



# IP Spoofing

(what is it, how it allows largest attacks and how to fix it)

Marek Majkowski

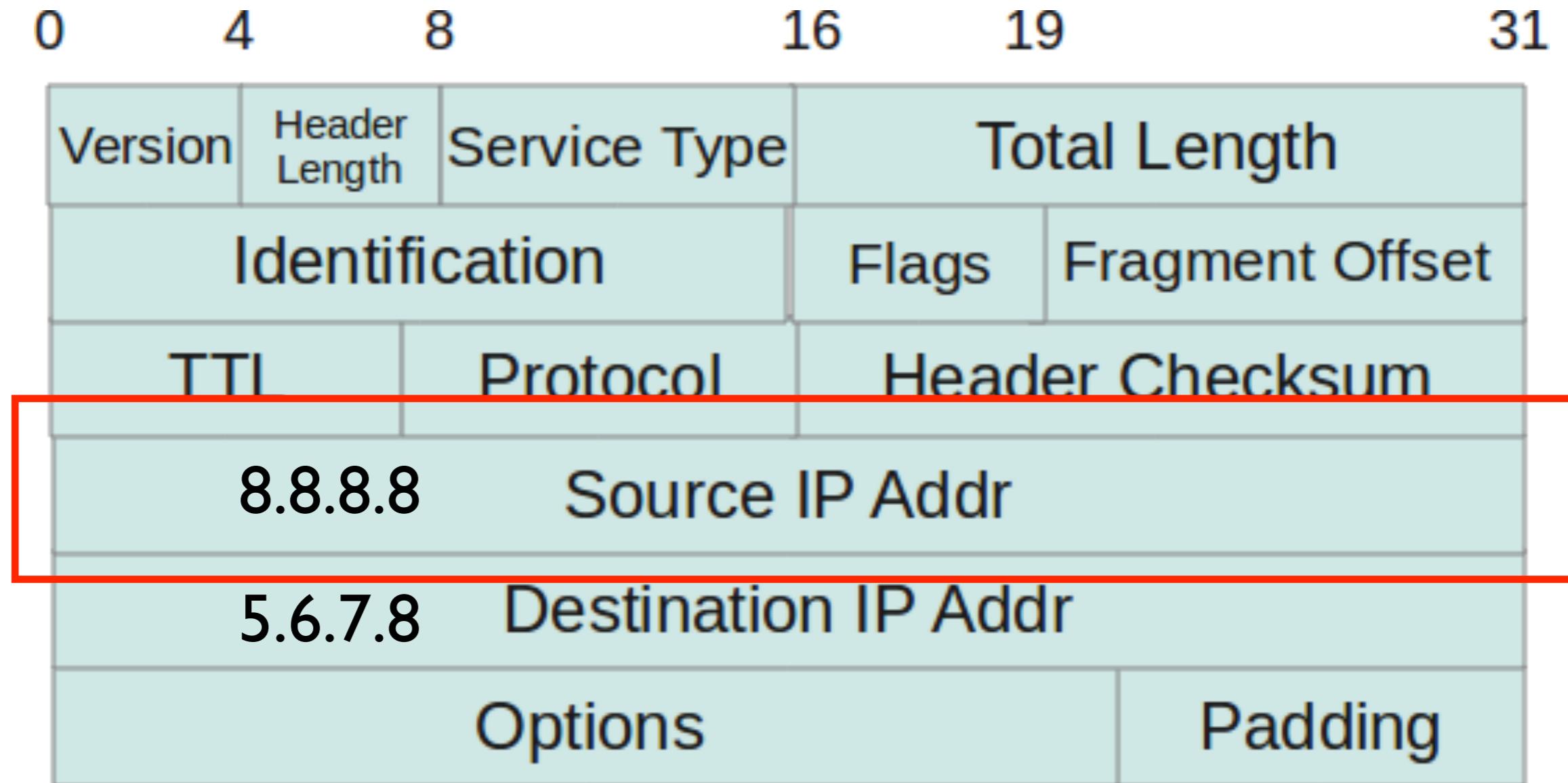
What is it?

A photograph of two pugs sitting on a grassy lawn. The pug on the left is white with black spots on its ears and a blue collar with a white tag that says 'BAILEY'. The pug on the right is mostly white with some dark spots on its face. They are both looking towards the camera.

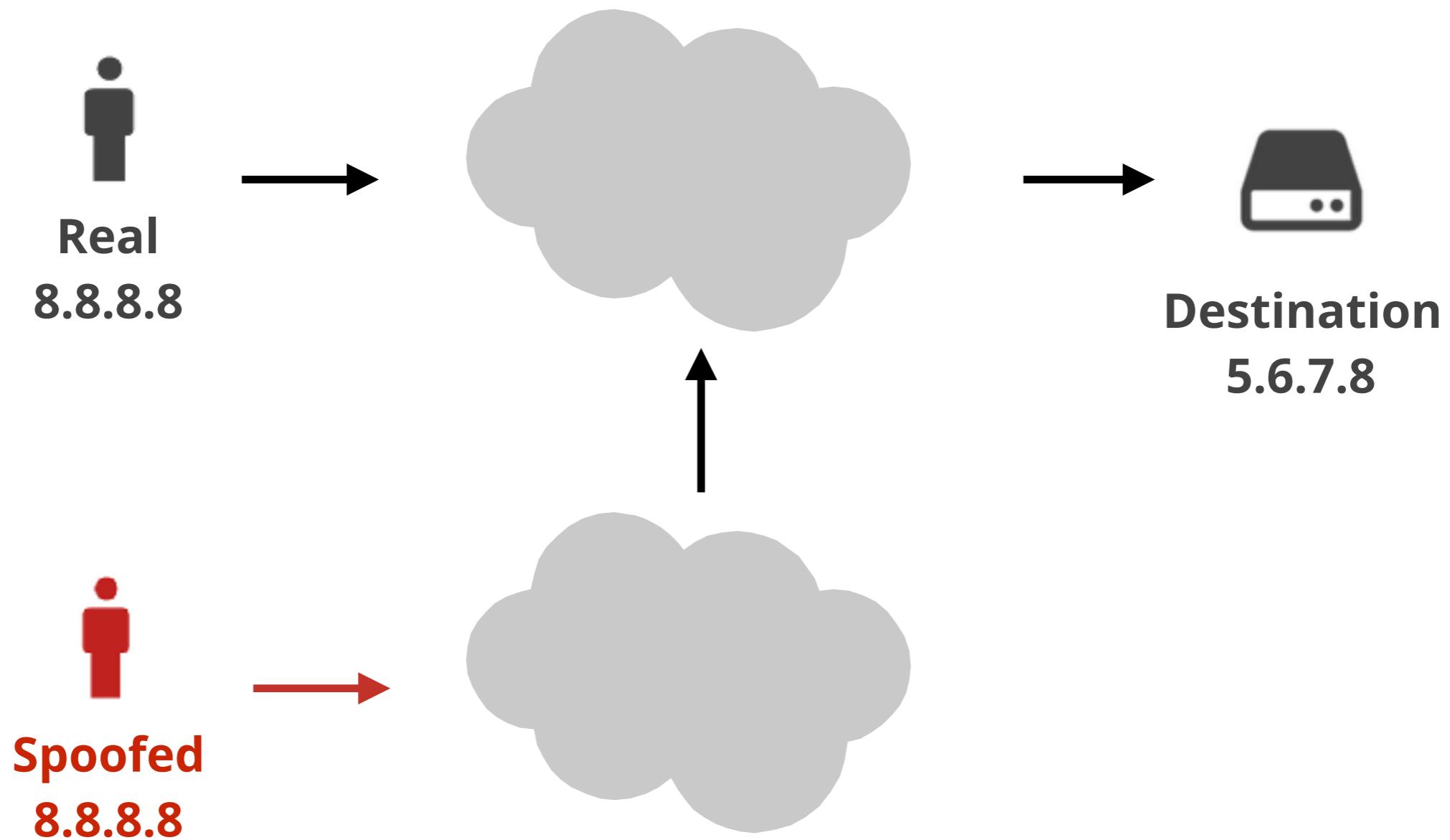
# IP Spoofing

(source: [DaPuglet](#))

# IP Spoofing



# Enables impersonation



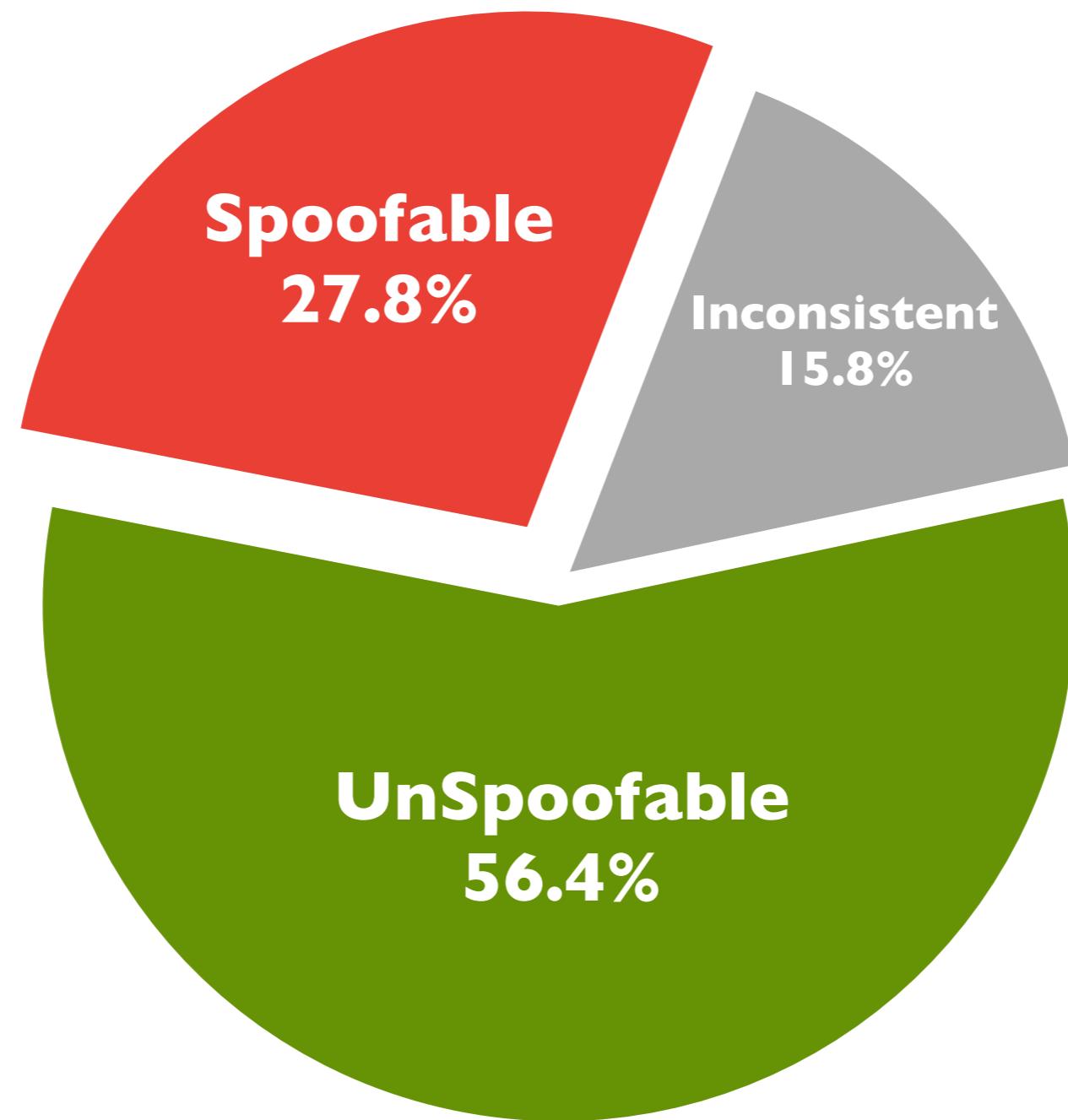


IP Spoofing is still a problem

# May 2000: BCP38

[spoofercaida.org](http://spoofercaida.org)

## Measured Autonomic Systems

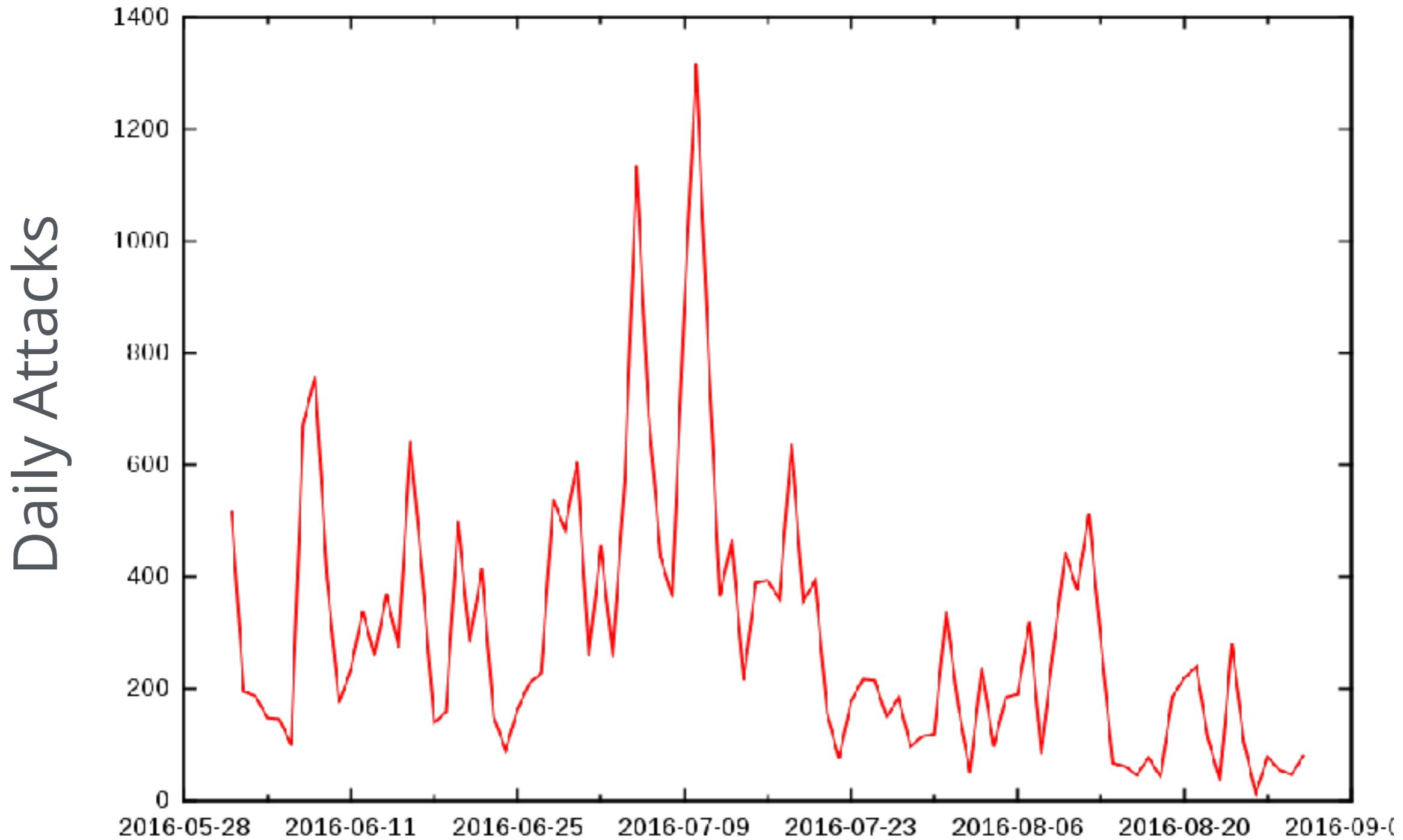


It allows largest attacks

# Global network



# Daily attacks



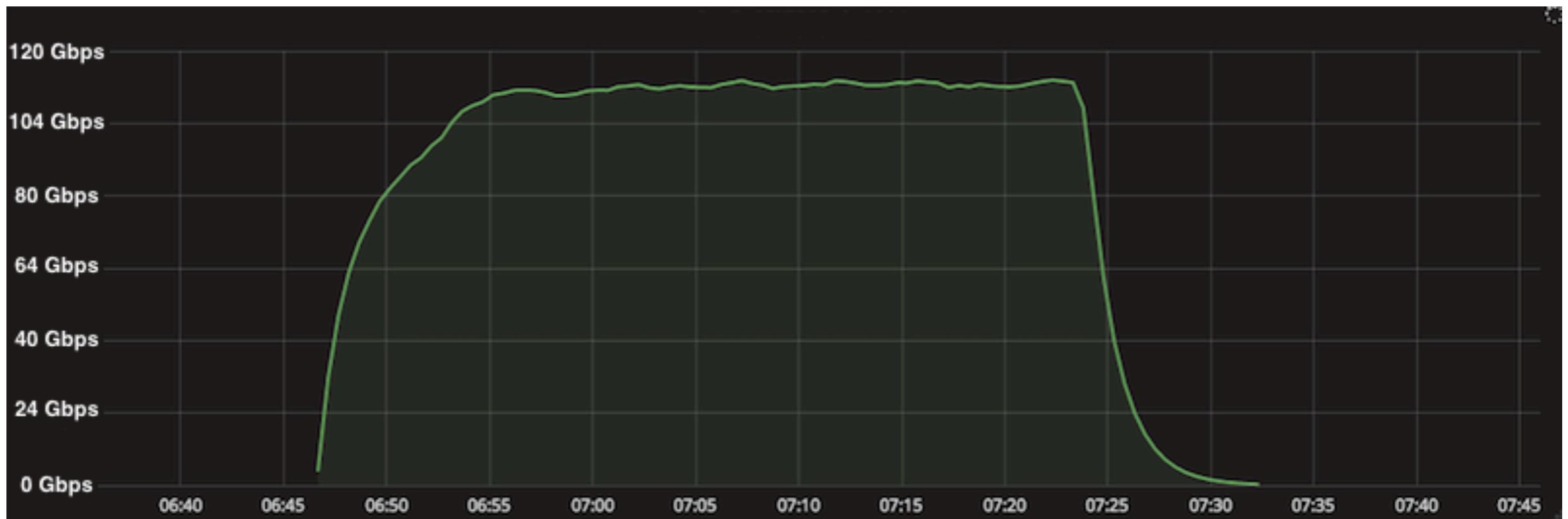
Some are super large

# Direct: SYN Flood

## "winter of attacks"



# Amplification: SSDP



A photograph of two pugs sitting on a grassy lawn. The pug on the left is white with black spots on its ears and a blue collar with a white tag that says 'BAILEY'. The pug on the right is mostly white with some dark spots on its face. They are both looking towards the camera.

# IP Spoofing

(source: [DaPuglet](#))

# IP Spoofing

1. Tracing back is impossible
2. Allows sophisticated attacks

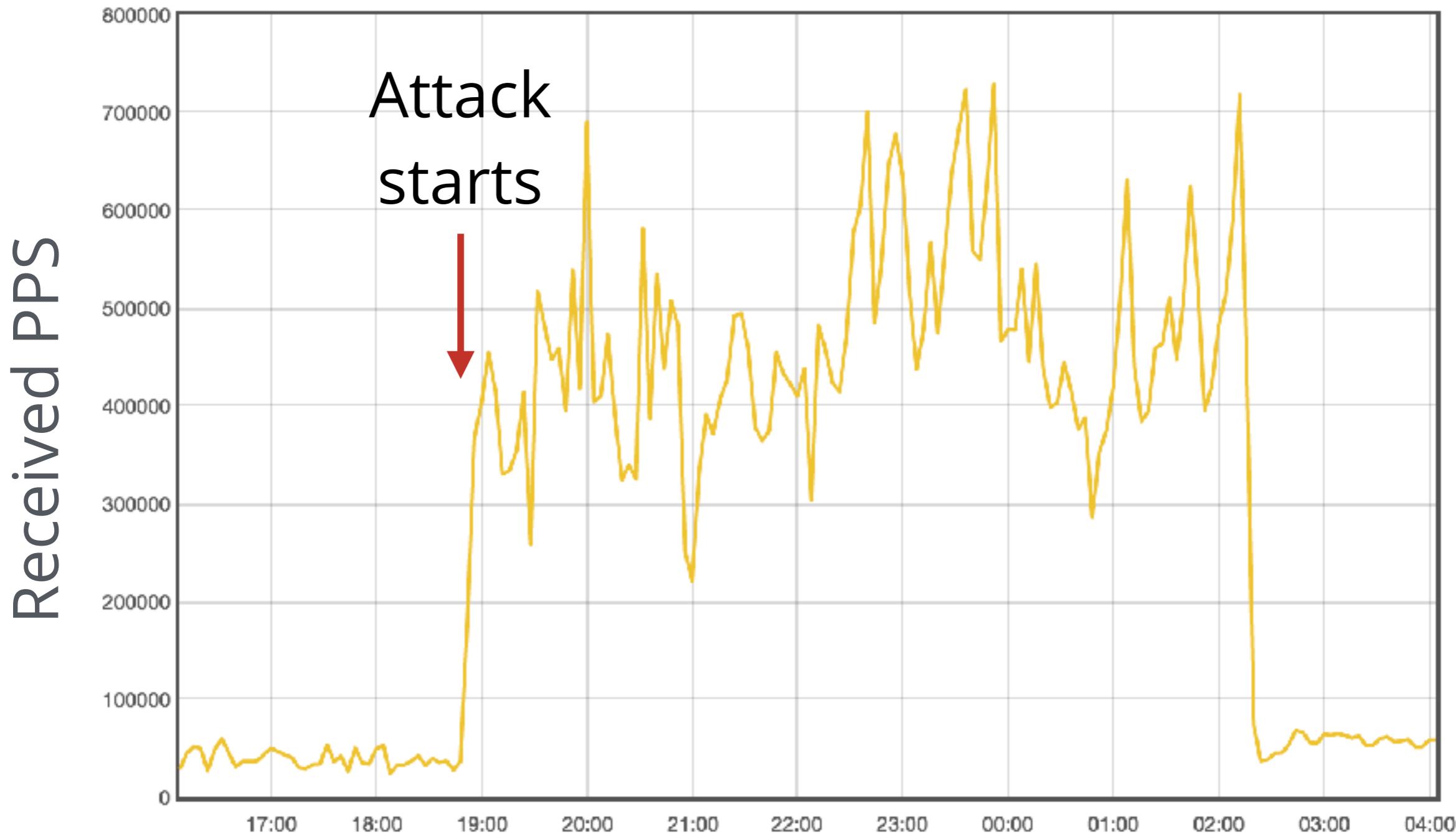
# Direct: SYN Flood



## 2. "Winter of attacks"



# Tracing the attack

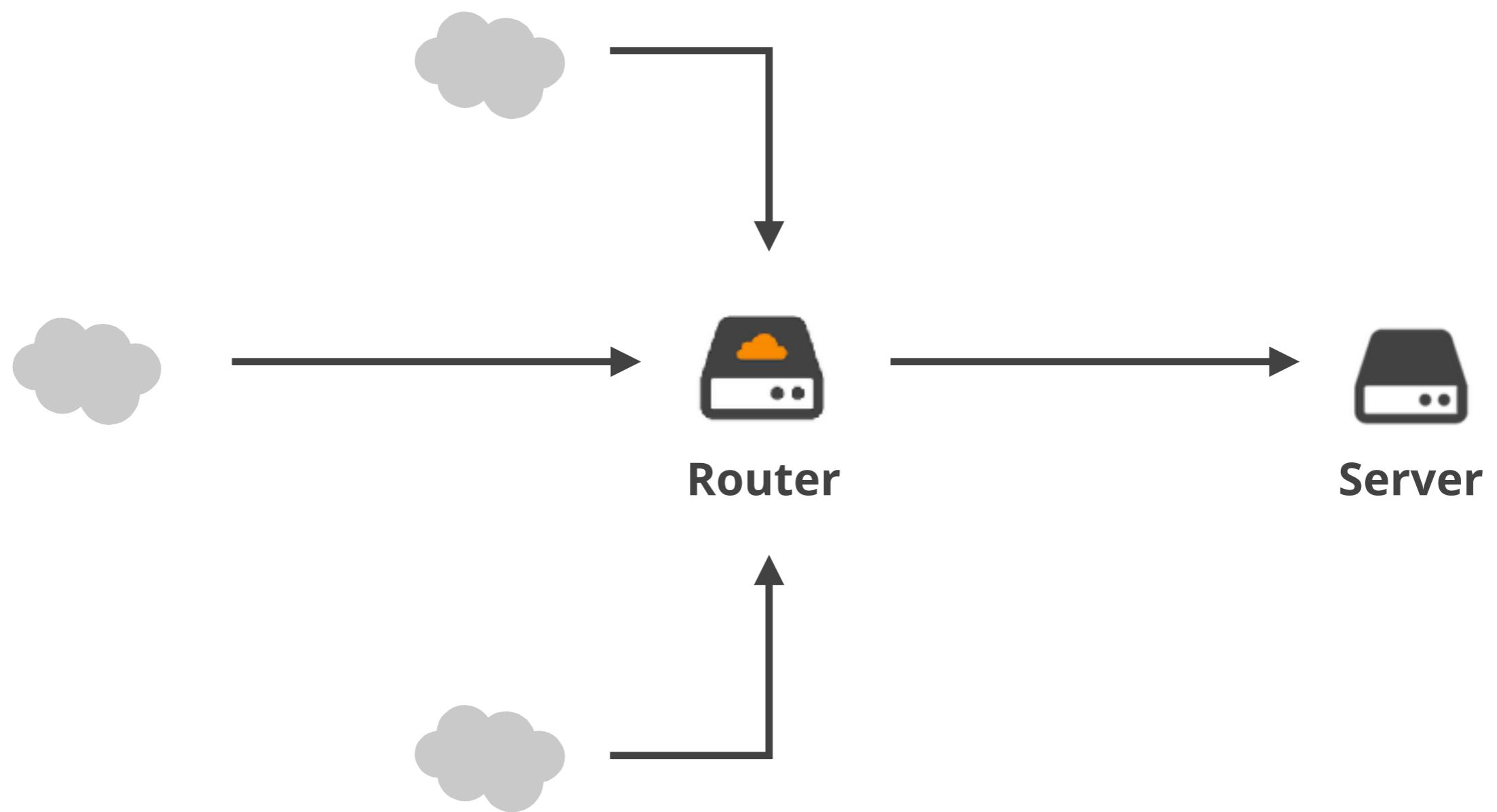


# Tcpdump

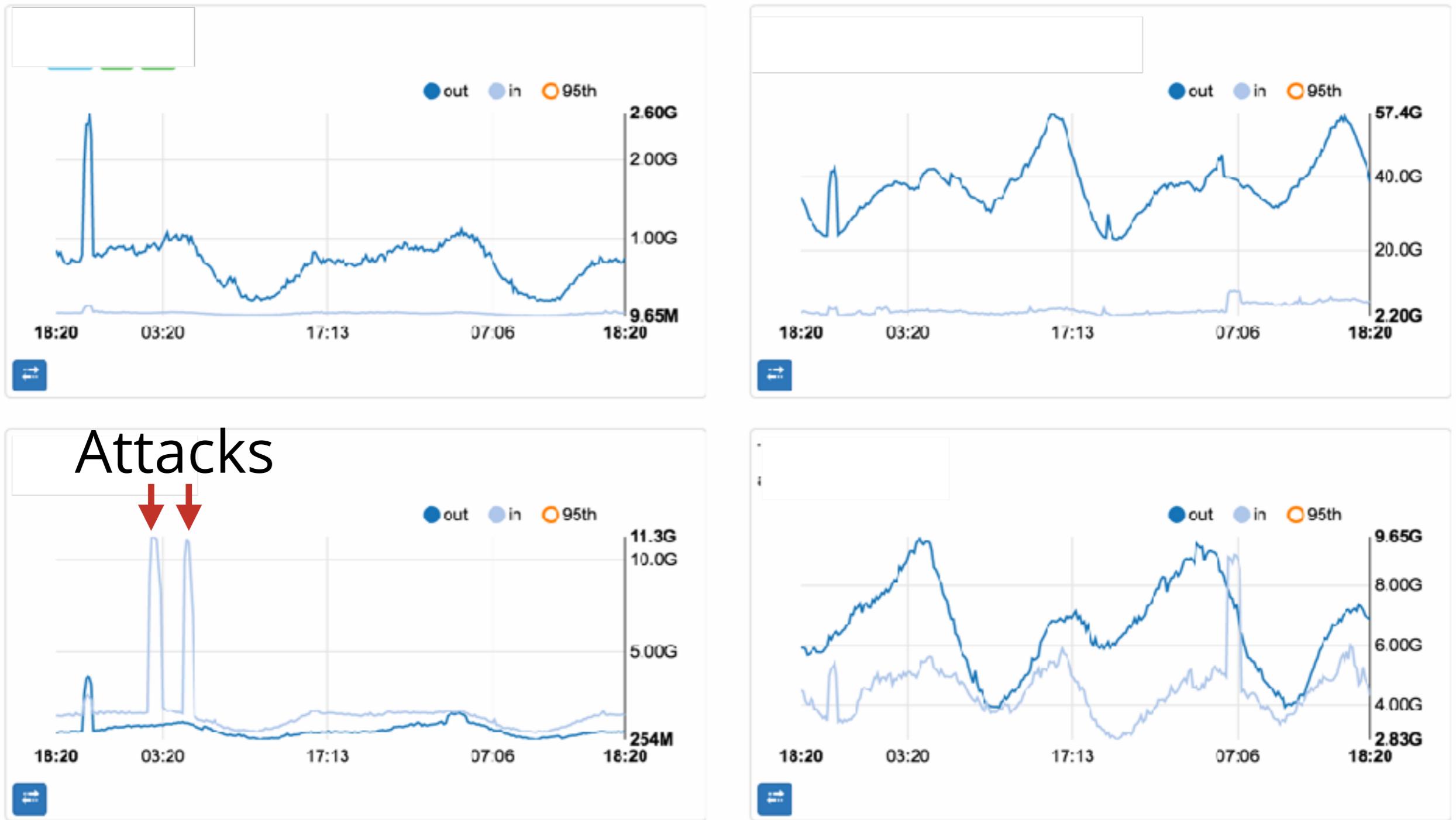
```
$ tcpdump -ni eth0 -c 100

IP 94.242.250.109.47330 > 1.2.3.4:80: Flags [S], seq 1444613291, win 63243
IP 188.138.1.240.61454 > 1.2.3.4:80: Flags [S], seq 1995637287, win 60551
IP 207.244.90.205.17572 > 1.2.3.4:80: Flags [S], seq 1523683071, win 61607
IP 94.242.250.224.65127 > 1.2.3.4:80: Flags [S], seq 928944042, win 61778
IP 207.244.90.205.43074 > 1.2.3.4:80: Flags [S], seq 137074667, win 63891
IP 64.22.81.44.23865 > 1.2.3.4:80: Flags [S], seq 838596928, win 63808
IP 188.138.1.137.23373 > 1.2.3.4:80: Flags [S], seq 593106072, win 60272
IP 207.244.90.205.39653 > 1.2.3.4:80: Flags [S], seq 47289666, win 63210
IP 208.66.78.204.64197 > 1.2.3.4:80: Flags [S], seq 1850809890, win 62714
IP 207.244.90.205.33108 > 1.2.3.4:80: Flags [S], seq 319707959, win 63351
IP 207.244.90.205.6937 > 1.2.3.4:80: Flags [S], seq 1591500126, win 63902
IP 213.152.180.151.60560 > 1.2.3.4:80: Flags [S], seq 1902119375, win 62511
IP 64.22.79.127.11061 > 1.2.3.4:80: Flags [S], seq 1456438676, win 62148
```

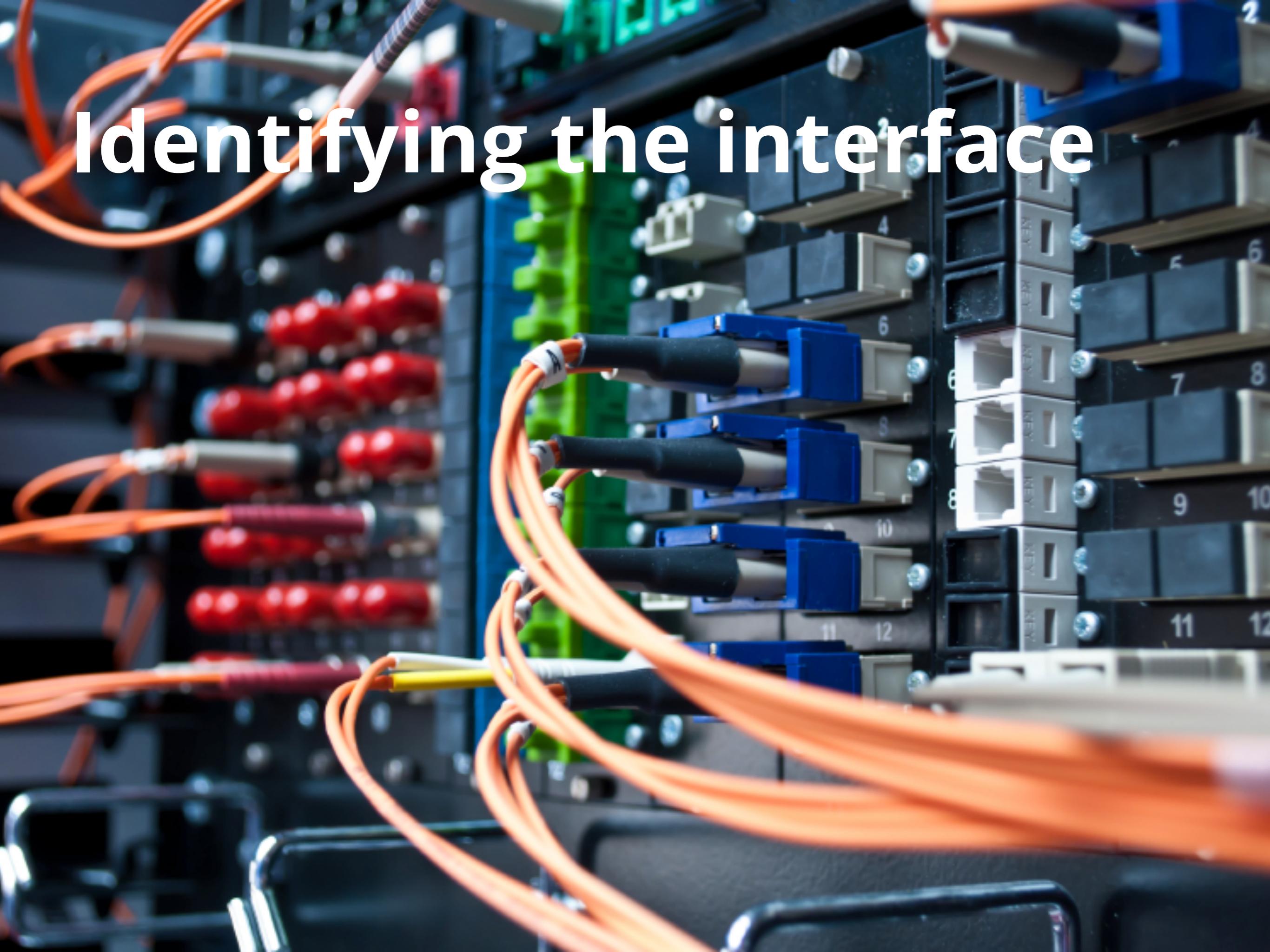
# Which router iface is it from?



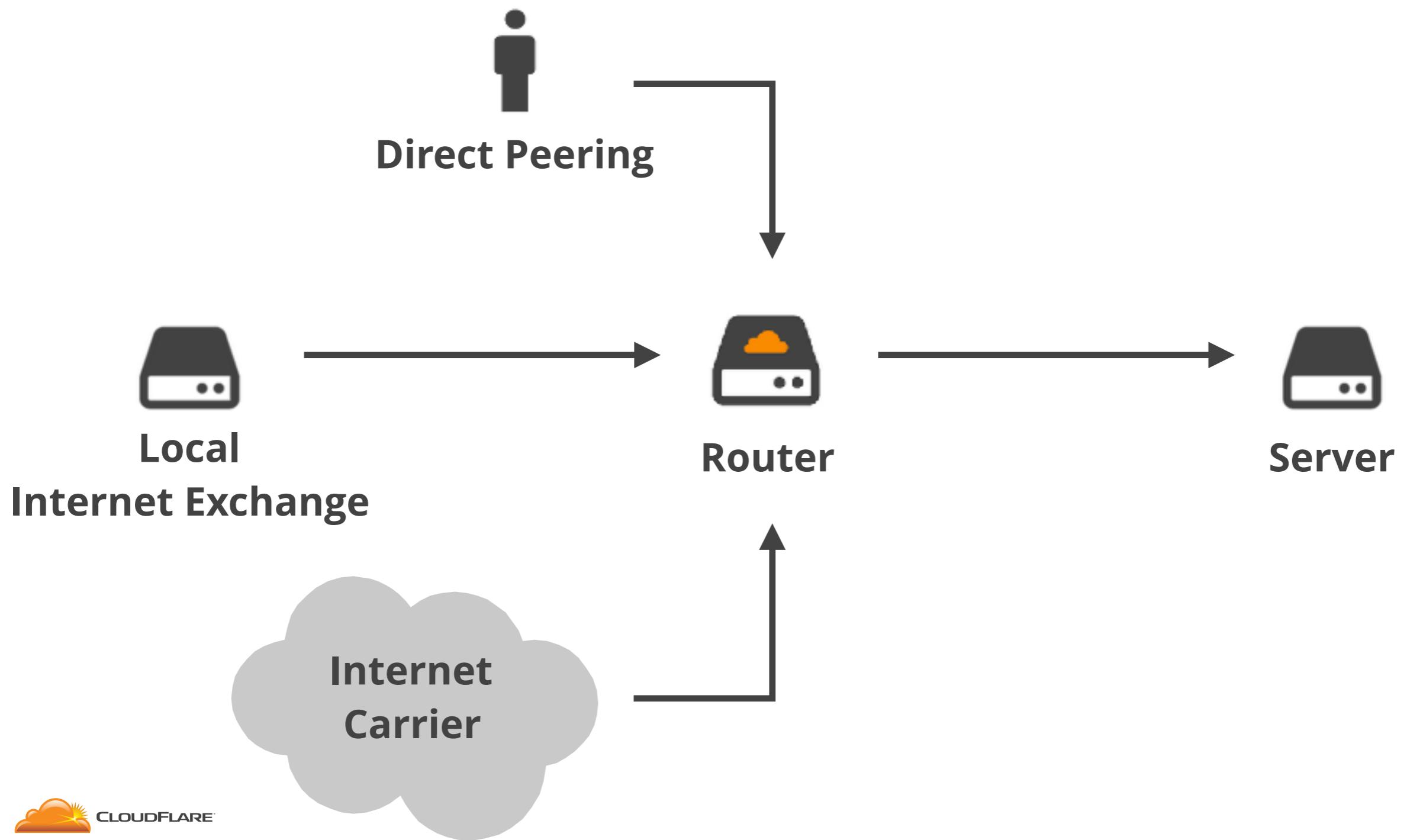
# Identifying interface



# Identifying the interface



# Other side of the cable



# 1. Direct Peering



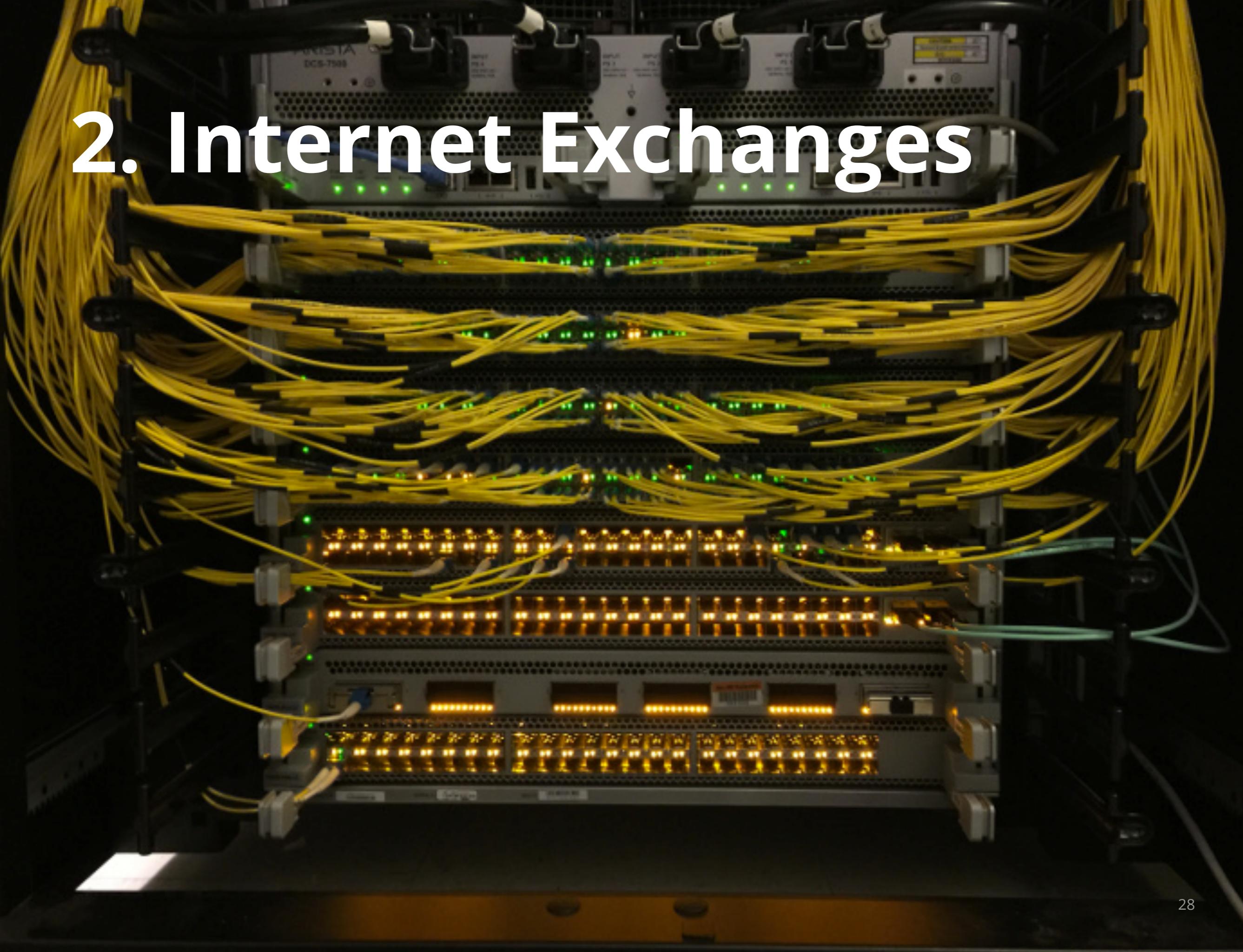
## 2. Internet Exchange



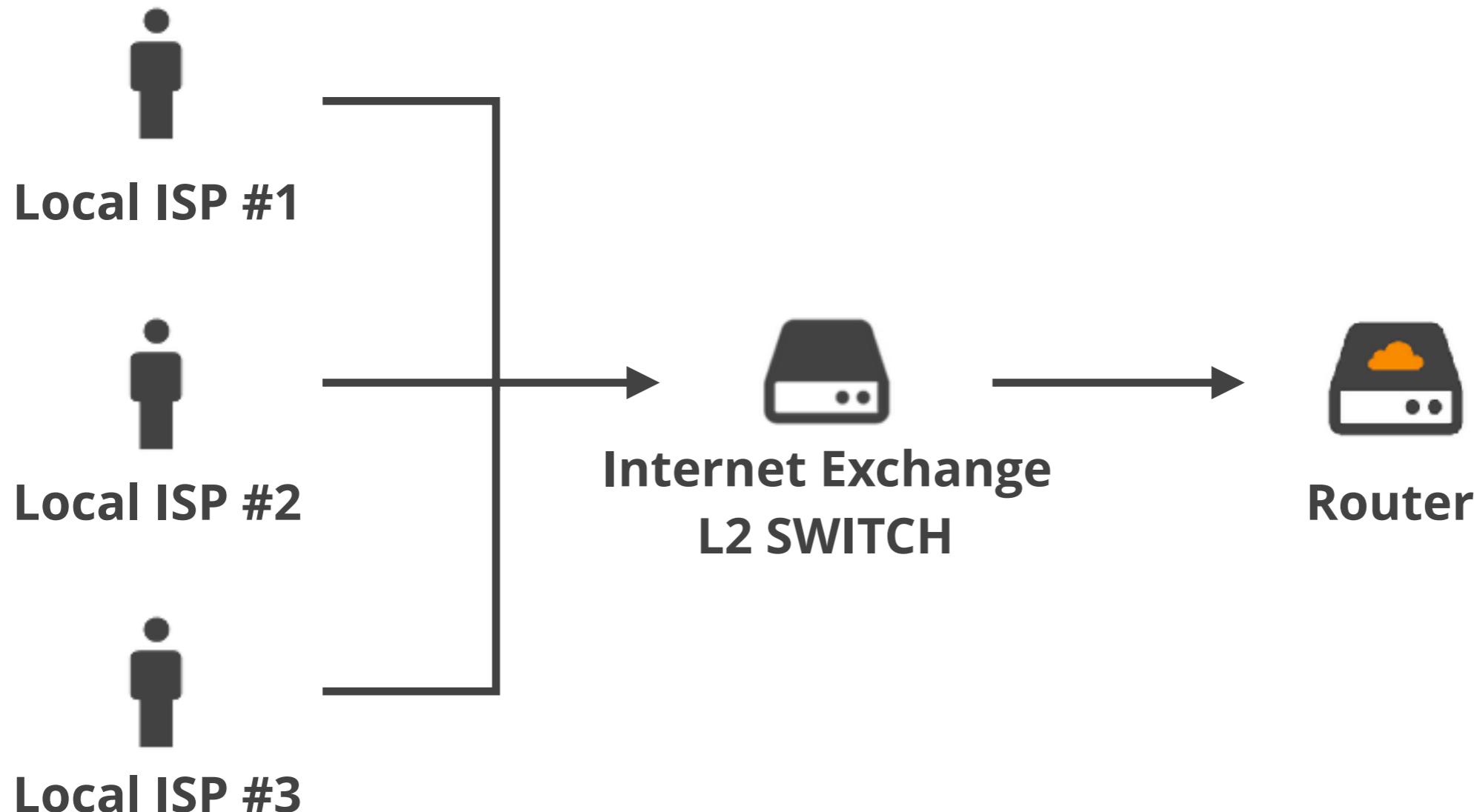
## 3. Internet Carrier



## 2. Internet Exchanges



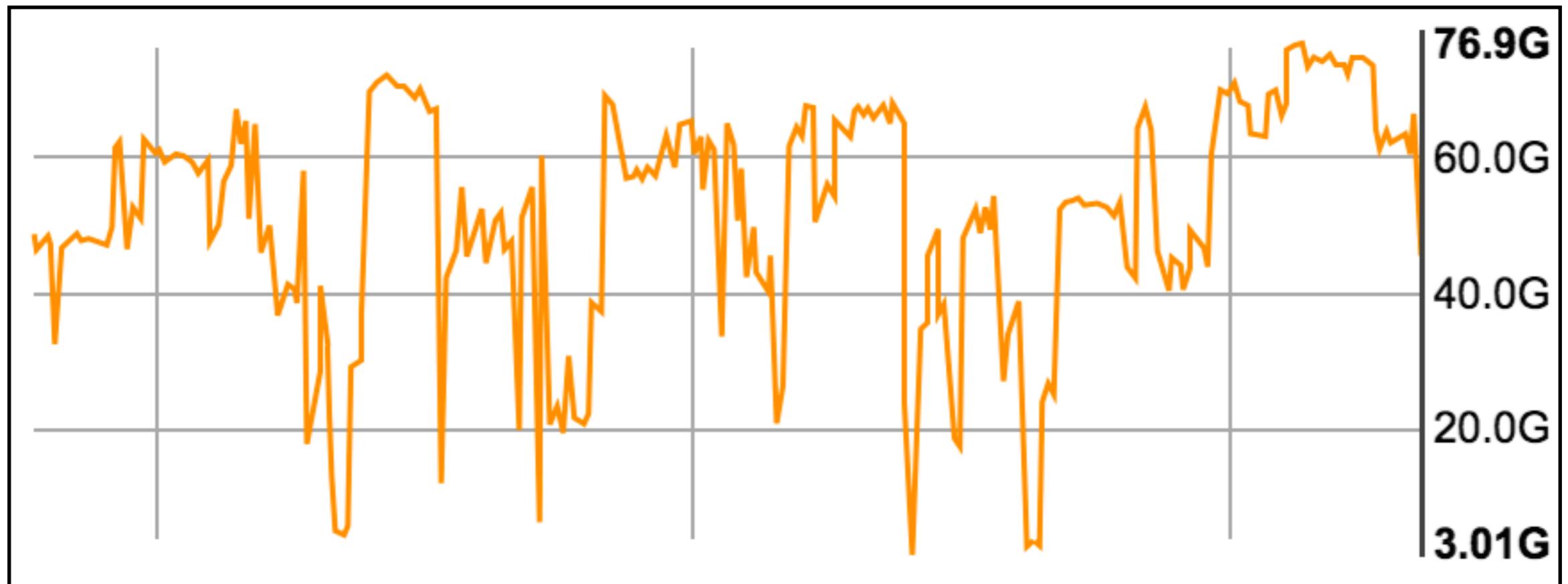
## 2. Internet Exchanges



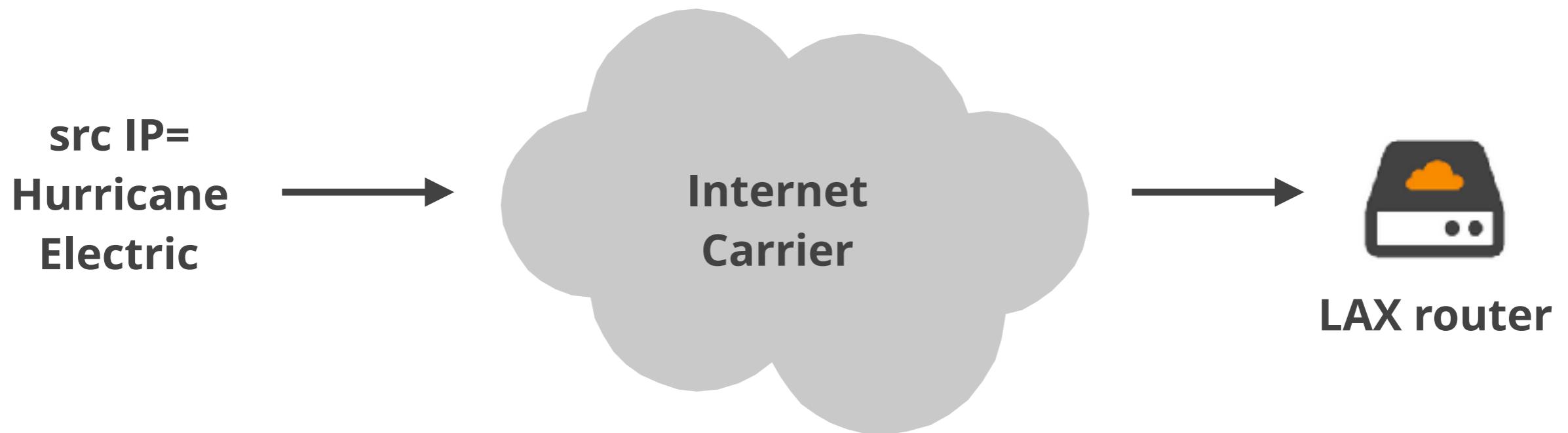
# 3. Internet Carriers



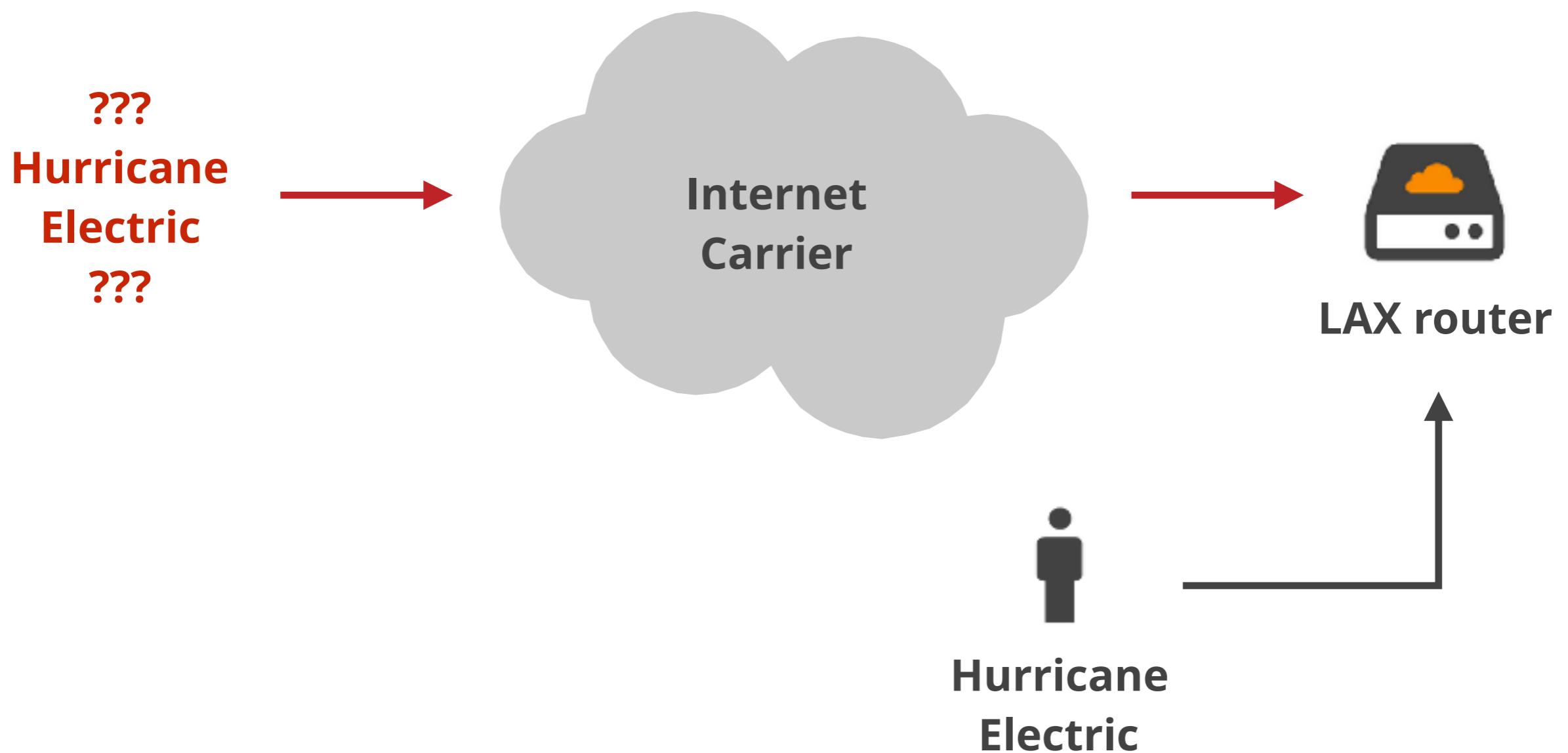
# “Winter of attacks”



# “Winter of attacks”



# “Winter of attacks”



Lack of attribution

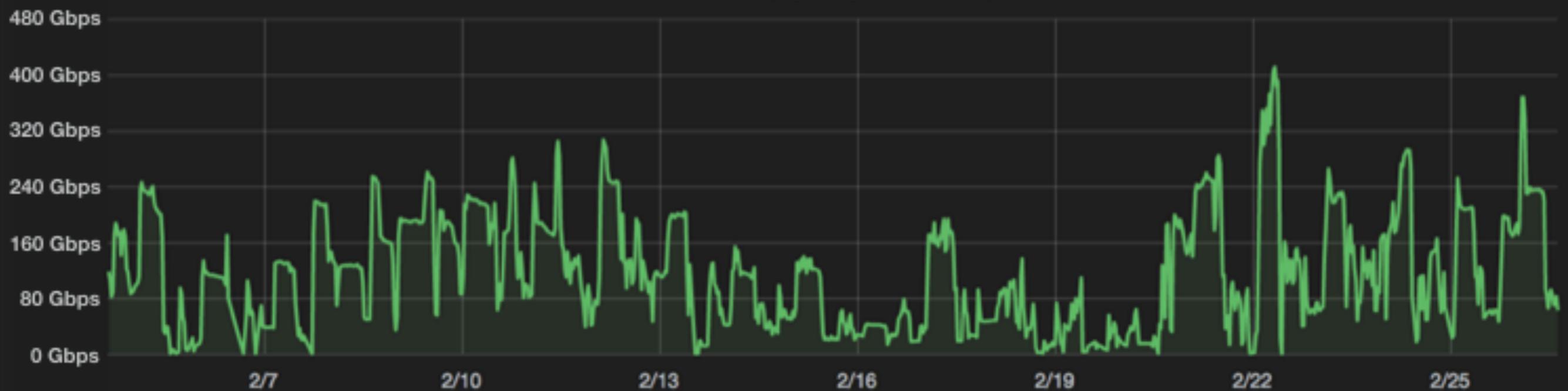
==

impossible to fight

Attack traffic PPS



Attack traffic (Bytes per second)



# Blocked with BPF

```
iptables -A INPUT \
--dst 1.2.3.4 \
-p udp --dport 53 \
-m bpf --bytecode "14,0 0 0 20,177 0 0 0,12 0 0 0,7
0 0 0,64 0 0 0,21 0 7 124090465,64 0 0 4,21 0 5
1836084325,64 0 0 8,21 0 3 56848237,80 0 0 12,21 0 1
0,6 0 0 1,6 0 0 0" \
-j DROP
```

# BPF bytecode

```
lidx 4*([14]&0xf)
ld #34
add x
tax
lb_0:
ldb [x + 0]
add x
add #1
tax
ld [x + 0]
jneq #0x07657861, lb_1
ld [x + 4]
jneq #0x6d706c65, lb_1
ld [x + 8]
jneq #0x03636f6d, lb_1
ldb [x + 12]
jneq #0x00, lb_1
ret #1
lb_1:
ret #0
```

# Introducing the BPF Tools

03 Jul 2014 by [Marek Majkowski](#).

In a recent article I described the basic concepts behind the use of Berkeley Packet Filter (aka BSD Packet filter or BPF) bytecode for high performance packet filtering, and the `xt_bpf` iptables module. In this post I'll explain how we use BPF and `xt_bpf` as one tool to deal with large scale DDoS attacks.

And, today, CloudFlare is open sourcing the tools we've created to generate and deploy BPF rules.

## The Code

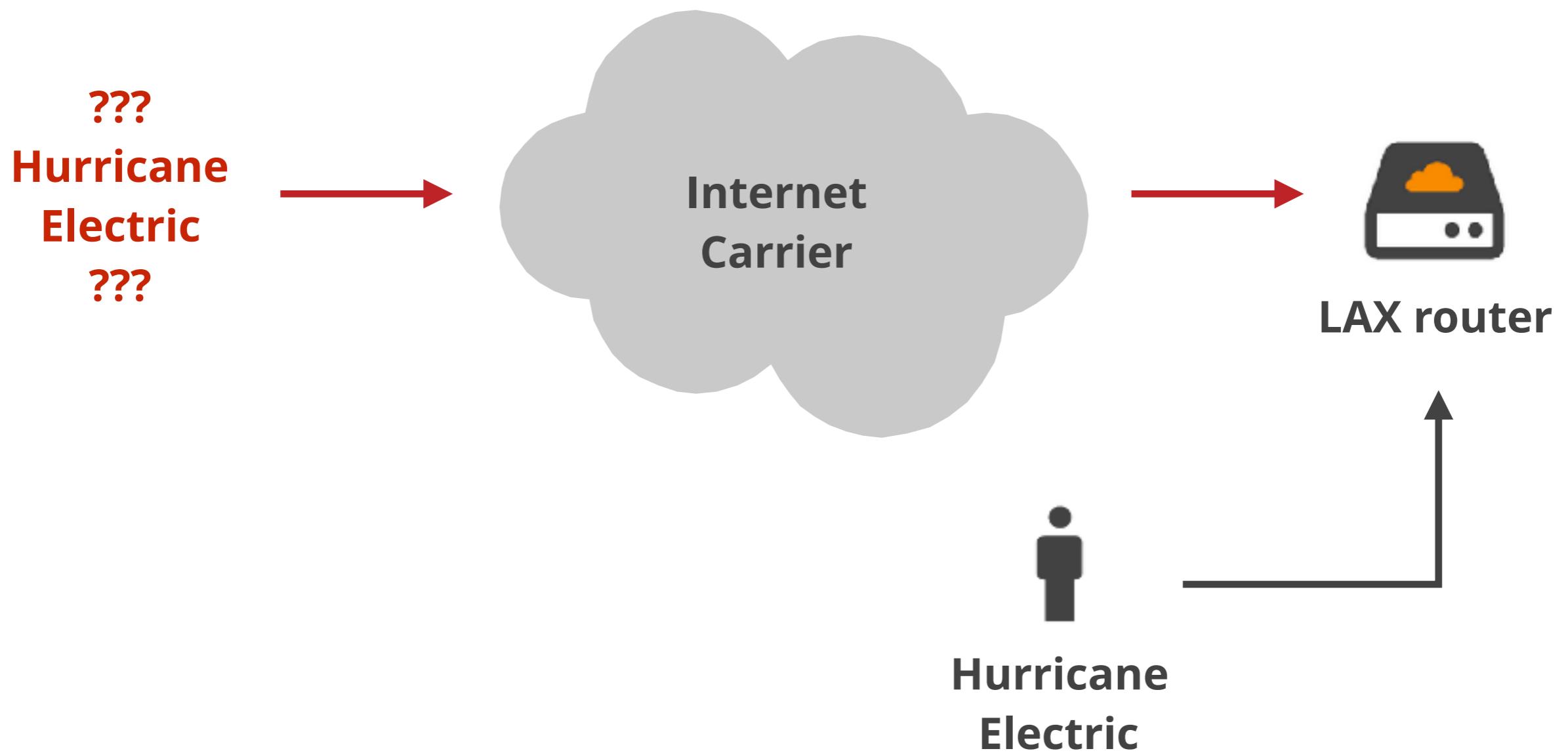
Our BPF Tools are now available on the CloudFlare Github:

<https://github.com/cloudflare/bpftools>

For installation instructions review the [README](#), but typing `make` should do most of the work:

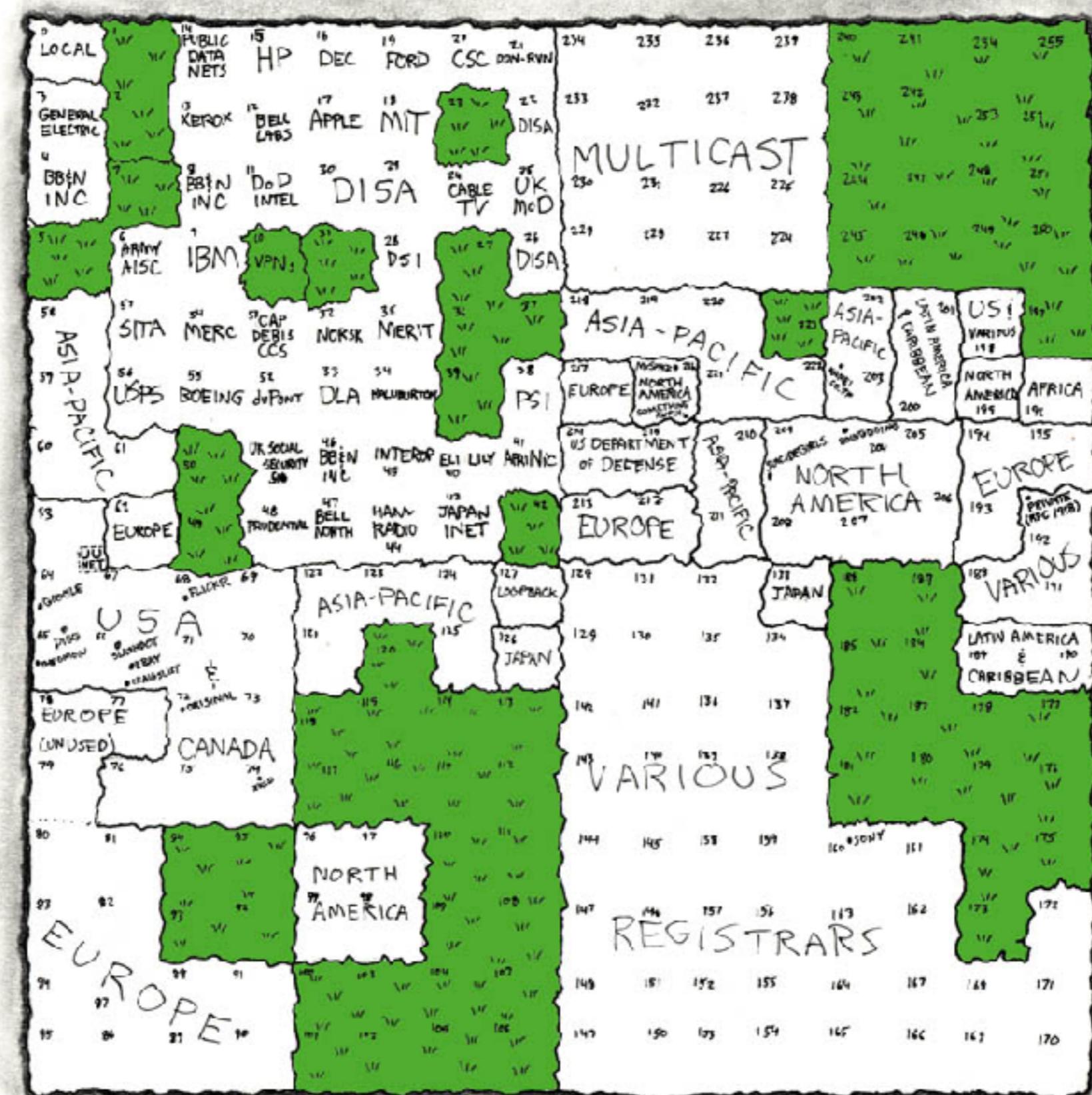
```
$ git clone https://github.com/cloudflare/bpftools.git  
$ cd bpftools  
$ make
```

# Source IP addresses



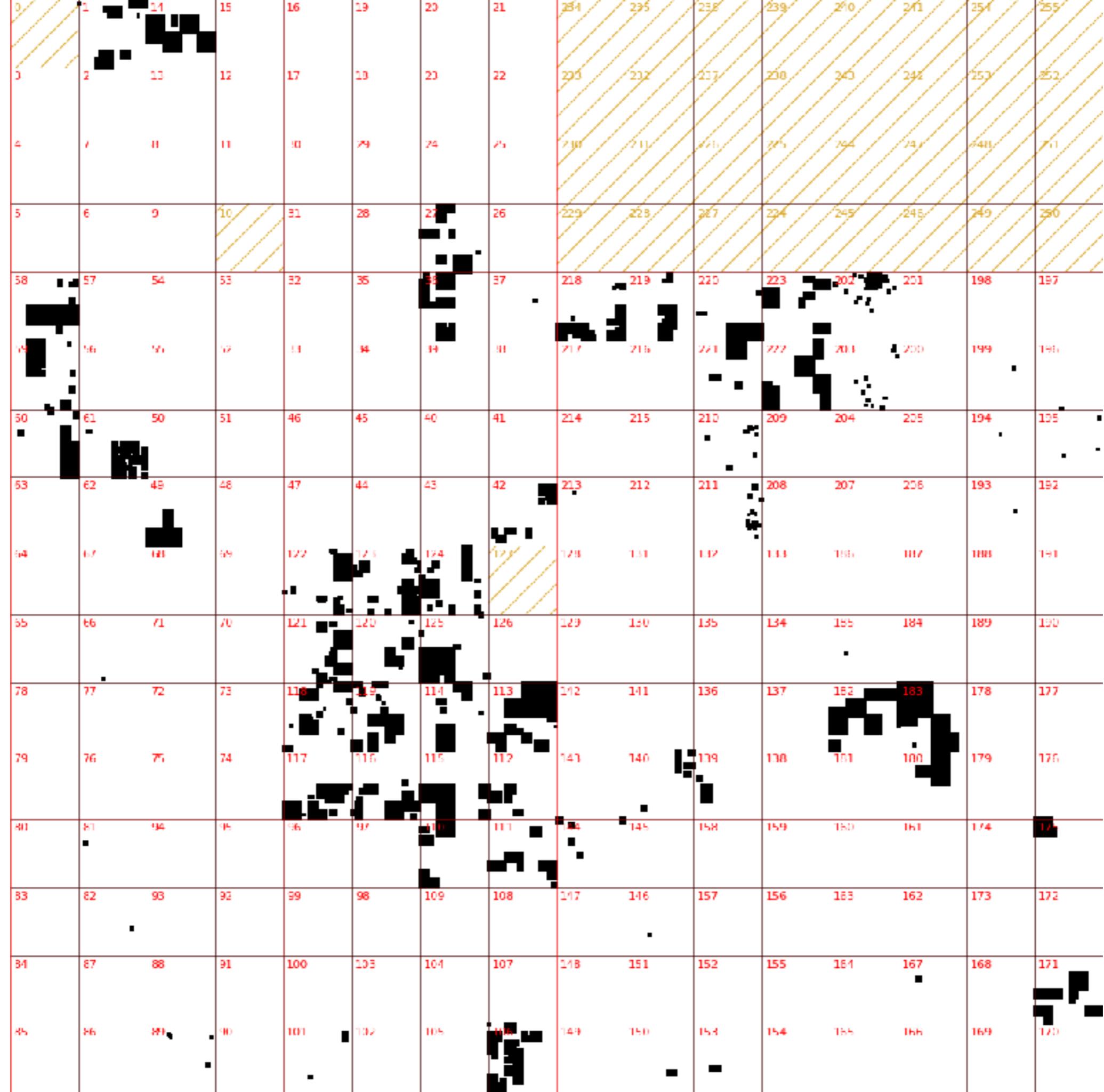
# MAP of THE INTERNET

THE IPv4 SPACE, 2006



0	1	24	15	16	19	20	21	254	295	238	239	240	241	254	255
0	2	13	12	17	10	20	22	295	292	237	200	345	242	252	252
4	7	11	11	30	29	24	23	230	211	211	265	264	244	248	211
5	6	9	10	31	28	27	26	229	228	267	234	245	245	249	290
58	57	54	53	32	35	35	37	218	219	220	223	202	231	198	197
47	46	45	46	44	44	44	41	217	216	221	222	201	200	199	196
50	61	50	51	46	45	40	41	214	215	210	209	204	205	194	125
53	62	49	48	47	44	43	42	213	212	211	208	207	206	193	192
54	67	60	69	122	123	124	125	127	128	131	142	144	135	117	181
55	66	71	70	121	120	125	126	129	120	135	134	185	184	189	120
78	77	72	73	118	119	114	113	142	141	136	137	182	183	178	177
79	76	75	74	117	115	115	112	141	140	179	138	151	100	179	175
80	81	94	95	96	97	101	101	144	145	158	159	160	161	174	170
83	82	93	92	99	98	109	108	147	146	157	156	163	162	173	172
84	87	88	91	100	103	104	107	148	151	152	155	161	167	168	171
85	86	89	90	101	102	105	106	149	150	153	154	155	166	169	170





1	14	15	16	19	20	21	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275		
2	12	13	16	17	18	19	20	21	22	23	24	25	26	27	28	29	29	29	29	29	29	29	29	29	29	29	29	29		
3	7	8	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	29	29	29	29	29	29		
4	6	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	29	29	29	29	29	29		
5	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	29	29	29	29	29	29		
6	17	54	53	32	35	36	37	38	219	220	223	202	201	198	167	166	165	164	163	162	161	160	159	158	157	156	155	154	153	
7	16	17	18	19	20	21	22	23	24	25	26	27	28	29	29	29	29	29	29	29	29	29	29	29	29	29	29	29		
8	61	50	51	46	45	40	41	214	213	210	205	204	229	194	165	164	163	162	161	160	159	158	157	156	155	154	153	152	151	
9	63	49	48	47	46	43	42	213	212	211	208	207	206	193	162	161	160	159	158	157	156	155	154	153	152	151	150	149	148	
10	10	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	29	29	29	29	29	29	29	29	29	29	29		
11	46	73	70	121	120	125	126	129	130	136	135	134	133	132	131	130	129	128	127	126	125	124	123	122	121	120	119	118		
12	77	72	73	118	119	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100	109	108	107	106	105	104	103	102	101	
13	76	75	74	117	116	115	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100	109	108	107	106	105	104	103	102	101
14	81	82	84	85	86	87	88	89	89	109	108	107	106	105	104	103	102	101	100	109	108	107	106	105	104	103	102	101	100	
15	82	83	84	85	86	87	88	89	89	109	108	107	106	105	104	103	102	101	100	109	108	107	106	105	104	103	102	101	100	
16	87	88	91	100	103	104	107	104	107	118	151	152	155	151	154	153	152	151	150	159	158	157	156	155	154	153	152	151	150	
17	86	87	88	89	90	91	92	93	94	105	106	107	108	109	108	107	106	105	104	103	102	101	100	109	108	107	106	105	104	

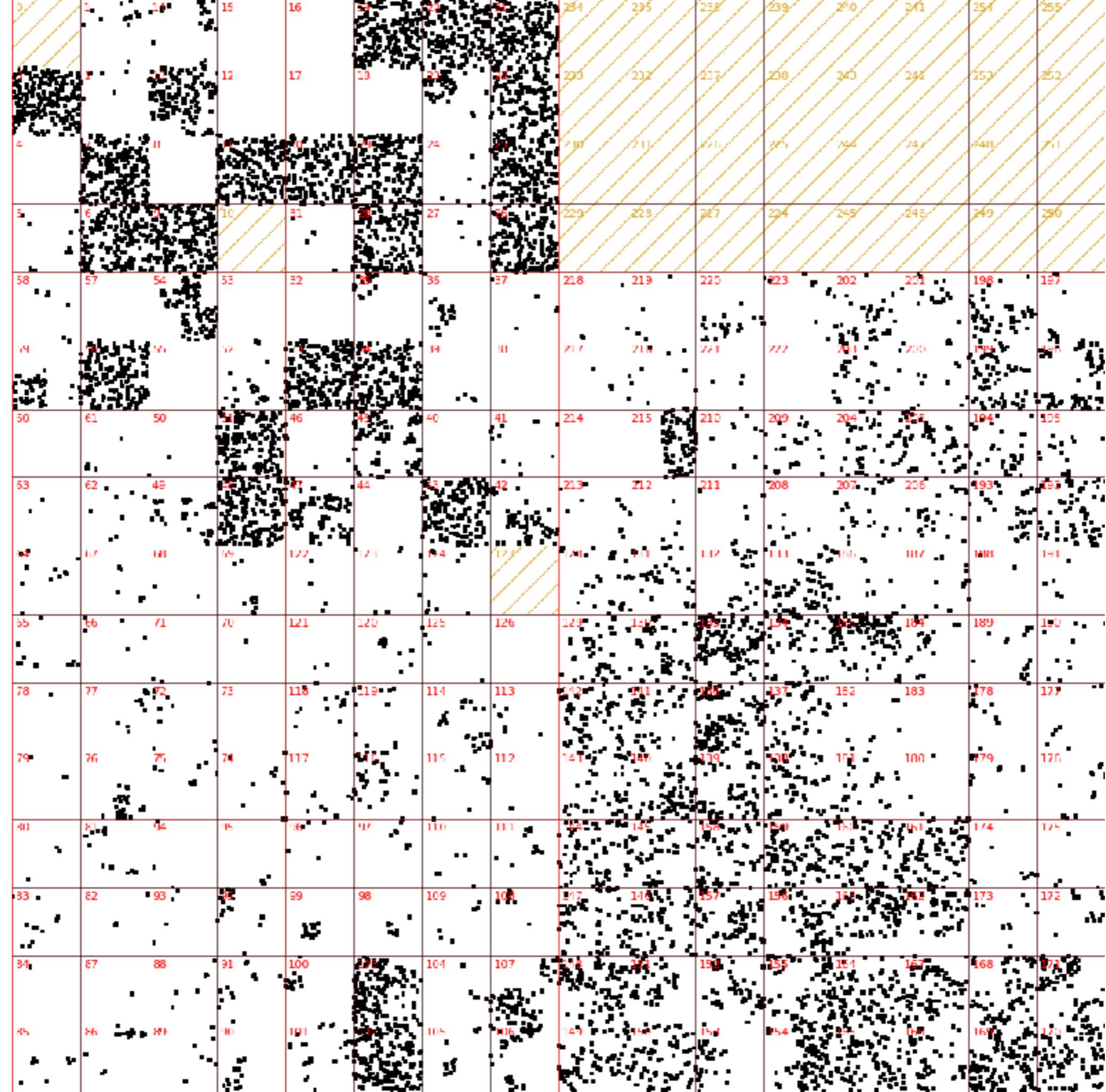


0	14	15	16	17	18	19	20	21	254	295	238	239	240	241	251	255	
1	2	13	14	17	18	19	20	21	22	295	296	237	238	240	242	252	
2	3	4	10	11	18	24	24	25	296	297	239	240	242	243	252	253	
3	5	6	9	10	24	28	27	26	229	228	237	234	245	246	249	250	
4	18	17	52	53	32	35	35	36	218	219	225	223	242	241	198	197	
5	19	16	51	51	46	45	46	47	40	217	216	221	222	243	244	199	198
6	50	61	50	51	46	45	46	47	41	224	215	213	209	244	225	194	195
7	58	42	49	48	47	44	45	42	53	212	211	208	207	224	193	192	
8	59	14	12	11	10	72	72	73	72	73	131	132	133	134	135	136	
9	55	16	6	30	121	125	125	126	126	129	130	131	144	152	184	189	
10	78	77	72	73	148	119	114	113	122	141	136	137	142	163	178	167	
11	79	76	77	74	117	116	115	112	141	142	179	146	154	104	179	140	
12	40	4	8	8	4	9	9	10	11	12	148	159	160	161	174	175	
13	53	61	2	92	29	98	102	108	112	216	157	156	183	162	171	182	
14	54	87	88	91	100	103	104	107	118	151	152	155	181	167	168	171	
15	45	46	45	46	101	112	105	106	108	154	150	147	148	145	166	169	



0	1	14	15	16	19	20	21	254	295	238	239	240	241	251	255
0	2	13	12	17	10	23	22	295	292	237	200	340	242	252	252
4	7	11	11	30	29	24	23	210	211	211	245	244	241	211	211
5	6	9	10	31	28	27	26	229	228	227	234	245	245	249	290
58	57	54	53	32	35	35	37	228	219	219	221	206	45	38	151
64	56	50	50	14	14	14	11	214	215	215	221	247	241	241	241
60	61	50	51	46	45	40	41	214	215	215	209	204	205	194	194
53	62	49	48	47	44	43	42	213	212	212	208	207	207	193	192
64	67	140	69	70	73	73	73	213	212	212	209	204	204	190	189
55	66	71	70	71	74	74	74	126	126	126	120	120	124	184	185
78	77	72	73	78	79	74	74	113	113	113	101	136	137	136	177
79	76	75	74	75	75	74	74	121	121	121	140	149	149	161	179
80	81	94	95	96	97	100	100	11	11	11	12	154	153	154	176
83	82	93	92	99	98	109	108	17	145	152	156	163	162	173	182
84	87	88	91	100	103	104	107	26	26	26	152	155	174	167	168
85	86	89	90	101	102	105	106	41	41	41	144	144	144	168	169

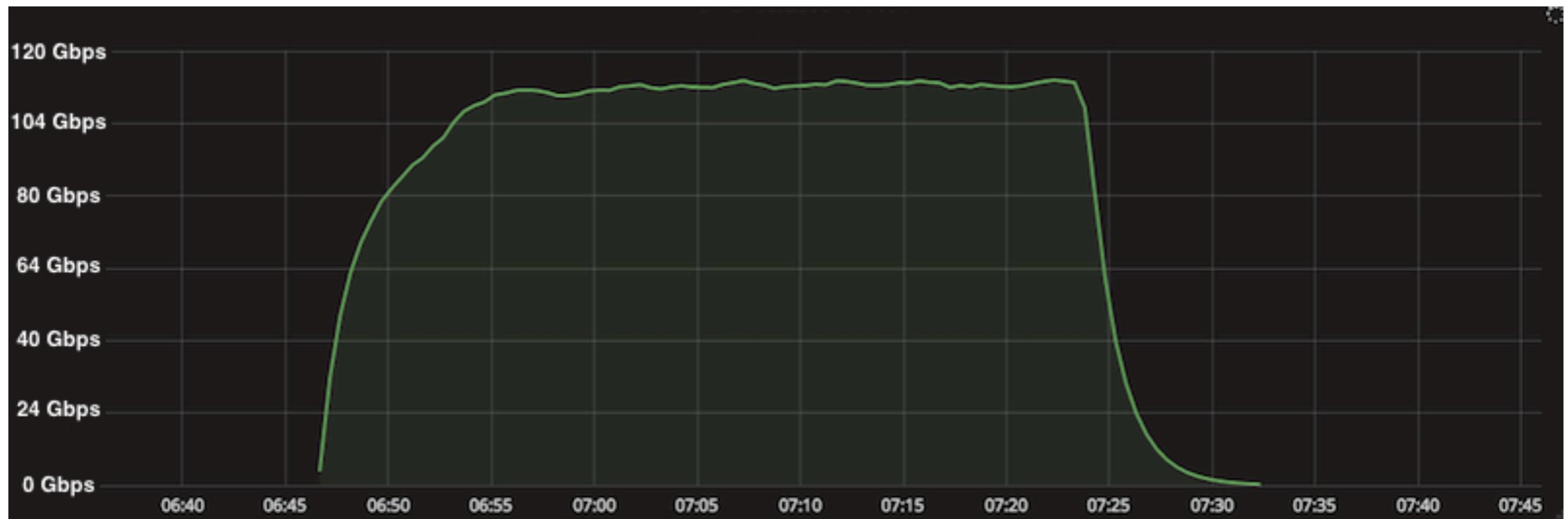




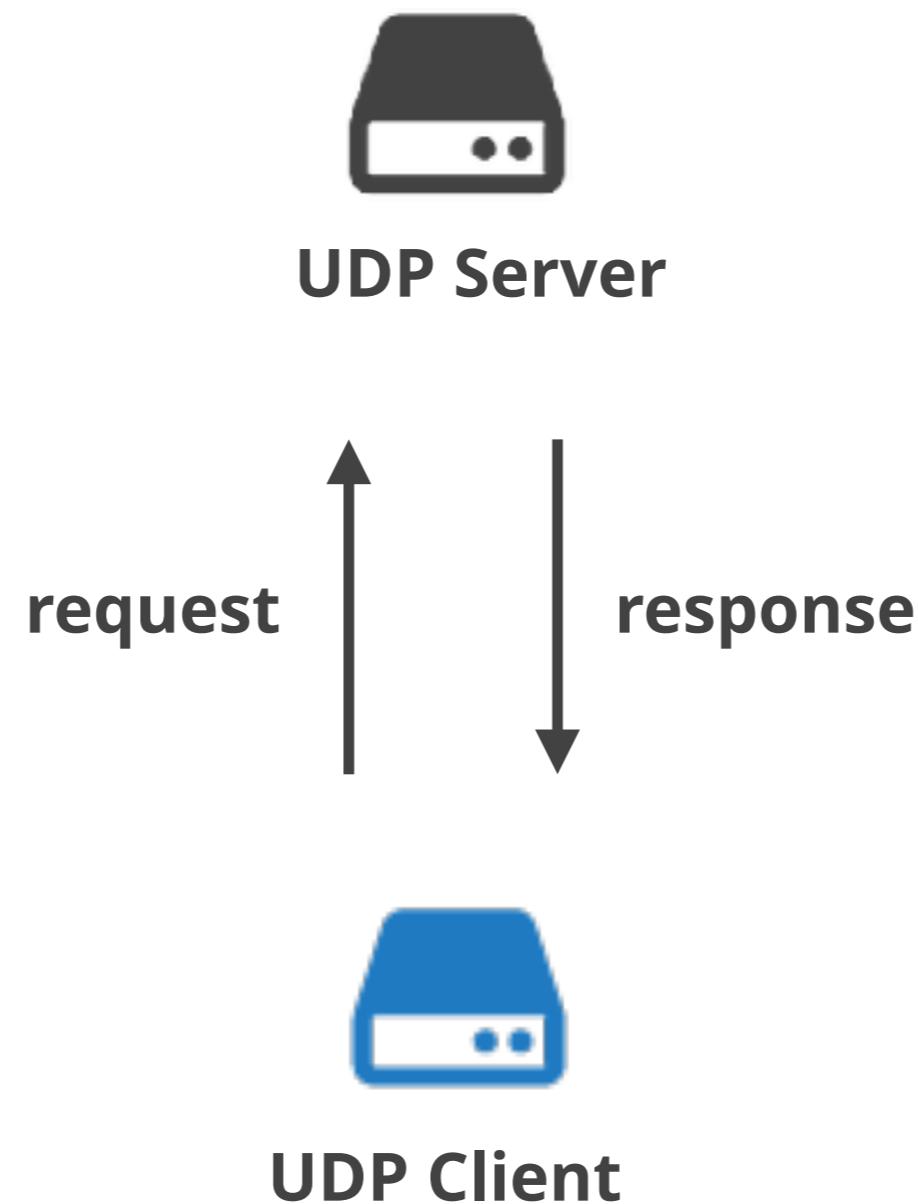
# IP Spoofing

1. Tracing back is impossible
2. Allows sophisticated attacks

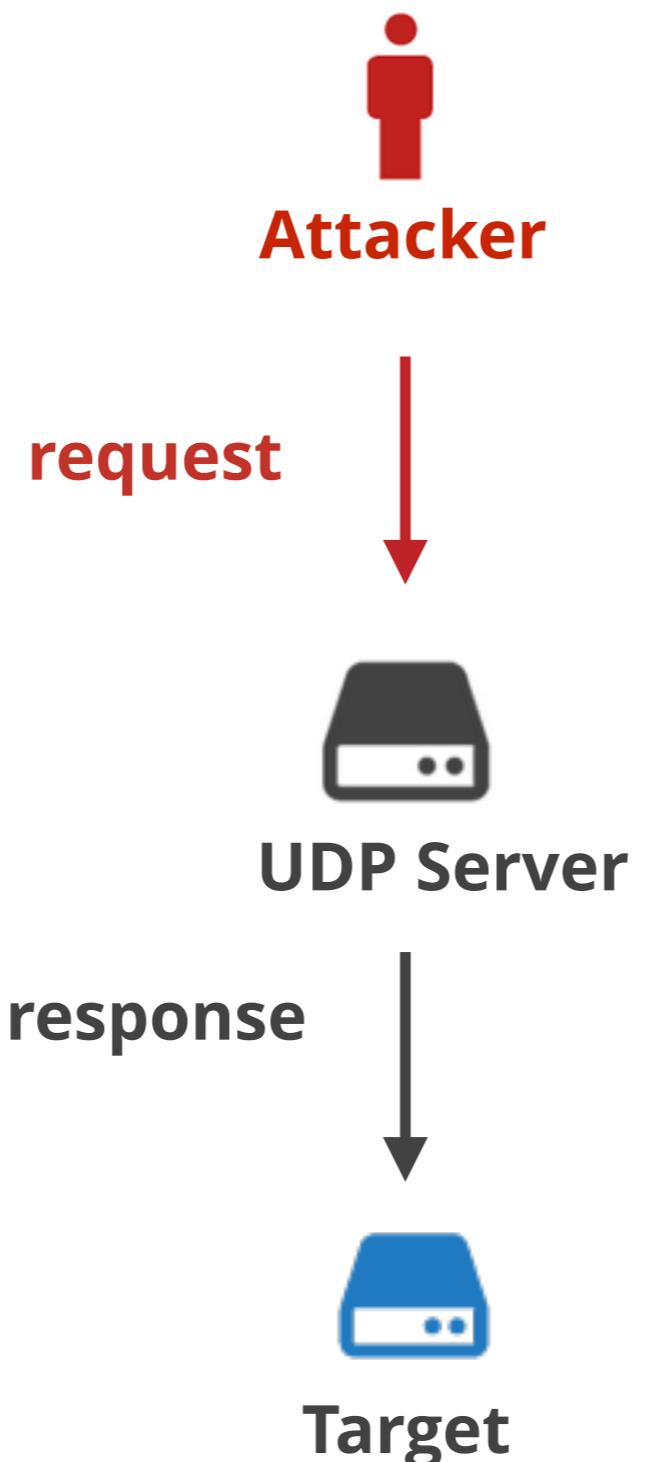
# Sophistication



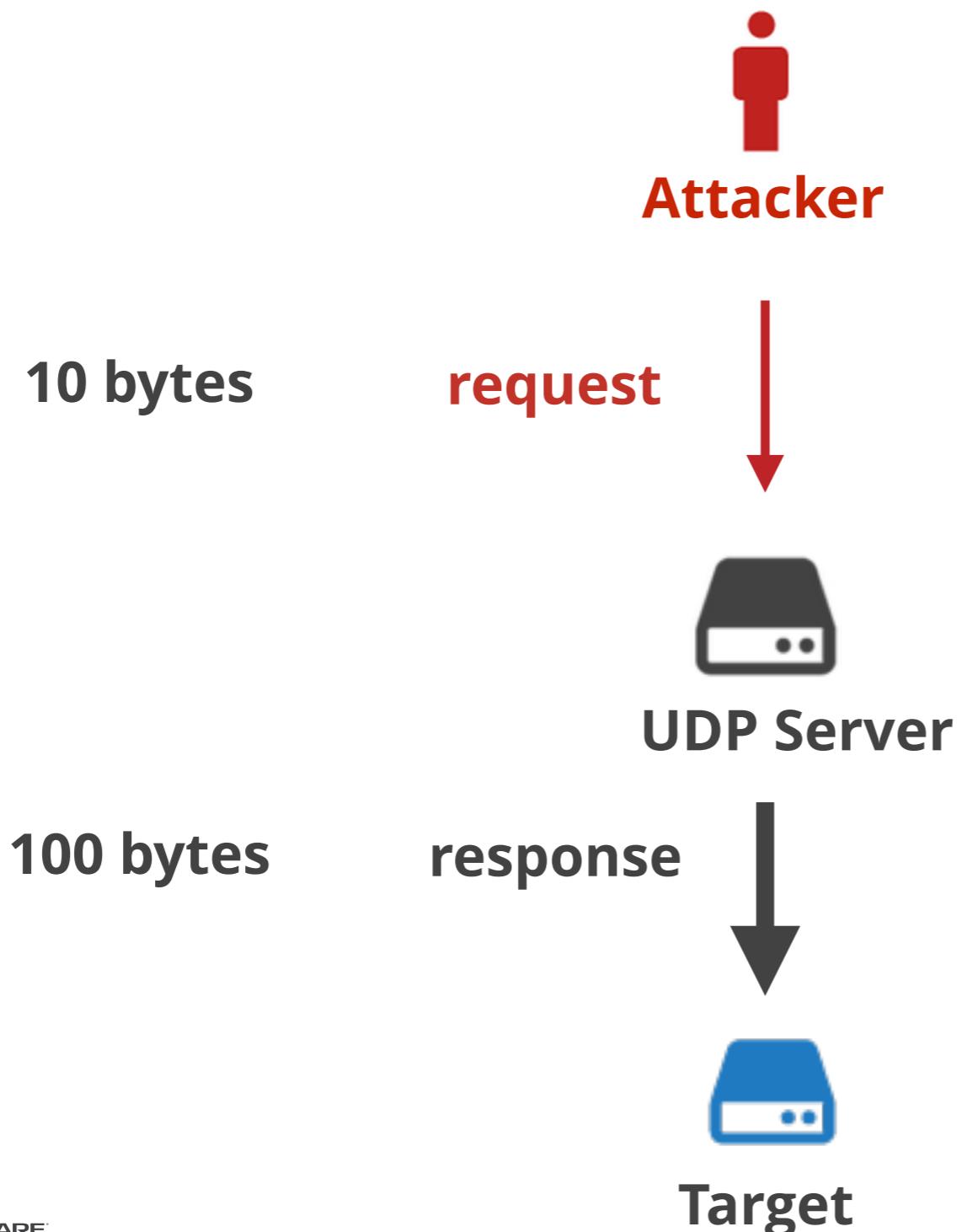
# 1. UDP request-response



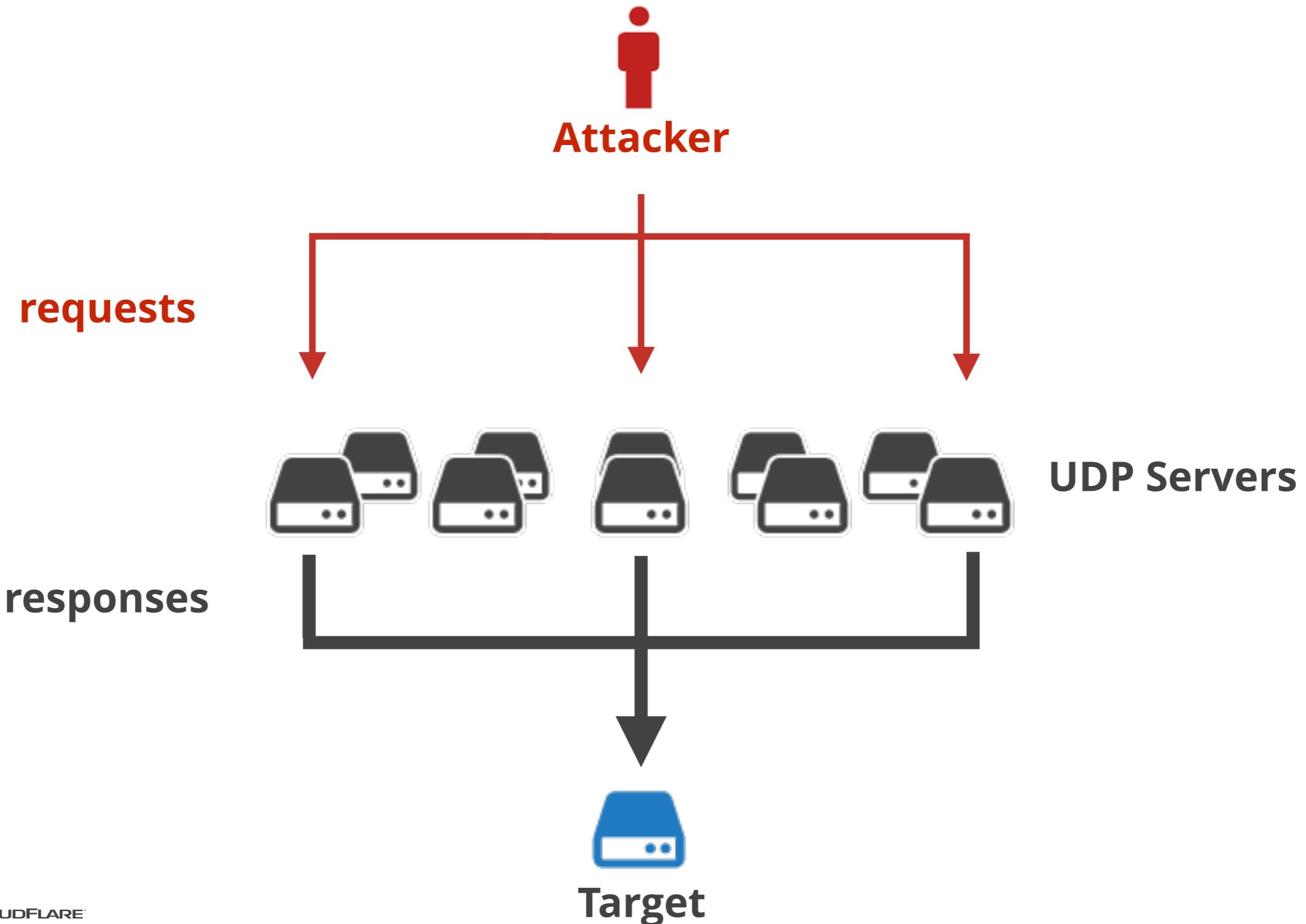
# 1. Amplification



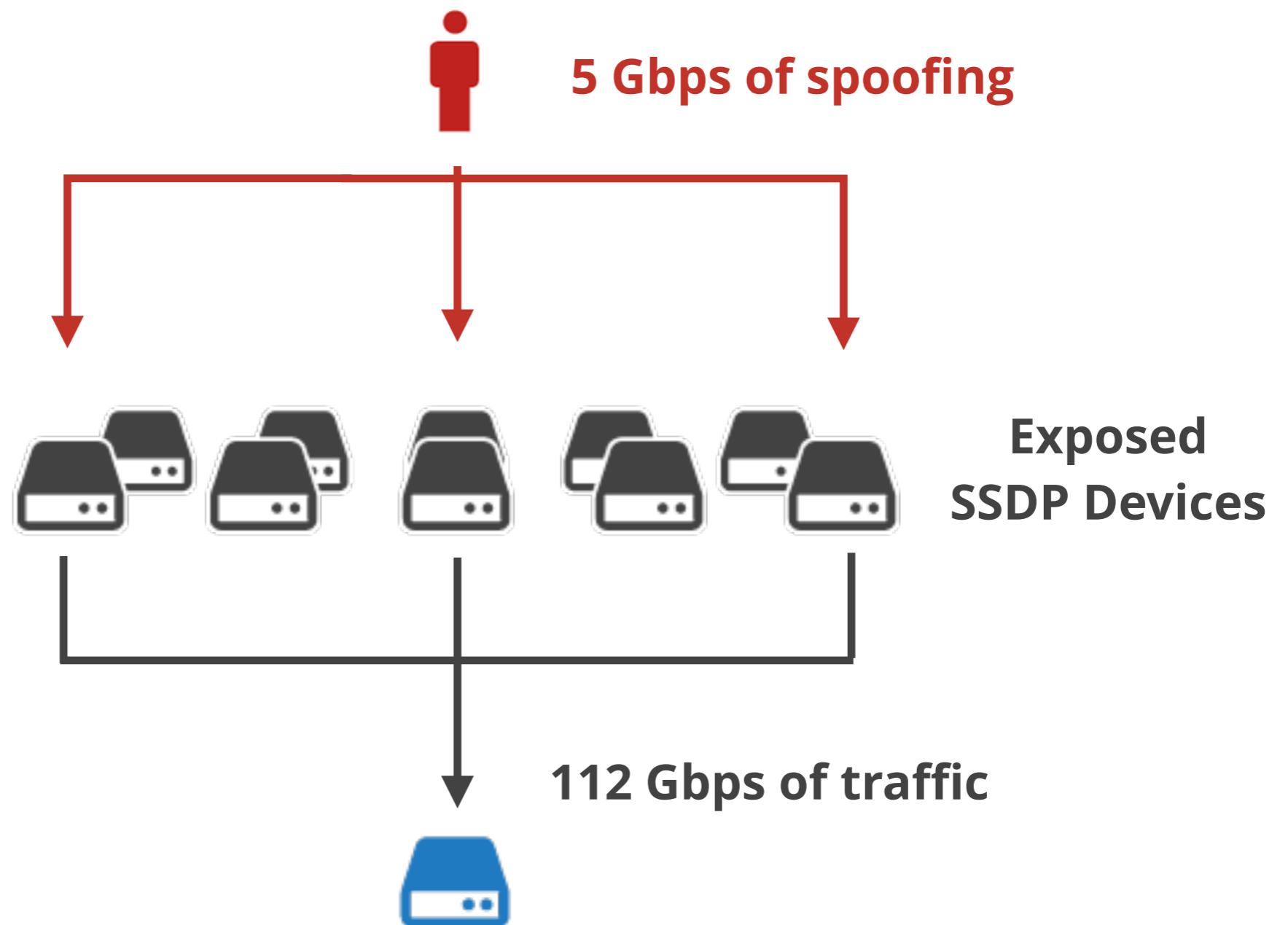
# 1. Amplification factor



# 1. Scale up!



# June 2017: SSDP



# Amplification easy to block

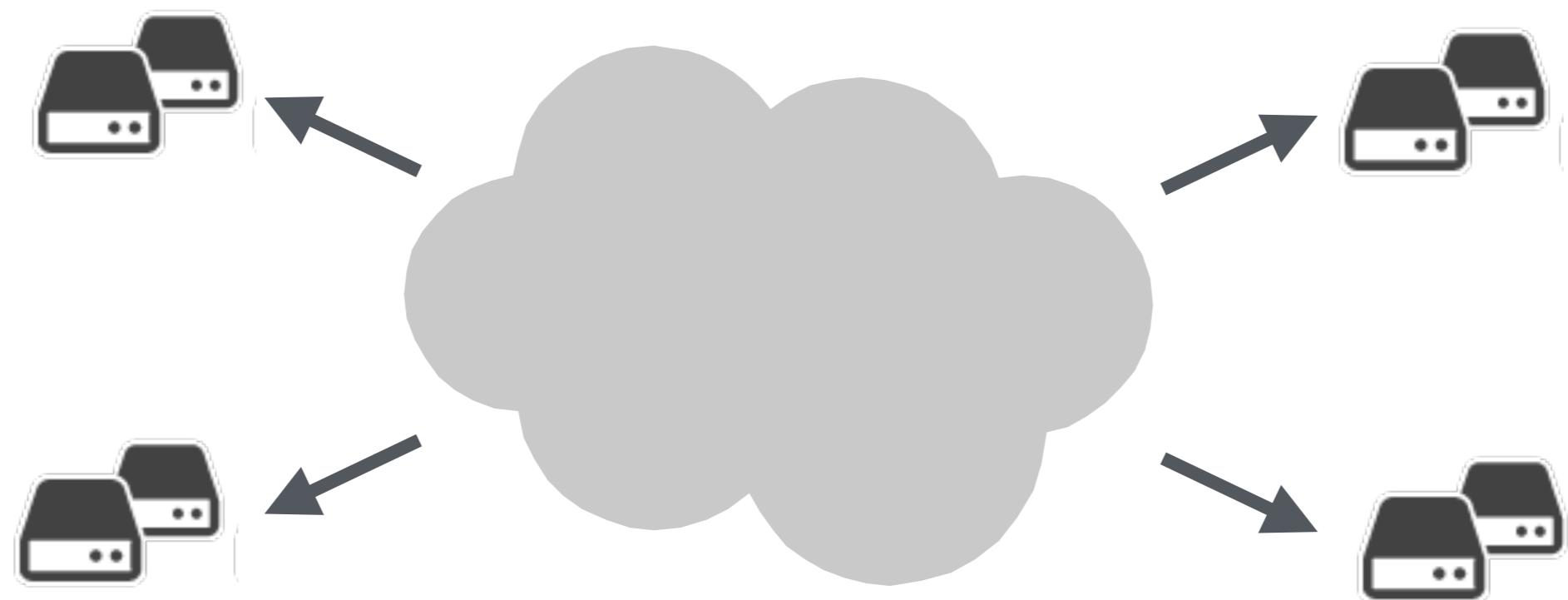
- Easy to block on firewall
  - udp and src port 1900
- Nicely dispersed geographically
- But tracing is impossible

The only way to keep online  
is to absorb the attack

# Receive and process



# Centralisation



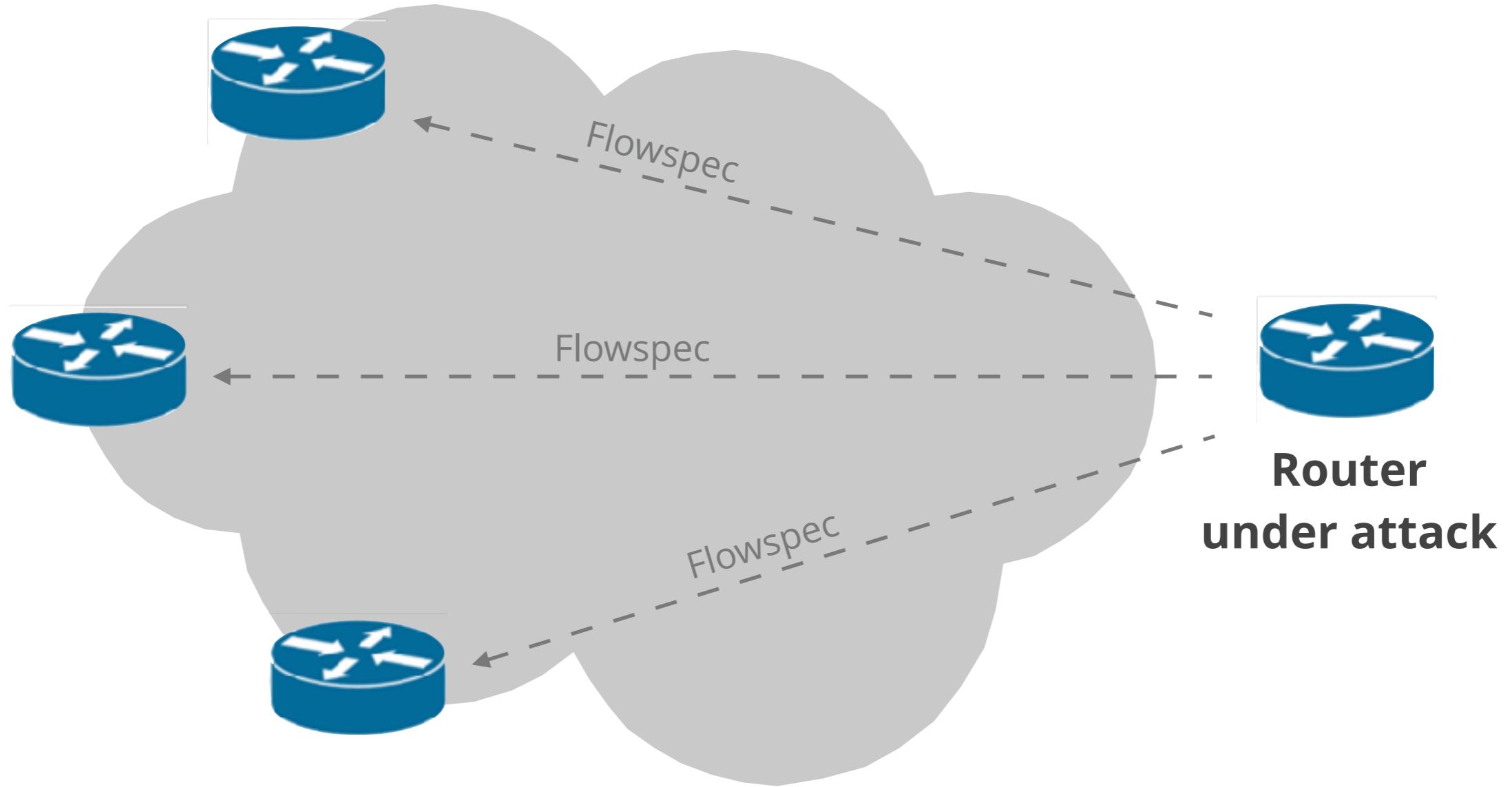
# Solution

# Technical solutions

- BGP Flowspec
- Prevent IP spoofing - BCP38
- Netflow

# BGP Flowspec is awesome

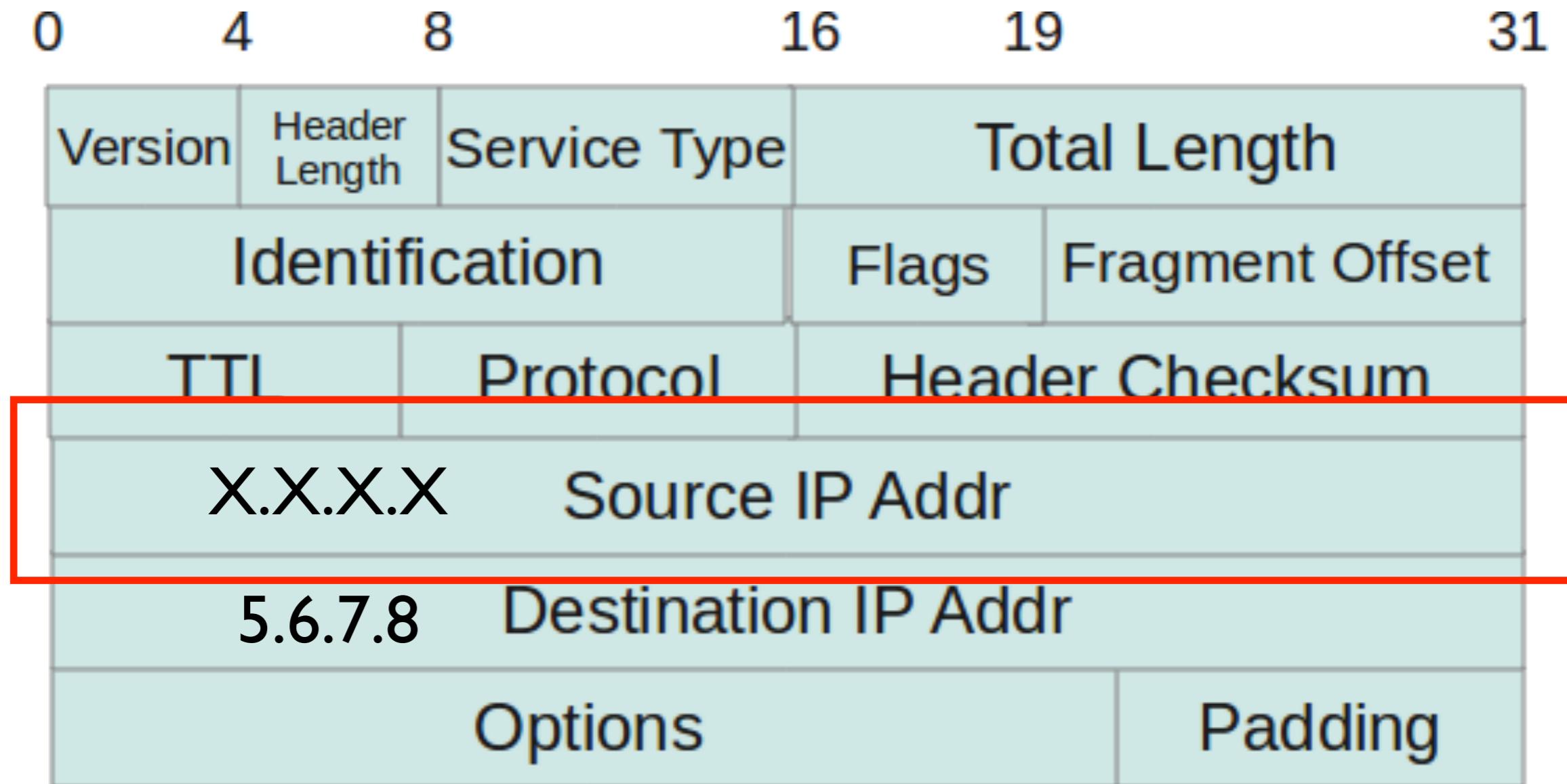
# Flowspec



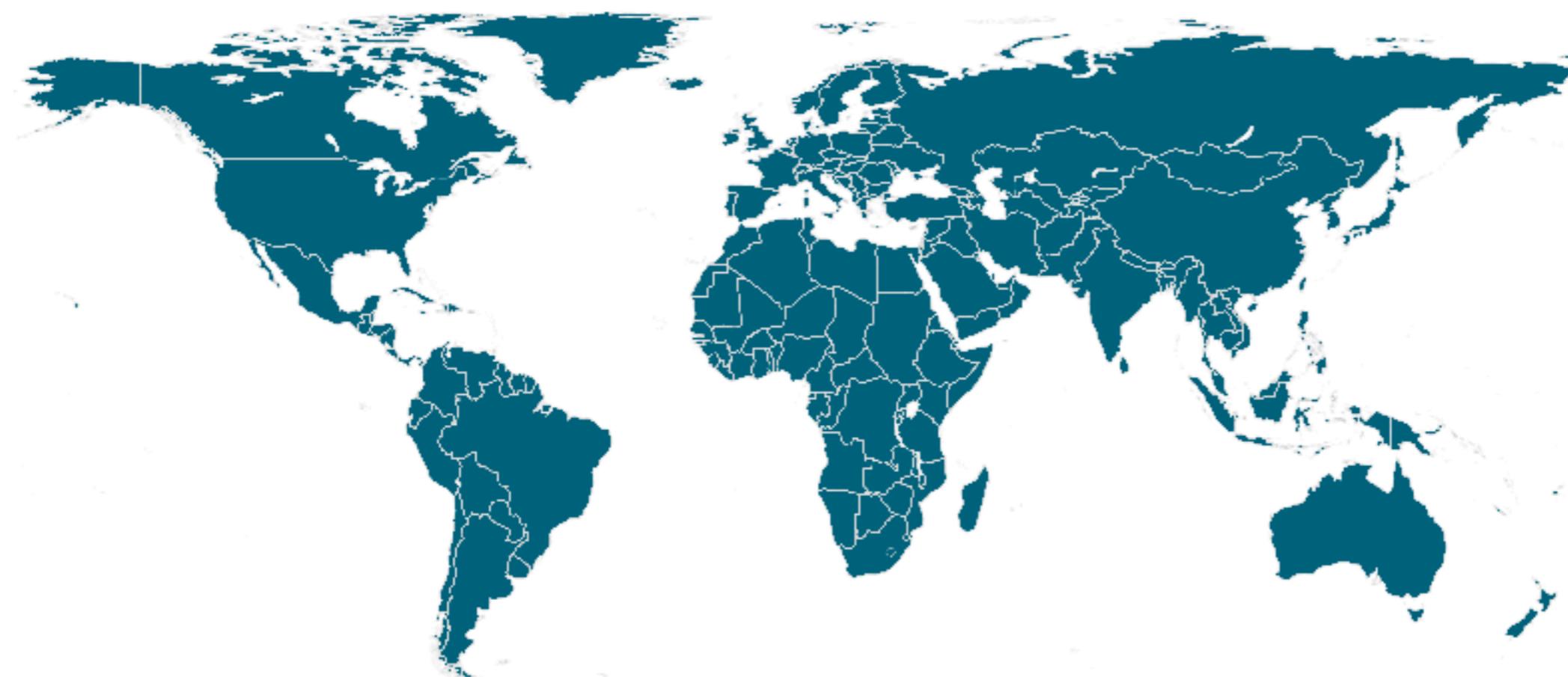
Adoption = nil

# Prevent IP spoofing - BCP38

# Failed! Live with it!



# We're left with incompetent



Don't solve the IP spoofing!

Solve the attribution!

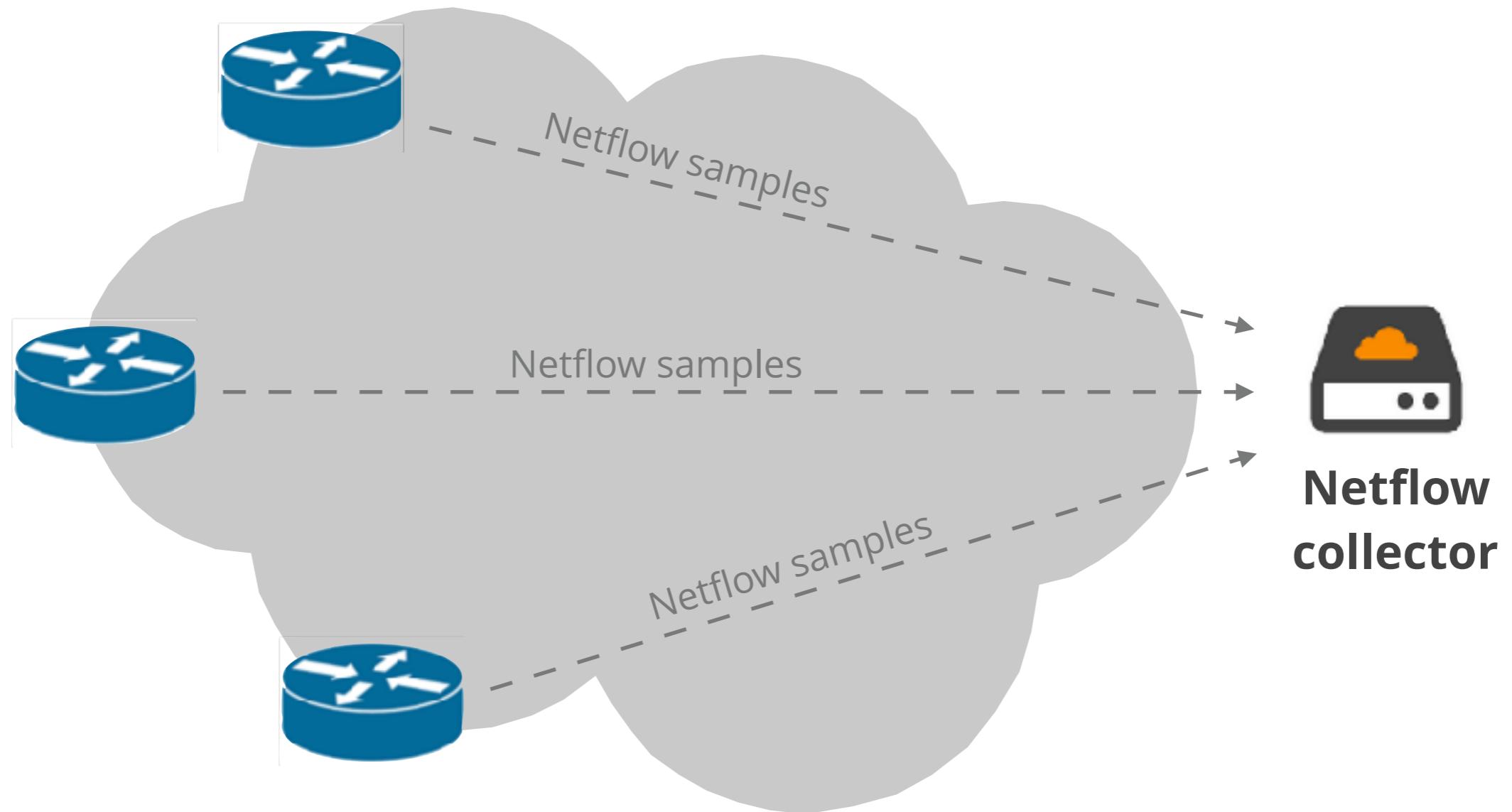
# Netflow

# NETFLIX



## ALL THE THINGS

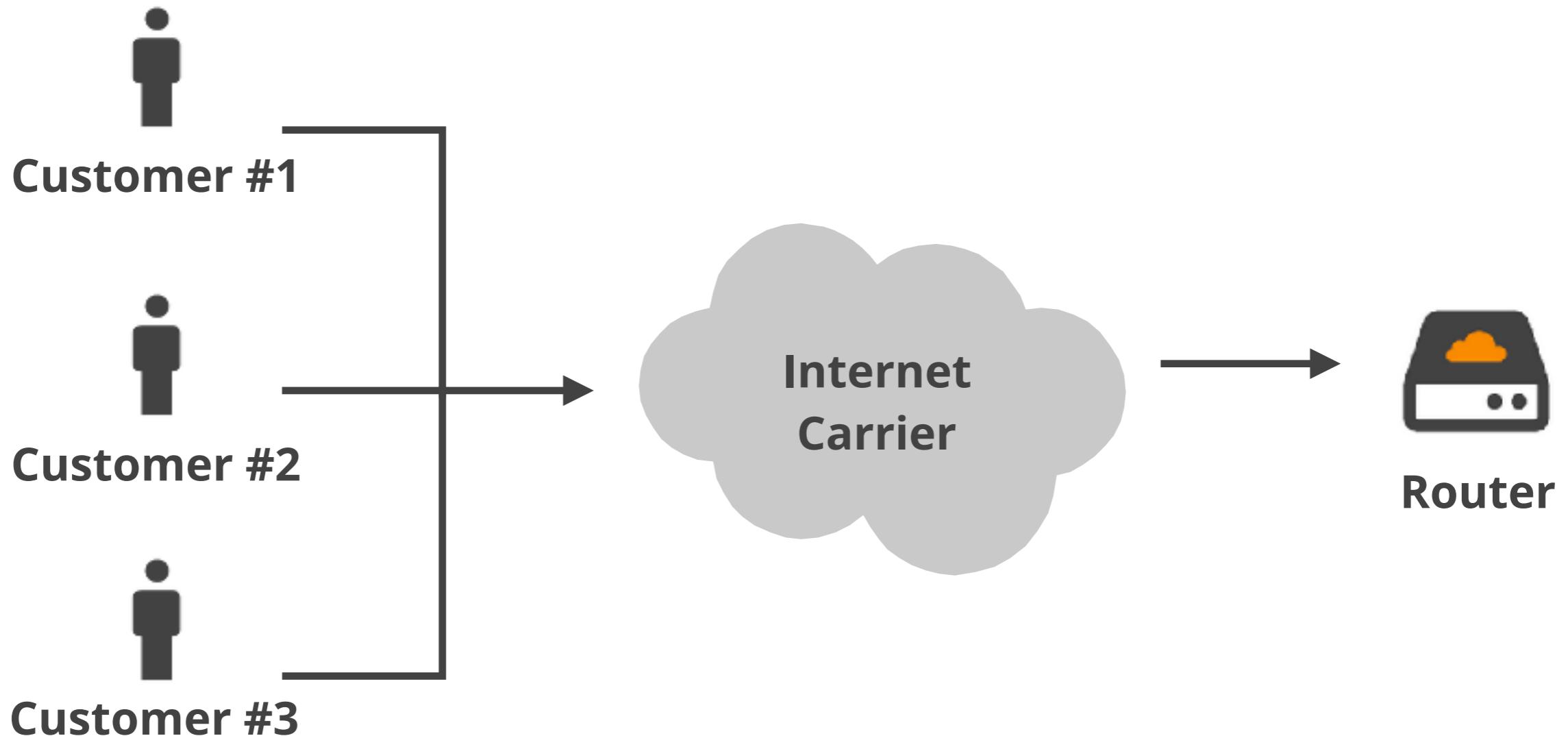
# Netflow



# Netflow

- Open source netflow toolchain is great
- Scales well
- To avoid privacy issues
  - Rotate logs often
  - Set high sampling rate - 1/64k connections

# Internet Carriers



# Netflow

```
(netops) # nfdump -M db/waw01:1hr01 -R . -n2 -t -300 -s dstip/packets "in if 731"
Top 2 Dst IP Addr ordered by packets:
Dst IP Addr Flows(%) Packets(%) Bytes(%) pps bps bpp
173.245.58.40 1.0 M(77.0) 17.6 G(75.8) 1.1 T(22.6) 59.0 M 30.7 G 65
173.245.59.15 54962( 4.0) 910.3 M( 3.9) 75.5 G( 1.5) 3.1 M 2.0 G 82

Summary: total flows: 1361108, total bytes: 5087980650496, total packets:
23271079936, avg bps: 135599480319, avg pps: 77524526, avg bpp: 218
Total flows processed: 2457140, Blocks skipped: 0, Bytes read: 177251772
Sys: 0.210s flows/second: 11700666.7 Wall: 0.210s flows/second: 11654603.2
```

# Recap

- BGP flowspec firewall
  - A stop gap, useful for amplification attacks
- Prevent IP spoofing - BCP38
  - The root of all evil, but unfixable in short time
- Netflow sampling
  - Required for attack attribution
  - Supported by some competent ISP's / tier 1
  - Still not supported by majority!

Attribution allows  
informed discussion

The internet will be better for everyone.

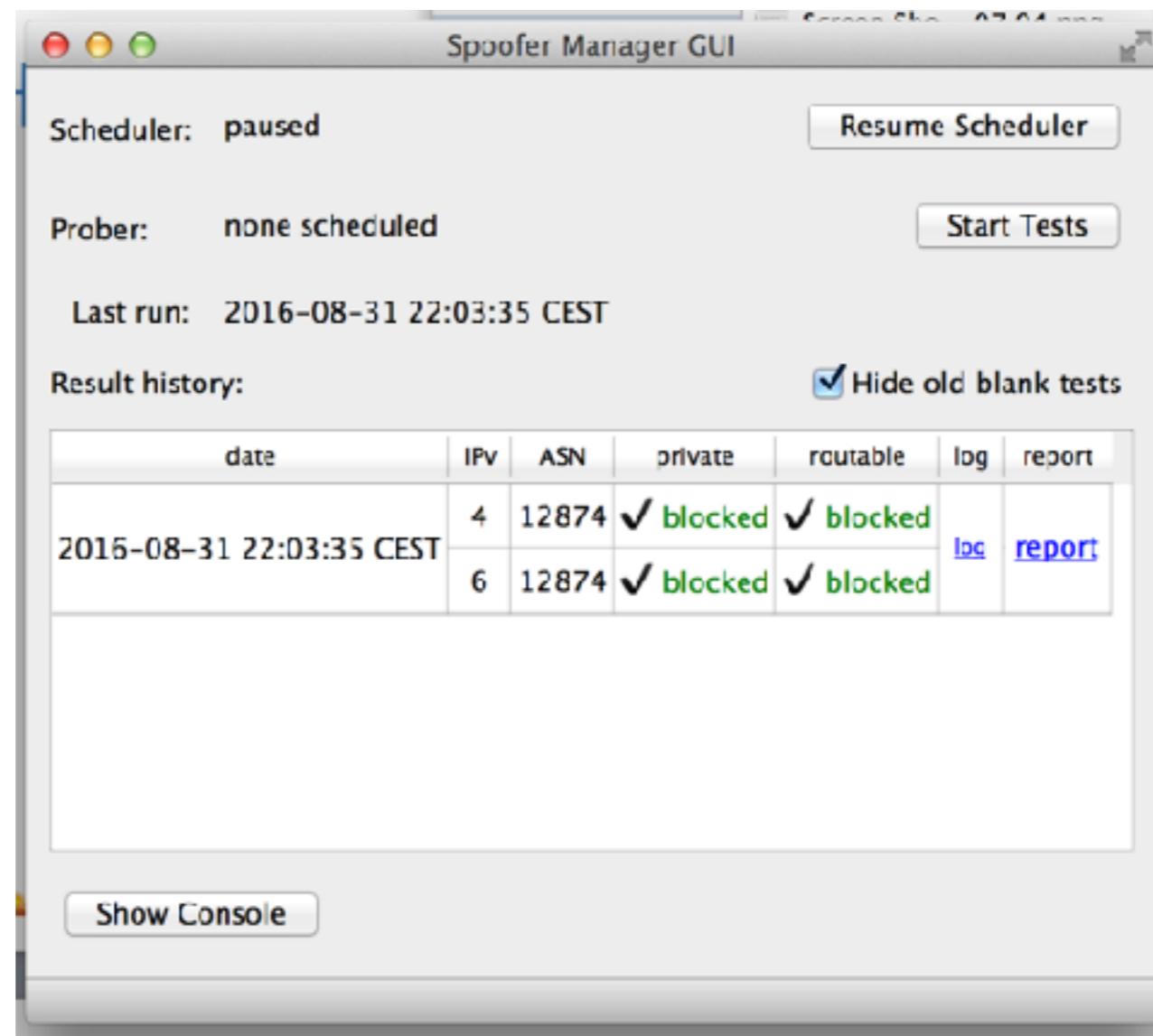
marek@cloudflare.com



# How to help?

# Help: report IP spoofing

- From [spoofercaida.org](http://spoofercaida.org)



# Help: close NTP and DNS

- Scan your network for open NTP and DNS servers
- <http://openntpproject.org/>
- <http://openresolverproject.org>
- <http://www.team-cymru.org/Open-Resolver-Challenge.html>
- <https://www.shodan.io/>

# Help: press for attribution

- When under attack
- Collect evidence
- Ask where the traffic came from!

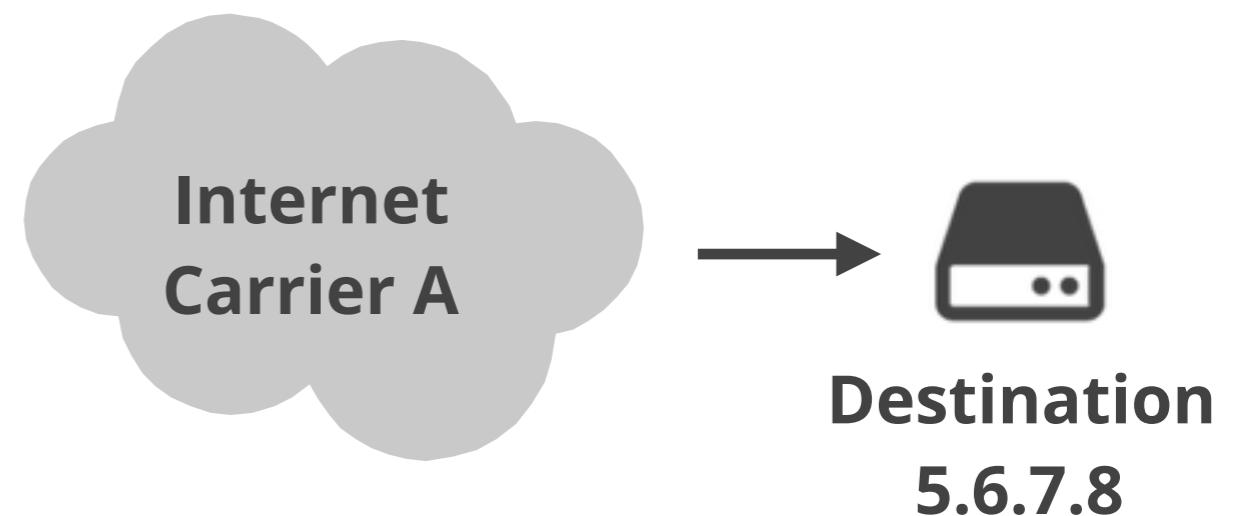
Is amplification in decline?

# Is amplification in decline?

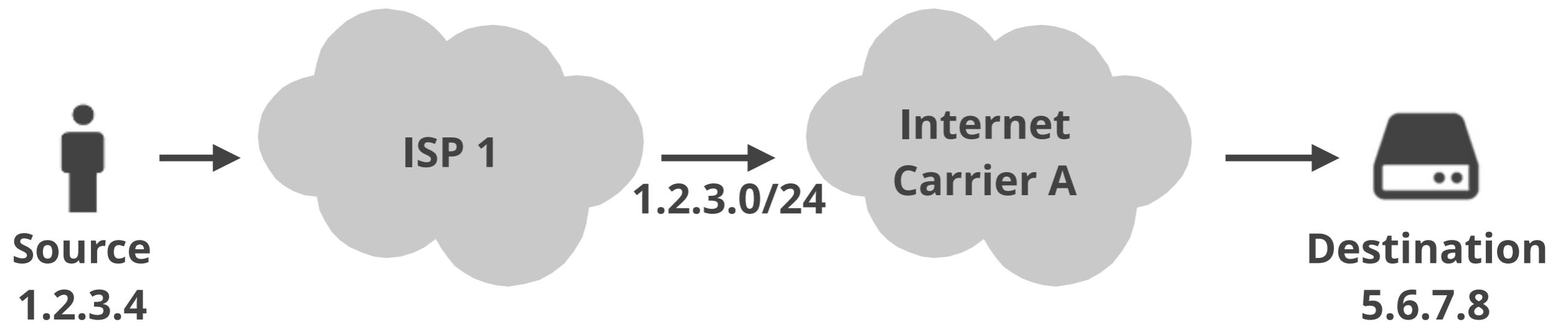
- Very easy to block on firewall
  - udp and src port 123 == NTP attack
  - udp and src port 53 == DNS attack
- DDoS mitigation vendors have FAT pipes
- Amplification is bouncing off *real* servers
- Therefore geographically distributed
- Not effective against anycast

Why IP Filtering must be on  
the edge

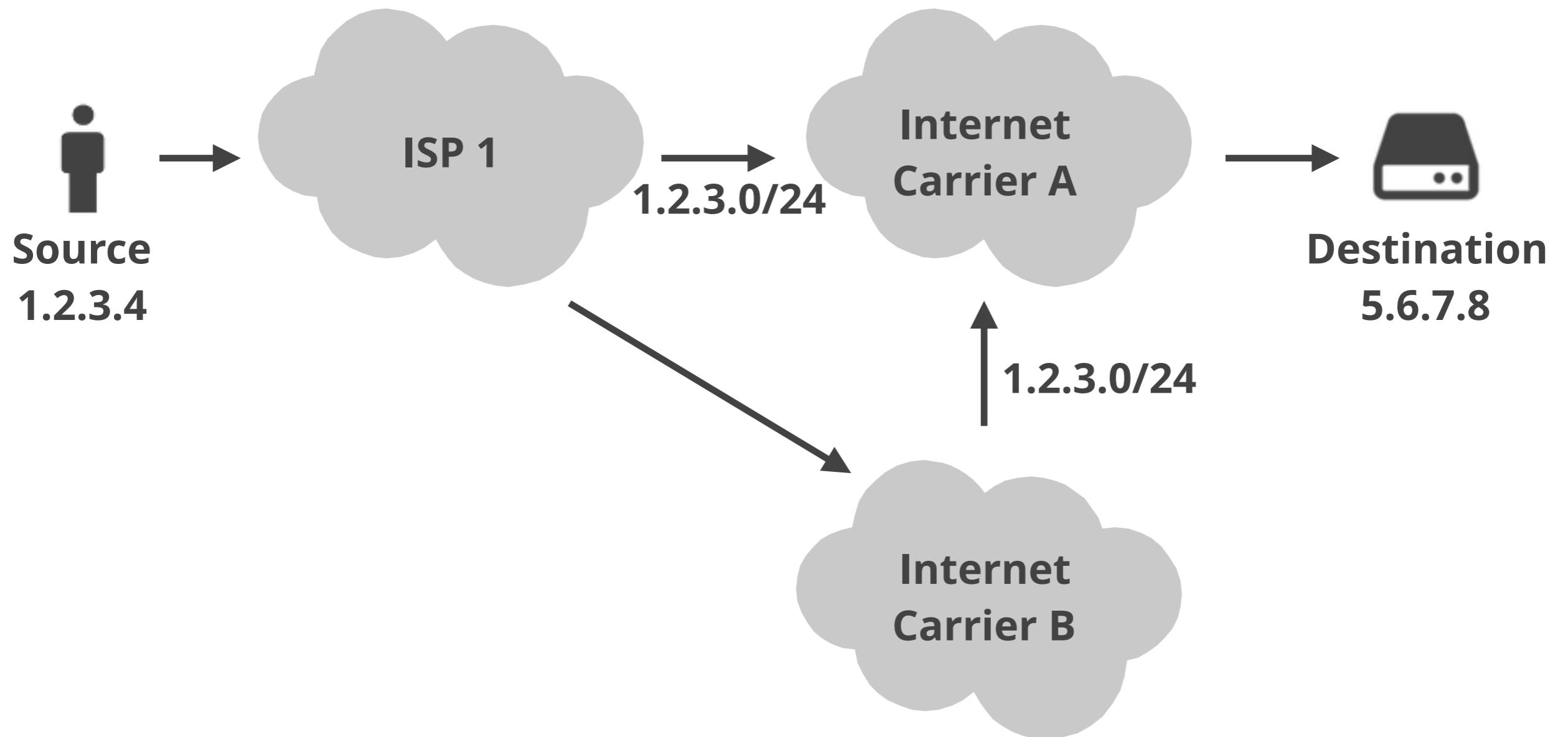
# Filtering is hard



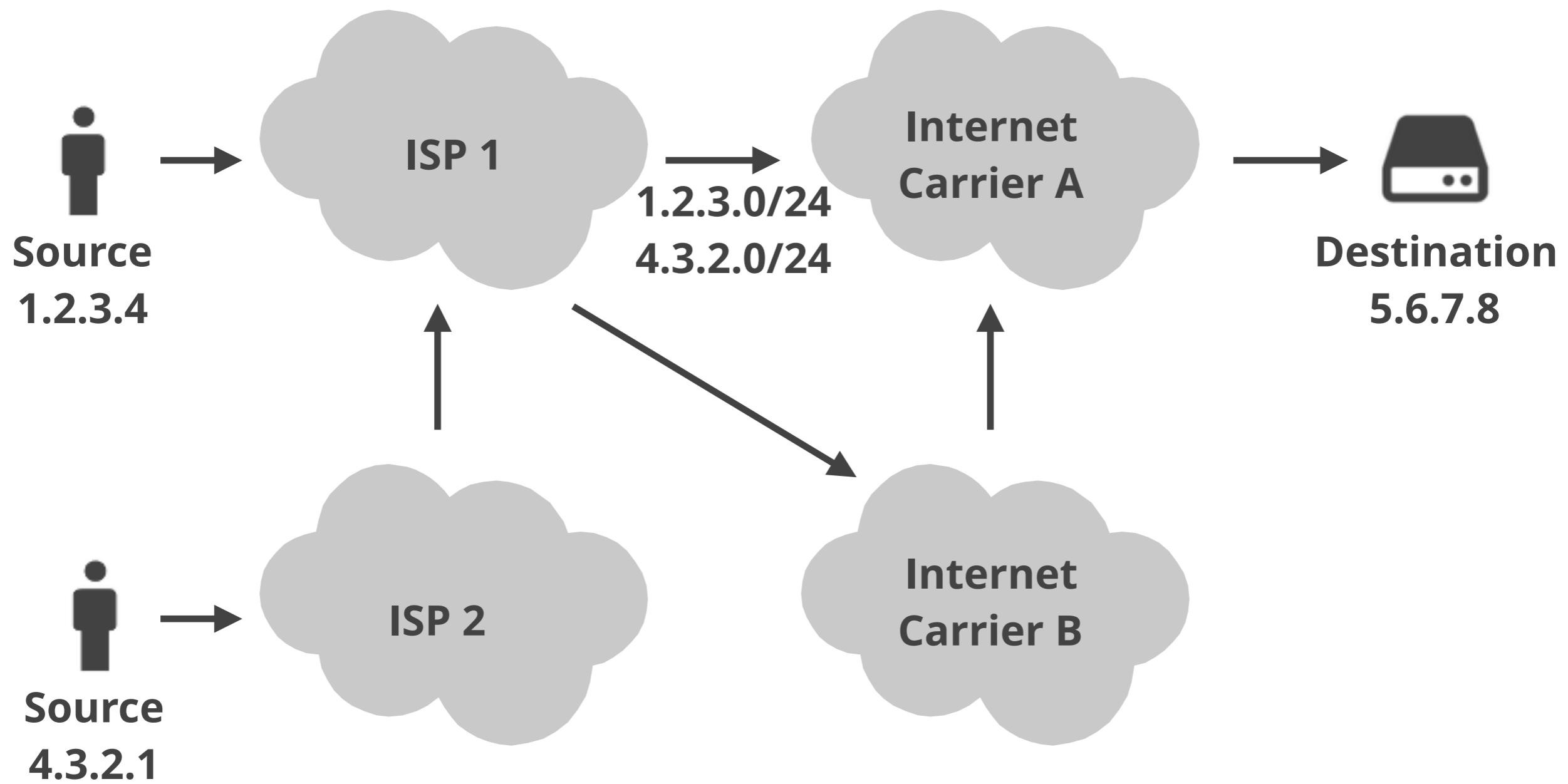
# Filtering is hard



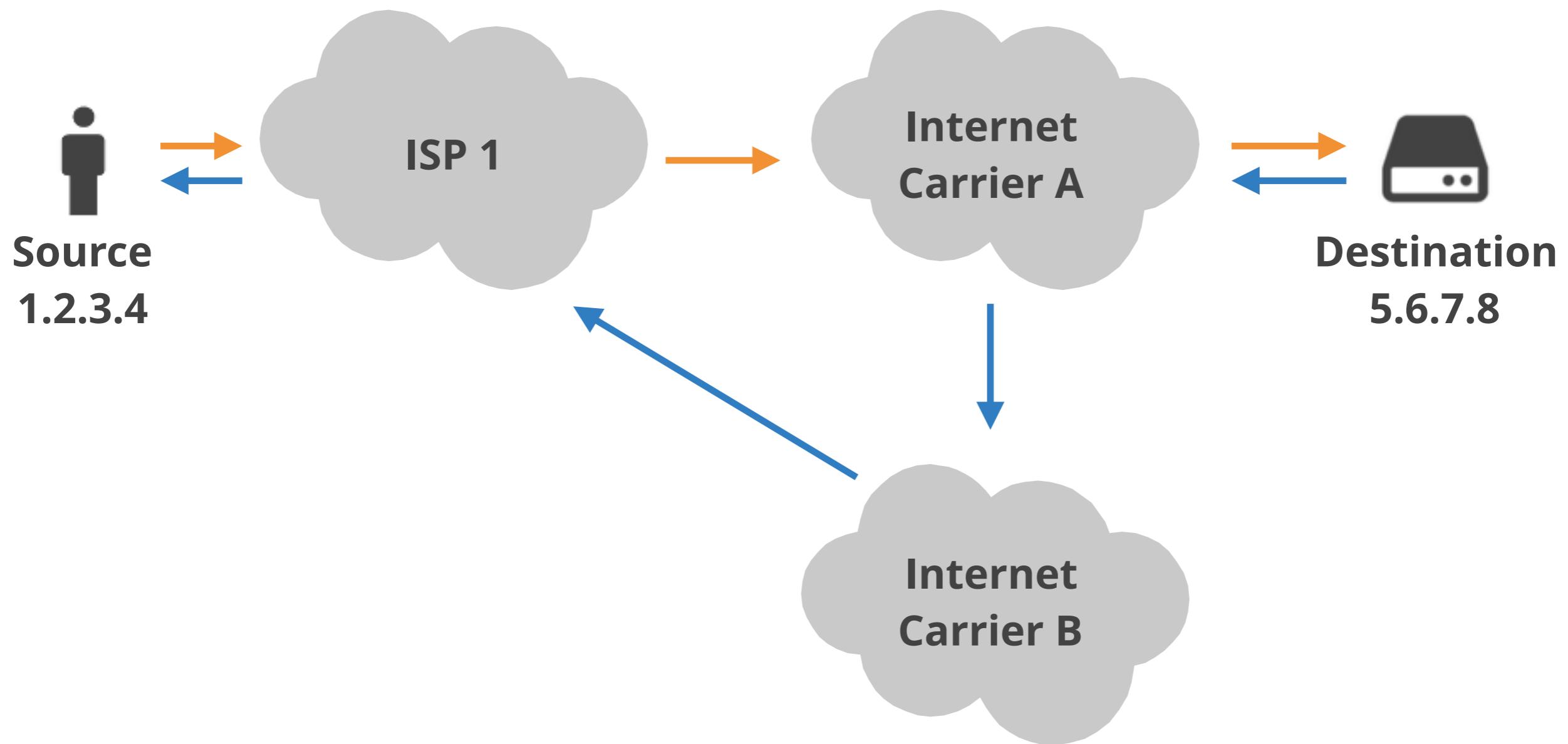
# Filtering is hard



# Filtering is hard



# Internet is asymmetric



# Filter close to the source

