Software Defined Networking

Dr. Nick Feamster
Associate Professor

In this course, you will learn about software defined networking and how it is changing the way communications networks are managed, maintained, and secured.

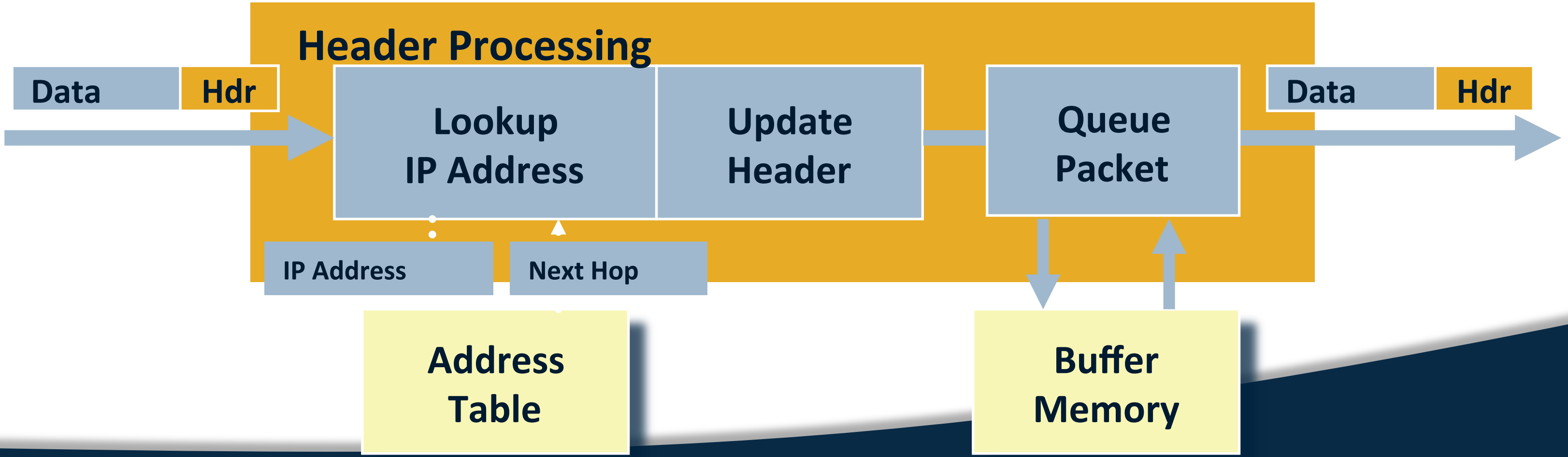
School of Computer Science

Module 5.1: Programmable Data Plane

- Two Lessons
  - Programming the data plane: Click
  - Scaling programmable data planes
- **Optional** programming assignment (in Click)
- Quiz on Concepts

Data Plane Review

- Router gets packet
- Looks at packet header for destination
- Looks up forwarding table for output interface
- Modifies header (TTL, IP header checksum)
- Passes packet to appropriate output interface



Data Plane

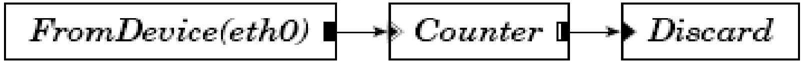
- Streaming algorithms that act on packets
  - Matching on some bits, taking a simple action
  - … at behest of control and management plane
- Wide range of functions
  - Forwarding
  - Access control
  - Mapping header fields
  - Traffic monitoring
  - Buffering and marking
  - Shaping and scheduling
  - Deep packet inspection

Motivation for Software Data Plane

- Network devices are diverse!
  - Must do much more than forward/route packets
  - Adding functions difficult
  - **Match/Action is only one type of data plane**
- Data plane design goals
  - Flexible
  - Extensible
  - Clean interfaces

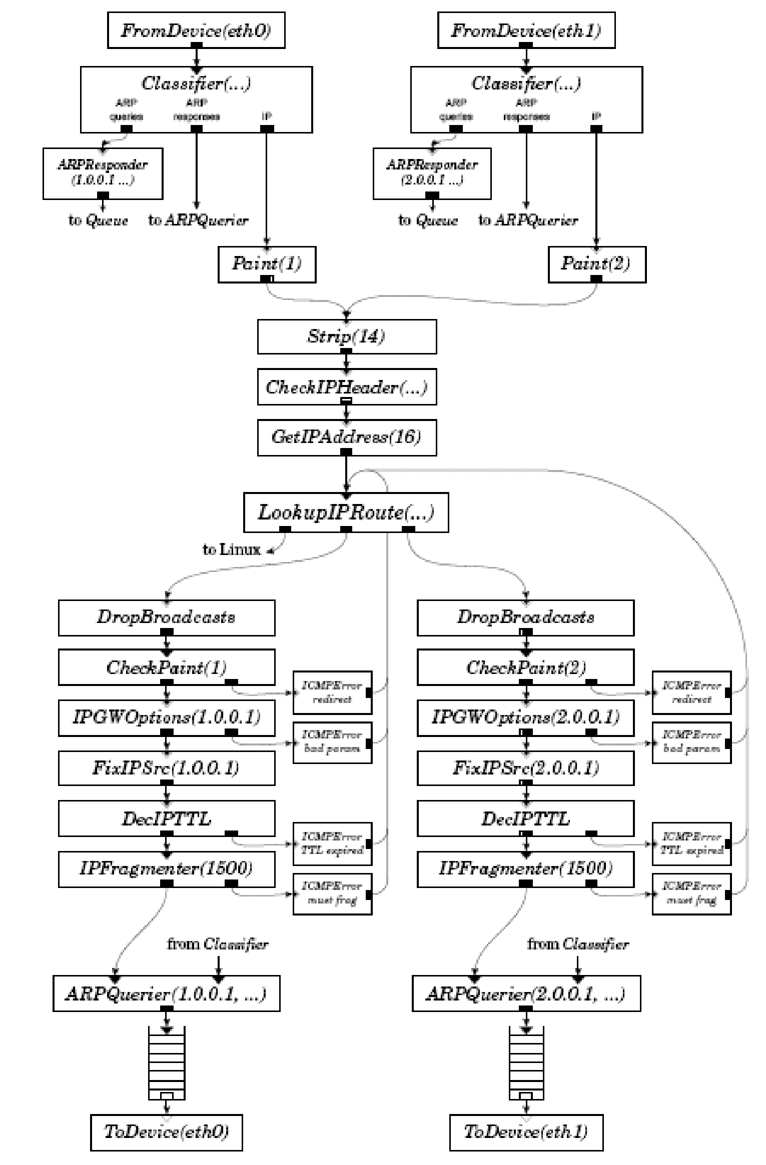
Click: A Software Data Plane

- **Elements** (building blocks)
  - Each individual element provides unique function
    - Packet switching
    - Lookup and Classification
    - Dropping



- **Implement functions:** assemble building blocks

http://www.read.cs.ucla.edu/click/


FromDevice(eth0)
FromDevice(eth1)
Classifier(...)
ARP queries
ARP responses
IP
Classifier(...)
ARP queries
ARP responses
IP
ARPResponder (1.0.0.1 ...)
to Queue
to ARPQuerier
ARPResponder (2.0.0.1 ...)
to Queue
to ARPQuerier
Paint(1)
Paint(2)
Strip(14)
CheckIPHeader(...)
GetIPAddress(16)
LookupIPRoute(...)
to Linux
DropBroadcasts
CheckPaint(1)
IPGWOptions(1.0.0.1)
FixIPSrc(1.0.0.1)
DecIPTTL
IPFragmenter(1500)
ICMPError redirect
ICMPError bad param
ICMPError TTL expired
ICMPError must frag
DropBroadcasts
CheckPaint(2)
IPGWOptions(2.0.0.1)
FixIPSrc(2.0.0.1)
DecIPTTL
IPFragmenter(1500)
ICMPError redirect
ICMPError bad param
ICMPError TTL expired
ICMPError must frag
from Classifier
ARPQuerier(1.0.0.1, ...)
from Classifier
ARPQuerier(2.0.0.1, ...)
ToDevice(eth0)
ToDevice(eth1)

Aspects of an Element

- **Class:** The code that should be executed when an element processes a packet
- **Ports:** Connections go from output port of one element to input port on another element
- **Configuration:** Additional arguments that are passed to the element at configuration time
- **Method:** Additional functions (e.g., reporting queue length)

Connecting Elements: Push and Pull

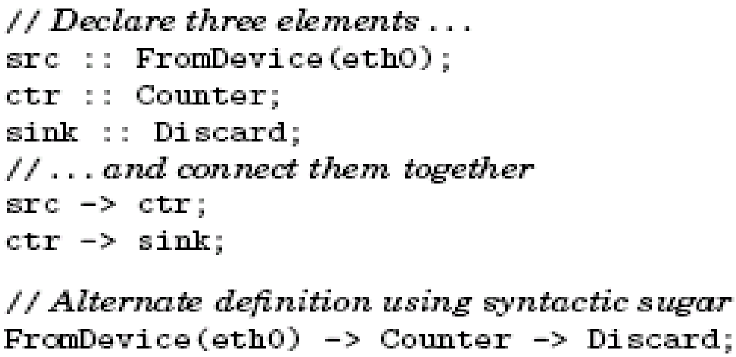
- Edges between two elements that could be possible data paths for packets
  - **Push:** Upstream element hands over a packet to a downstream element
    - packet-arrival element where the data is handed over to the next unit of processing
  - **Pull:** Downstream element requests data from the upstream element
    - transmit-side elements where the transmit ports will request for a packet from the previous element

Packet Storage: Queues

- Elements need to either store packets, discard them, or forward them to the next element.
- **Data storage necessary:** a push input and a pull output necessitates storage of pushed data until it is requested.
  - Packet storage at element is not implicit.
- Queues implemented as elements so that their insertion/ deletion becomes more configurable.
  - Need to be explicitly put at elements.

Configuration Language

- Two constructs
  - Declarations create elements
  - Connections say how they are connected
- Configuration string passed as is, as a list separated by commas to the element
- Other elements used as primitives to define **compound elements**



```
// Declare three elements . . .
src :: FromDevice(eth0);
ctr :: Counter;
sink :: Discard;
// . . . and connect them together
src -> ctr;
ctr -> sink;

// Alternate definition using syntactic sugar
FromDevice(eth0) -> Counter -> Discard;
```

Summary

- The data plane must also be programmable!
- Click: Open, extensible, configurable router framework.
- The example router configuration proves that a complex router can be designed using simple building blocks.
- Performance is acceptable for prototyping.
  - Click is still 90% as fast as the base Linux system