software defined networking

CS 6410 Fall 2017
Eric Campbell and Rolph Recto
software defined networking
software defined networking
(OpenFlow, originally)
TR10: Software-Defined Networking

Nick McKeown believes that remotely controlling network hardware with software can bring the Internet up to speed.

4 comments

KATE GREENE
Tuesday, February 24, 2009
“Stanford computer scientist Nick McKeown and colleagues developed a standard called OpenFlow that essentially opens up the Internet to researchers, allowing them to define data flows using software—a sort of ‘software-defined networking.’ Installing a small piece of OpenFlow firmware (software embedded in hardware) gives engineers access to flow tables, rules that tell switches and routers how to direct network traffic.”
software defined radio
software defined networking
software defined storage
software defined data center
software defined everything
software defined anything
software defined storage for dummies
software defined architecture
software defined definition
software defined storage solutions
control plane
  routing
  isolation
  traffic engineering
data plane
  packet forwarding
  packet scheduling
traditional networking
Software Defined Networking
(programmable)
Active Networks

Separating the Data and Control Planes

OpenFlow

SDN Today
use pulls

technology pushes

network ossification

desire for unified middlebox interface
use pulls

technology pushes

lower compute costs

advances in programming languages

DARPA Active Networks
programmable switches
with `sniff.java` installed, switch maintains table of packet counts by source IP
<table>
<thead>
<tr>
<th>srcIP</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0.0.1</td>
<td>1</td>
</tr>
<tr>
<td>10.0.0.2</td>
<td>1</td>
</tr>
</tbody>
</table>
capsules
packet contains instructions to push switch info at every hop
whither active networks?
whither active networks?

performance and security concerns

no “killer app”

no practical deployment plan

“The misconception that packets would necessarily carry Java code written by end users made it possible to dismiss active network research as too far removed from real networks and inherently unsafe.”

“The Road to SDN,” Feamster et al 2014
whither active networks?

performance and security concerns

no “killer app”

no practical deployment plan

“The misconception that packets would necessarily carry Java code written by end users made it possible to dismiss active network research as too far removed from real networks and inherently unsafe.”

“The Road to SDN,” Feamster et al 2014
Active Networks

Separating the Data and Control Planes

OpenFlow

SDN Today
use pulls

technology pushes

burgeoning network speeds

insufficient network reliability

specialized services (VPNs)
use pulls

technology pushes

open interface between control and data planes

logically centralized control
Open Shortest Path First (OSPF)
SoftRouter

Network Entity (NE)

Control Element (CE)

Forwarding Entity (FE)
Shortest Path Routing
vendors didn’t adopt ForCES (and others)

not general enough

no practical deployment plan
Active Networks

Separating the Data and Control Planes

OpenFlow

SDN Today
OpenFlow: enabling innovation in campus networks

Nick McKeown, Tom Anderson, Hari Balakrishnan,
Guru Parulkar, Larry Peterson,
Jennifer Rexford, Scott Shenker, Jonathan Turner

SIGCOMM 2008
Nick McKeown
Stanford

Jennifer Rexford
Princeton

Scott Shenker
Berkeley
use pulls

technology pushes

networking research

market factors

datacenter networks
use pulls

technology pushes

backwards compatible

general packet processing
(more fields to match on)
OpenFlow protocol

<table>
<thead>
<tr>
<th>fields</th>
<th>counter</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>srcIp=10.0.0.*, ipProto=TCP</td>
<td>10</td>
<td>pt = 2</td>
</tr>
<tr>
<td>dstPort=80</td>
<td>0</td>
<td>drop</td>
</tr>
</tbody>
</table>
learning switch
controller maintains hash table of MAC to port number mappings
src: spongebob
dst: patrick
msg: I'M READY
packet_out

MAC

spongebob

port

1
install rule: **forward packets** for **spongebob** to port 1
**src**: patrick
**dst**: spongebob
**msg**: i’m patrick
no flooding required!
install rule: **drop packets** destined for squidward
src: spongebob
dst: squidward
msg: hello
Active Networks

Separating the Data and Control Planes

OpenFlow/SDN

SDN Today
Frenetic

Abstractions for Network Update

NetKAT

Propane
NetKAT

Forwarding Policy (F)

if sw==A then
  (if pt==1 then pt=2 elif pt==2 then pt=1)
elif sw==B then
  (if pt==1 then p=2 elif pt=2 then pt=1)
else drop

if sw==A and pt==2 then (sw=B; pt=1)
elif sw==B and pt==1 then (sw=A; pt=2)
else drop

Network Behavior
run(F;T)

Topology (T)
NetKAT

Forwarding Policy (F)

\[
\begin{align*}
\text{if } \text{sw} &= \text{A} \text{ then} \\
&(\text{if } \text{pt} = 1 \text{ then } \text{pt} = 2 \text{ elif } \text{pt} = 2 \text{ then } \text{pt} = 1) \\
\text{elif } \text{sw} &= \text{B} \text{ then} \\
&(\text{if } \text{pt} = 1 \text{ then } \text{pt} = 2 \text{ elif } \text{pt} = 2 \text{ then } \text{pt} = 1) \\
\text{else drop}
\end{align*}
\]

Topological Policy (T)

\[
\begin{align*}
\text{if } \text{sw} &= \text{A} \text{ and } \text{pt} = 2 \text{ then } (\text{sw} = \text{B}; \text{pt} = 1) \\
\text{elif } \text{sw} &= \text{B} \text{ and } \text{pt} = 1 \text{ then } (\text{sw} = \text{A}; \text{pt} = 2) \\
\text{else drop}
\end{align*}
\]

Intuition: These are functions from packets to sets of packets.

Network Behavior

\[
\text{run}(F; T)
\]
Want to show: 
SSH packets sent from H1 get to H2

Invariant true when P1 is equivalent to P2.

let @h1 be sw==A and pt==1
let @h2 be sw==B and pt==2

\[
\text{if typ==SSH and } @h1 \\
\text{then run(\text{F} ; \text{T}) ;} \\
\text{eventually}(\text{@h2})
\]

\[
\text{if typ==SSH and } @h1 \\
\text{then run(\text{F} ; \text{T})}
\]
Want to show:
SSH packets sent from H1 get to H2

Invariant true when P1 is equivalent to P2.

P1
if \( \text{typ} == \text{SSH} \) and \( \@h_1 \)
then run(F,T);
eventually(\@h_2)

P2
if \( \text{typ} == \text{SSH} \) and \( \@h_1 \)
then run(F,T)

SSH packets from H1
Want to show:
SSH packets sent from H1 get to H2

Invariant true when P1 is equivalent to P2.

P1
if typ==SSH and @h1
then run(F;T);
eventually(@h2)

P2
if typ==SSH and @h1
then run(F;T)

“run” the network
Want to show:
SSH packets sent from H1 get to H2

Invariant true when P1 is equivalent to P2.

if typ==SSH and @h1 then run(F;T);
eventually(@h2)

packets received by H2

if typ==SSH and @h1 then run(F;T)
if sw == A then:
    if port == 1 then port := 2
    elif port == 2 then port := 3
    else drop
elif sw == B then:
    if port == 1 drop
    else port := 1
elif sw == C
    if port == 1 then port := 3
    elif port == 3 then port := 2
    elif port == 2 drop;
NetKAT

if sw == A then:
    if port == 2 then alert_ctrl
    else port := 3

if sw == B then:
    if port == 1 then drop
    elif port == 2 then port := 3
    elif port == 3 then port := 2

if sw == C
    if port == 1 then alert_ctrl
    elif port == 2 then port := 1
    elif port == 3 then port := 2
Why VMware is spending $1B-plus to buy Nicira

VMware makes strong software-defined networking play by purchasing Nicira
Silicon Valley Makes a Rare Bet on Silicon

By Ian King
September 13, 2017, 5:00 AM EDT  Corrected September 15, 2017, 2:22 PM EDT

- Barefoot Networks emerges as leading startup chipmaker
- Company has big backers in Goldman Sachs, Google, Alibaba
questions?
thanks!