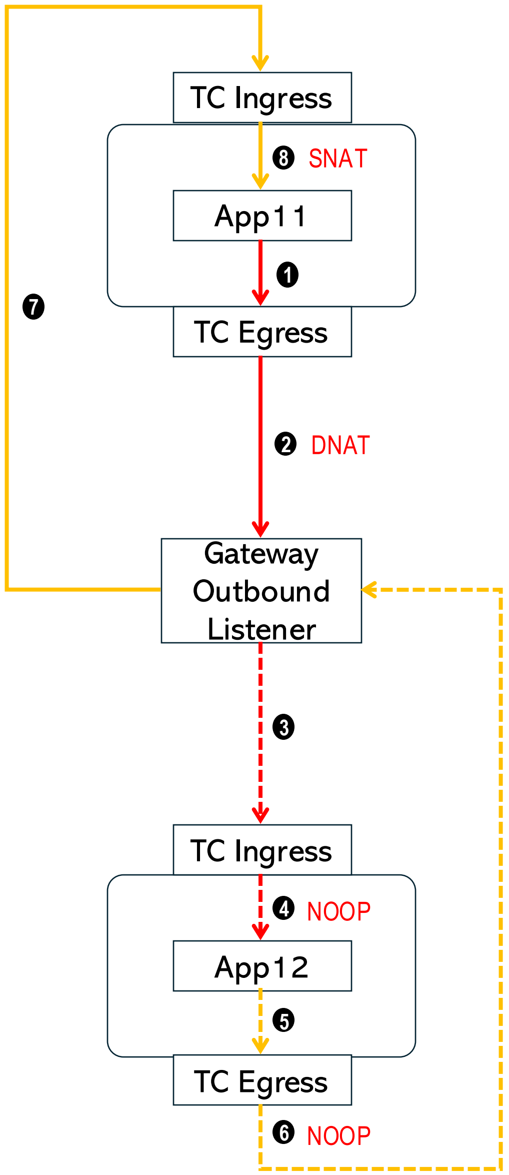
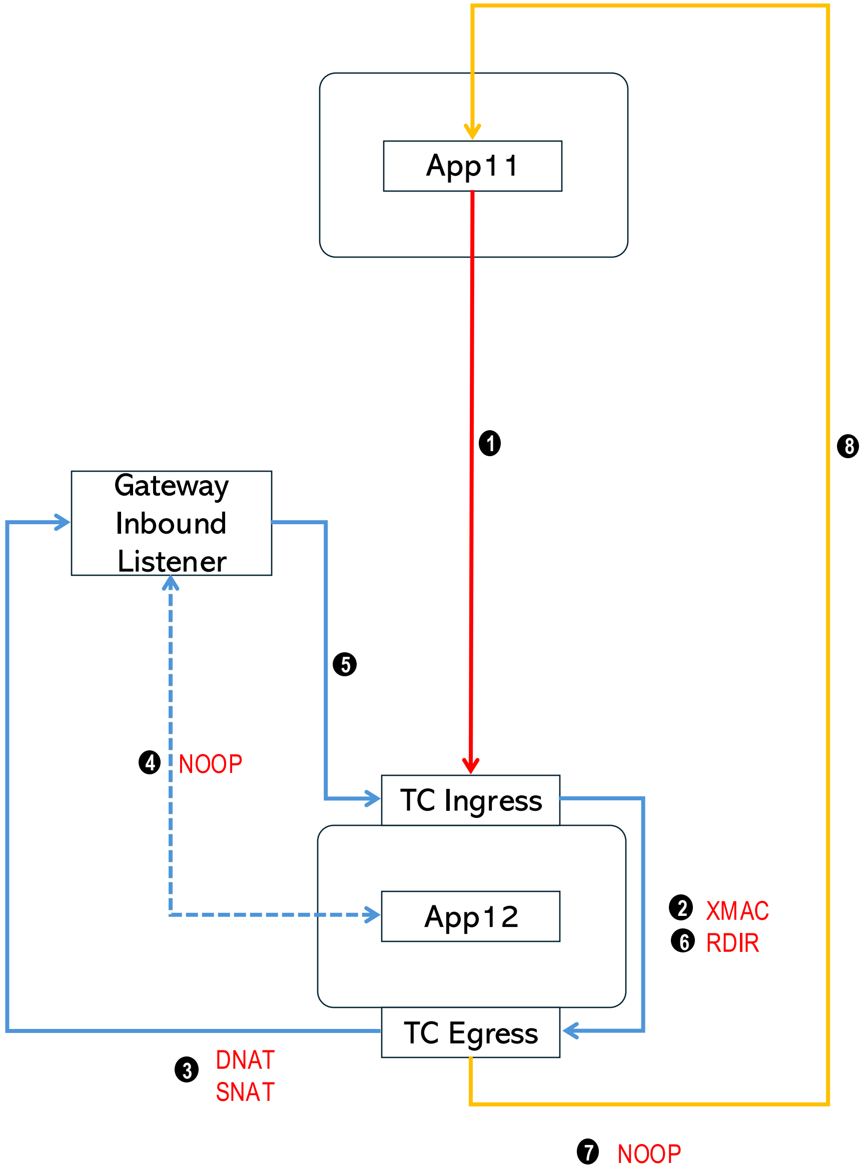
一 基于信任 Gateway Outbound 的流量模型

App进/出的流量只会被拦截给

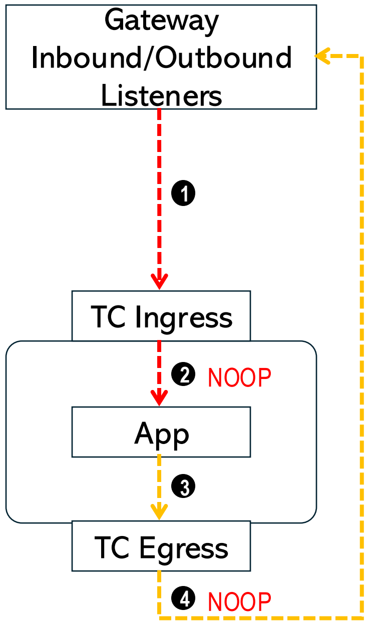
同 Node 上的 FGW Listeners



Mesh内互访

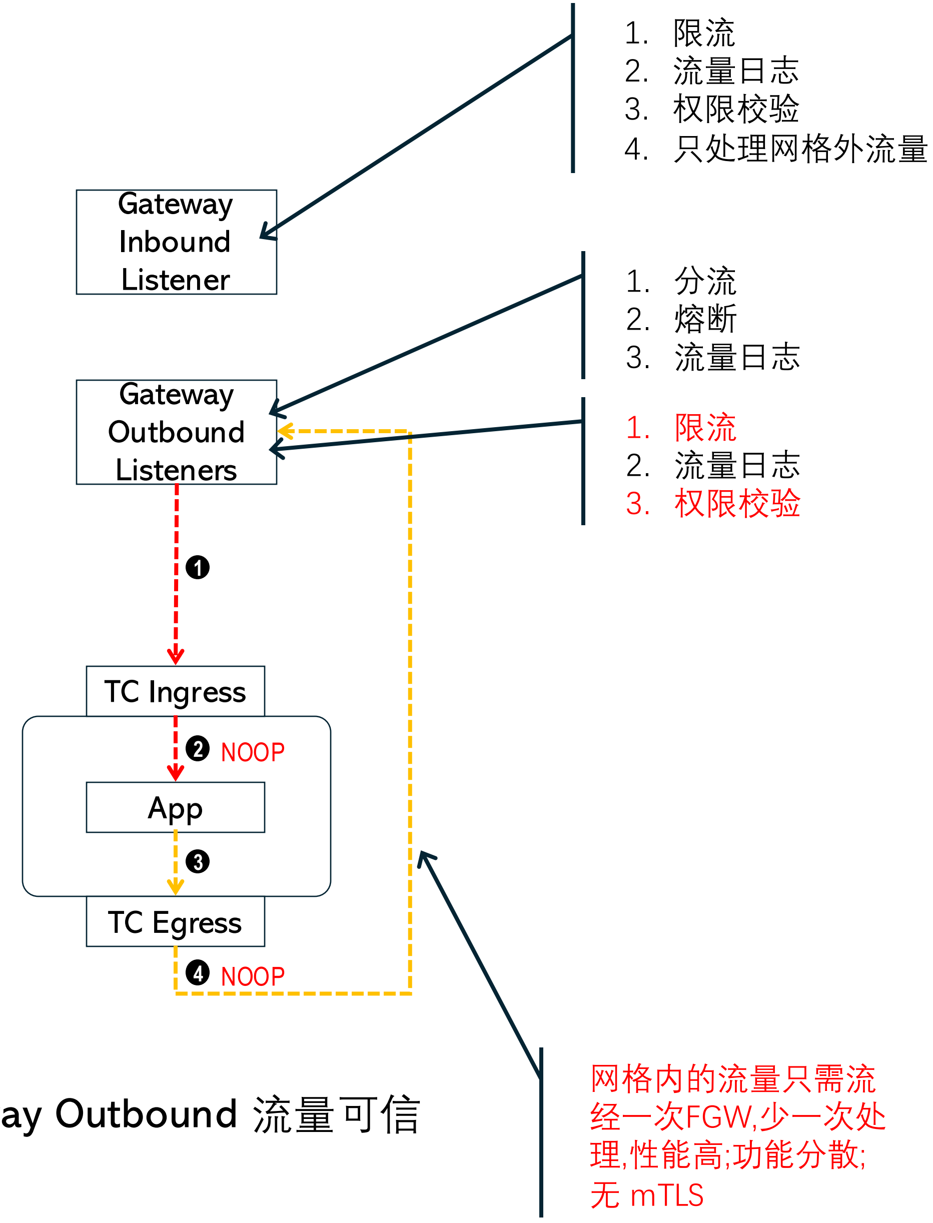


Mesh外访问Mesh内



Gateway访问Mesh内

二 Gateway Outbound 信任与否的流量模型比较



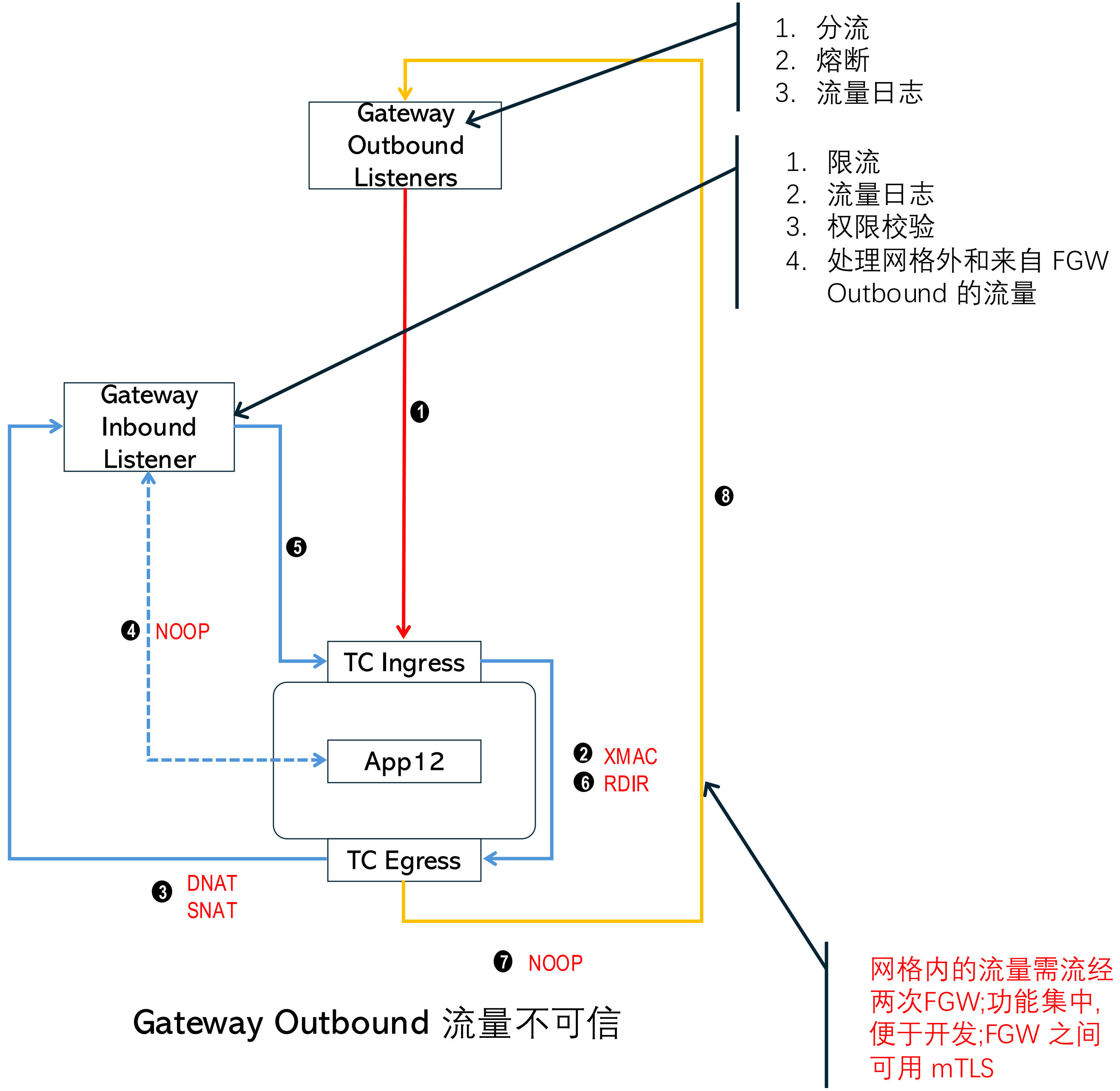
1. 限流
2. 流量日志
3. 权限校验
4. 只处理网格外流量

1. 分流
2. 熔断
3. 流量日志

1. 限流
2. 流量日志
3. 权限校验

网格内的流量只需流经一次FGW,少一次处理,性能高;功能分散;无 mTLS

Gateway Outbound 流量可信



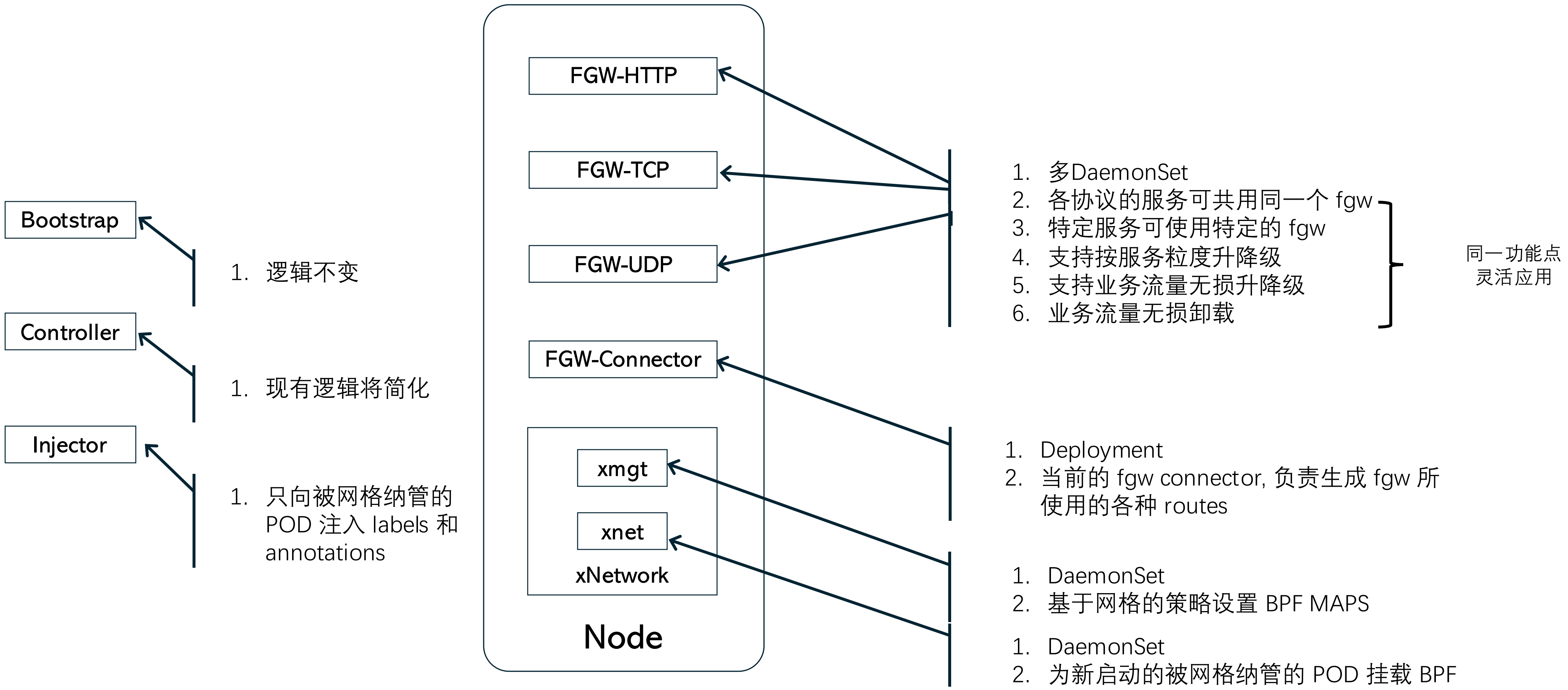
1. 分流
2. 熔断
3. 流量日志

1. 限流
2. 流量日志
3. 权限校验
4. 处理网格外和来自 FGW Outbound 的流量

网格内的流量需流经两次FGW;功能集中,便于开发;FGW 之间可用 mTLS

Gateway Outbound 流量不可信

三 Mesh 各控制层服务部署示意图



1. 逻辑不变

1. 现有逻辑将简化

1. 只向被网格纳管的 POD 注入 labels 和 annotations

1. 多DaemonSet
2. 各协议的服务可共用同一个 fgw
3. 特定服务可使用特定的 fgw
4. 支持按服务粒度升降级
5. 支持业务流量无损升降级
6. 业务流量无损卸载

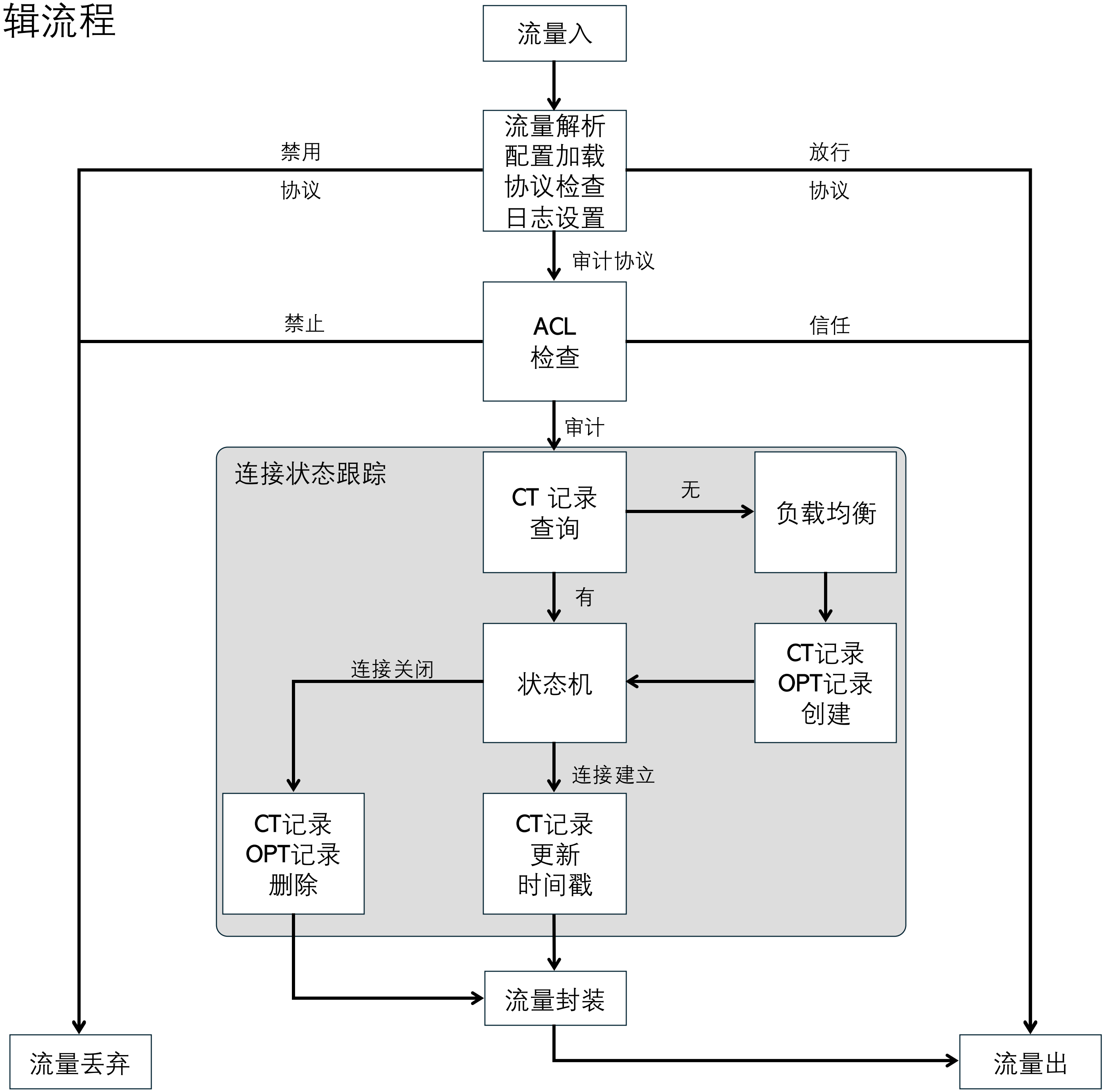
同一功能点
灵活应用

1. Deployment
2. 当前的 fgw connector, 负责生成 fgw 所使用的各种 routes

1. DaemonSet
2. 基于网格的策略设置 BPF MAPS

1. DaemonSet
2. 为新启动的被网格纳管的 POD 挂载 BPF

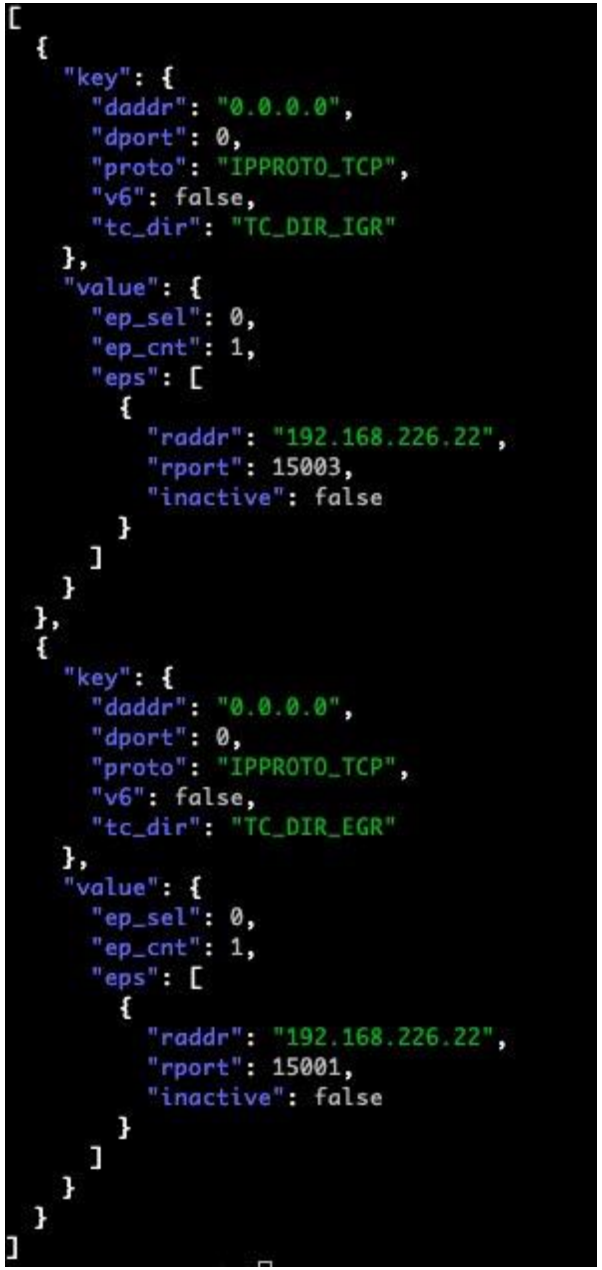
四 eBPF流量处理逻辑流程



TC 的 Ingress 和 Egress 相同

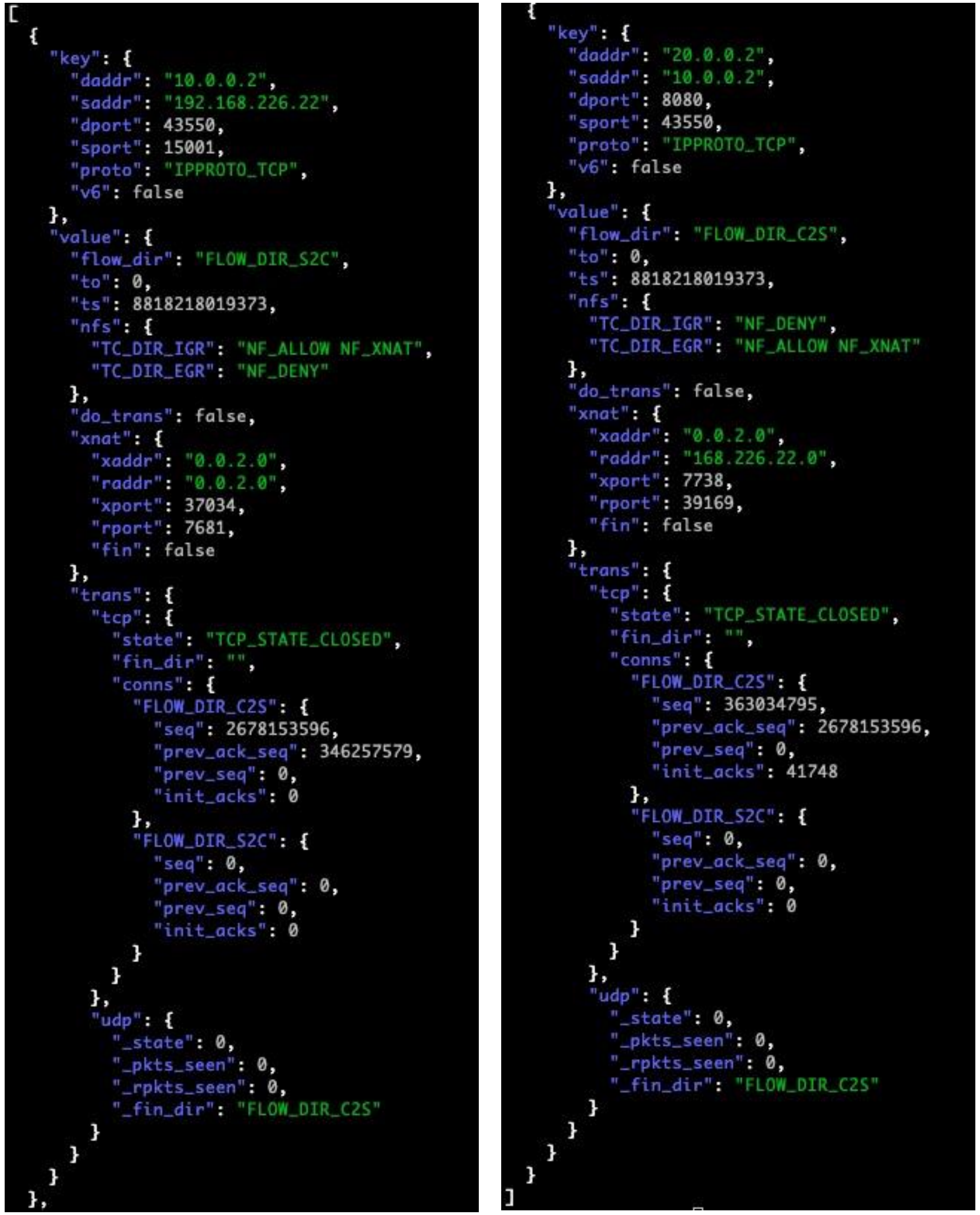
的处理流程, 处理的方向相反

五 主要的 eBPF maps



```
[
  {
    "key": {
      "daddr": "0.0.0.0",
      "dport": 0,
      "proto": "IPPROTO_TCP",
      "v6": false,
      "tc_dir": "TC_DIR_IGR"
    },
    "value": {
      "ep_sel": 0,
      "ep_cnt": 1,
      "eps": [
        {
          "raddr": "192.168.226.22",
          "rport": 15003,
          "inactive": false
        }
      ]
    }
  },
  {
    "key": {
      "daddr": "0.0.0.0",
      "dport": 0,
      "proto": "IPPROTO_TCP",
      "v6": false,
      "tc_dir": "TC_DIR_EGR"
    },
    "value": {
      "ep_sel": 0,
      "ep_cnt": 1,
      "eps": [
        {
          "raddr": "192.168.226.22",
          "rport": 15001,
          "inactive": false
        }
      ]
    }
  }
]
```

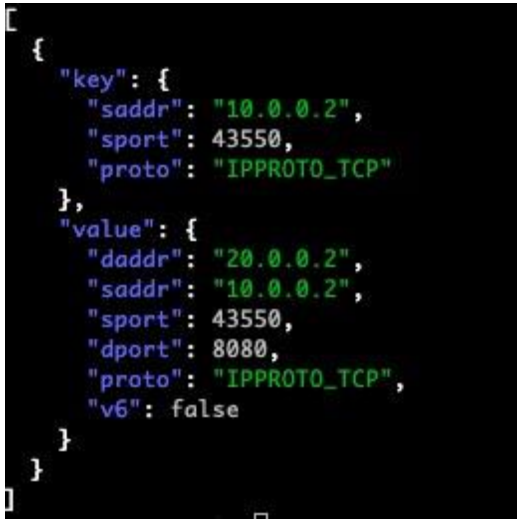
fsm_xnat



```
[
  {
    "key": {
      "daddr": "10.0.0.2",
      "saddr": "192.168.226.22",
      "dport": 43550,
      "sport": 15001,
      "proto": "IPPROTO_TCP",
      "v6": false
    },
    "value": {
      "flow_dir": "FLOW_DIR_S2C",
      "to": 0,
      "ts": 8818218019373,
      "nfs": {
        "TC_DIR_IGR": "NF_ALLOW NF_XNAT",
        "TC_DIR_EGR": "NF_DENY"
      },
      "do_trans": false,
      "xnat": {
        "xaddr": "0.0.2.0",
        "raddr": "0.0.2.0",
        "xport": 37034,
        "rport": 7681,
        "fin": false
      },
      "trans": {
        "tcp": {
          "state": "TCP_STATE_CLOSED",
          "fin_dir": "",
          "conns": {
            "FLOW_DIR_C2S": {
              "seq": 2678153596,
              "prev_ack_seq": 346257579,
              "prev_seq": 0,
              "init_acks": 0
            },
            "FLOW_DIR_S2C": {
              "seq": 0,
              "prev_ack_seq": 0,
              "prev_seq": 0,
              "init_acks": 0
            }
          }
        },
        "udp": {
          "_state": 0,
          "_pkts_seen": 0,
          "_rpkts_seen": 0,
          "_fin_dir": "FLOW_DIR_C2S"
        }
      }
    }
  },
```

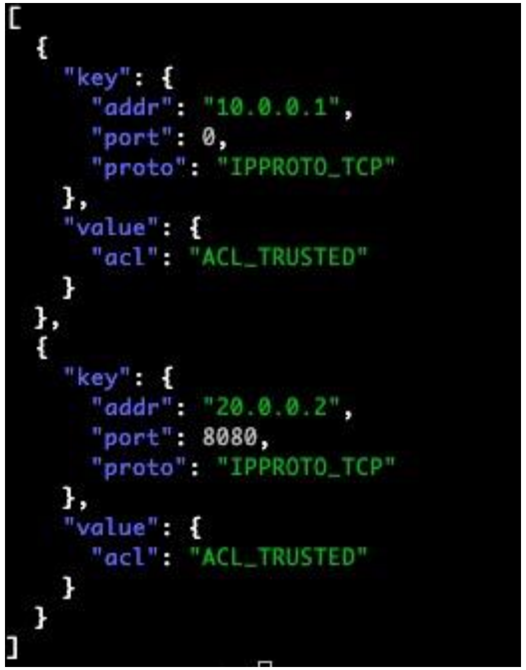
```
  {
    "key": {
      "daddr": "20.0.0.2",
      "saddr": "10.0.0.2",
      "dport": 8080,
      "sport": 43550,
      "proto": "IPPROTO_TCP",
      "v6": false
    },
    "value": {
      "flow_dir": "FLOW_DIR_C2S",
      "to": 0,
      "ts": 8818218019373,
      "nfs": {
        "TC_DIR_IGR": "NF_DENY",
        "TC_DIR_EGR": "NF_ALLOW NF_XNAT"
      },
      "do_trans": false,
      "xnat": {
        "xaddr": "0.0.2.0",
        "raddr": "168.226.22.0",
        "xport": 7738,
        "rport": 39169,
        "fin": false
      },
      "trans": {
        "tcp": {
          "state": "TCP_STATE_CLOSED",
          "fin_dir": "",
          "conns": {
            "FLOW_DIR_C2S": {
              "seq": 363034795,
              "prev_ack_seq": 2678153596,
              "prev_seq": 0,
              "init_acks": 41748
            },
            "FLOW_DIR_S2C": {
              "seq": 0,
              "prev_ack_seq": 0,
              "prev_seq": 0,
              "init_acks": 0
            }
          }
        },
        "udp": {
          "_state": 0,
          "_pkts_seen": 0,
          "_rpkts_seen": 0,
          "_fin_dir": "FLOW_DIR_C2S"
        }
      }
    }
  }
]
```

fsm_xflow



```
[
  {
    "key": {
      "saddr": "10.0.0.2",
      "sport": 43550,
      "proto": "IPPROTO_TCP"
    },
    "value": {
      "daddr": "20.0.0.2",
      "saddr": "10.0.0.2",
      "sport": 43550,
      "dport": 8080,
      "proto": "IPPROTO_TCP",
      "v6": false
    }
  }
]
```

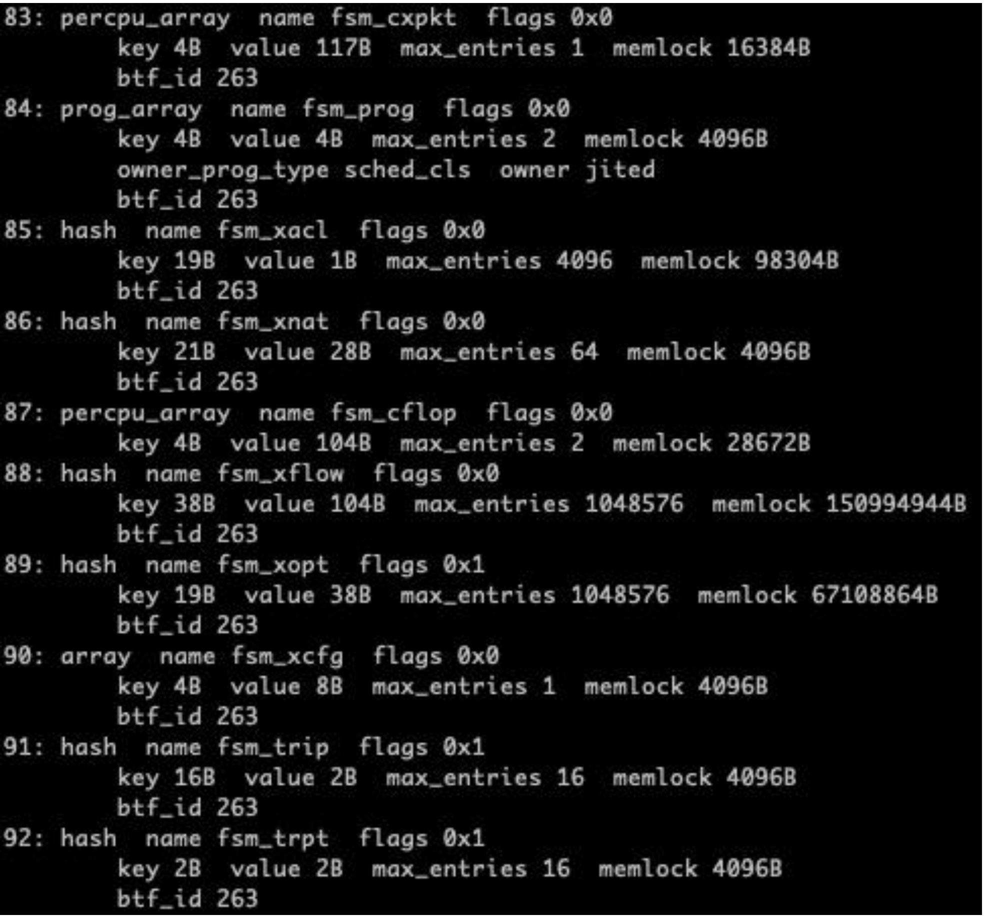
fsm_xopt



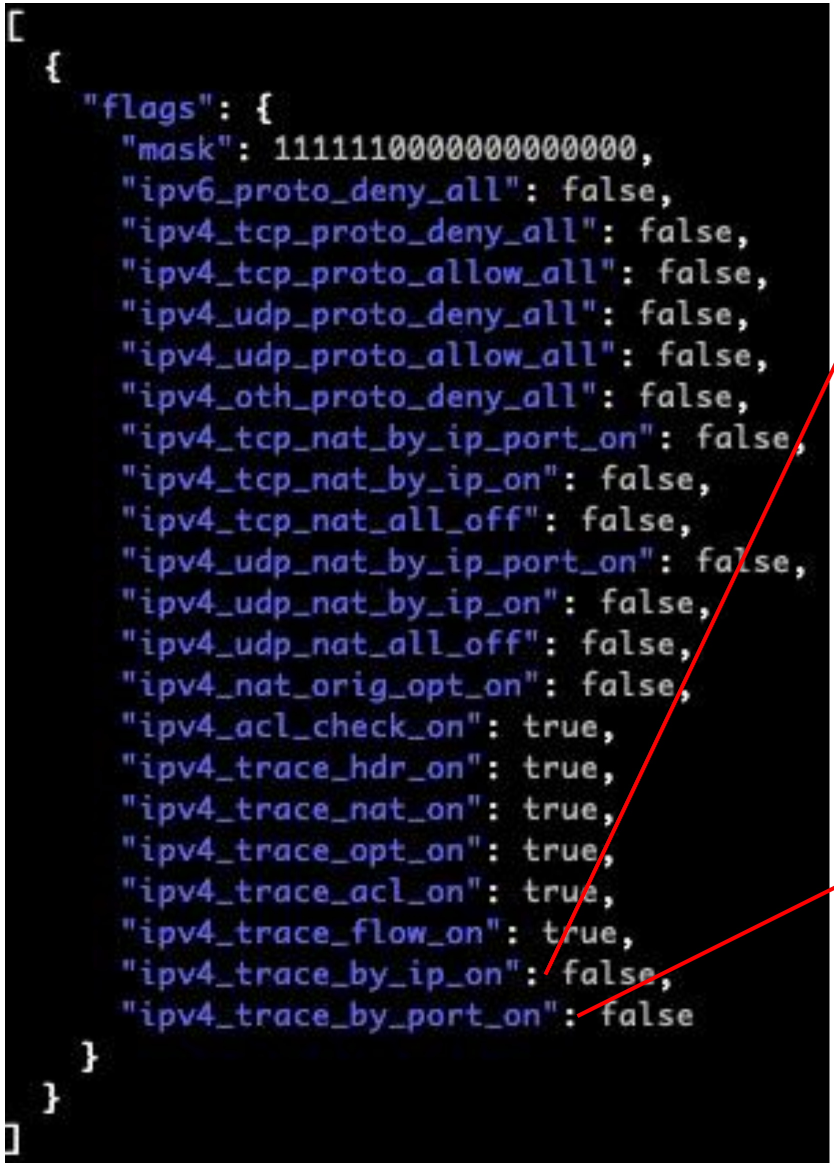
```
[
  {
    "key": {
      "addr": "10.0.0.1",
      "port": 0,
      "proto": "IPPROTO_TCP"
    },
    "value": {
      "acl": "ACL_TRUSTED"
    }
  },
  {
    "key": {
      "addr": "20.0.0.2",
      "port": 8080,
      "proto": "IPPROTO_TCP"
    },
    "value": {
      "acl": "ACL_TRUSTED"
    }
  }
]
```

fsm_xacl

五　主要的 eBPF maps

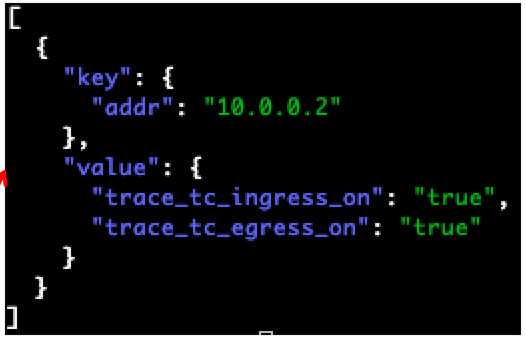


```
83: percpu_array  name fsm_cxpkt  flags 0x0
        key 4B  value 117B  max_entries 1  memlock 16384B
        btf_id 263
84: prog_array  name fsm_prog  flags 0x0
        key 4B  value 4B  max_entries 2  memlock 4096B
        owner_prog_type sched_cls  owner jited
        btf_id 263
85: hash  name fsm_xacl  flags 0x0
        key 19B  value 1B  max_entries 4096  memlock 98304B
        btf_id 263
86: hash  name fsm_xnat  flags 0x0
        key 21B  value 28B  max_entries 64  memlock 4096B
        btf_id 263
87: percpu_array  name fsm_cflop  flags 0x0
        key 4B  value 104B  max_entries 2  memlock 28672B
88: hash  name fsm_xflow  flags 0x0
        key 38B  value 104B  max_entries 1048576  memlock 150994944B
        btf_id 263
89: hash  name fsm_xopt  flags 0x1
        key 19B  value 38B  max_entries 1048576  memlock 67108864B
        btf_id 263
90: array  name fsm_xcfg  flags 0x0
        key 4B  value 8B  max_entries 1  memlock 4096B
        btf_id 263
91: hash  name fsm_trip  flags 0x1
        key 16B  value 2B  max_entries 16  memlock 4096B
        btf_id 263
92: hash  name fsm_trpt  flags 0x1
        key 2B  value 2B  max_entries 16  memlock 4096B
        btf_id 263
```



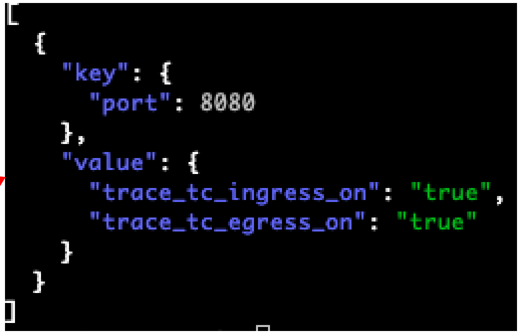
```
[
  {
    "flags": {
      "mask": 1111110000000000000,
      "ipv6_proto_deny_all": false,
      "ipv4_tcp_proto_deny_all": false,
      "ipv4_tcp_proto_allow_all": false,
      "ipv4_udp_proto_deny_all": false,
      "ipv4_udp_proto_allow_all": false,
      "ipv4_oth_proto_deny_all": false,
      "ipv4_tcp_nat_by_ip_port_on": false,
      "ipv4_tcp_nat_by_ip_on": false,
      "ipv4_tcp_nat_all_off": false,
      "ipv4_udp_nat_by_ip_port_on": false,
      "ipv4_udp_nat_by_ip_on": false,
      "ipv4_udp_nat_all_off": false,
      "ipv4_nat_orig_opt_on": false,
      "ipv4_acl_check_on": true,
      "ipv4_trace_hdr_on": true,
      "ipv4_trace_nat_on": true,
      "ipv4_trace_opt_on": true,
      "ipv4_trace_acl_on": true,
      "ipv4_trace_flow_on": true,
      "ipv4_trace_by_ip_on": false,
      "ipv4_trace_by_port_on": false
    }
  }
]
```

fsm_xcfg



```
[
  {
    "key": {
      "addr": "10.0.0.2"
    },
    "value": {
      "trace_tc_ingress_on": "true",
      "trace_tc_egress_on": "true"
    }
  }
]
```

fsm_trip



```
[
  {
    "key": {
      "port": 8080
    },
    "value": {
      "trace_tc_ingress_on": "true",
      "trace_tc_egress_on": "true"
    }
  }
]
```

fsm_trpt

六 访问控制策略

1. 优先按 IP+PORT查询, 其次是按 IP 查询
2. FGW 的 inbound 和 outbound 端口也要设置 ACL,策略为 AUDIT, 以便能做反向的 DNAT/SNAT

七 后续改进点

1. TC Ingress 前增加 XDP, 处理 ACL, 放行的流量不进入内核网络层, 直接转发出去
2. App 到 FGW的 Inbound 和 Outbound Listeners 的流量都是同节点流量, 适合加速处理; 需要较高的内核版本, 不适用一些国产 OS