SDN Programming using Algorithmic Policies

Andreas Voellmy

Michael F. Nowlan, Junchang Wang, Lewen Yu,
Bryan Ford, Paul Hudak, Y. Richard Yang

Yale University

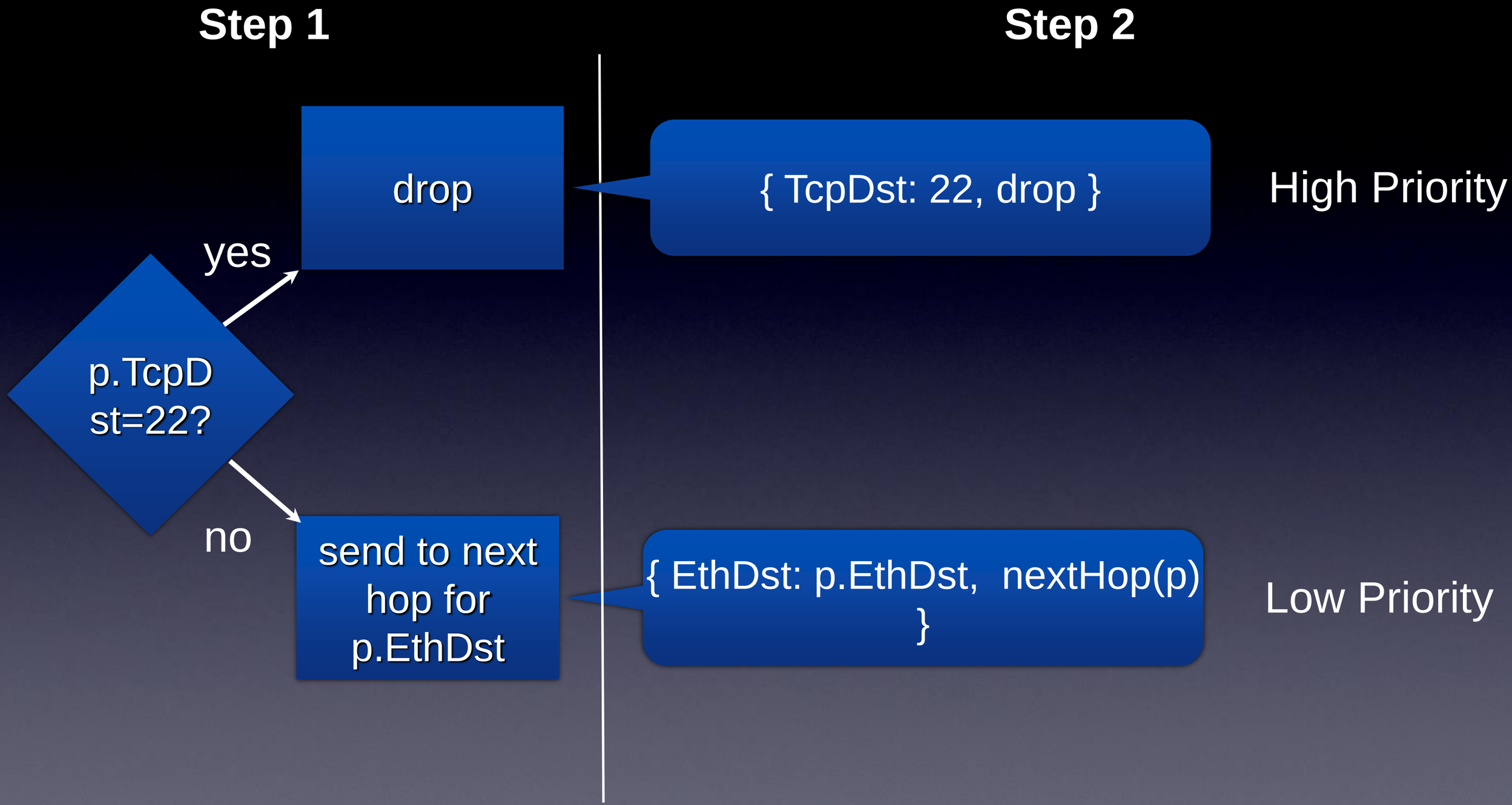
A Key Source of SDN Complexity

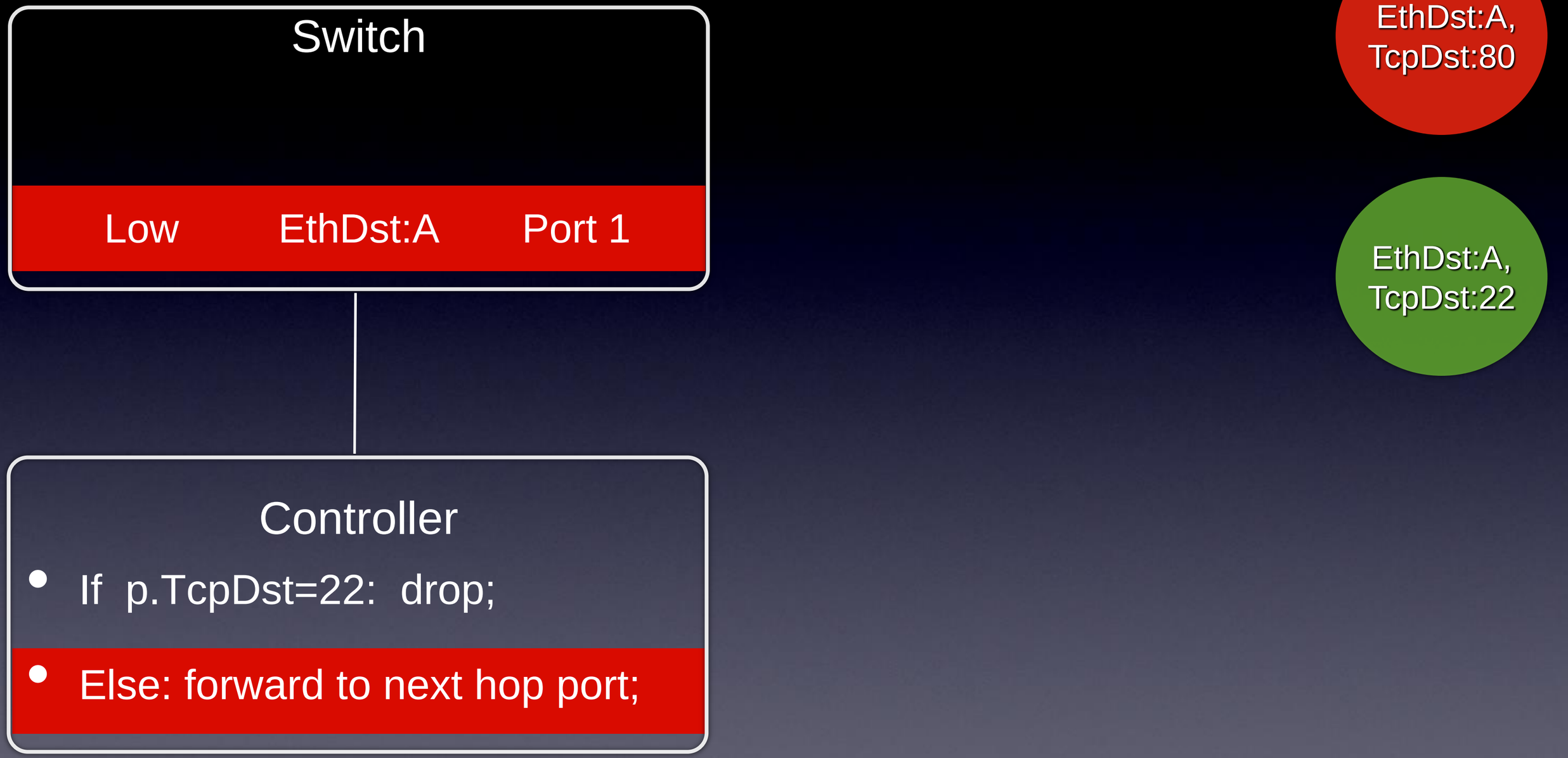
- onPacketIn(p):

Step 1 examine p and decide what to do with p.

Step 2 try to construct OF rules that mimic (1) so that similar packets are processed at switches.

Source of errors


Step 1
Step 2
drop
{ TcpDst: 22, drop }
High Priority
yes
p.TcpDst=22?
no
send to next hop for p.EthDst
{ EthDst: p.EthDst, nextHop(p) }
Low Priority


Switch
Low EthDst:A Port 1
Controller
If p.TcpDst=22: drop;
Else: forward to next hop port;
EthDst:A, TcpDst:80
EthDst:A, TcpDst:22

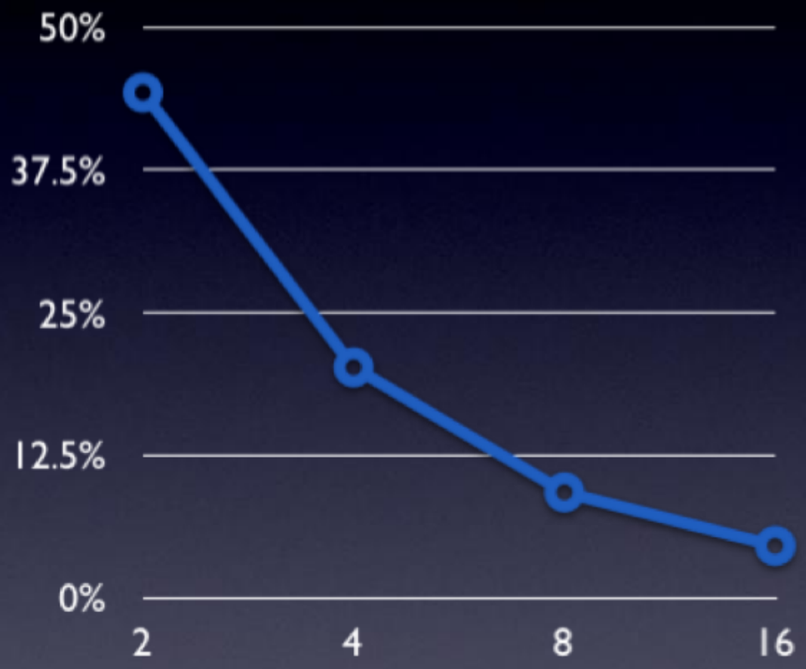
A Trivial Solution

- onPacketIn(p):

Step 1 examine p and decide what to do with p.

Step 2’ use “exact match” rules, i.e. match on ALL attributes can be observed.

Learning switch controller using exact matches



Flow table miss rate

Average packets per TCP flow

Control Plane

Step 1. Make Decisions

Step 2. Generate Rules

OF Controller Library

Data Plane

OF Switches

Maple

Control Plane

Step 1. Make Decisions

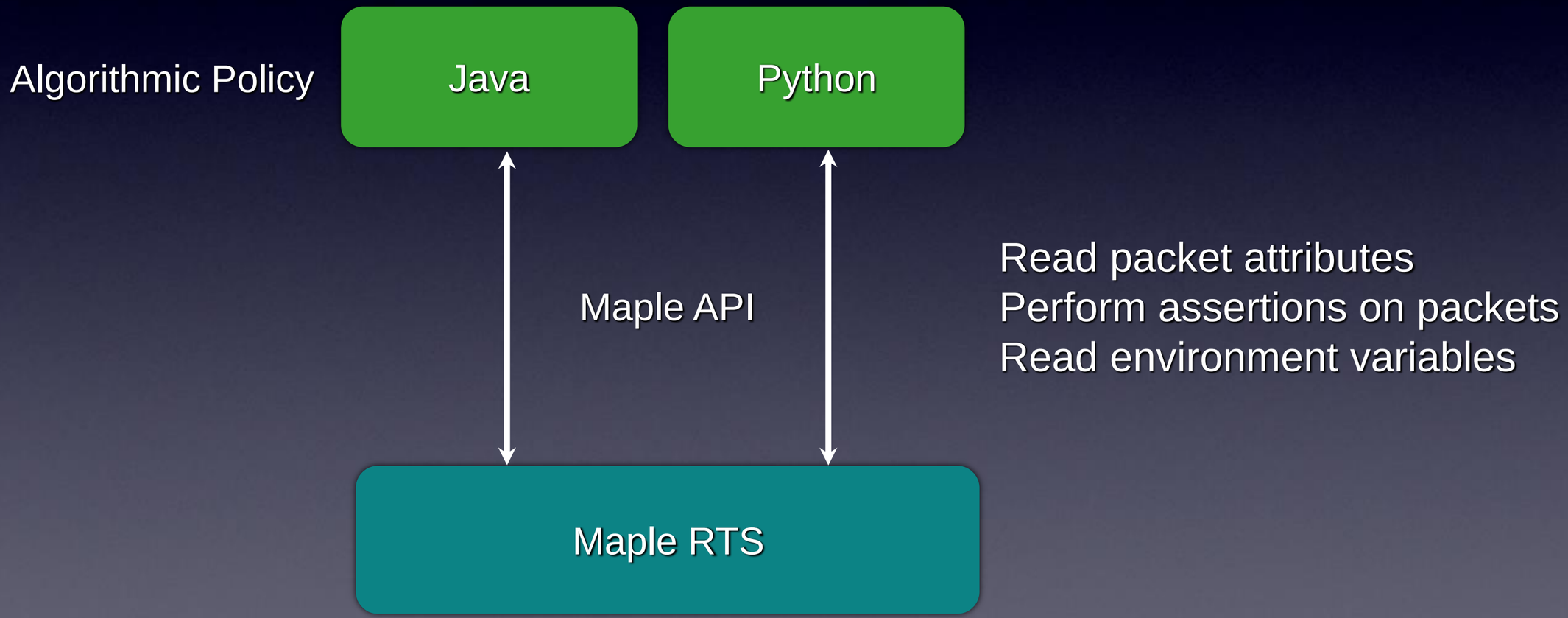
Step 2. Translate Rules

OF Controller Library

Data Plane

OF Switches

Algorithmic Policies in Familiar Languages



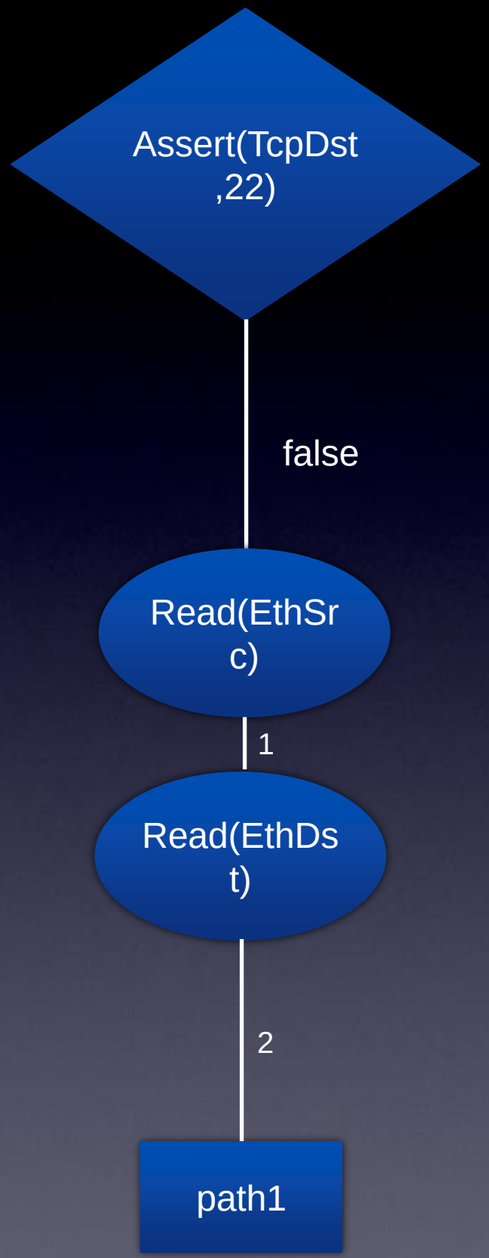
Maple in Java

```
Route f(Packet p) {
  if (p.tcpDstIs(22))
    return drop();
  else {
    Location sloc = location(p.ethSrc());
    Location dloc = location(p.ethDst());
    Path path = minPath(links(), sloc,dloc);
    return unicast(sloc,dloc,path);
  }
}
```

No rules specified

Policy

```
Route f(Packet p) {

 if (p.tcpDstIs(22))


  return drop();


 else {

  Location sloc =
   location(p.ethSrc());

  Location dloc =
   location(p.ethDst());

  Path pth     =
   minPath(links()
     ,sloc,dloc);

  return
   unicast(sloc,dloc,pth);
 }
}
```

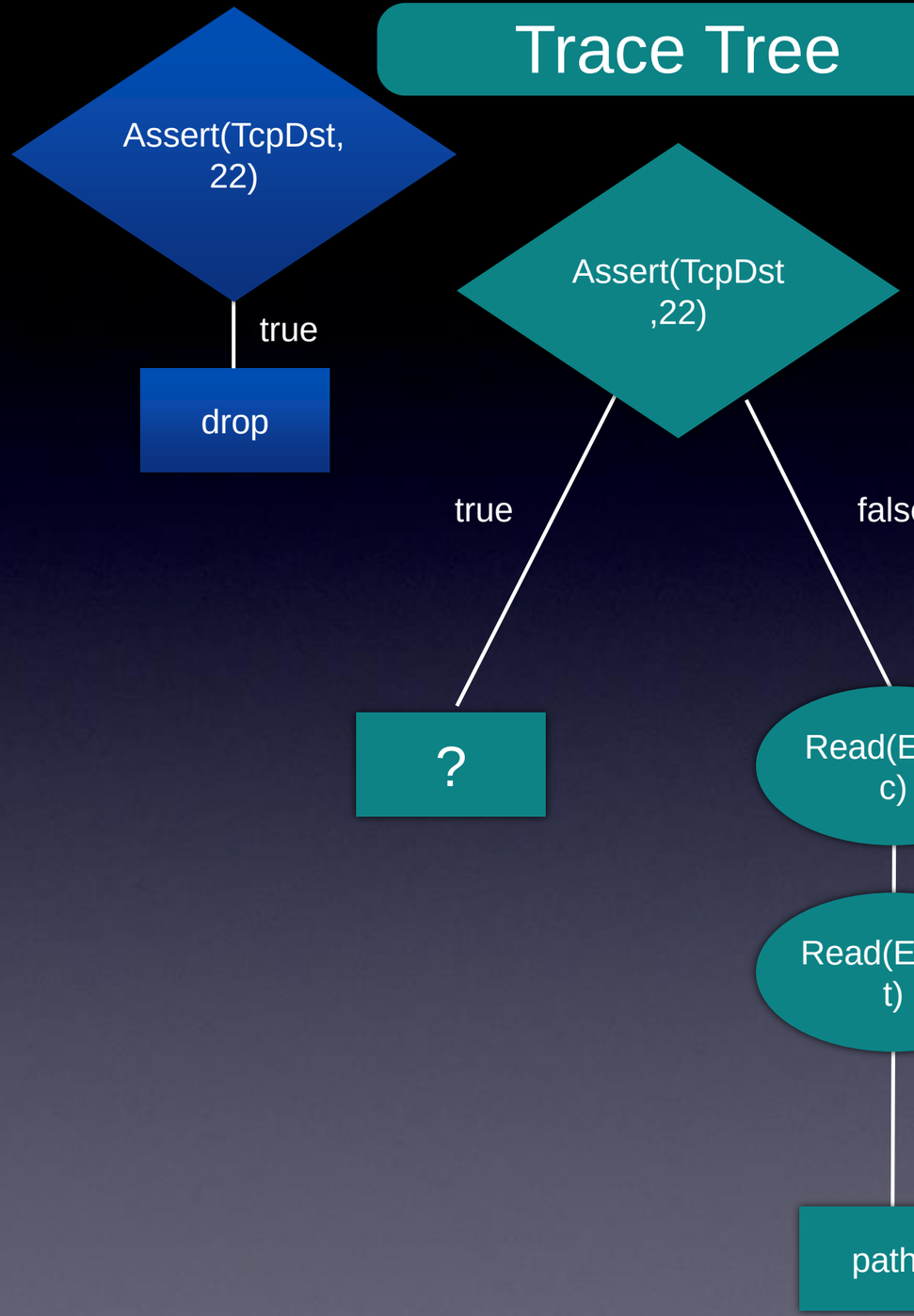


EthDest:A
,
TcpDst:80

Policy

```
Route f(Packet p) {

  if (p.tcpDstIs(22))


    return drop();


  else {

    Location sloc =
      location(p.ethSrc());

    Location dloc =
      location(p.ethDst());

    Path pth     =
      djkstrPath(sloc,dloc);

    return
      unicast(sloc,dloc,pth);
  }
}
```

Trace Tree



Generated Rules

EthSrc:A, TcpDst:22

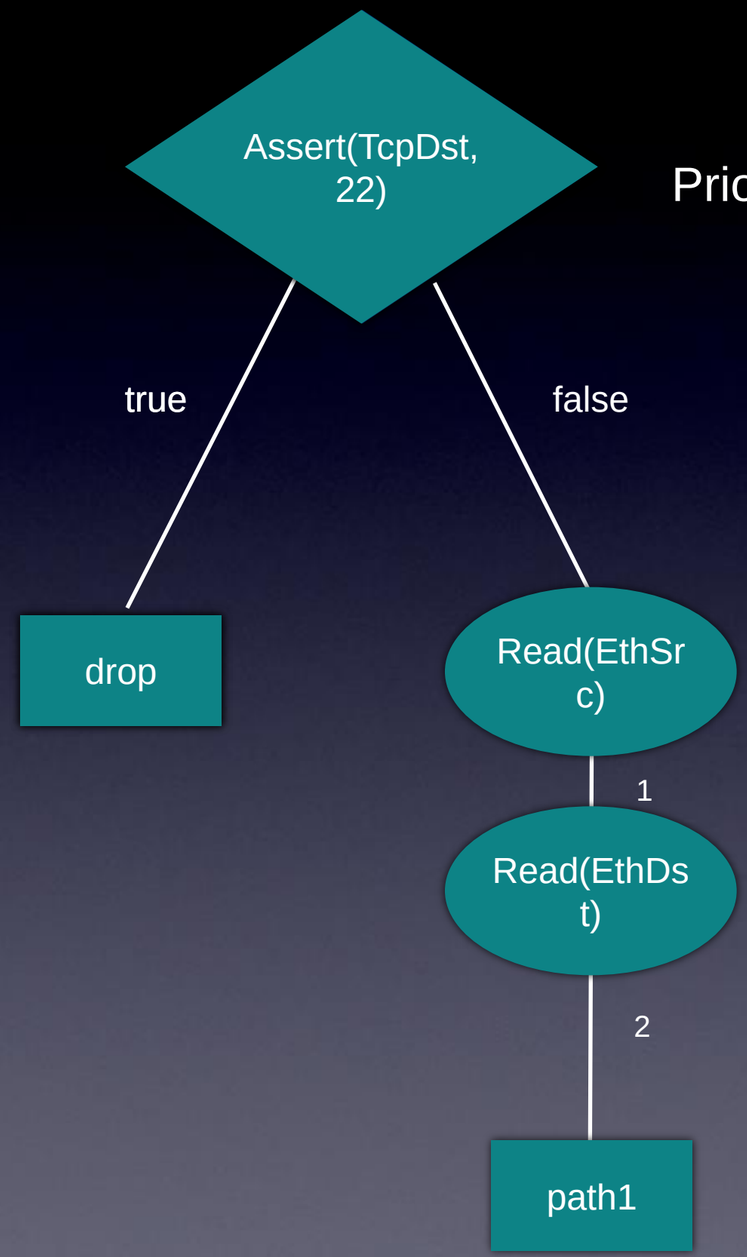
Prio:1, TcpDst:22, ?

Prio:0, EthSrc:B, EthDst:A, p1

Policy

```
Route f(Packet p) {

  if (p.tcpDstIs(22))


    return drop();


  else {

    Location sloc =
      location(p.ethSrc());

    Location dloc =
      location(p.ethDst());

    Path pth     =
      djkstrPath(sloc,dloc);

    return
      unicast(sloc,dloc,pth);
  }
}
```

Trace Tree



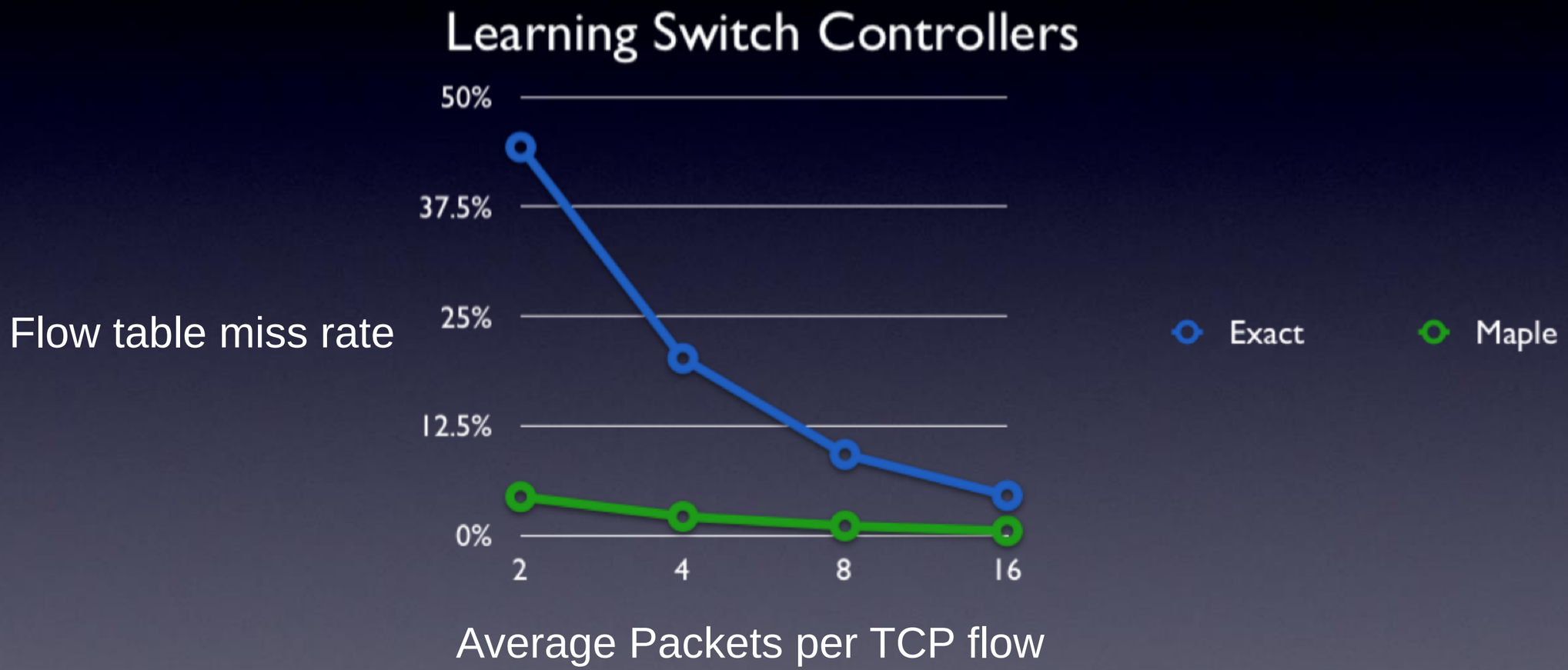
Generated Rules

Prio:2, TcpDst:22, drop

Prio:1, TcpDst:22, ?

Prio:0, EthSrc:B, EthDst:A, p1

Maple Generates Efficient Rules



Maple: SDN using Algorithmic Policies

- Programmer writes algorithmic policies in a familiar language.
- Maple automatically generates optimized rules implementing those policies.
- Go to www.maplecontroller.com for this presentation.
- Thank you.