Super Fast Packet Filtering with eBPF and XDP

Helen Tabunshchyk, Systems Engineer, Cloudflare

@advance_lunge
Agenda

1. Background.
2. A tiny bit of theory about routing.
3. Problems that have to be solved.
4. Overview of existing solutions.
5. DDoS mitigation pipeline.
6. eBPF and XDP.
A bit of context about the work I do
What does Cloudflare do?

**CDN**
- Moving content physically closer to visitors with our CDN
- Intelligent caching
- Unlimited DDOS mitigation

**Website Optimisation**
- TLS 1.3 (with 0-RTT)
- HTTP/2 + QUIC
- Server push
- AMP
- Origin load-balancing
- Smart routing
- Workers
- Post quantum crypto
- Many more

**DNS**
- Cloudflare is the fastest managed DNS providers in the world.
- 1.1.1.1 and 2606:4700:4700::1111
- DNS over TLS
• 154 data centres in 74 countries
• More than 10 million domains
• 10% of all Internet requests
• 7.5M requests per second on average, 10M at peak
• 1.6M DNS queries per second
• 2.8 billion people served each month
• Biggest DDoS attack - 942 Gbps
• 20 Tbps network capacity and growing
Life of a packet
A long time ago in a galaxy far, far away...
The OPTE Project Internet 2015 Map

http://www.opte.org
Load Balancing Between Data Centres

- Locality and congestion control
- DNS
- BGP
- Anycast

https://www.cloudflare.com/learning/dns/what-is-dns/
**BGP — A Tale of Two Napkins**

At an Internet Engineering Task Force (IETF) conference last January, Kirk Longheed and Len Bosack of cisco and Yakov Rechter of IBM sat down in the meeting hall cafeteria and wrote a new routing protocol. What has since become RFC 1163, the Border Gateway Protocol (BGP), is still known to some as the “Two-Napkin Protocol,” in reference to the handy medium upon which the engineers first drafted it.

According to Longheed, cisco’s director of software engineering, BGP developed as a solution to the deficiencies of EGP. The problem evolved with the exponential increase in the number of Internet hosts, and with its expanding topology. “The Internet Protocol suite succeeded beyond anyone’s expectations,” Longheed explains. “EGP was simply not designed to handle networks of this size.”

With the Internet’s diversification and expanding routing domains, network managers soon needed to execute some control over their resources by introducing different types of user policies. EGP made no provisions for such policies. Nor did it scale to large numbers of networks. The networking community began to express a degree of concern that the core routing system would simply fail at some point. Moreover, EGP showed further signs of weakness as increasingly large routing updates were sent over the Internet. Protocol data containing these updates outgrew the ARPANET’s maximum transport size of 500 bytes, thus requiring fragmentation before transmission.

**Cisco Makes Bold Entry to OSI Marketplace, Designs Largest OSI Network Demo to Date**

The most complex OSI network ever assembled ran throughout the Interop 89 tradeshow this year in the San Jose Convention Center, Northern California. All together, about 14 vendors supporting the OSI network protocol successfully interconnected their systems to form the Interop OSI demo network.

cisco played a major role in the triumph of the OSI demo. Routers from cisco—running the ISO CLNS (Connectionless Network Services) protocol—managed...
Types of Routing

- **Anycast**: A single IP address is associated with multiple servers. A single request can be sent to any of the servers.

- **Unicast**: Each request is sent to a specific server, ensuring a more direct and potentially faster connection.

![Diagram illustrating anycast and unicast routing](image-url)
Okay, our little packet is inside the DC
Problems
1. Uneven load
Problems

2. Different kinds of traffic
Problems

3. Per packet load balancing

Image credit: https://flic.kr/p/imuUKx
Problems

4. Heterogenous hardware

Image credit: computer animated film Madagascar
Problems

5. Locality (e.g. for cache) and transport affinity

Image credit: https://www.flickr.com/photos/10361931@N06/4259933727/
Problems

6. DDoS

Image credit: Lockheed Martin F-22 Raptor https://youtu.be/UxgiBJATe9M
# Types of DDoS Attacks

<table>
<thead>
<tr>
<th></th>
<th>Volumetric Attack</th>
<th>Protocol Attack</th>
<th>Application Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>What is it?</strong></td>
<td>Saturating the bandwidth of the target.</td>
<td>Exploiting a weakness in the Layer 3 and Layer 4 protocol stack.</td>
<td>Exploiting a weakness in the Layer 7 protocol stack.</td>
</tr>
<tr>
<td><strong>How does it cripple the target?</strong></td>
<td>Blocks access to the end-resource.</td>
<td>Consume all the processing capacity of the attacked-target or intermediate critical resources.</td>
<td>Exhaust the server resources by monopolising processes and transactions.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>NTP Amplification, DNS Amplification, UDP Flood, TCP Flood, QUIC HelloRequest amplification</td>
<td>Syn Flood, Ping of Death, QUIC flood</td>
<td>HTTP Flood, Attack on DNS Services</td>
</tr>
</tbody>
</table>
Problems

7. Group change
Problems

8. Graceful connection draining

Image credit: German airforce
Load balancing techniques
ECMP

ID (packet) mod N,
ID - some function that produces connection ID, e.g. 5-tuple flow;
N - the number of configured backends.

Uneven load
Different kinds of traffic
Per packet load balancing
Heterogenous hardware
Locality
DDoS
Group change
Graceful connection draining
ECMP-CH

populating the ECMP table not simply with next-hops, but with a slotted table that's made up of redundant next-hops

- Uneven load
- Different kinds of traffic
- Per packet load balancing
- Heterogenous hardware
- Transport affinity
- DDoS
- Group change
- Graceful connection draining
Stateful Load Balancing

- Uneven load
- Different kinds of traffic
- Per packet load balancing
- Heterogenous hardware
- Transport affinity
- DDoS
- Group change
- Graceful connection draining
Google Maglev

https://ai.google/research/pubs/pub44824
• Beamer muxes do not keep per-connection state; each packet is forwarded independently.
• When the target server changes, connections may break.
• Beamer uses state stored in servers to redirect stray packets.
Beamer at work

MUX

<table>
<thead>
<tr>
<th>Bkt</th>
<th>DIP</th>
<th>PDIP</th>
<th>TS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Server₁

Server₂
Beamer at work

- Packets contain previous server and time of reassignment
Beamer at work

- New connections are handled locally
Beamer at work

- Daisy chained connections die off in time
Daisy Chaining a.k.a Beamer

- Uneven load
- Different kinds of traffic
- Per packet load balancing
- Heterogenous hardware
- Transport affinity
- DDoS
- Group change
- Graceful connection draining
- Performance

Spoilers: could be even better

https://www.usenix.org/conference/nsdi18/presentation/olteanu
https://github.com/Beamer-LB
Also FPGA Packet Processing

- Early experiments with FPGAs
- Smart NICs
- P4 language
Fun (?) Facts

https://www.fastly.com/blog/anatomy-an-iot-botnet-attack
An average IoT device gets infected with malware and launches an attack within 6 minutes of being exposed to the internet.
Over the span of a day an average of over 400 login attempts per device; 66 percent of them on average are successful.
Over the span of a day, IoT devices are probed for vulnerabilities 800 times per hour.
DDoS Mitigation
Disclaimer
OH: "In any team you need a tank, a healer, a damage dealer, someone with crowd control abilities, and another who knows iptables"
BPF and eBPF

- Low overhead sandboxed user-defined bytecode running in kernel
- Written in a subset of C, compiled by clang llvm
- It can never crash, hang or interfere with the kernel negatively
- If you run Linux 3.15 or newer, you already have it
- Great intro from Brendan Gregg: http://www.brendangregg.com/ebpf.html
Limitations

- Verifier is picky
- Instructions limit
- Difficult to debug
- No standard library
- Tricky with synchronisation primitives
End of disclaimer
DDoS Mitigation Pipeline
Gatebot
iptables

- Initially it was the only tool to filter traffic
- Leveraged modules ipsets, hashlimit, connlimit
- With the xt_bpf module it was possible to specify complex filtering rules
- But we soon started experiencing IRQ storms during big attacks
- All CPUs were busy dropping packets, userspace applications were starving of CPU
Userspace Offload a.k.a. Kernel Bypass

• Network traffic is offloaded to userspace before it hits the Linux network stack
• Allows to run BPF in userspace
• An order of magnitude faster than iptables (5M pps)

• Requires one or more CPUs to busy poll the NIC event queue
• Reinjecting packets in the network stack is expensive
• Hardware dependent
XDP to the rescue!
XDP Packet Processing Overview

+AF_XDP since 4.19

https://www.iovisor.org/technology/xdp
Limitations

- Only one program per interface (solved by tail-calling)
- Driver support
- And all of the limitations of eBPF
Sample usage
SEC("xdp1")
int xdp_prog(struct xdp_md * ctx)
{
  void * data   = (void *)(long)ctx->data;
  void * data_end = (void *)(long)ctx->data_end;
  int ret;

  ret = rule_1(data, data_end);
  if (ret != XDP_PASS) // multiple if statements
  {
    return ret;
  }

  ret = rule_2(data, data_end); // one for each rule
  if (ret != XDP_PASS)
  {
    return ret;
  }
...
  return XDP_PASS; // if none of them match - the packet is accepted
}
static inline int rule_1 (void * data, void * data_end) {
    if (!condition_1) {
        return XDP_PASS;
    }
    if (!condition_2) // multiple if conditions
    {
        return XDP_PASS;
    }
    ...
    update_rule_counters(1); // do additional work
    sample_packet(data, data_end);
    return XDP_DROP;
}
// eBPF map shared with userpace

struct bpf_map metrics_map __section("maps") = {
    .type = BPF_MAP_TYPE_PERCPU_ARRAY,
    .key_size = sizeof(unsigned int),
    .value_size = sizeof(rule_metrics_t),
    .max_entries = STATS_MAP_SIZE,
};

static inline void update_rule_counters(int rule_id) {
    long * value = bpf_map_lookup_elem(&metrics_map, &rule_id);
    if (value) {
        *value += 1;
    }
}
A perfect match: XDP both for load balancing and DDoS mitigation 😄
XDP L4LB with daisy chaining using encapsulation

- Uneven load
- Different kinds of traffic
- Per packet load balancing
- Heterogenous hardware
- Transport affinity
- DDoS
- Group change
- Graceful connection draining
- Performance
And They Lived Happily Ever After
But… DPDK?
Advantages of XDP over DPDK

- Allows option of busy polling or interrupt driven networking
- No need to allocate huge pages
- Dedicated CPUs are not required, user has many options on how to structure the work between CPUs
- No need to inject packets into the kernel from a third party user space application
- No special hardware requirements
- No need to define a new security model for accessing networking hardware
- No third party code/licensing required
- More expensive in, surprise, passing packets to the network stack
Bonus part
Quick UDP Internet Connections

HTTP/2
TLS
TCP

HTTP over QUIC
QUIC
TLS 1.3
TCP-like congestion control, loss recovery
UDP

IP
L4LB to the rescue!
Thank you!

@advance_lunge