Packet-In Message Control for Reducing CPU Load and Control Traffic in OpenFlow Switches

Daisuke Kotani, Yasuo Okabe
(Kyoto University, Japan)
OpenFlow

- OpenFlow: One Component to Realize Software Defined Networks
  - Flow Tables allow us to define how to process packets in switches
  - Centralized Management of Flow Tables in a Controller
Packet Forwarding in OpenFlow

Matched With Flow Entry: Follow Actions in Flow Entry

<table>
<thead>
<tr>
<th>Flow Header</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dest Host=B</td>
<td>Output to Port 3</td>
</tr>
<tr>
<td>Dest Host=A</td>
<td>Output to Port 1</td>
</tr>
</tbody>
</table>

Missing Flow Entry: Forward to Controller as Packet-In message

Set up Flow Entries **Reactively**

- **Flow Header**
- **Action**
  - Dest Host=B: Output to Port 3
  - Dest Host=A: Output to Port 1

Host A

OpenFlow Switch

Flow Table

Host B

OpenFlow Controller

Control Program

October 26, 2012

Kyoto University
A Problem occurred by many Packet-In messages

Set up Flow Entries Reactively & A host sends “Heavy Flow”

↓

Many Packet-In messages are sent to the controller

- Heavy Flows: like video streams over RTP
  - A host suddenly starts to send many packets in the same flow without any negotiation in advance

![Diagram showing packets going from Host A to Host B through OpenFlow Switch and messages being sent to OpenFlow Controller]

Consume many computing resources in switches and controller

Increase Traffic in Management Network
A Problem occurred by many Packet-In messages

- Normal Flows: like TCP sessions
  - Establish a session between hosts

- Limiting the overall bandwidth of Packet-In messages drops Packet-In messages of “Normal Flows”
  - Delay for insertion of flow entries of such flows

We need a method to limit only Packet-In messages from “Heavy Flows” in switches
Overview of Our Work

- A Problem by many Packet-In messages of “Heavy Flow”
  - Many Packet-Ins are sent to a controller before a flow entry is set (Consume CPU/Memory/bandwidth for management)
  - Limiting the overall bandwidth of Packet-Ins is not a good idea
    - Such method drops many Packet-Ins of “Normal Flows”

- Categorize Packet-In messages based on the processing in controllers
  - State Change, Flow Setup (Important), Forward (Less Important)
  - Most of Packet-Ins by “Heavy Flows” would be in “Forward”

- Propose a method to limit Packet-Ins in “Forward” group
  - Switches record flows whose packets are sent to the controller
Related Work

- Improve performance and stability of controllers
  - ONIX [Koponen 2010], HyperFlow [Tootoonchian 2010]
  - Loads of switches and management networks would not be reduced

- Enable more complex operations in the forwarding plane
  - DIFANE [Yu 2010], DevoFlow [Curtis 2011]
  - Packet-In messages would be reduced

- Meter (introduced in OpenFlow 1.3)
  - Bandwidth / Rate limit mechanism in OpenFlow
  - Can control the rate/bandwidth of Packet-In messages per pre-defined group of flows
Categorization and Discrimination of Packet-In messages

① **State Change:** *Important* for control
   Change data in controllers
   Ex: Obtain location of hosts using ARP
   Set Flow Entries manually
   to send to controllers in advance

② **Flow Setup:** *Important* for control
   Trigger insertion of flow entries
   The first packet of a flow

③ **Forward:** *Less Important* for control
   Forward packets to other switches
   (Most of packets in Heavy Flows)
   Second or later packets of a flow

Note: A message would be categorized to two or more categories.

Propose a method to limit only “③ Forward” in switches
Record flows whose packets are sent to the controller

- Add “Pending Flow Table” in switches
- When a switch sends a Packet-In message to controllers,
  - Check whether a flow of the Packet-In message is matched with an entry of Pending Flow Table
  - Record a flow of the Packet-In message to Pending Flow Table

Sender

OpenFlow Switch

Packet-In message

OpenFlow Controller

Packets

Look up & Insert

Look up

Delete

Pending Flow Table

Bandwidth Control

Flow Entry
Pending Flow Table

- Contain flow information whose packets are sent to controllers
- An entry of Pending Flow Table consists of:
  - Flow Header: Same as Match Fields in Flow Entry, Exact Match
  - Hard Timeout: Expiration of the Entry
- Add an Entry when a Packet-In message is generated
- Delete an Entry when:
  - A flow entry that cover the flow is inserted
  - The entry has timed out

<table>
<thead>
<tr>
<th>Flow Header</th>
<th>Hard Timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPsrc=A, IPdst=B, protocol=TCP, srcport=1025, dstport=80</td>
<td>YYYY/MM/DD HH:MM:SS</td>
</tr>
<tr>
<td>IPsrc=A, IPdst=C, protocol=TCP, srcport=1026, dstport=80</td>
<td>YYYY/MM/DD HH:MM:SS</td>
</tr>
</tbody>
</table>
Implementation

- Extend Open vSwitch 1.0.1
- When a switch sends Packet-In messages to controllers:
  - Insert a flow entry to forward packets to the controller through another bandwidth control mechanism
- When a controller adds a new flow entry:
  - Delete entries of Pending Flow Table which are covered by the flow entry, and corresponding flow entries
Evaluation

- How many Packet-Ins to trigger insertion of flow entries ("Flow Setup" category) are sent to the controller?
- Overhead in a Switch

- Compare two methods for limiting Packet-In messages
  - Our Method: Limit both all and "Forward" category of Packet-Ins
    - The overall bandwidth of Packet-Ins to 4Mbps
    - The bandwidth of "Forward" type of Packet-Ins to 1Mbps
  - Rate Limit Method: Limit only all Packet-Ins
    - The overall bandwidth of Packet-Ins to 4Mbps

Flow Generator

2000 new flows/sec + One 30Mbps flow (10sec -20sec from the beginning)

Open vSwitch

Packet-In messages (Except heavy flow)

Insert flow entries

Count flows

OpenFlow Controller
Evaluation

- Flow Generator sends two kinds of traffic
  
  **Normal Flows:** 2000 new flows/sec, one 64 bytes packet in one flow
  - Emulate the first packets of flows that negotiate between hosts ("Flow Setup" category)
  
  **Heavy Flow:** 30Mbps (about 2600 packets/sec, 1500 bytes each)
  - Emulate a flow to send many packets suddenly in the same flow with no negotiation in advance ("Forward" category)
  - Send from 10 sec to 20 sec after starting to send Normal Flows

- Controller
  
  - Count flows in Packet-In messages
  - Insert flow entries for Normal Flows

---

**Diagram:**

- **Flow Generator**
  - 2000 new flows/sec + One 30Mbps flow (10sec -20sec from the beginning)

- **Open vSwitch**

- **Packet-In messages**
  - Flow Entries (Except Heavy Flow)

- **Insert flow entries**
  - Count flows

---

**OpenFlow Controller**
Evaluation Results: the Number of Flows

The number of flows in Packet-In messages
(Flow Setup type of Packet-In messages)
(flows/sec)

Rate Limit Method: Reduced
- Packet-Ins of Heavy Flow (Forward category) occupy most of the bandwidth

Our Method: Not Reduced
- Packet-Ins in Heavy Flow (Forward category) are well filtered out

- Trade-off
- Our method increases packet loss in heavy flow

Send the Heavy Flow (30Mbps)
Send Normal Flow (2000 new flows/sec)
Evaluation Results: Overhead in Switches

CPU and Memory Usage in Switches

CPU Usage: A little difference if the number of flow entries are almost same

- Processing of our method has high affinity with that of OpenFlow
- Costs heavily on processing of OpenFlow

Memory Usage: Increased
- due to Pending Flow Table

- Our Method: CPU Usage
- Our Method: Memory Usage
- Rate Limit Method: CPU Usage
- Rate Limit Method: Memory Usage

Send the Heavy Flow (30Mbps)

Send Normal Flow (2000 new flows/sec)
Summary and Future Work

- Propose a method for a problem occurred by many Packet-In messages of heavy flows

- Categorize Packet-In messages
  - State Change (Important): Change data in the controller
  - Flow Setup (Important): Trigger insertion of new flow entries
  - Forward (Less Important): Forward packets to other switches

- Filter out Packet-Ins in “Forward” using Pending Flow Table
  - Record flows whose packets are sent to the controller
  - Limit bandwidth of packets that are matched with recorded flows

- Evaluation Results suggests switches with our method forward more packets of new flows with a little overhead

- Future Work
  - Implementation and Evaluation using hardware switches
  - Other methods for the similar situation occurred by many flows