Outgoing flow packets (dst ->src)
MIN_TTL	Min flow TTL
MAX_TTL	Max flow TTL
ICMP_TYPE	ICMP Type * 256 + ICMP code

IPFIX templates

Export template: DNS

Field	Description
IPV4_SRC_ADDR	IPv4 source address
IPV4_DST_ADDR	IPv4 destination address
PROTOCOL	IP protocol byte
IN_BYTES	Incoming flow bytes (src ->dst)
IN_PKTS	Incoming flow packets (src ->dst)
OUT_BYTES	Outgoing flow bytes (dst ->src)
OUT_PKTS	Outgoing flow packets (dst ->src)
MIN_TTL	Min flow TTL
MAX_TTL	Max flow TTL
DNS_QUERY	DNS query
DNS_QUERY_ID	DNS query transaction Id
DNS_QUERY_TYPE	DNS query type (e.g. 1=A, 2=NS..)
DNS_RET_CODE	DNS return code (e.g. 0=no error)

IPFIX templates

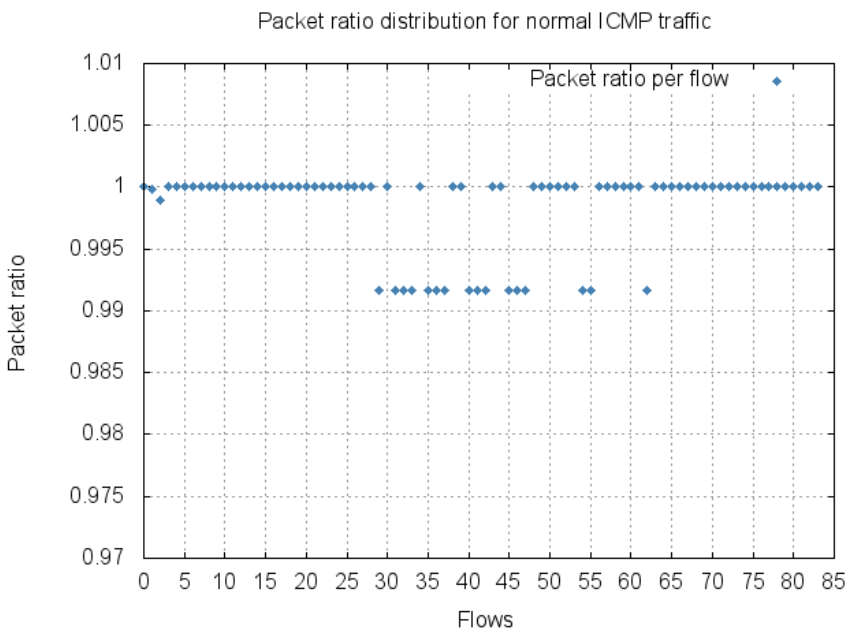
Export template: HTTP

Field	Description
IPV4_SRC_ADDR	IPv4 source address
IPV4_DST_ADDR	IPv4 destination address
PROTOCOL	IP protocol byte
IN_BYTES	Incoming flow bytes (src->dst)
IN_PKTS	Incoming flow packets (src->dst)
OUT_BYTES	Outgoing flow bytes (dst->src)
OUT_PKTS	Outgoing flow packets (dst->src)
MIN_TTL	Min flow TTL
MAX_TTL	Max flow TTL
TCP_FLAGS	Cumulative of all flow TCP flags
HTTP_URL	HTTP URL
HTTP_METHOD	HTTP METHOD
HTTP_RET_CODE	HTTP return code (e.g. 200, 304...)

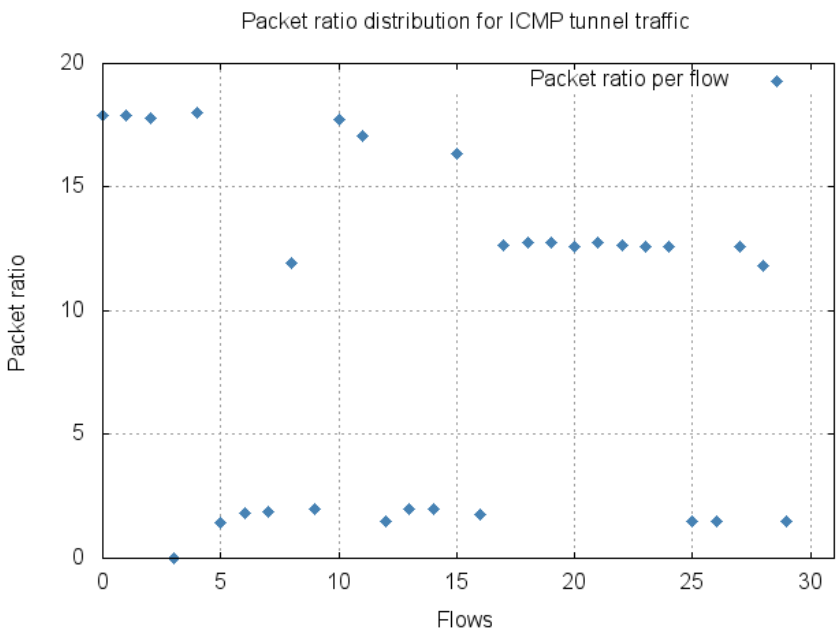
Outline

ICMP tunnel

Packet ratio distribution



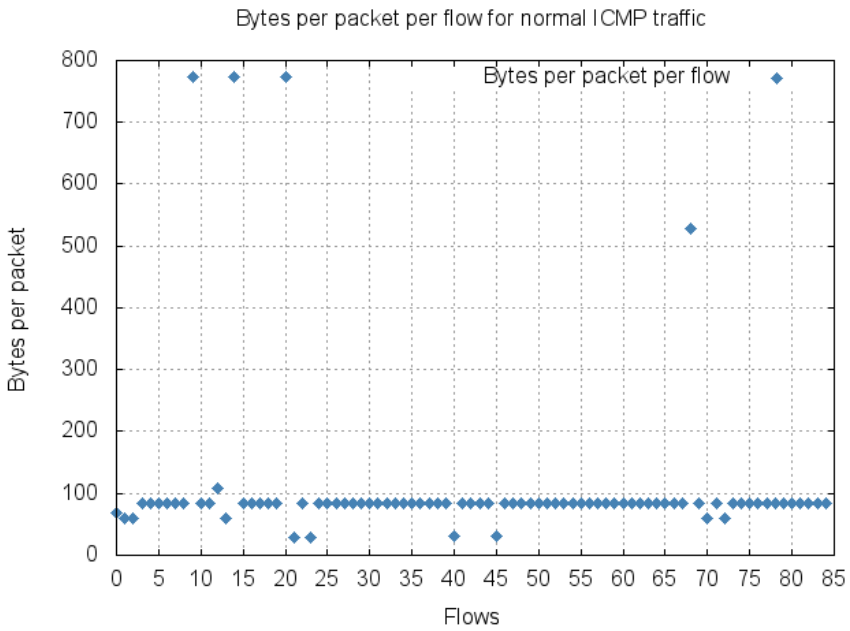
Regular ICMP



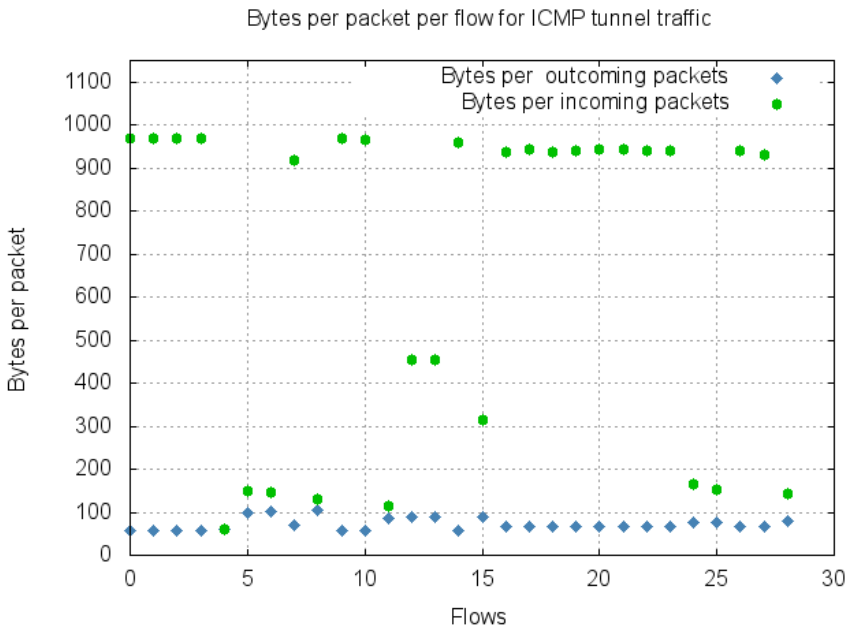
ICMP tunnel

ICMP tunnel

Bytes per packet distribution



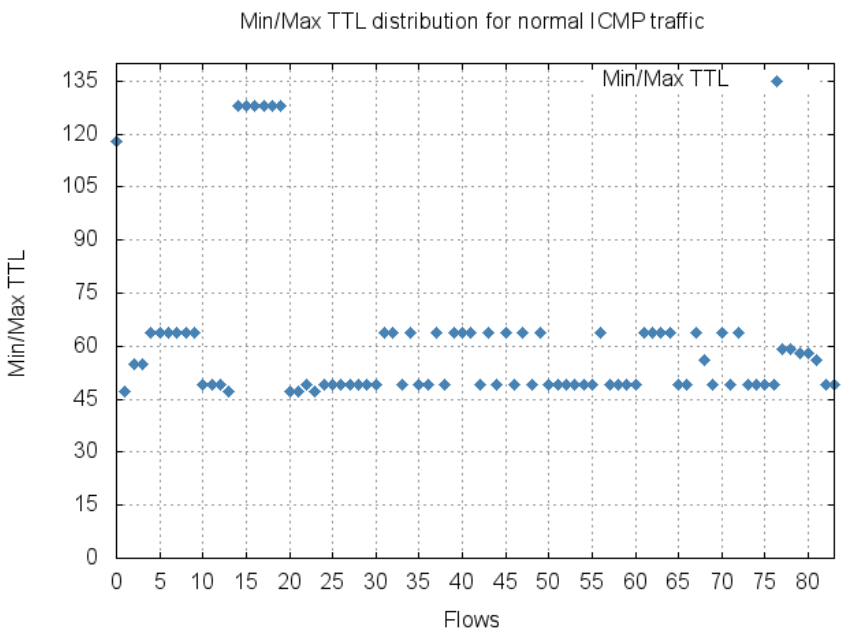
Regular ICMP



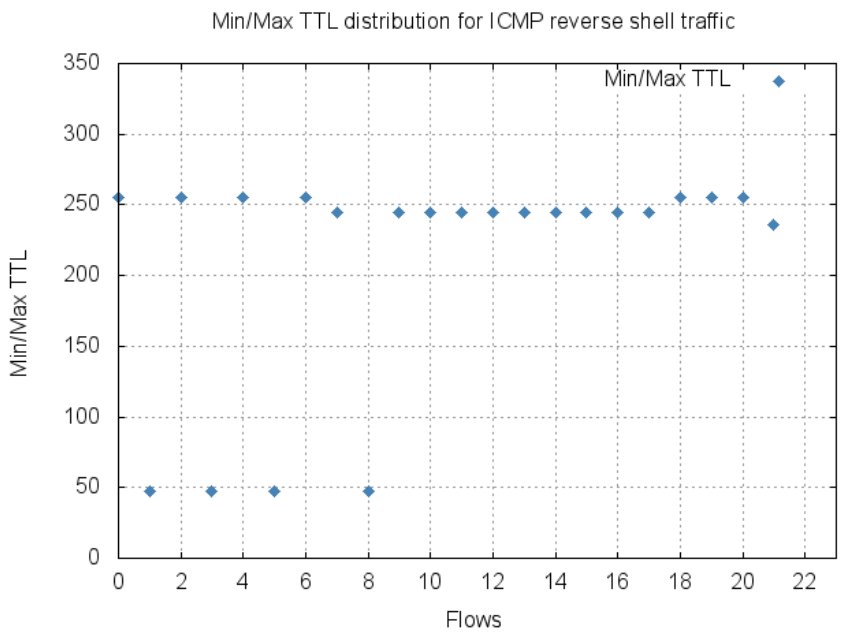
ICMP tunnel

ICMP reverse shell

TTL distribution



Regular ICMP

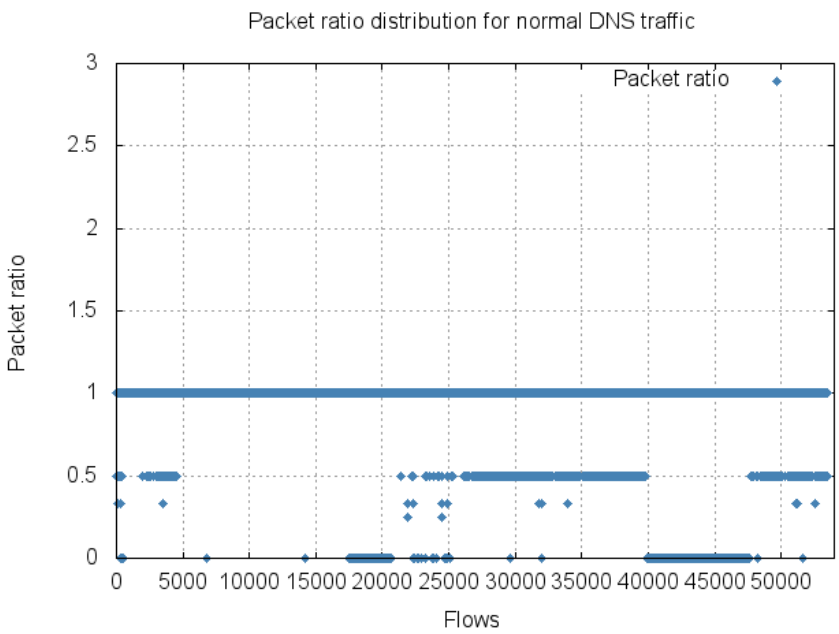


ICMP reverse shell

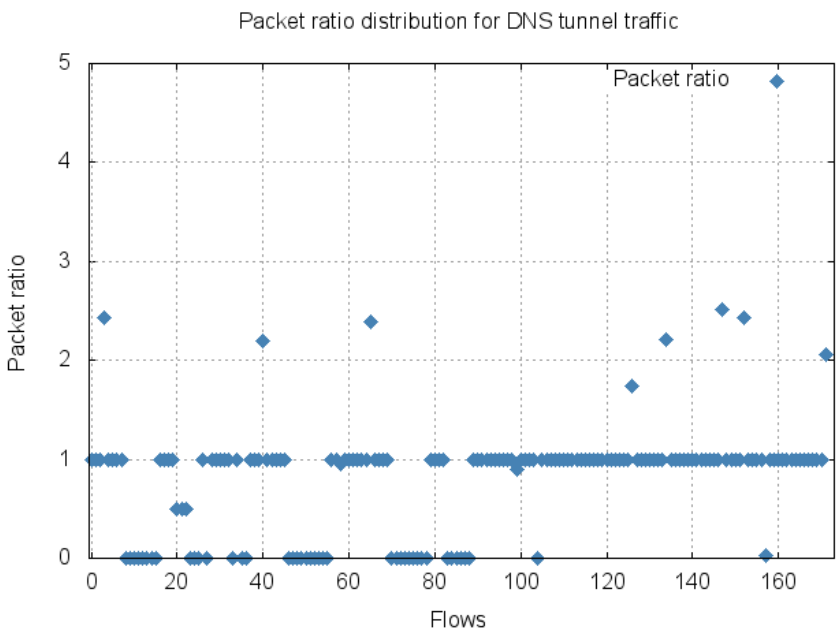
Outline

DNS tunnel

Packet ratio distribution



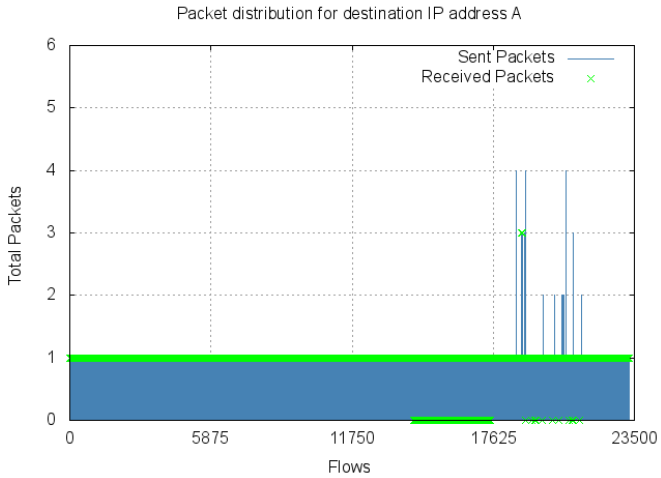
Regular DNS

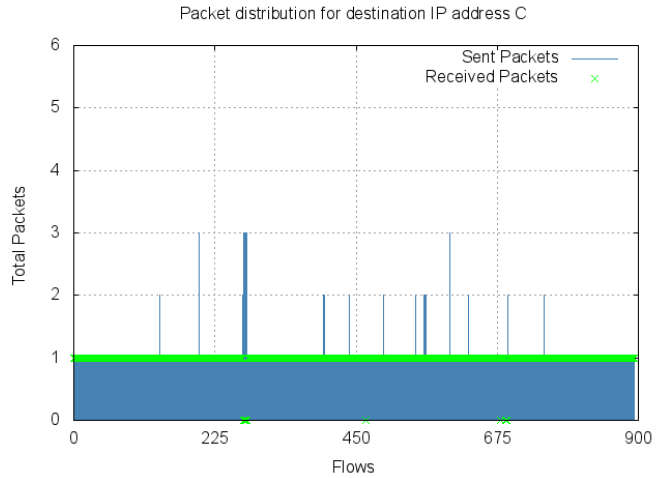


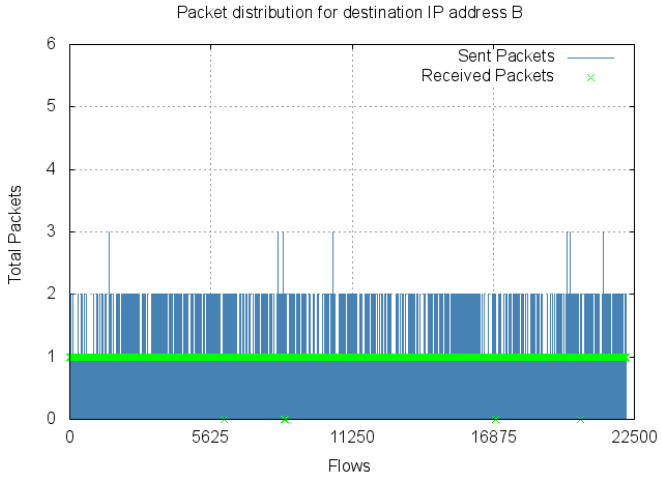
DNS tunnel

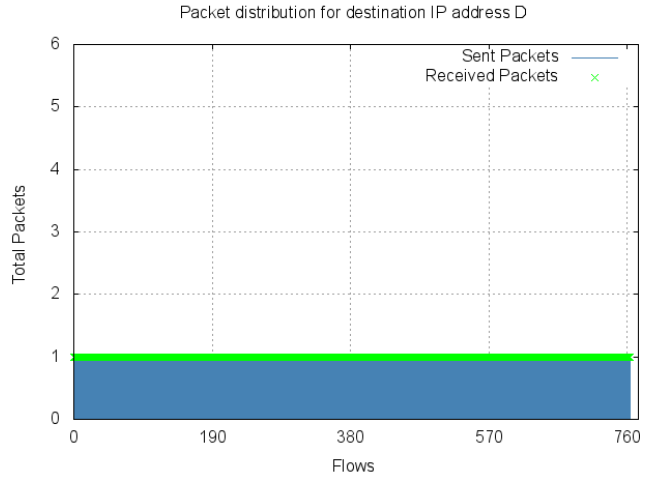
Regular DNS

Packet distribution per unique destination IP



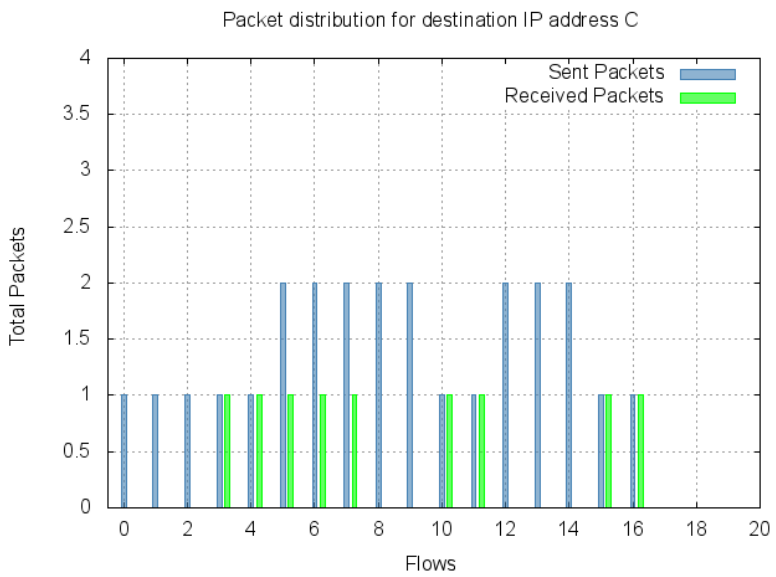




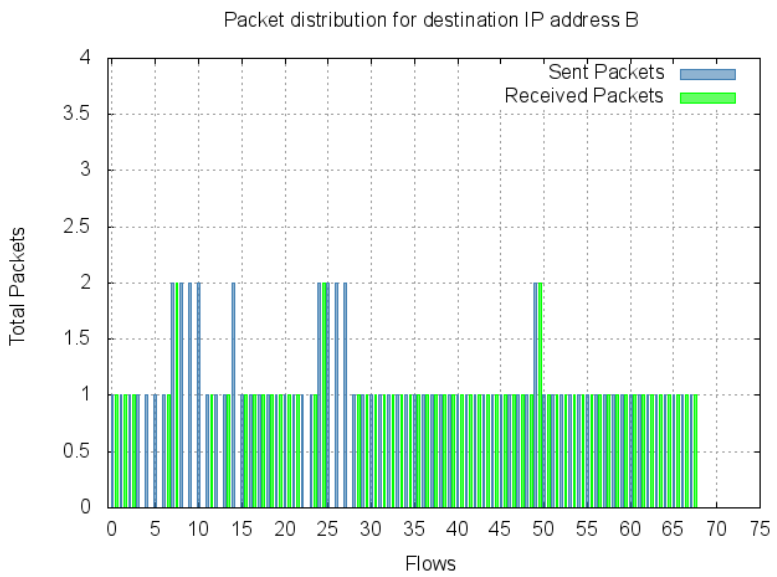


DNS tunnel

Packet distribution per unique destination IP



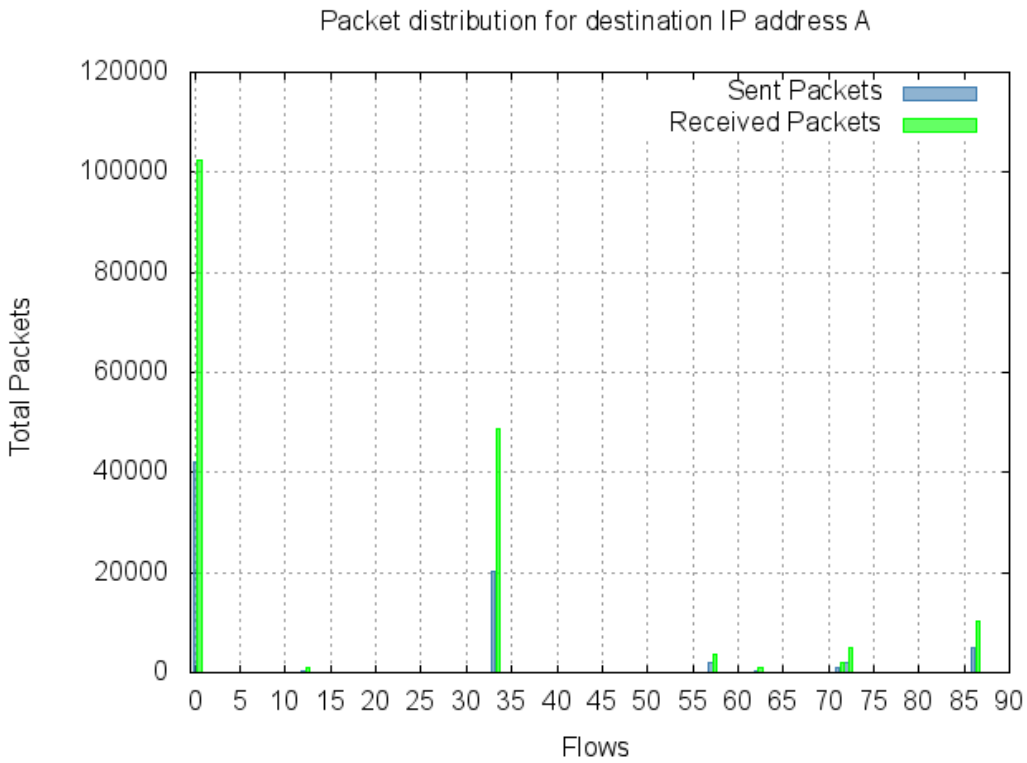
Destination IP A



Destination IP B

DNS tunnel

Packet distribution per unique destination IP



Destination IP C (Tunnel server)

Regular DNS

DNS_QUERY_TYPE analysis

DNS_QUERY_TYPE	# of flows	%	Type
1	40395	75.5	A
2	1807	3.39	NS
6	4	0.007	SOA
12	438	0.08	PTR
16	1	0.002	TXT
28	2461	4.6	AAAA
33	18	0.03	SRV
43	723	1.35	DS
48	8083	15.03	DNSKEY

DNS tunnel

DNS_QUERY_TYPE analysis

DNS_QUERY_TYPE	# of flows	%
12	60	34.88
10	57	33.14
1	26	15.12
0	13	7.56
16	5	2.92
5	3	1.74
15	3	1.74
33	3	1.74
255	1	0.58
28	1	0.58

Outline

HTTP

TCP_FLAGS analysis

- Cumulative OR-ed of TCP_FLAGS for all packets in one flow.
- For regular HTTP traffic, this value is well distributed.
- But, for malicious HTTP traffic, every flow has the TCP_FLAGS value $= 27$

Regular HTTP

TCP_FLAGS analysis

TCP_FLAG	# of flows	Meaning	%
24	22088	ACK+PUSH	55,0727
26	10284	ACK+PUSH+SYN	25,6414
27	5039	ACK+PUSH+SYN+FIN	12,5639
19	2223	ACK+FIN+SYN	5,5427
17	163	ACK+FIN	0,4064
31	162	ACK+PUSH+RST+SYN+FIN	0,4039
30	93	ACK+PUSH+RST+SYN	0,2319
23	38	ACK+RST+SYN+FIN	0,0947
25	15	ACK+PSH+FIN	0,0374
21	1	ACK+RST+FIN	0,0025
18	1	ACK+SYN	0,0025

HTTP

HTTP_METHOD analysis per unique destination IP

Destination IP address	# of Flows with method:			
	GET	POST	HEAD	EMPTY
A	104	-	1722	105
B	114	-	1482	107
C	267	25	849	94
D	-	-	-	979
E	18	-	729	3
F	700	-	-	10
G	628	-	-	33
H	-	-	-	618
I	-	-	555	4
J	371	136	-	39

HTTP

HTTP_METHOD analysis per unique destination IP

- For HTTP reverse shell traffic, the amount of POST and GET methods per unique destination IP address is about 50% each.

Algorithms

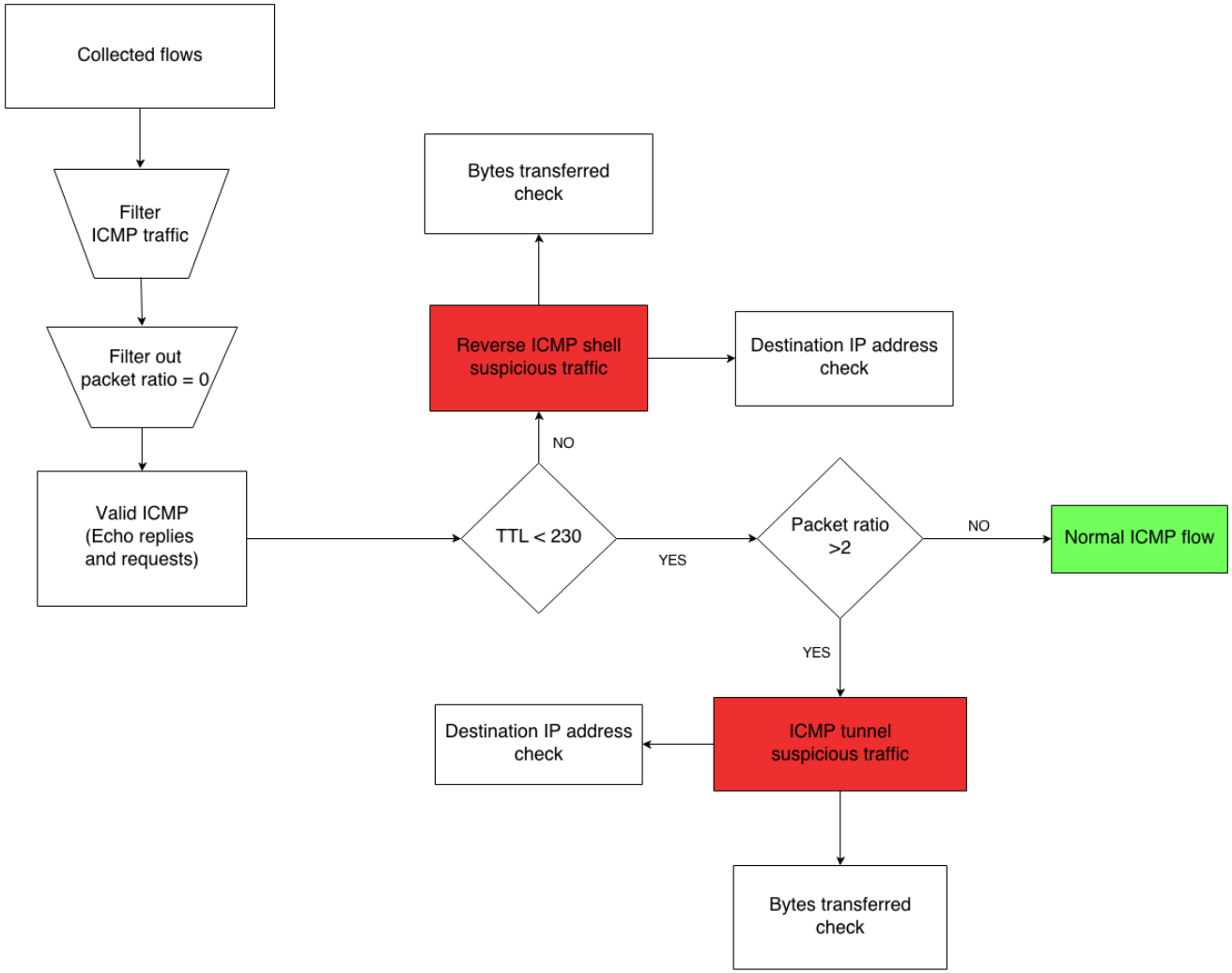
Description

Using a data-set provided by the sponsoring company.

- HTTP traffic generated by 150 different web crawlers (64095 flows)
- DNS traffic (35219 flows)
- ICMP traffic (12352 flows)

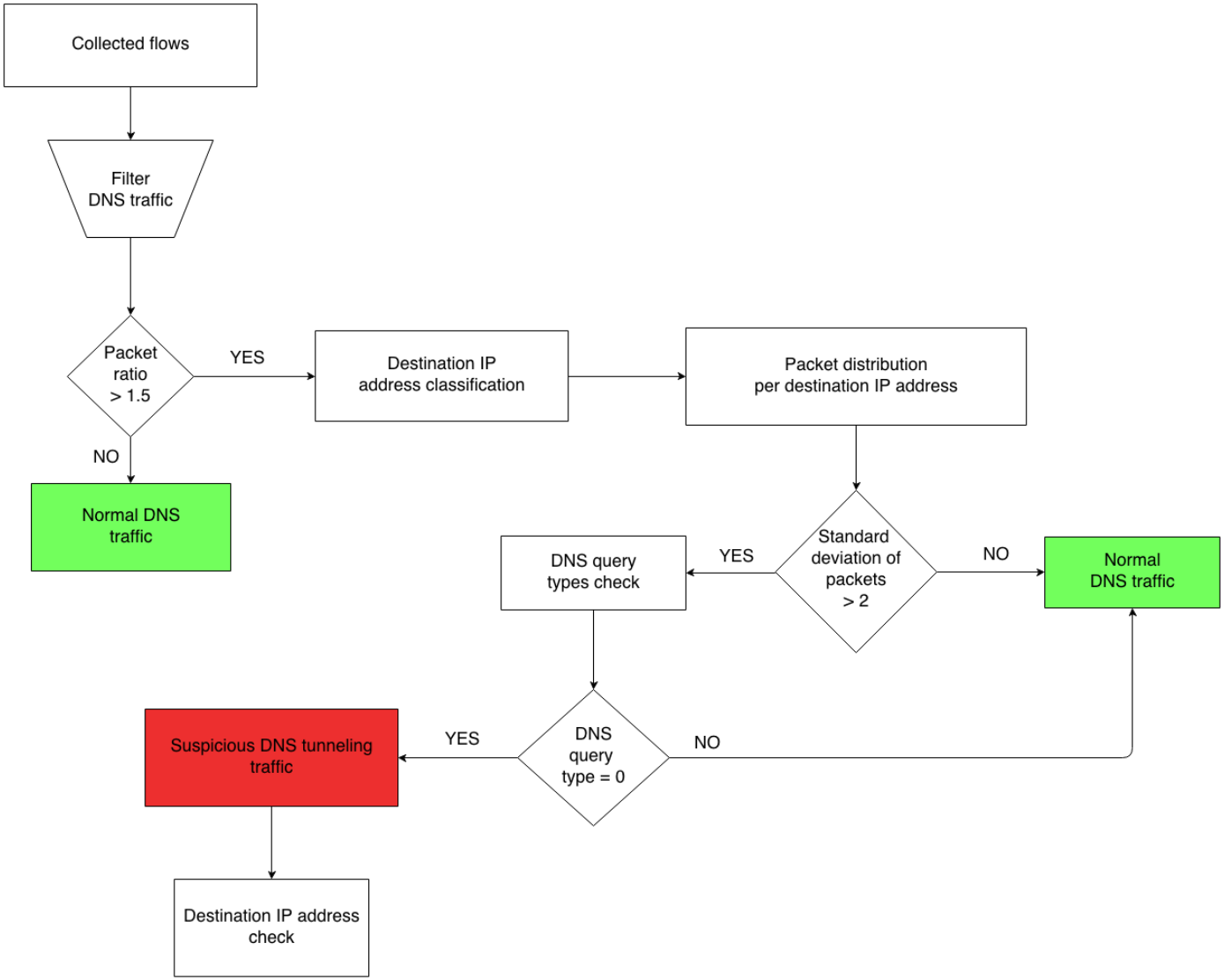
Proposed alorithms

ICMP



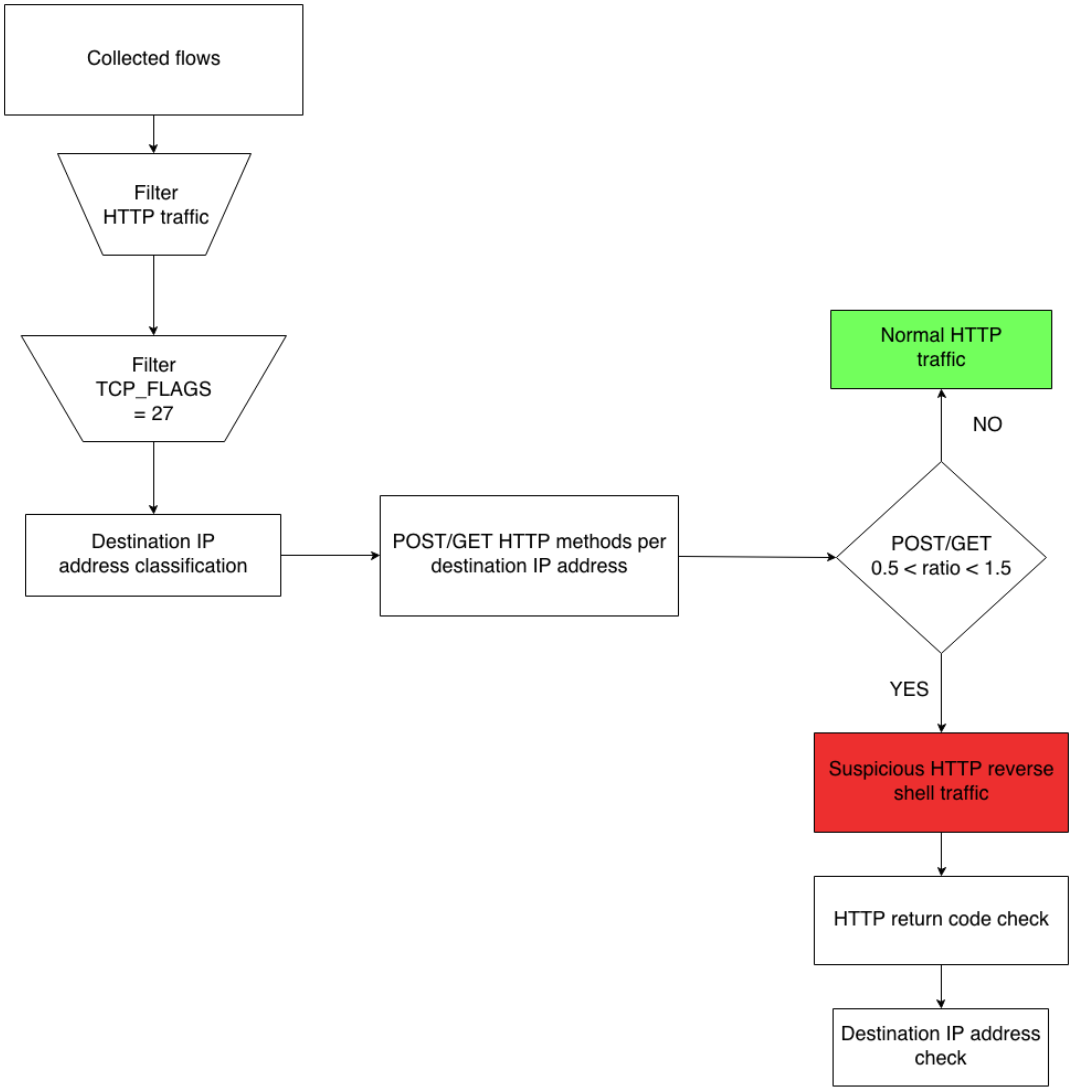
Proposed alorithms

DNS



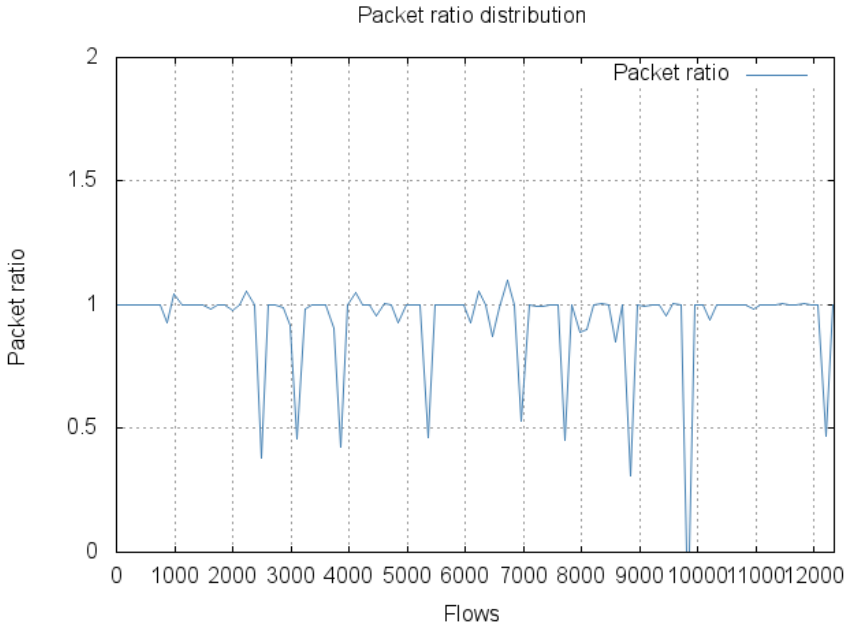
Proposed alorithms

HTTP

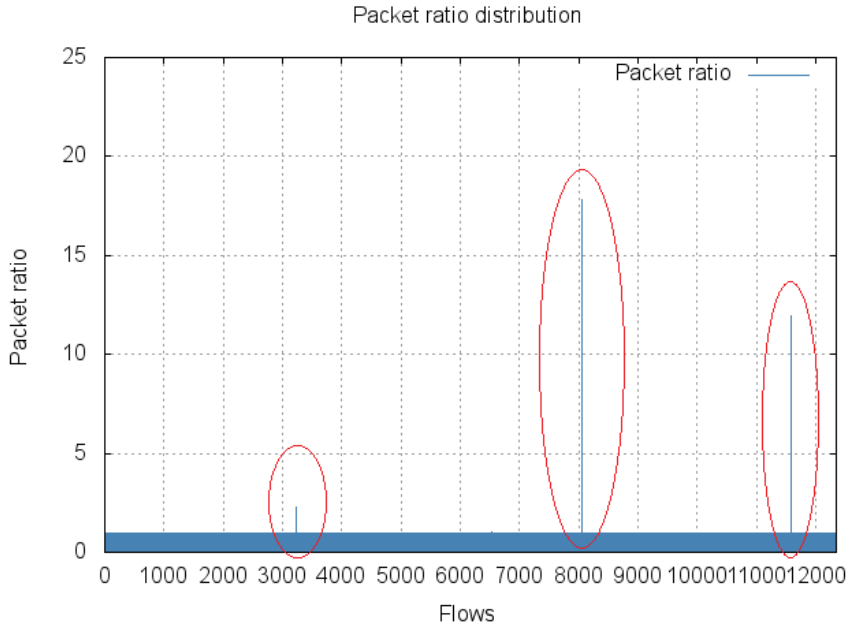


Outline

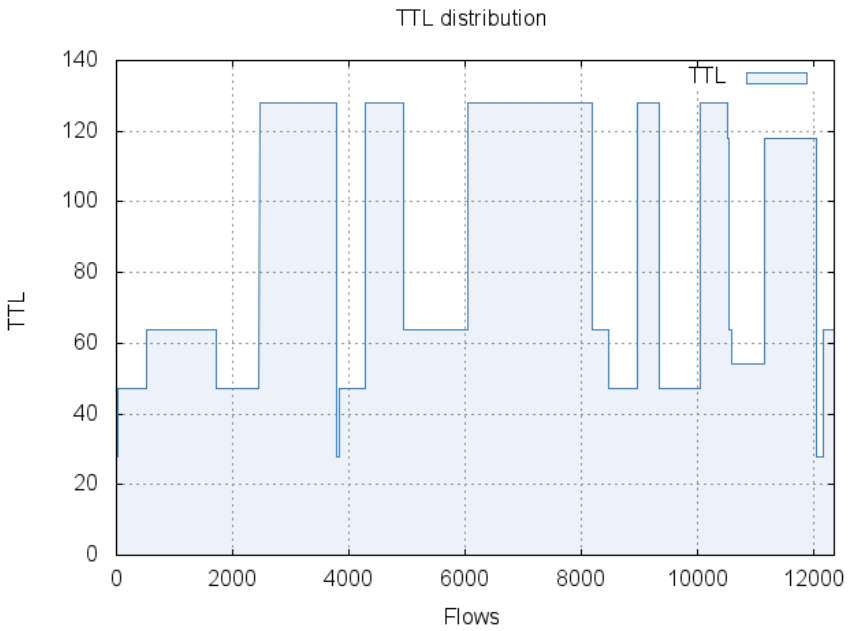
ICMP Tunnel



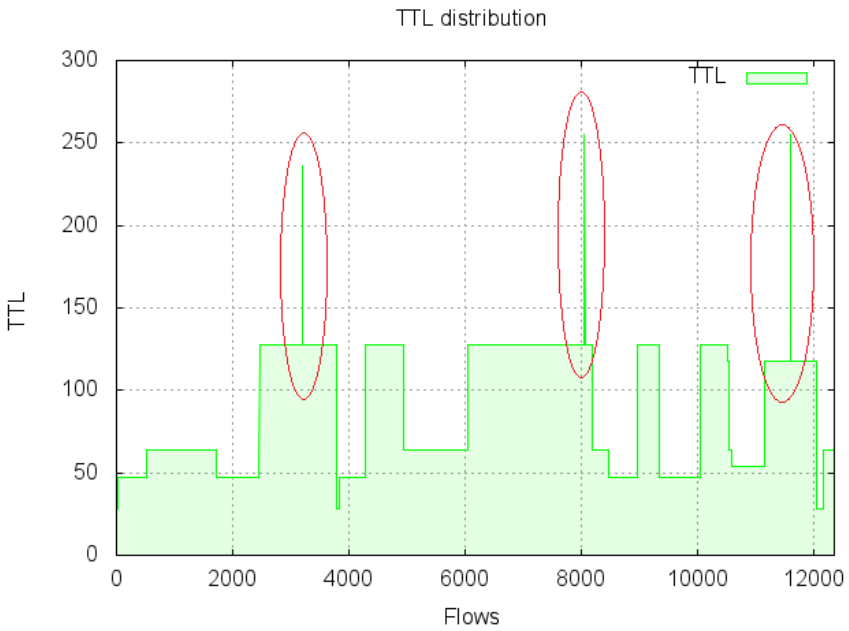
Before injection



After injection



Before injection



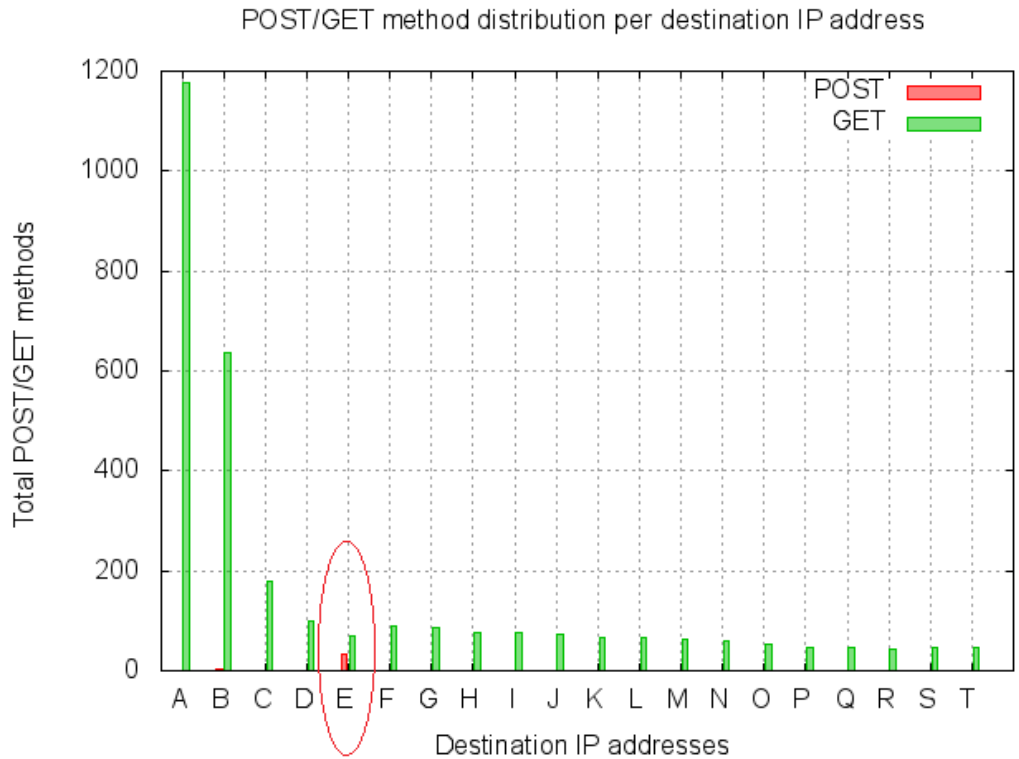
After injection

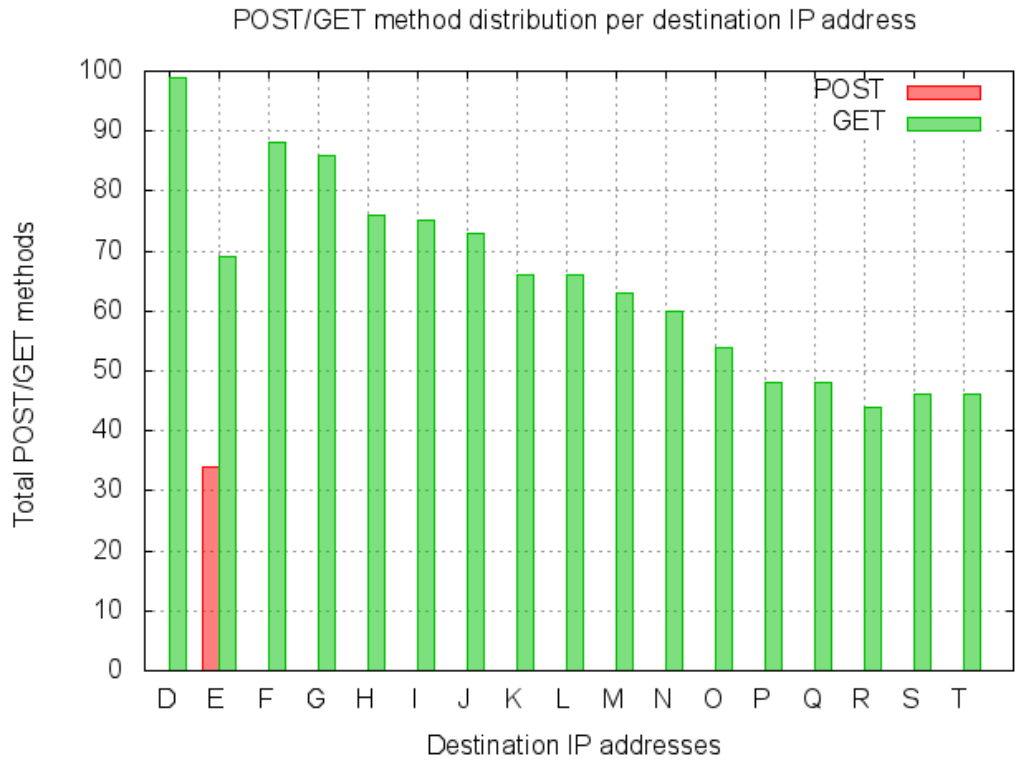
Outline

- Analysis on the packet distribution per unique destination IP address shows suspicious standard deviation values for specific flows.
- DNS_QUERY_TYPE field is effective to retrieve unusual values.

Outline

- After filtering every flow with TCP_FLAGS field = 27


POST/GET method distribution per destination IP address
Total POST/GET methods
1200
1000
800
600
400
200
0
POST
GET
A B C D E F G H I J K L M N O P Q R S T
Destination IP addresses


POST/GET method distribution per destination IP address
Total POST/GET methods
Destination IP addresses
POST
GET
0
10
20
30
40
50
60
70
80
90
100
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T

Conclusions

- It is possible to detect the tested network-based covert channels by using flow data.
- By establishing a base line behaviour, it is possible to compare between regular and suspicious behavior.
- Even though, flow-data does not give an insight on the payload of a packet, is still a powerful tool for security analysis.

Future work

- Implement the proposed algorithms as a script or programming language and with live flow-data
- Test more tools for similar behaviour patterns
- Test other protocols
- Test a bigger data-set for possible false positives
- Compare results with other types of malicious traffic
- Investigate flow-data with network-based covert timing channels

Q&A

Questions?

- Covert Storage Channel
  - Carries information inside protocol fields
- Covert Timing Channel
  - They use time emission between packets.
  - A time interval can be defined: if a packet is sent during the interval, this codes a one, if no packet is sent this codes a zero.