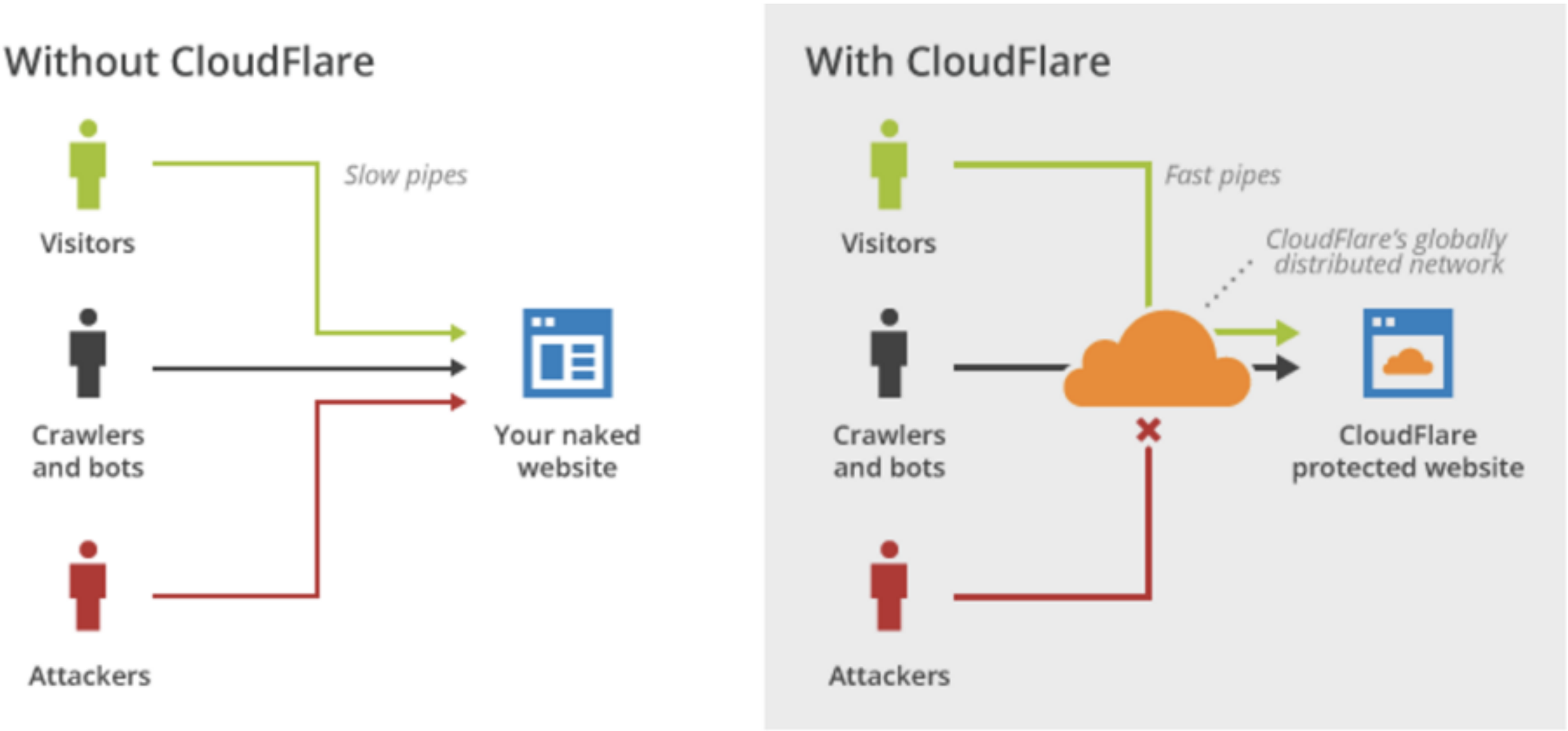


Building a DDoS Mitigation Pipeline

Marek Majkowski

"Help Build a Better Internet"



Content neutral

































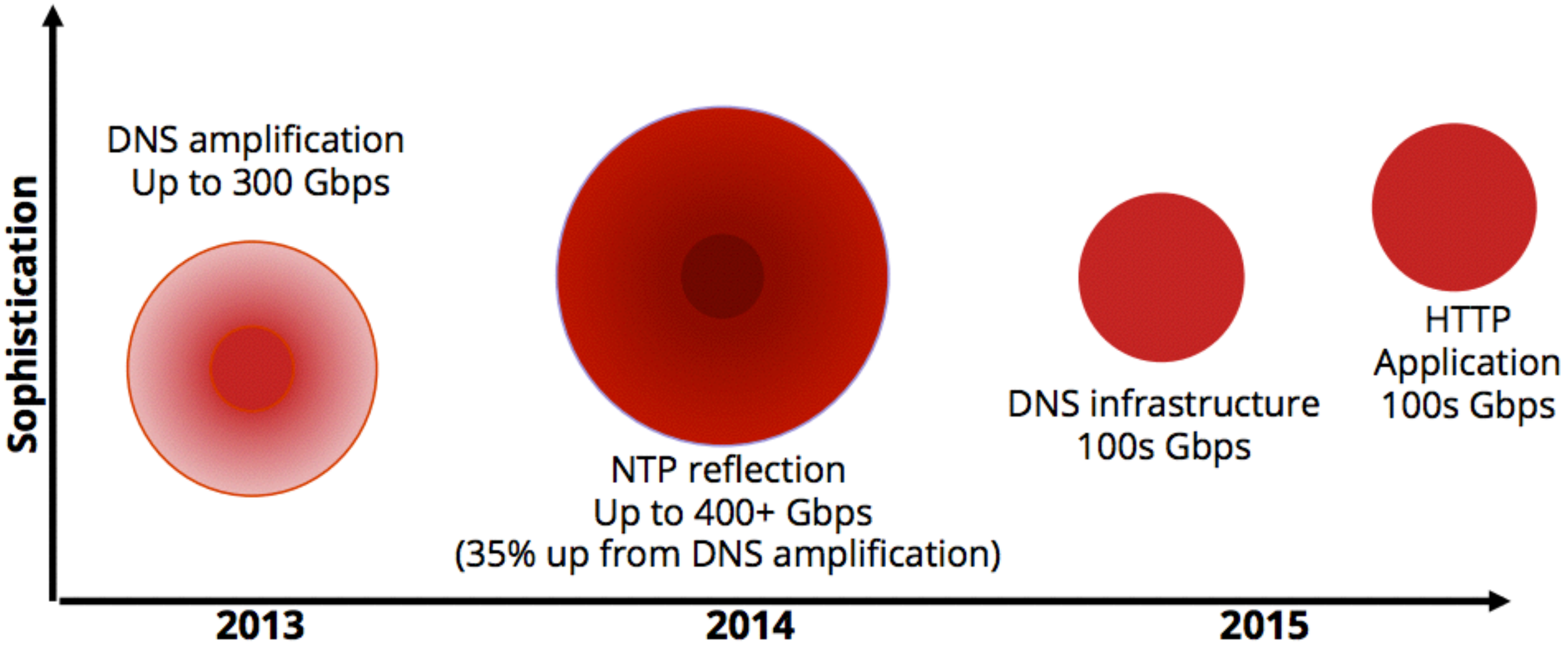








DDoS is a threat



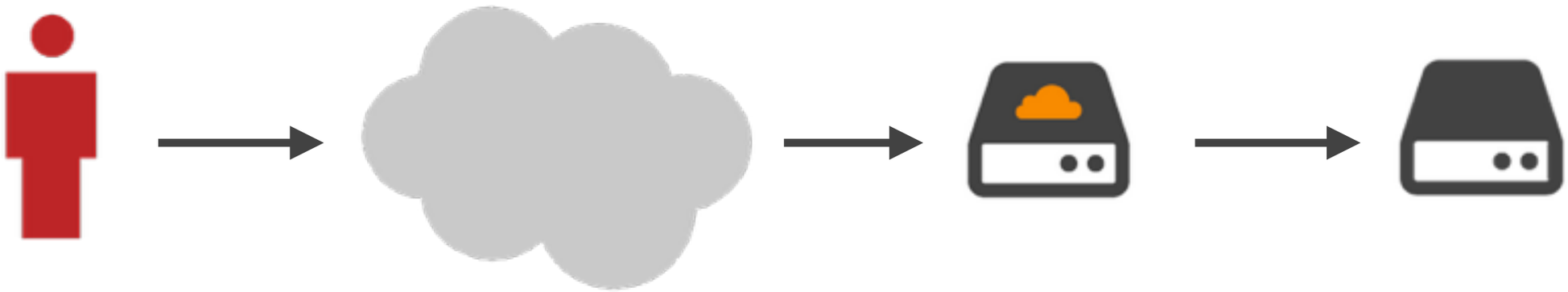
Big effort

trust & safety team

working with operators

improving our infrastructure

public outreach



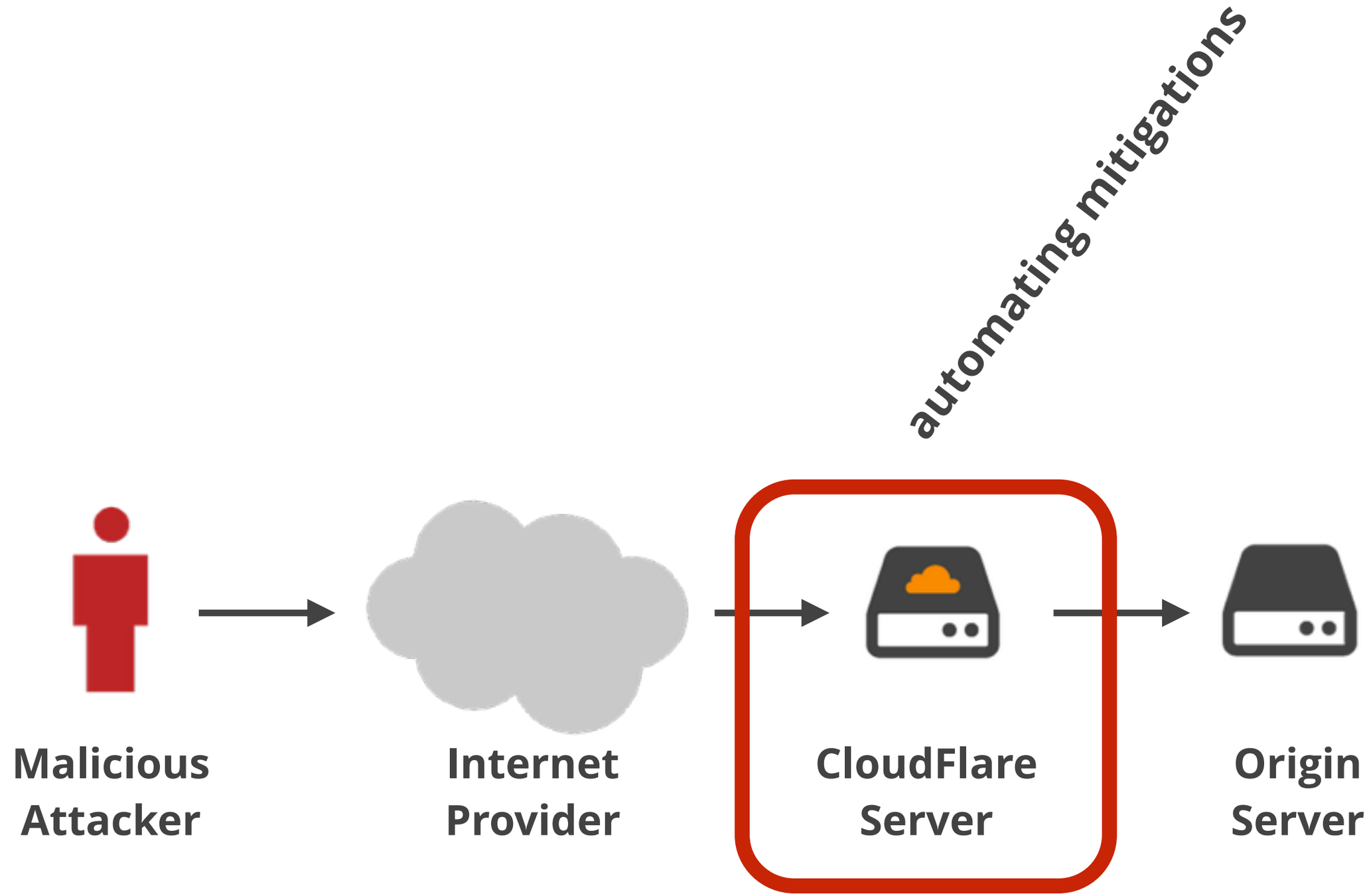
Malicious Attacker

Internet Provider

CloudFlare Server

Origin Server

Automated DDoS Mitigations



attack volume > CloudFlare network capacity



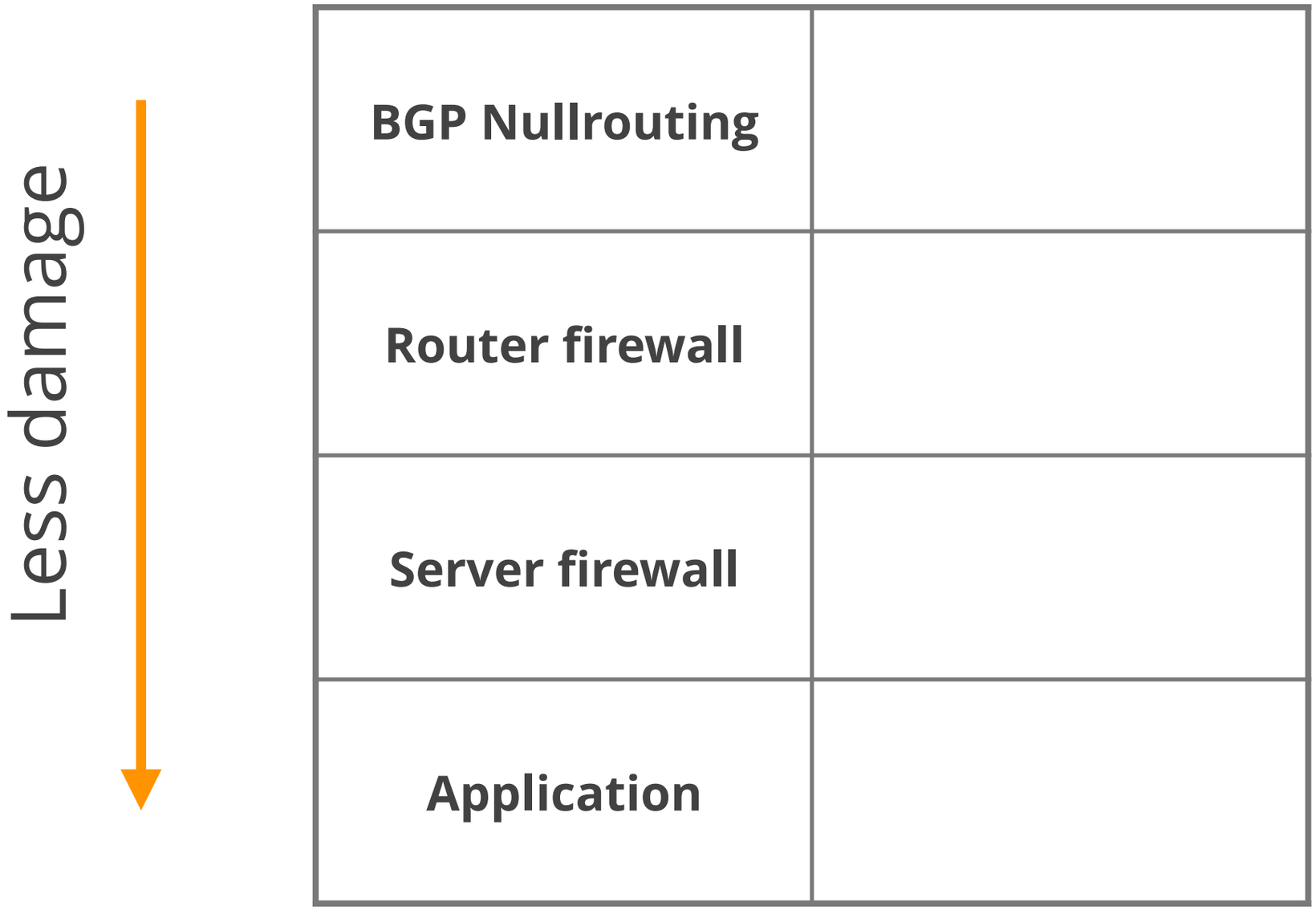
BGP Nullroute and move on

```
route 1.2.3.4/32 {
    discard;
    community [ 13335:666 13335:668 13335:36006 ];
}
```

attack volume < CloudFlare network capacity



Reducing damage



Reducing damage

More precision

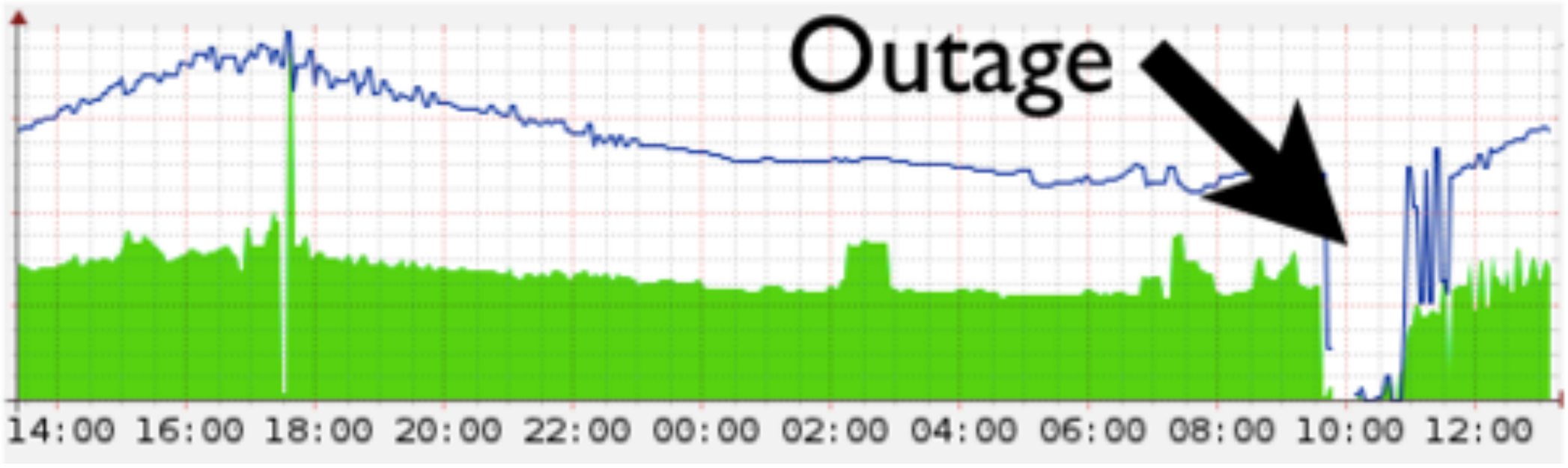
BGP Nullrouting	IP
Router firewall	IP, port, packet length
Server firewall	all above + stateless DPI parameters
Application	all above + application logic

Speed



Precision

Operator


Outage
14:00
16:00
18:00
20:00
22:00
00:00
02:00
04:00
06:00
08:00
10:00
12:00

Speed



Precision

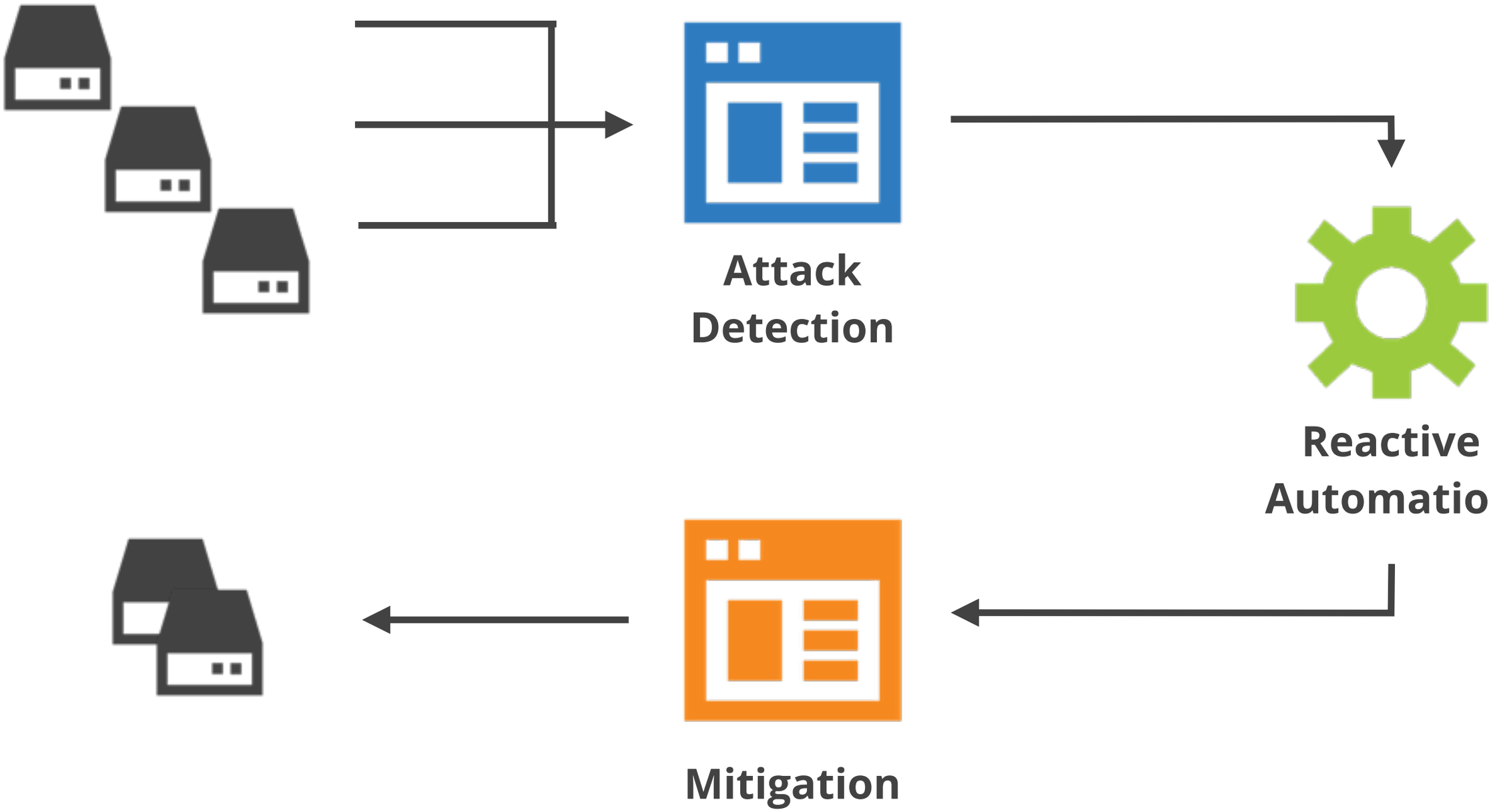
Automation

Automatic attack handling

Speed ⟵  ⟶ **Precision**

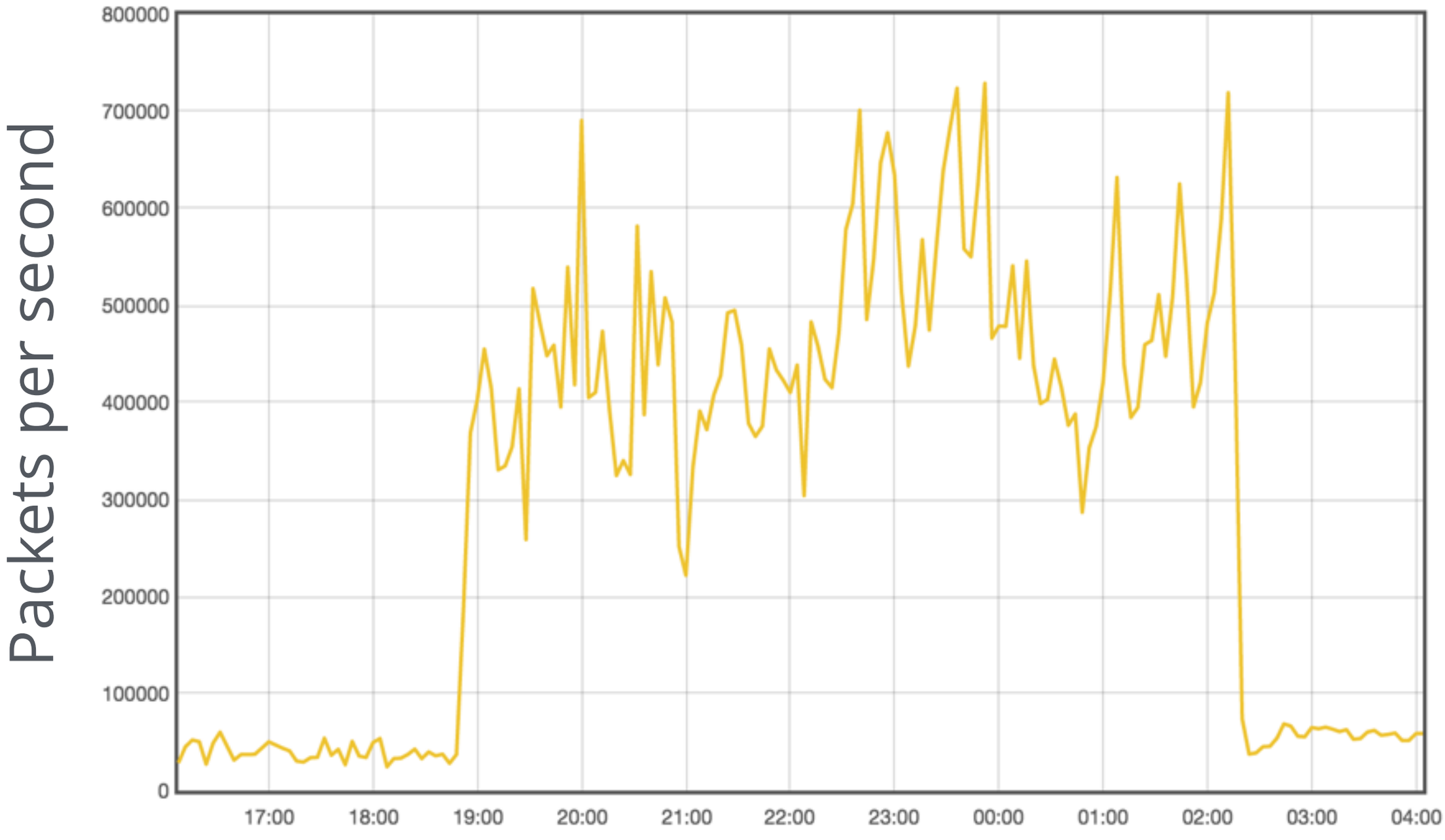
Gatebot

Automatic attack handling



The attack

High volume packet floods



DNS packet flood

```
$ tcpdump -ni eth2 inbound and port 53 -c 100

IP 202.194.181.95.15443 > 1.2.3.4:53: 63476% [1au] A? example.com. (50)
IP 221.12.236.115.6570  > 1.2.3.4:53: 11406% [1au] A? example.com. (50)
IP 203.94.134.43.18473  > 1.2.3.4:53: 8559%  [1au] A? example.com. (50)
IP 203.196.66.75.32573  > 1.2.3.4:53: 47971% [1au] A? example.com. (50)
IP 124.240.198.136.2333 > 1.2.3.4:53: 61152% [1au] A? example.com. (50)
IP 218.247.70.185.11679 > 1.2.3.4:53: 16360% [1au] A? example.com. (50)
IP 202.109.218.98.27549 > 1.2.3.4:53: 17829% [1au] A? example.com. (50)
IP 203.148.240.82.21825 > 1.2.3.4:53: 22590% [1au] A? example.com. (50)
IP 211.167.108.67.25782 > 1.2.3.4:53: 17663% [1au] A? example.com. (50)
IP 203.209.60.18.20221  > 1.2.3.4:53: 38257% [1au] A? example.com. (50)
IP 203.81.181.168.12749 > 1.2.3.4:53: 53492% [1au] A? example.com. (50)
```

1 in 10k packets is "real"



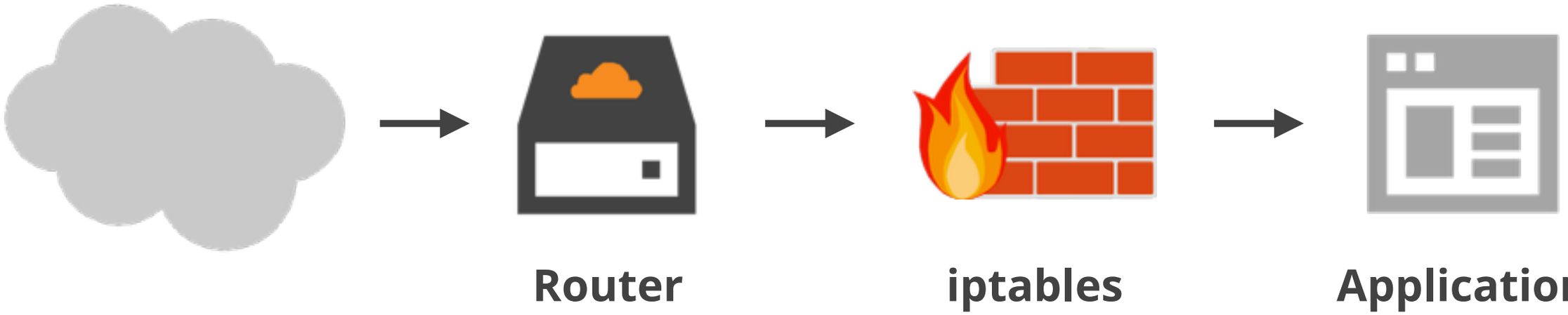
Finding attack parameters

```
IP 202.194.181.95.15443 > 1.2.3.4:53: 63476% [1au] A? example.com. (50)
IP 221.12.236.115.6570  > 1.2.3.4:53: 11406% [1au] A? example.com. (50)
IP 203.94.134.43.18473  > 1.2.3.4:53: 8559%  [1au] A? example.com. (50)
IP 203.196.66.75.32573  > 1.2.3.4:53: 47971% [1au] A? example.com. (50)
IP 124.240.198.136.2336 > 1.2.3.4:53: 61152% [1au] A? example.com. (50)
IP 218.247.70.185.11679 > 1.2.3.4:53: 16360% [1au] A? example.com. (50)
IP 202.109.218.98.27549 > 1.2.3.4:53: 17829% [1au] A? example.com. (50)
IP 203.148.240.82.21825 > 1.2.3.4:53: 22590% [1au] A? example.com. (50)
IP 211.167.108.67.25782 > 1.2.3.4:53: 17663% [1au] A? example.com. (50)
IP 203.209.60.18.20221  > 1.2.3.4:53: 38257% [1au] A? example.com. (50)
IP 203.81.181.168.12749 > 1.2.3.4:53: 53492% [1au] A? example.com. (50)
```



Mitigation

Where to DROP?



Router **iptables** **Application**

Traffic matching 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

```
    ldx 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
```

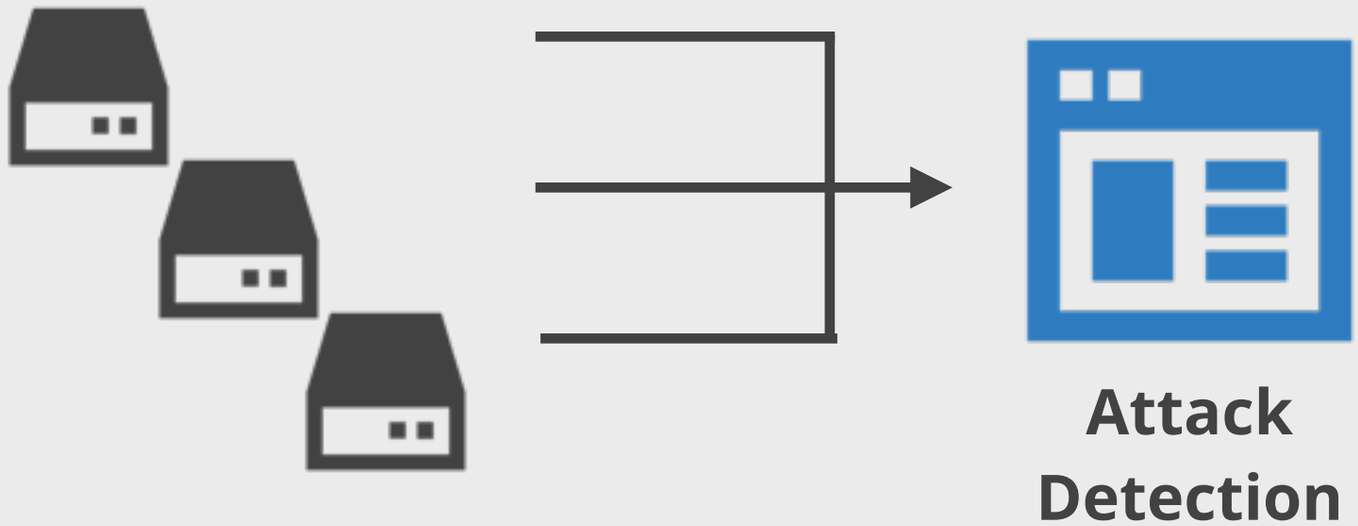
Deployment



Mitigation database

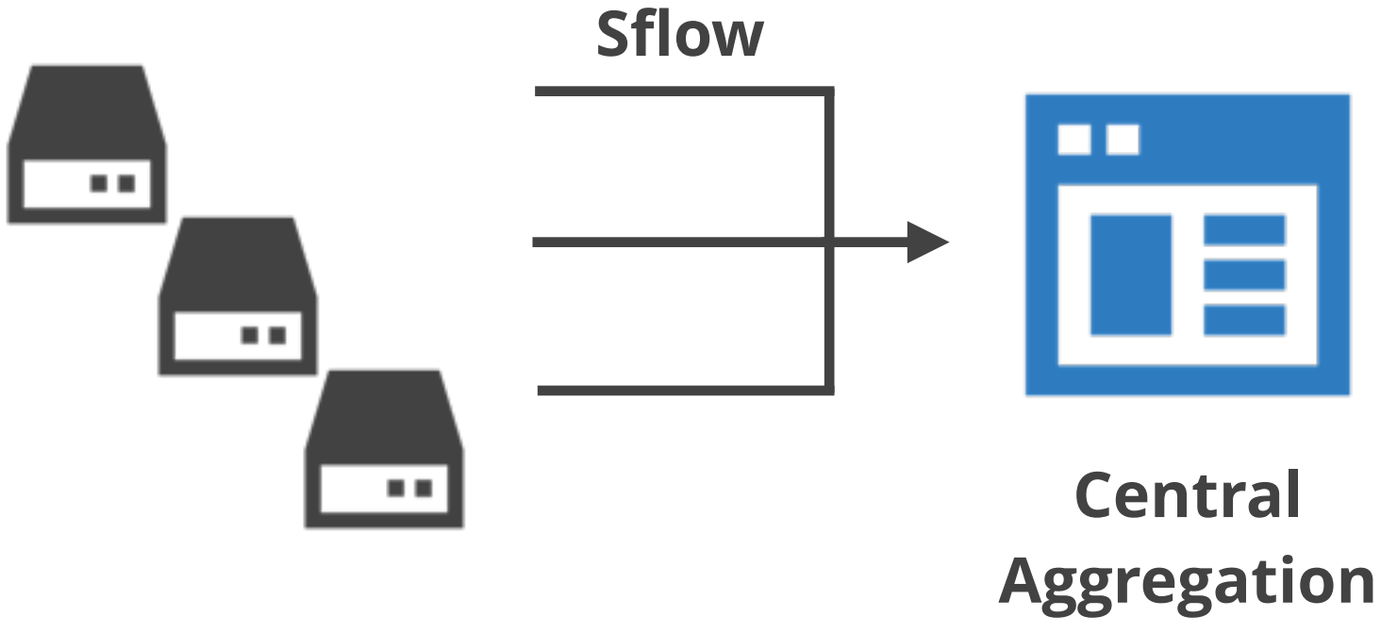
```
$ gatekeeper dnsbpf list
--ip=1.2.3.4 *.example.com
--ip=4.3.2.1 www.test.de *.www.test.de
--ip=4.3.4.4 *.cloudflare.com --except=www.** --except=ns1.**
--ip=2.3.1.4 www.onedomain.com,wwww.seconddomain.com
--ip=1.2.3.0/24 test.com

$ gatekeeper dnsbpf add -- --ip=4.3.2.1 *.newattack.com
```



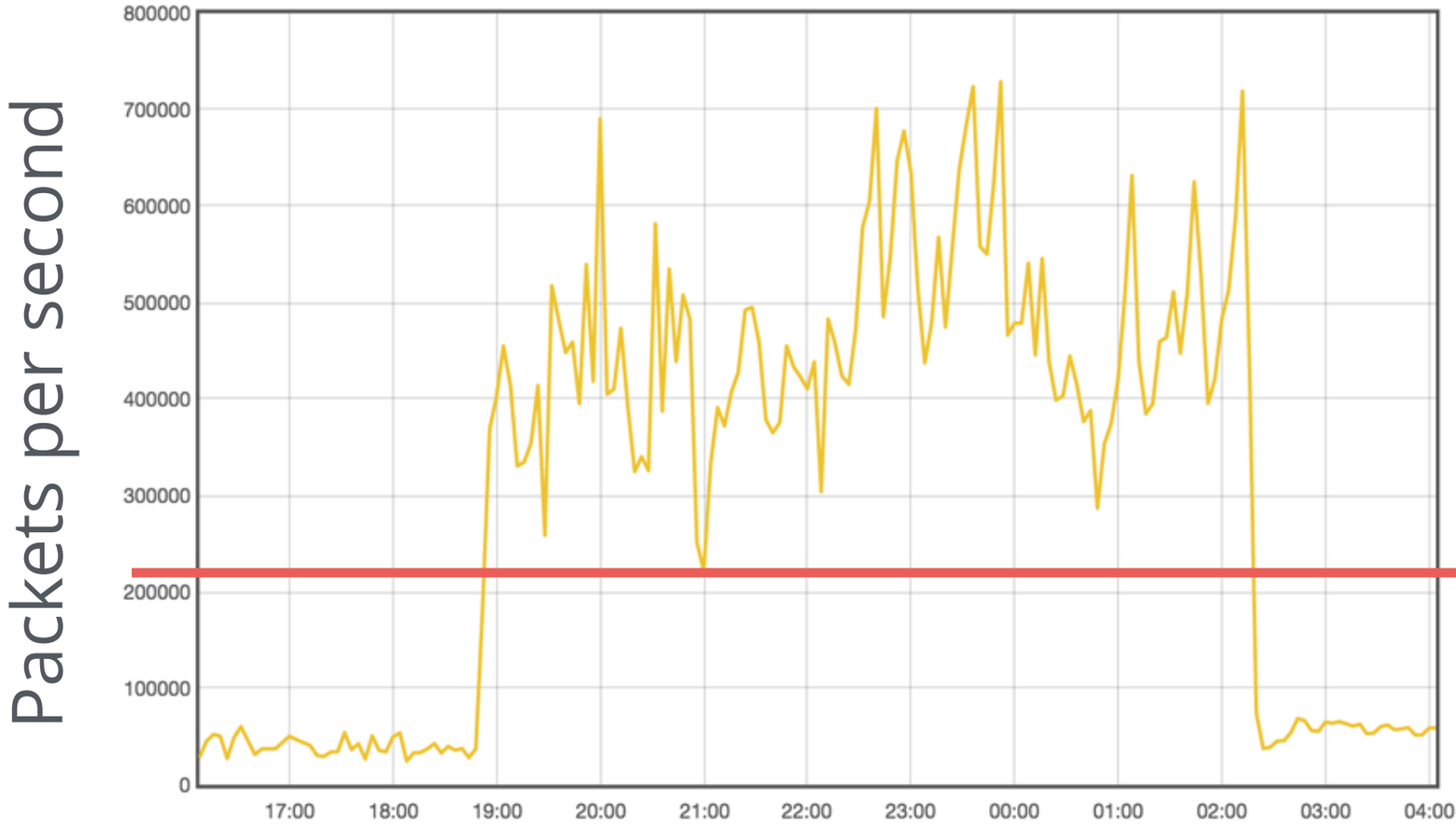
Detection

Sflow



What is an "attack"?

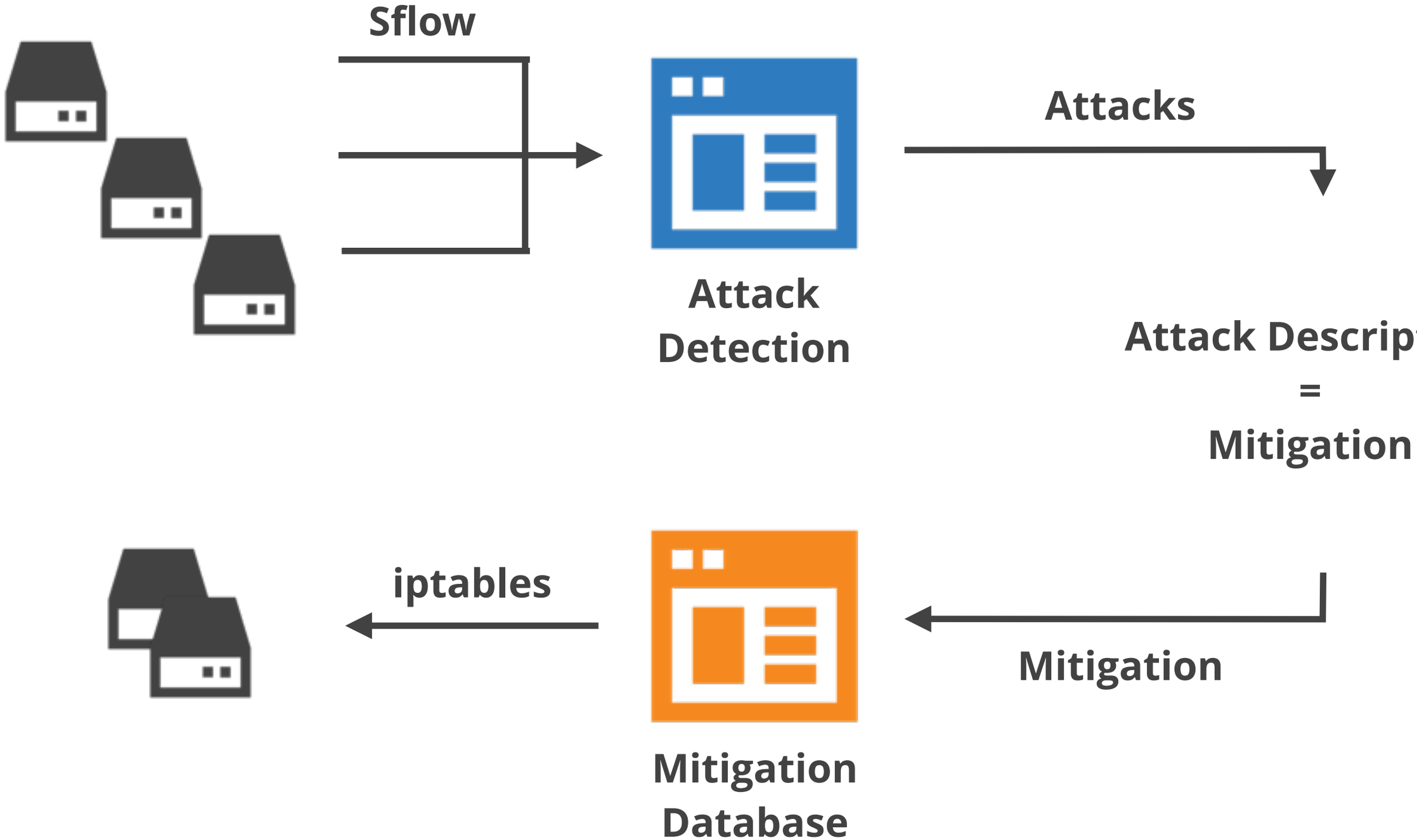
"Attack" is large



Large attacks

Small attacks

"Attack" can be mitigated



"Attacks" shall be aggregated

```
Mpps   Descr
3.878 --ip=141.245.59.191/32
2.878 --ip=141.245.59.192/32
1.878 --ip=141.245.59.193/32
1.878 --ip=141.245.59.194/32
1.878 --ip=141.245.59.195/32
1.878 --ip=141.245.59.196/32
1.878 --ip=141.245.59.197/32
1.878 --ip=141.245.59.198/32
1.878 --ip=141.245.59.199/32
...
```

VS

```
Mpps   Descr
 35.878 --ip=141.245.59.0/24
```

An attack-finding algorithm

Top N / Heavy hitters

- Fixed memory size; Algorithm: Space Saving
- https://github.com/cloudflare/golibs

pps	IP
12.2M	1.2.3.4
2.4M	42.1.2.4
0.01M	2.4.3.1
0.01M	192.168.1.1

Multiple dimensions

pps	IP:port
12.2M	1.2.3.4:53
2.4M	42.1.2.4:80
0.01M	2.4.3.1:80
0.01M	192.168.1.1:443

pps	IP
12.2M	1.2.3.4
2.4M	42.1.2.4
0.01M	2.4.3.1
0.01M	192.168.1.1

pps	subnet
12.2M	1.2.3.0/24
2.4M	42.1.2.0/24
0.01M	2.4.3.0/24
0.01M	192.168.1.0/24

Multiple dimensions

incoming sample: **42.1.2.4:80**

pps	IP:port
12.2M	1.2.3.4:53
2.4M	**42.1.2.4:80**
0.01M	2.4.3.1:80
0.01M	192.168.1.1:443

pps	IP
12.2M	1.2.3.4
2.4M	**42.1.2.4**
0.01M	2.4.3.1
0.01M	192.168.1.1

pps	subnet
12.2M	1.2.3.0/24
2.4M	**42.1.2.0/24**
0.01M	2.4.3.0/24
0.01M	192.168.1.0/24

Multiple dimensions

reporting threshold: 1M

pps	IP:port
12.2M	**1.2.3.4:53**
2.4M	**42.1.2.4:80**
0.01M	2.4.3.1:80
0.01M	192.168.1.1:443

pps	IP
12.2M	**1.2.3.4**
2.4M	**42.1.2.4**
0.01M	2.4.3.1
0.01M	192.168.1.1

pps	subnet
12.2M	**1.2.3.0/24**
2.4M	**42.1.2.0/24**
0.01M	2.4.3.0/24
0.01M	192.168.1.0/24

Attack report

```
Mpps    Descr
12.2 --ip=1.2.3.4 --port=53
 2.4 --ip=42.1.2.4 --port=80
12.2 --ip=1.2.3.4
 2.4 --ip=42.1.2.4
12.2 --ip=1.2.3.0/24
 2.4 --ip=42.1.2.0/24
```

Multiple dimensions

incoming sample: **42.1.2.4:80**

pps	IP:port
12.2M	1.2.3.4:53
2.4M	**42.1.2.4:80**
0.01M	2.4.3.1:80
0.01M	192.168.1.1:443

pps	IP
0.1M	1.2.3.4
0M	42.1.2.4
0.01M	2.4.3.1
0.01M	192.168.1.1

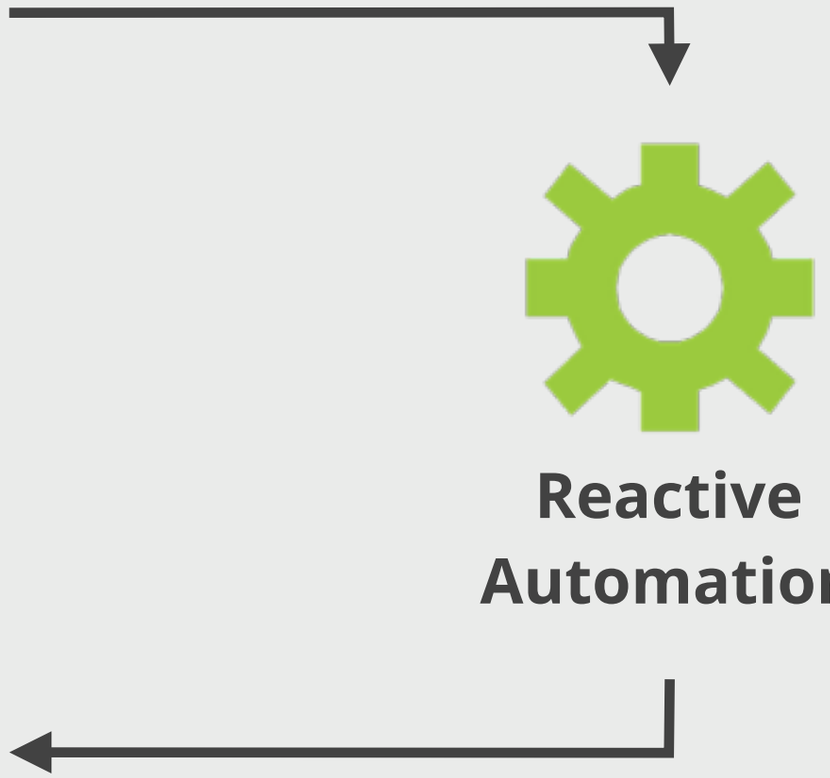
pps	subnet
0.1M	1.2.3.0/24
0M	42.1.2.0/24
0.01M	2.4.3.0/24
0.01M	192.168.1.0/24

Attack report

```
  Mpps    Descr
 12.2 --ip=1.2.3.4 --port=53
  2.4 --ip=42.1.2.4 --port=80
```

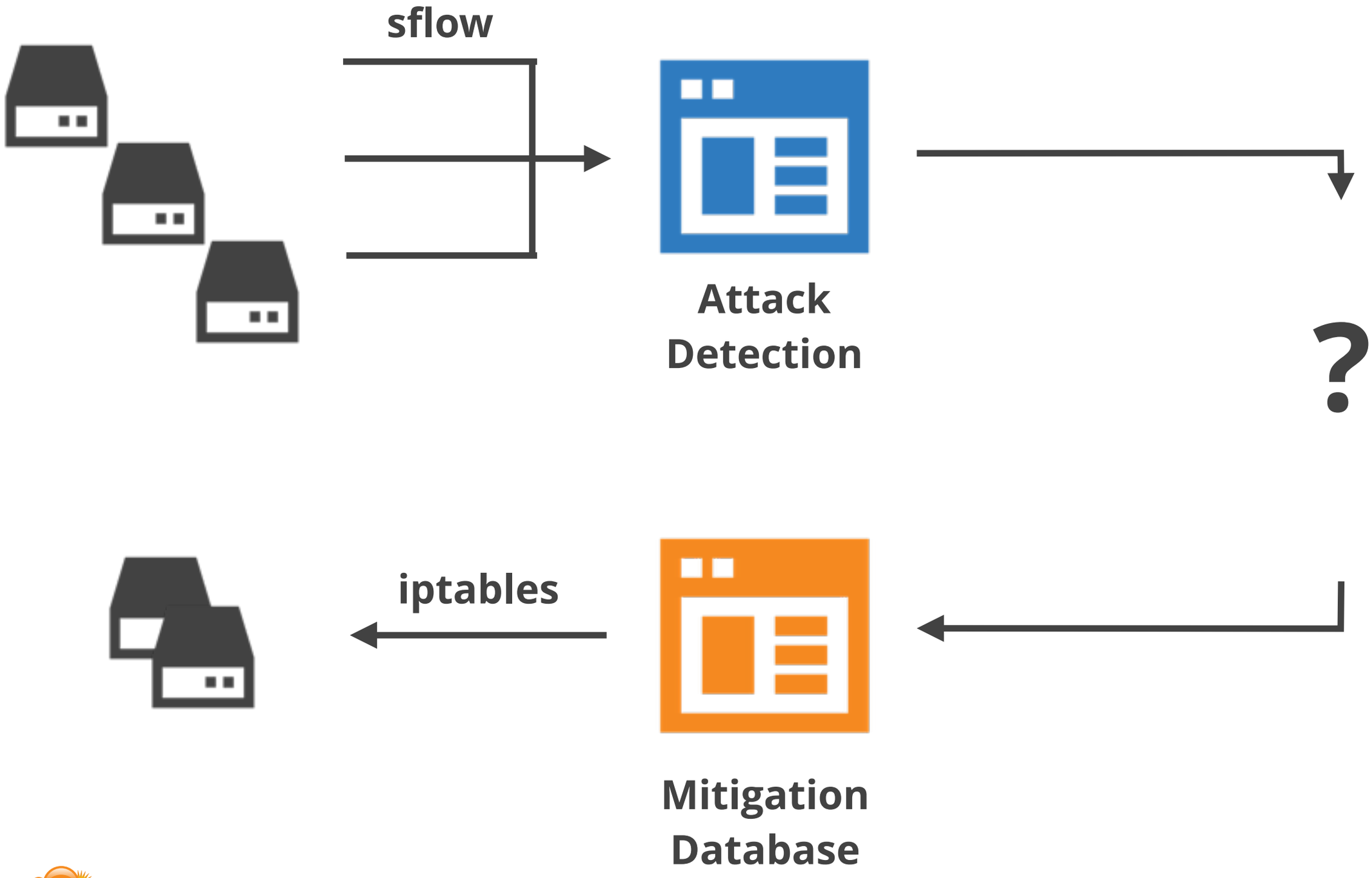
Scales well

```
INCOMING/UDP  samples=11836 pps={total= 65.176M, accounted= 65.174M}
   Mpps   rule hint
  1.160     dnsbpf -- --ip=173.245.59.
  3.953     dnsbpf -- --ip=173.245.59.
  3.892     dnsbpf -- --ip=173.245.58.
  4.169     dnsbpf -- --ip=173.245.58.
  3.750     dnsbpf -- --ip=173.245.58.
  4.098     dnsbpf -- --ip=173.245.59.
  0.965     dnsbpf -- --ip=173.245.59.
  3.910     dnsbpf -- --ip=173.245.59.
  0.973     dnsbpf -- --ip=173.245.59.
  3.811     dnsbpf -- --ip=173.245.58.
  3.770     dnsbpf -- --ip=173.245.59.
  3.879     dnsbpf -- --ip=173.245.58.
  3.857     dnsbpf -- --ip=173.245.58.
  3.975     dnsbpf -- --ip=173.245.58.
  0.545     dnsbpf -- --ip=
  3.406     dnsbpf -- --ip=
  0.315     dnsbpf -- --ip=173.245.58.
  0.415     dnsbpf -- --ip=173.245.59.
```

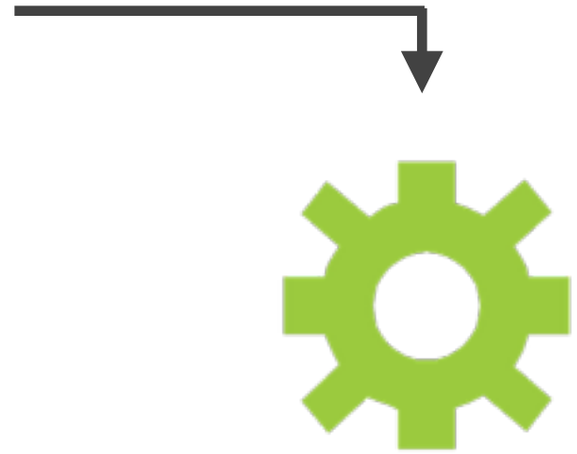


Reactive automation

Connecting the pieces



```
--ip=1.2.3.4 example.com
```

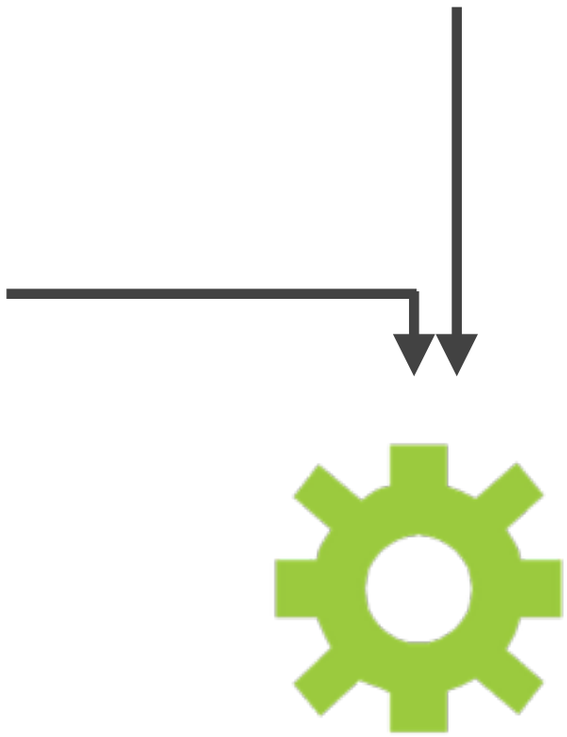


Reactive Rule



```
--ip=1.2.3.4 example.com --qps=100
```

```
example.com = FREE | PAID
```



Reactive Rule

```
--ip=1.2.3.4 example.com
```



```
--ip=1.2.3.4 example.com --qps=500
```

```
example.com subdomains:
(www, ns1, ns2)
```

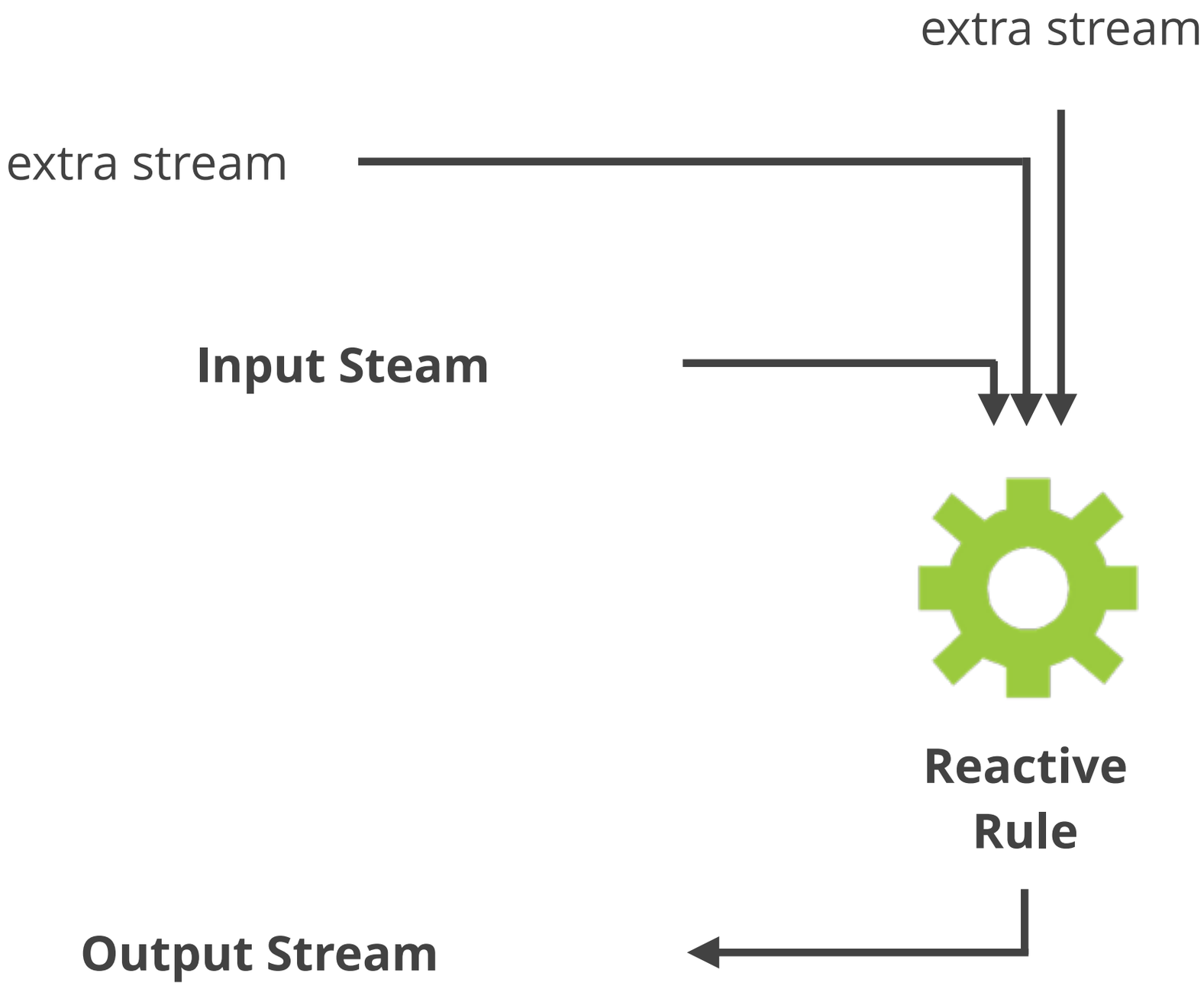
```
example.com = FREE | PAID
```

```
--ip=1.2.3.4 example.com
```



Reactive Rule

```
--ip=1.2.3.4 example.com --except www,n1,ns2 --qps=500
```


extra stream
extra stream
Input Steam
Reactive
Rule
Output Stream

Chain of transformations

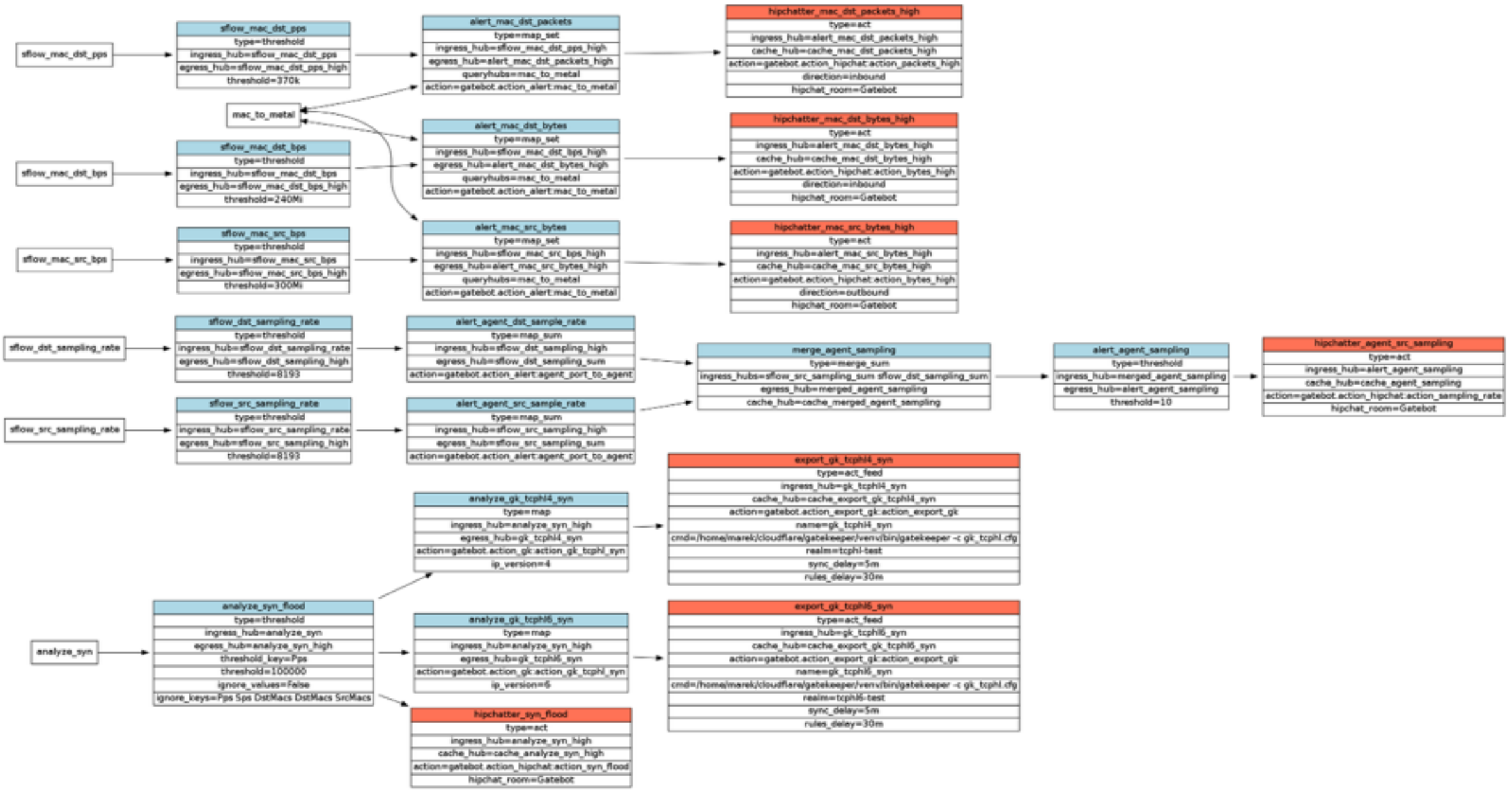
```
def dns_mitigation(attack, plan, subdomains):
    domain = attack['domain']

    qps = 100
    if plan[domain] == 'business':
        qps = 500

    mitigation =
        attack['description'] + \
        ' --qps=%s' % qps + \
        ' --except=%s'.join(subdomains[domain])

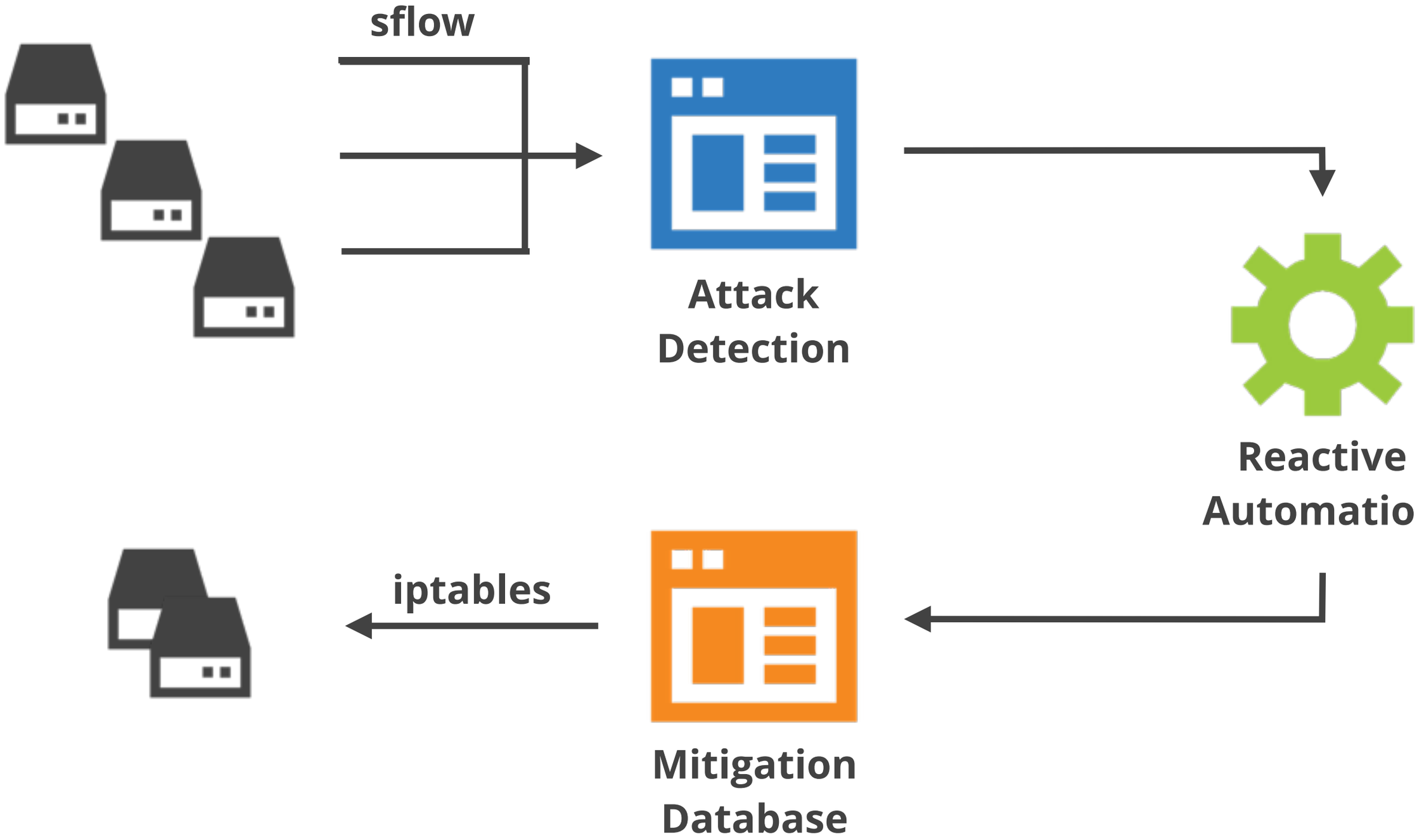
    return mitigation
```

Fully composable

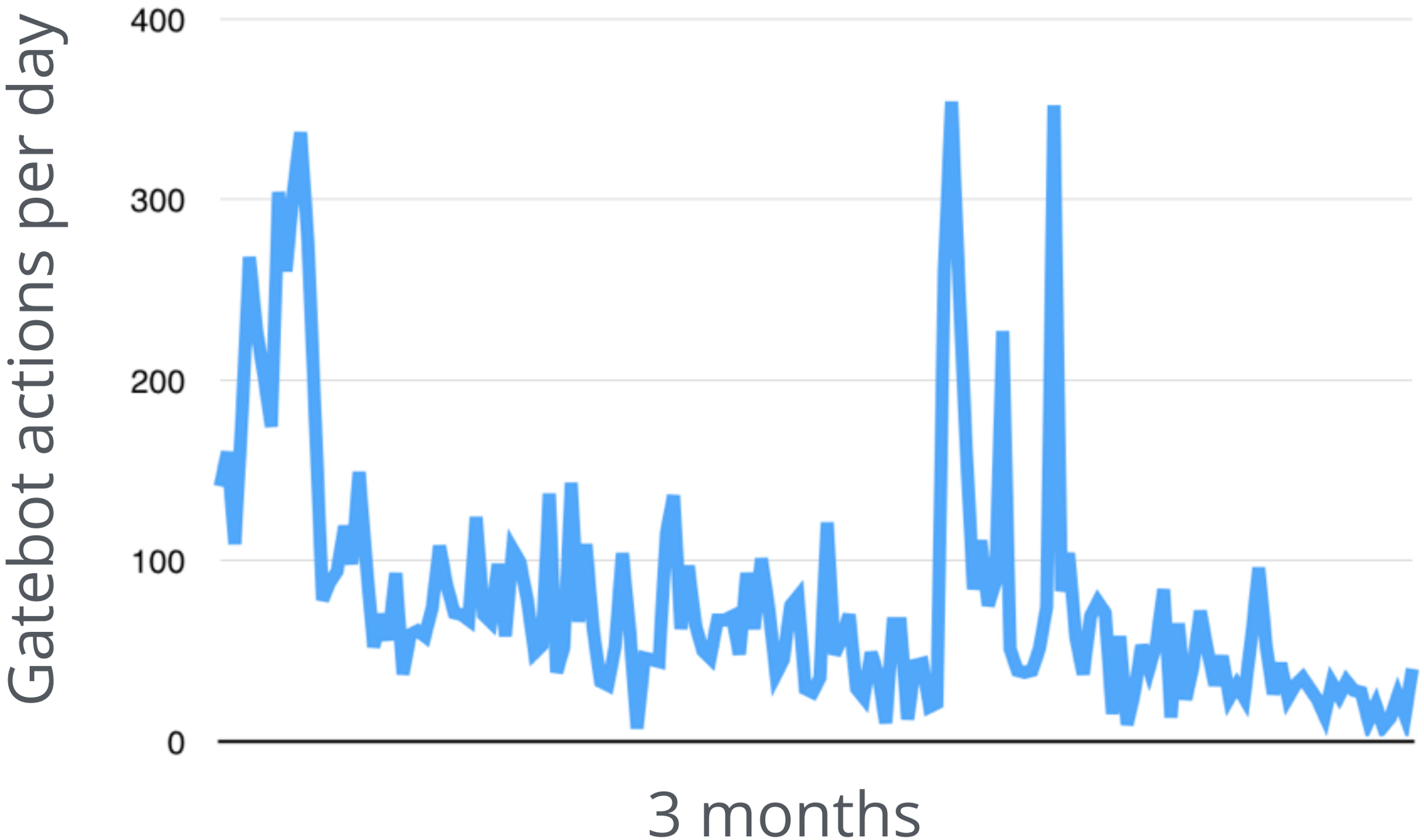


Putting it all together

Putting it all together



Gatebot: frequency



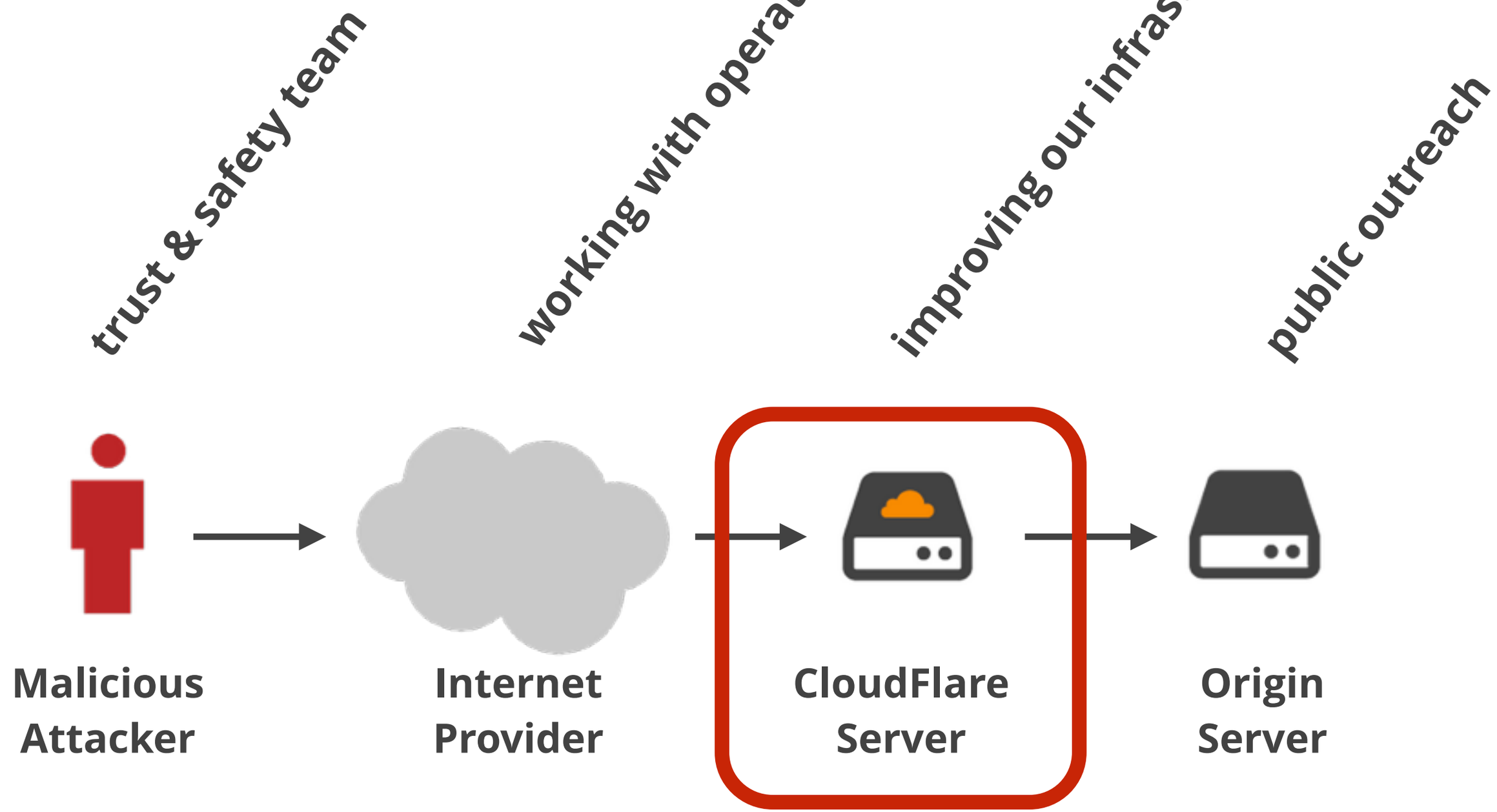
Gatebot: volume



1 week

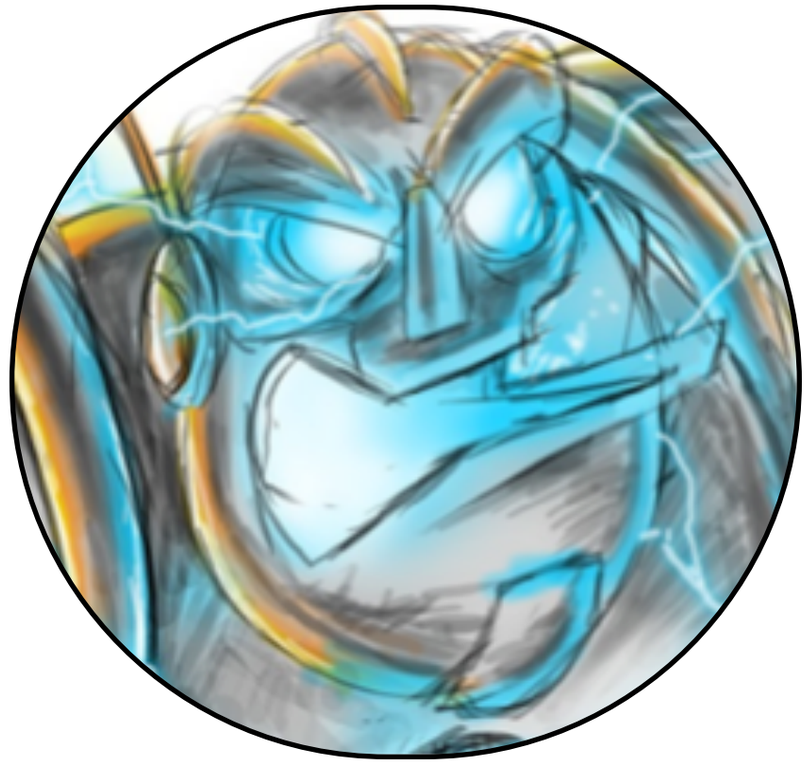
Summary

The fight goes on



Thanks!

and good luck!



@cfgatebot

- https://blog.cloudflare.com
- https://github.com/cloudflare

marek@cloudflare.com
@majek04