BBSakuraNetworks

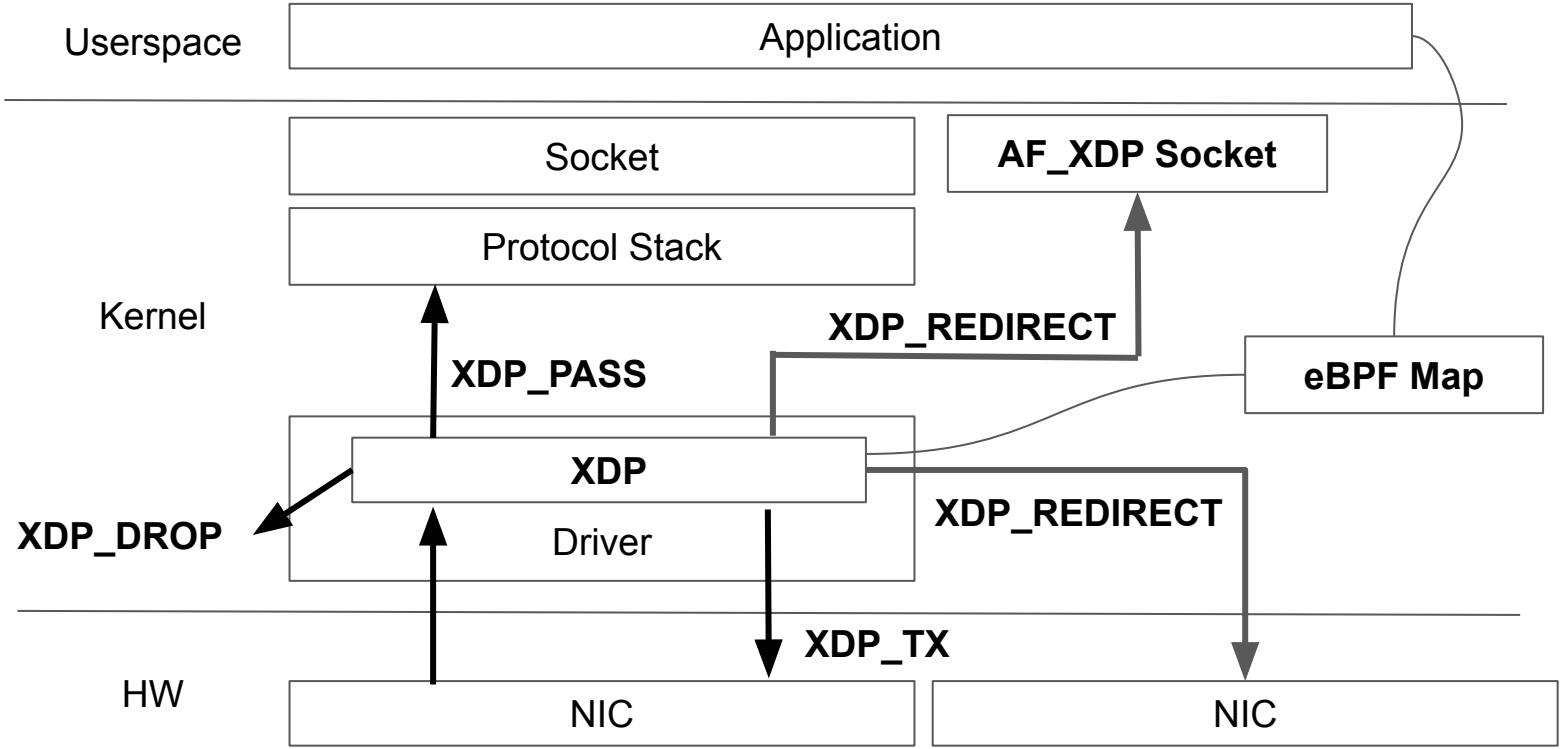
パケット処理の独自実装や高速化手法の比較と実践

独自パケット処理の実装方法の解説（XDP）

日下部雄也
BBSakura Networks株式会社
小樽市在住

XDPとは

BBSakuraNetworks



※XDP_REDIRECTはbpf_redirect(), bpf_redirect_map()を経由して使う

ユースケース（再掲）

BBSakuraNetworks

- bpfilter: iptablesをXDPにオフロード
- Cloudflare(DDoS Mitigation)
- Facebook(Katran): L4LB
- Cilium: Kubernetes上での透過的なセキュリティ、ネットワーク、LB
- XFLAG(Static NATなど)
- LINE(L4LB, SRv6)
- **BBSakura Networks(Mobile Core)**
  - さくらのクラウドでXDPを使えるようになっている
  - サーバのタグに **@nic-double-queue** と入れると使える
- OvS-AF_XDP
- DPDK-AF_XDP

XDPで使える機能(関数)

BBSakuraNetworks

- **bpf_map_{lookup,update,delete}_elem**
  - eBPF Mapの参照、更新、削除
- **bpf_fib_lookup**
  - FIB(Forwarding Information Base、ルーティングテーブル)の参照
- **bpf_xdp_adjust_head**
  - パケットデータの先頭をずらす
  - encap/decapで使う
- **bpf_xdp_adjust_tail**
  - パケットデータの末尾をずらす
  - パケットのヘッダだけ欲しいときに使うらしい (例: XFLAG ペイロードカッター)

XDPで使える機能(関数)

BBSakuraNetworks

- **bpf_xdp_adjust_meta**
  - パケットデータの先頭にメタデータの領域を確保する
  - XDP_PASS後にtc-bpfでskb->markに入れてtcやiptablesと連携するのによく使う
  - ちなみにvirtio_netなど非対応NICが多いので注意
    - xdp_set_data_meta_invalidでdriversフォルダをgrepするとわかる
    - virtio_netのパッチはある
  - 4.18まではVLANのパケットだと動かなかったが 4.19から直ってる
- **bpf_tail_call**
  - 別のプログラムを呼んで引き続きパケットを処理させる
  - xdpcapのようなキャプチャ用のプログラムや、メトリクス収集用、フィルタだけ先にやりたいので独立させるなどの用途がある
  - プログラムの命令数制限が緩和されたのと有限ループがサポートされたのでプログラムが大きくて渋々分けてtail_callというのはなくなりそう

XDPで使えるeBPF Mapの種類

BBSakuraNetworks

- **普通のMap**
  - BPF_MAP_TYPE_HASH
  - BPF_MAP_TYPE_ARRAY
  - BPF_MAP_TYPE_PERCPU_HASH: PERCPUなのでロックがなくなり性能が良い
  - BPF_MAP_TYPE_PERCPU_ARRAY
  - BPF_MAP_TYPE_LRU_HASH: LRU(マップが溢れたときにあまり使われないデータから消してくれる)
  - BPF_MAP_TYPE_LRU_PERCPU_HASH
- **Map in Map**
  - BPF_MAP_TYPE_ARRAY_OF_MAPS
  - BPF_MAP_TYPE_HASH_OF_MAPS
- **特殊なMap**
  - BPF_MAP_TYPE_LPM_TRIE: ルートテーブルのようなテーブルを作りたいときに使う
  - BPF_MAP_TYPE_PROG_ARRAY: bpf_tail_callで使う
  - BPF_MAP_TYPE_PERF_EVENT_ARRAY: ユーザスペースに対してイベントを送りたいときに使う
  - BPF_MAP_TYPE_DEVMAP: XDP_REDIRECTで別のNICにパケットを渡すときに使う
  - BPF_MAP_TYPE_CPUMAP: XDP_REDIRECTで別のCPUにパケットを渡すときに使う
  - BPF_MAP_TYPE_XSKMAP: AF_XDPで使う
- 使い方はLinuxの samples/bpf

開発の流れ

BBSakuraNetworks

1. **わりと制限のあるC**でソースコードを書く
2. LLVMを使ってバイトコードにコンパイル
3. コンパイルしたオブジェクトファイルをNICにアタッチ

わりと制限のあるC

BBSakuraNetworks

- **命令数制限**
  - 4.13までは4096
  - 4.14から128K https://patchwork.ozlabs.org/patch/798665/
  - 5.3から1M https://patchwork.ozlabs.org/patch/1073794/
  - ※iproute2でロードしたときは今のところ、常に 4096まで
    https://git.kernel.org/pub/scm/network/iproute2/iproute2.git/tree/lib/bpf.c?h=v5.4.0&#n305
- **ループ制限**
  - 無限ループは禁止
  - 5.2から有限ループOK https://lwn.net/Articles/794934/
  - 5.1まではunrollする必要がある←プログラムサイズが大きくなりがち
- **その他**
  - 到達不可能な命令がないこと
  - 有効なメモリにのみアクセスする（明示的に有効かどうかのチェックが必要）

コンパイル方法

BBSakuraNetworks

- **シンプルなプログラムの場合**
  - clang -O2 -target bpf -c example.c -o example.o
  - ただし、linux/if_ether.h などをincludeしたらエラーになるため、ほぼ使えない
- **実用的なプログラムの場合**
  - clang -I./include -DDEBUG -D__KERNEL__ -Wno-unused-value -Wno-pointer-sign -Wno-compare-distinct-pointer-types -O2 -emit-llvm -c -g src/example.c -o -| llc -march=bpf -filetype=obj -o src/example.o
  - サンプル https://github.com/higebu/ebpf-template
- **BCCの場合はソースを読み込むのが普通**
  - 詳細はリンク先参照

テスト

BBSakuraNetworks

- **BPF_PROG_TEST_RUN**を使うと実際にNICにアタッチすることなくテストすることができる
- libbpfを使った場合の例
  - https://github.com/higebu/ebpf-template/blob/master/test/test_xdp.c
- ただし、カーネルのバージョンや、NICのドライバによって、対応している機能が異なっていたり、特有のバグがあったりするので、本番で動かす前には必ず本番と同じ環境でテストする

デバッグ方法

BBSakuraNetworks

- **llvm-objdump -S --no-show-raw-insn example.o**
  - Verifierに怒られたときはとりあえずこれ
  - clangでコンパイルするときに -g を付けていればバイトコードに対応したソースコードも表示される
- **bpf_trace_printk()**
  - 性能に影響があるので、本番用のプログラムでは実行されないようにする
  - 普段は **trace-cmd record -e 'xdp:*' -O trace_printk** という感じで出力された trace.dat を**kernelshark**で見ている
  - trace-cmd stream は動かないことが多い。。。

```
Disassembly of section xdp_prog:

0000000000000000 prog:
; int prog(struct xdp_md *ctx) {
       0:       r0 = 1
;   void *data_end = (void *)(long)ctx->data_end;
       1:       r2 = *(u32 *)(r1 + 4)
;   void *data = (void *)(long)ctx->data;
       2:       r1 = *(u32 *)(r1 + 0)
;   if (data + nh_off > data_end)
       3:       r3 = r1
       4:       r3 += 14
       5:       if r3 > r2 goto +16 <LBB0_3>
;   h_proto = (__u32)eth->h_proto;
       6:       r2 = *(u8 *)(r1 + 12)
       7:       r1 = *(u8 *)(r1 + 13)
       8:       r1 <<= 8
       9:       r1 |= r2
      10:       *(u32 *)(r10 - 4) = r1
      11:       r2 = r10
      12:       r2 += -4
;   value = bpf_map_lookup_elem(&rxcnt, &h_proto);
      13:       r1 = 0 ll
      15:       call 1
      16:       r1 = r0
      17:       r0 = 2
;   if (value) {
      18:       if r1 == 0 goto +3 <LBB0_3>
;     *value += 1;
      19:       r2 = *(u64 *)(r1 + 0)
      20:       r2 += 1
      21:       *(u64 *)(r1 + 0) = r2

00000000000000b0 LBB0_3:
; }
      22:       exit
```

アタッチの前に気をつけること

BBSakuraNetworks

- 本番では **/proc/sys/net/core/bpf_jit_enable** を **1** にしておく
- フォワーディングするプログラムでは **/proc/sys/net/ipv4.ip_forward** や **/proc/sys/net/ipv6/conf/all/forwarding** を **1** にしておく
- **ethtool -L eth0 combined 2** のようにしてNICのキューの数を調整しておく
- ethtool -kで有効なオフロード機能を確認し、-Kでオフにしておく
- MapはLocked Memoryを使うため、**ulimit -l unlimited**やsystemdのunitファイルで**LimitMEMLOCK=infinity**しておく（デフォルトは64KB）
- オブジェクトをPinしたいときはbpffsをマウントしておく

  mount bpffs /sys/fs/bpf -t bpf

アタッチ/デタッチ

BBSakuraNetworks

- **iproute2の場合**
  - ip link set dev eth0 xdp obj example.o sec xdp_prog
  - ip link set dev eth0 xdp off
- **libbpfの場合**
  - bpf_prog_load()でロードして、bpf_set_link_xdp_fd()でアタッチ/デタッチ
  - 参考 https://github.com/xdp-project/xdp-tutorial/blob/master/basic01-xdp-pass/xdp_pass_user.c
- **bpftoolの場合**(5.4以上 https://patchwork.ozlabs.org/cover/1145981/
  - bpftool prog load ./example.o /sys/fs/bpf/prog
    - CONFIG_DEBUG_INFO_BTF=yなカーネルでないとlibbpfのエラー(22)が表示されるがロードはされる
  - bpftool prog list でidを確認
  - bpftool net attach xdp id 10 dev eth0
  - bpftool net detach xdp dev eth0

アタッチ/デタッチ

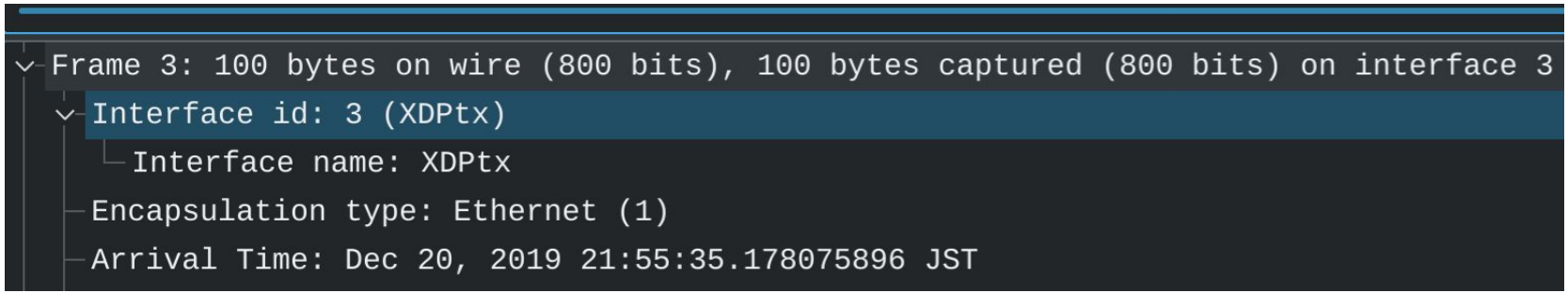
BBSakuraNetworks

- その他
  - BCC
    - 便利なマクロなどが定義されていて、少しプログラムを書きやすい
    - トレーシング方面で使われている
    - 新機能を追えてないことがあるので注意
  - iovisor/gobpf
    - Go言語で書かれたBCCのラッパー
    - cgoを使うことになるのと新機能を追えてないことがあるので注意
  - newtools/ebpf
    - pure Goで、libbpfと同様のことができるライブラリ
    - libbpf経由でなく、直接syscallを実行してロードしたりアタッチしたり Mapの読み書きをしたりできる
- **おすすめはbpftool、またはCなどからlibbpfを使う方法で、Cプレーンを Goで書く場合は、newtools/ebpf**

パケットキャプチャ

BBSakuraNetworks

- 問題が起きたときやデバッグのために XDPで処理されたパケットをキャプチャしたくなりますが、カーネルが処理する前に落としたり、返したりしてしまうので、**tcpdumpでは追えない**
- そこで、**xdpcap(https://github.com/cloudflare/xdpcap)** を使う
- 詳しくはリンク先を参照
- Wiresharkで見ると、プログラムのリターンコードもわかって便利



BBSakuraNetworks

コード解説（XDP編）

コード解説

BBSakuraNetworks

大変申し訳ありませんが、動くコードを実装できませんでした🙇🙇🙇

※開発はアルバイトで来ている東北学院大学3年生の早坂彪流さんにご協力いただきました。

ありがとうございました🙇🙇🙇

コード解説

BBSakuraNetworks

Endは動くのでEnd部分の解説をします

ソース:https://github.com/takehaya/srv6-gtp/blob/feature/t_m_GTP4_D/src/srv6.c

コード解説

BBSakuraNetworks

```
SEC("xdp_prog")
int srv6_handler(struct xdp_md *xdp)
{
    bpf_printk("srv6_handler start");
    void *data = (void *)(long)xdp->data;
    void *data_end = (void *)(long)xdp->data_end;
    struct ethhdr *eth = data;
    struct iphdr *iph = get_ipv4(xdp);
    struct ipv6hdr *v6h = get_ipv6(xdp);
    struct end_function *ef_table;
    struct transit_behavior *tb;
```

パケットの先頭と末尾のポインタ

ヘッダのパース

コード解説

BBSakuraNetworks

```
static inline struct ipv6hdr *get_ipv6(struct xdp_md *xdp)
{
    void *data = (void *)(long)xdp->data;
    void *data_end = (void *)(long)xdp->data_end;

    struct ipv6hdr *v6h = data + sizeof(struct ethhdr);

    if (data + sizeof(struct ethhdr) > data_end) {
        return NULL;
    }

    if (v6h + 1 > data_end) {
        return NULL;
    }

    return v6h;
};
```

パケットの先頭と末尾のポインタ

ヘッダのパース

コード解説

```
} else if (h_proto == bpf_htons(ETH_P_IPV6)) {
    // use nexthop and exec to function or decap or encap
    // checkSRv6
    if (v6h->nexthdr == NEXTHDR_ROUTING) {
        bpf_printk("match v6h->nexthdr == NEXTHDR_ROUTING)\n");
        ef_table = bpf_map_lookup_elem(&function_table, &(v6h->daddr));
        if (ef_table) {
            bpf_printk("ef_table check %u",ef_table);
            switch (ef_table->function) {
                case SEG6_LOCAL_ACTION_END:
                    bpf_printk("run action_end\n");
                    return action_end(xdp);
            }
        }
```

IPv6で、nexthdrがNEXTHDR_ROUTINGで、SEG6_LOCAL_ACTION_ENDのときにaction_end()を実行

コード解説(End)

BBSakuraNetworks

```
/* regular endpoint function */
static inline int action_end(struct xdp_md *xdp)
{
    void *data = (void *)(long)xdp->data;
    void *data_end = (void *)(long)xdp->data_end;

    struct ipv6_sr_hdr *srhdr = get_and_validate_srh(xdp);
    struct ipv6hdr *v6h = get_ipv6(xdp);

    if (!srhdr || !v6h) {
        return XDP_PASS;
    }
    if (advance_nextseg(srhdr, &v6h->daddr, xdp)) {
        return XDP_PASS;
    }

    return rewrite_nexthop(xdp);
}
```

SRヘッダのパースと内容チェック

segmentのパース

パケット書き換え及びREDIRECT

コード解説(REDIRECT辺り)

BBSakuraNetworks

```
    lookup_nexthop(xdp, &smac, &dmac, &ifindex);
    // bpf_printk("check_lookup_result\n");
    if (check_lookup_result(&dmac)) {
        bpf_printk("check_lookup_result match\n");
        set_src_dst_mac(data, &smac, &dmac);

        if (xdp->ingress_ifindex == ifindex) {
            bpf_printk("select TX\n");
            return XDP_TX;
        }
        bpf_printk("select bpf_redirect_map\n");
        // todo: fix bpf_redirect_map
        return bpf_redirect_map(&tx_port, ifindex, 0);
    }
    // bpf_printk("select XDP PASS\n");
    return XDP_PASS;
}
```

bpf_fib_lookupしてる

REDIRECT

コード解説(bpf_fib_lookup辺り)

BBSakuraNetworks

```
struct in6_addr *src = (struct in6_addr *)fib_params.ipv6_src;
struct in6_addr *dst = (struct in6_addr *)fib_params.ipv6_dst;
fib_params.family = AF_INET6;
fib_params.tos = 0;
fib_params.flowinfo = *(__be32 *)v6h & IPV6_FLOWINFO_MASK;
fib_params.l4_protocol = v6h->nexthdr;
fib_params.sport = 0;
fib_params.dport = 0;
fib_params.tot_len = bpf_ntohs(v6h->payload_len);
*src = v6h->saddr;
*dst = v6h->daddr;
fib_params.ifindex = xdp->ingress_ifindex;

rc = bpf_fib_lookup(xdp, &fib_params, sizeof(fib_params), BPF_FIB_LOOKUP_DIRECT | BPF_FIB_LOOKUP_OUTPUT);
```

これが動かない

コード解説

BBSakuraNetworks

プルリクエスト、アドバイス、お待ちしております🙇🙇🙇

XDPの開発でつらかったところ

BBSakuraNetworks

- **ロード時のエラーがわかりにくい**
  reference to missing symbolなどよくある
- **使っているカーネルのバージョンによってエラーの箇所が変わる**
- **bpf_trace_printkを足しただけでロードできなくなった**
  printkの行とは関係ない箇所でreference to missing symbolになる
- **netns内からbpffsをマウントできない**→開発しづらい
- **bpf_fib_lookupが動かないときに原因がわかりにくい**
  v6のlookup結局動かなかった。。。

BBSakuraNetworks

以下おまけ

最近のXDP関連の開発の様子

BBSakuraNetworks

- bpf: add bpf_ct_lookup_{tcp,udp}() helpers
  - conntrack tableの参照ができるようになりそう
- xdp: Introduce bulking for non-map XDP_REDIRECT Accepted
  - bpf_redirect_mapの方がbpf_redirectより速かったが、差がなくなる
- XDP in tx path RFC
- Introduce the BPF dispatcher Accepted
- xdp_flow: Flow offload to XDP RFC
- virtio_net XDP offload RFC
- Introduce XDP to ena
  - EC2のElastic Network AdapterのXDP対応
- sfc: Add XDP support
  - SolarflareのNICのXDP対応

XDPを使うときによく見る資料

BBSakuraNetworks

- Cilium BPF and XDP Reference Guide
- Suricata eBPF and XDP
- Linux Documantation/bpf
- Linux samples/bpf
  - 見るときは使うカーネルのバージョンに合わせて見る
- XDP Hands-On Tutorial
- BPF Features by Linux Kernel Version
  - この機能どのバージョンから使える？というときに見る
- Linux Observability with BPF(本) (sysdigのサイトに情報登録すると無料でダウンロードできる)
- yunazuno.log
  - 日本語で検索すると大抵このブログがヒットする
- @IT Berkeley Packet Filter（BPF）入門
- あとはドライバなどのソースコードを見る。。。

EOF