lecture 19:
debugging SDN
5590: software defined networking

anduo wang, Temple University

TTLMAN 401B, R 17:30-20:00

Where is the Debugger for my Software-Defined Network?

Teemu Koponen., et al. “Network Virtualization in Multi-tenant Datacenters”

debugging networks is hard

reconstruct the complex and distributed state of the network

- `ping, traceroute`
- end-hosts: `tcpdump`
- switches and routers: `netflow`

network state is constantly changing

- (by) L2 learning, L3 routing …
- 6k RFCs …

SDN presents an opportunity

protocols → controller program

- rethink debugging from the development of control programs, all the way to their deployment

from gdb to ndb

gdb

- `breakpoint`: pause execution
- `backtrace`: show history of function calls leading to the breakpoint

from gdb to ndb

from gdb to ndb

ndb

- `packet breakpoint`:
  an unforwarded packet
  - packet filter at one or more switches
- backtrace
  - returns forwarding details (history) along the path

from gdb to ndb

ndb

- `packet breakpoint`:
  an unforwarded packet
  - packet filter at one or more switches
- backtrace
  - returns forwarding details (history) along the path

```
packet [dl_src: 0x123, ...]:
    switch 1: { inport: p0, outports: [p1]
                mods: [dl_dst -> 0x345]
                matched flow: 23 [...]
                matched table version: 3 }
    switch 2: { inport: p0, outports: [p2]
                mods: []
                matched flow: 11 [...]
                matched table version: 7 }
    ...
    switch N: { inport: p0
                table miss
                matched table version: 8 }
```

ndb: SDN debugger

ndb

- (what) primitives: breakpoint and backtrace
- (how)
  - modify forwarding state and logs packet digests
  - rebuild sequence of forwarding actions leading to an errant packet

(what) using ndb

bug: packet matching no forwarding rule in the middle of the network

- breakpoint
  - packets with no flow entries on every switch
- backtrace
  - forwarding history of a packet until its last hop before it failed to match any flow entires

(what) using ndb

bug: why a client and server could not communicate

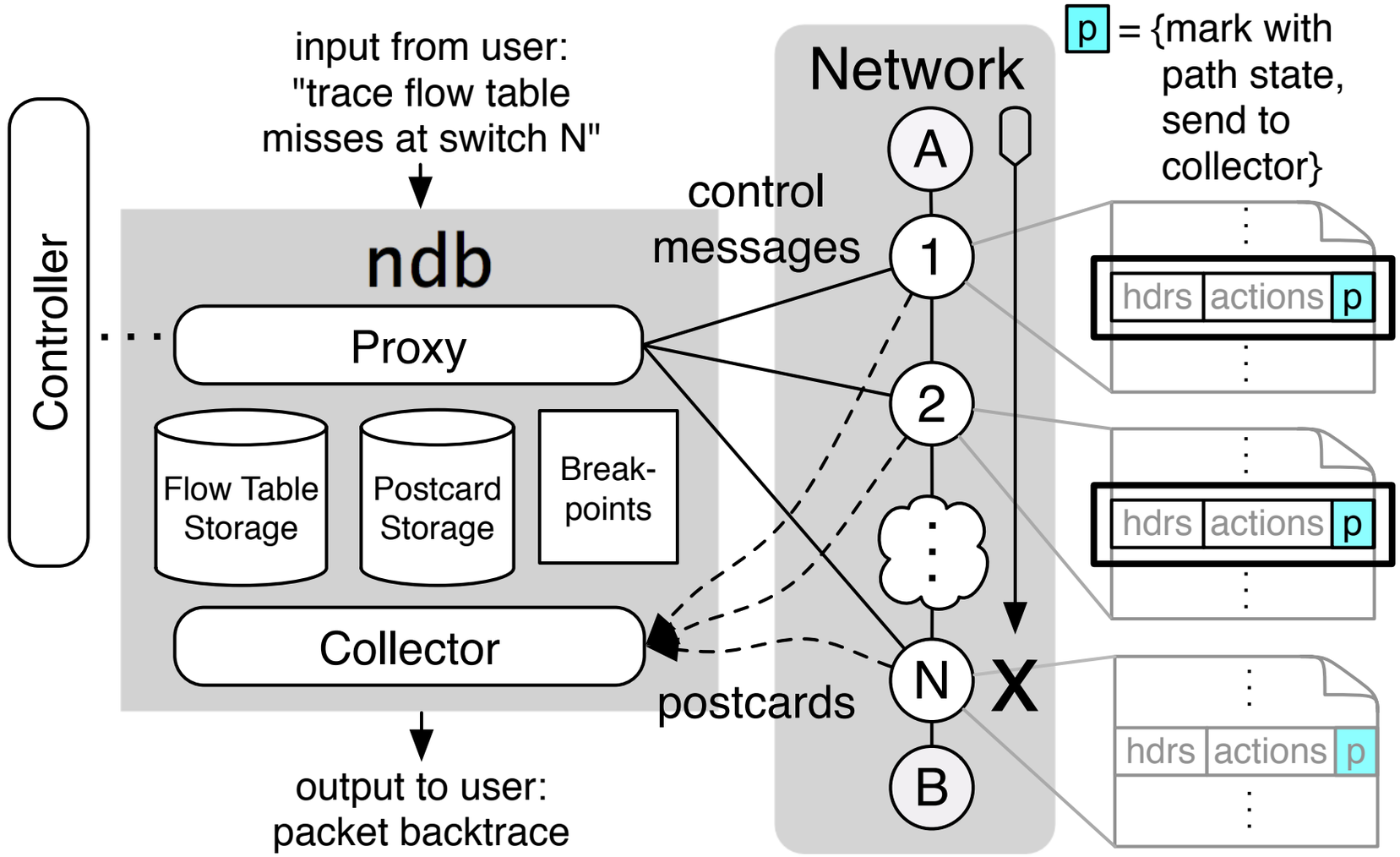
- breakpoint
  - all packets coming to the server
- backtrace
  - packets reaching the server along the right path, but corrupted

(how) building ndb

rebuild the path taken by a world traveler

- by stamps in his passport
- *alternative: from the postcards (sent at each hop) he sent home*

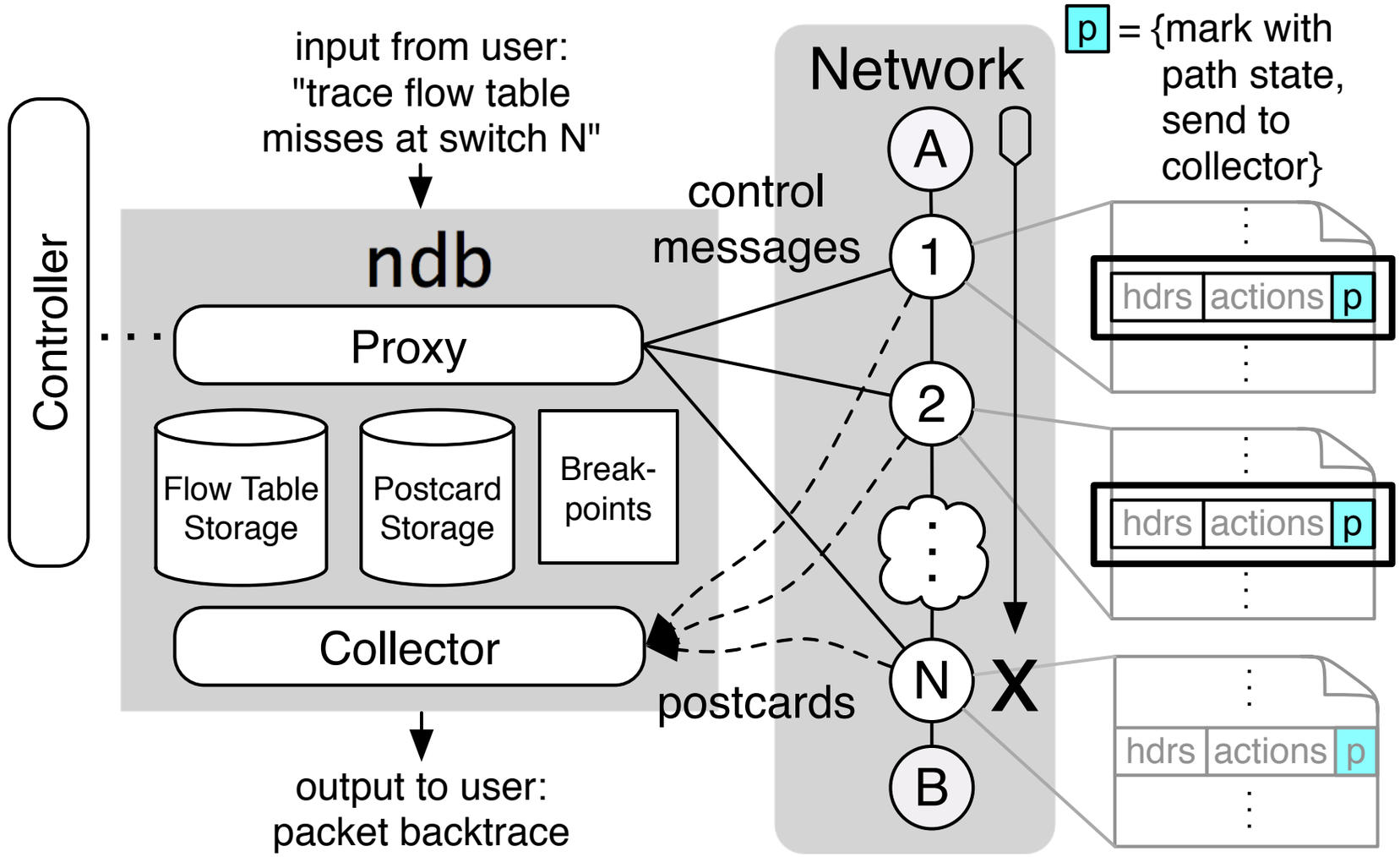
(how) building ndb



create postcards

- ndb proxy modify control message, instructing switches to create postcards everytime a packet passes by
- postcard: a small truncated copy of the packet header + switch, port …

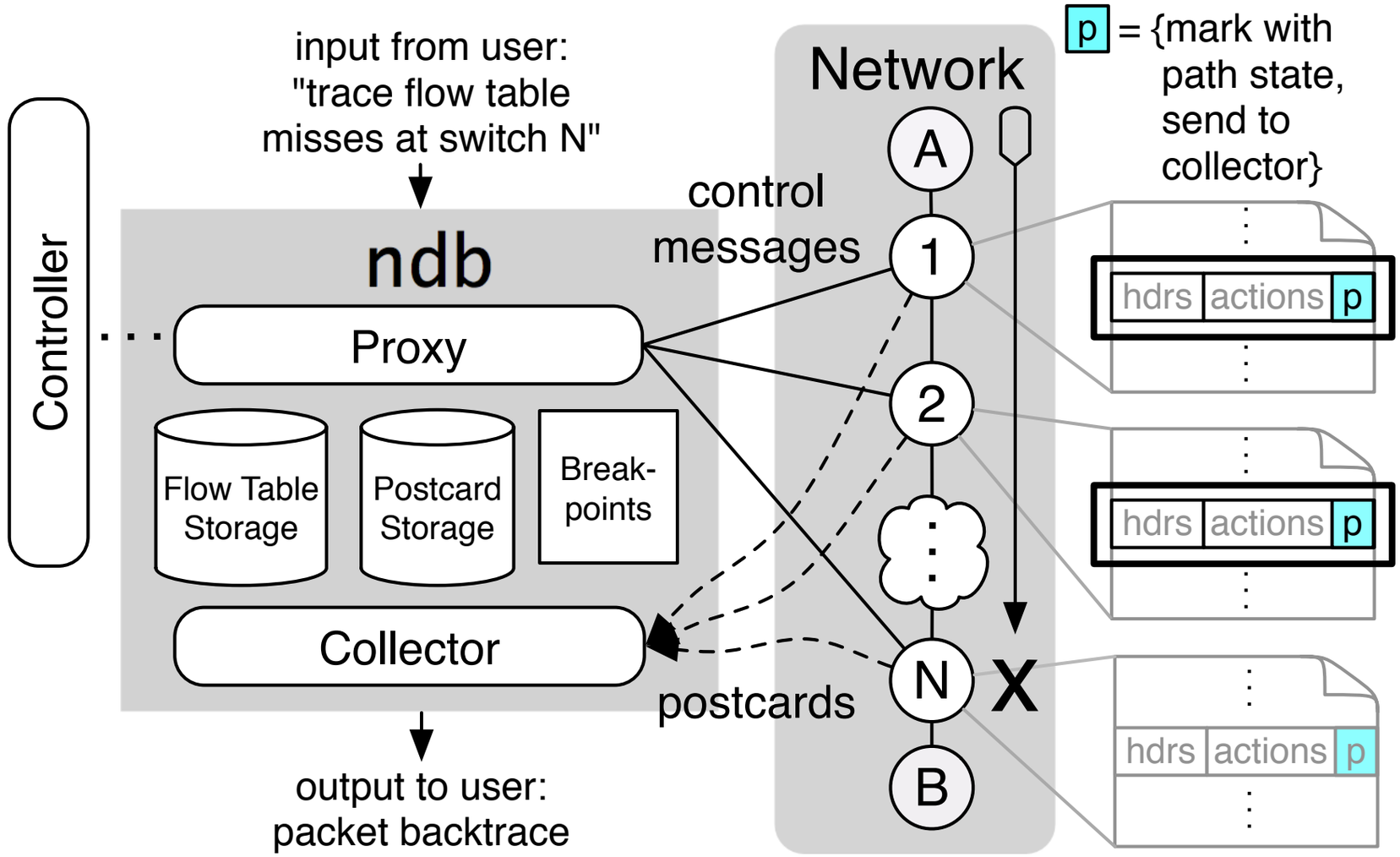
(how) building ndb



collect postcards

- store and retrieve postcards with a hash table
  - key: (immutable) packet header fileds

(how) building ndb



reconstruct backtrace

- when a packet triggers a breakpoint at the collector
- finding the matching postcards (packet headers)
- leverage topology — topological sort to rebuild the backtrace path

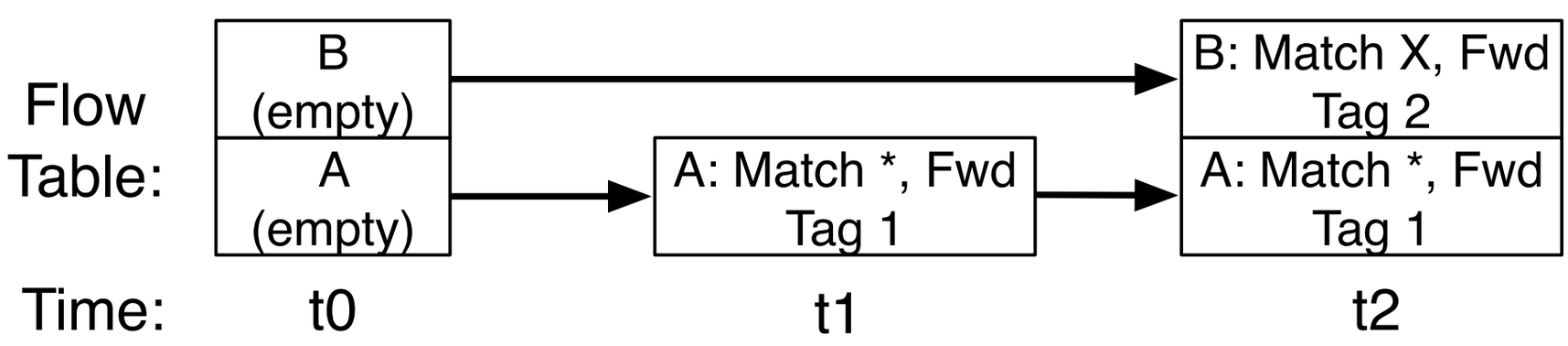
challenges — ambiguity

flow table ambiguity

- forwarding decided by the *full* flow table
- postcards only report the matching flow entires

solution

- ndb keeps every change to flow table state



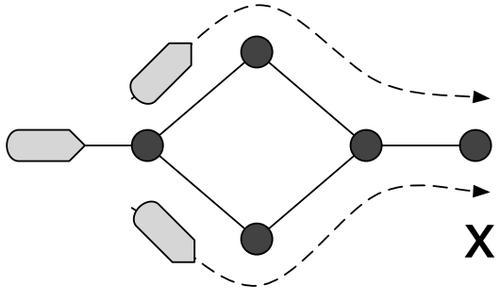
Packet matched A, postcard has Tag 1. What is the flow table state?

- really v1: B wasn't installed
- **? flow table state ambiguity**
- really v2: B didn't match

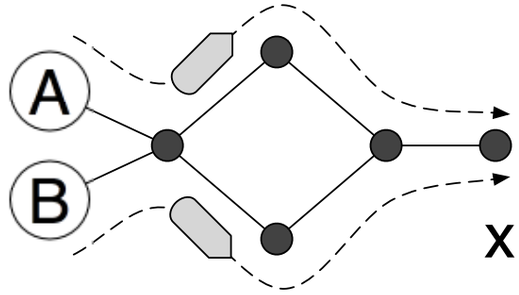
challenges — ambiguity

packet ambiguity

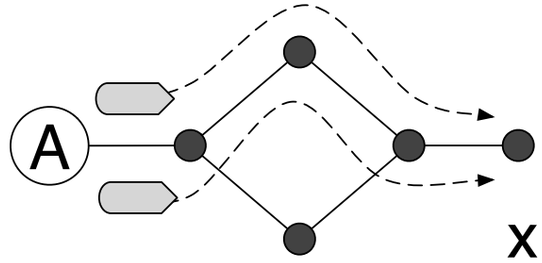
- forwarding decided by the *full* flow table
- postcards only report the matching flow entires



(a) Packet duplicated at the first switch



(b) A and B transmit identical packets



(c) A transmits identical packets

discussion

$$\frac{64B}{1031B} \times 5(hops) = 31\%.$$

discussion

performance overhead

- extra postcard bandwidth
  - every packet creates a postcard at every hop
- average packet size of 1031 bytes, minimum-size 64 postcards, network diameter of 5
- traffic increase of

$$\frac{64B}{1031B} \times 5(hops) = 31\%.$$

discussion

discussion

scaling the collector (at Stanford)

- provision packets rate of 7.8Gb/s, need 15.2M postcards/second
- hash table size: 1Gbyte of memory
- parallelized
  - yes
  - extreme: collector instances attached to individual switches

strength

strength

dataplane debugging

- network type
- control applications
- human roles
  - switch vendors, framework developers, network operators