Middlebox Deployments in Data Centers

Middleboxes, which provide layer 3-7 network services, are a crucial part of many enterprise data centers and cloud deployments. They help ensure security, improve performance, and meet other sophisticated goals.

Traditionally, enterprise data centers have included dedicated physical appliances at network chokepoints. However, middleboxes are now being deployed in a variety of additional forms—as VMs, in hypervisors, or as collections of processes—and placed at arbitrary points in the network. Unfortunately, current approaches for managing middleboxes are limited and clumsy, reducing enterprises’ abilities to create more dynamic deployments. We address this issue by designing a software-defined middlebox networking framework that simplifies management and engenders rich, new applications.

Motivating Scenarios

Live migration between software-defined data centers
1 & 2) Migration of application VMs & middlebox VMs
3) Transfer of middlebox state to physical appliance
Requires programmatic control over middlebox state

Middlebox scaling and load balancing
1) Clone middlebox VM
2) Divide flows evenly
3) Clone middlebox VM
4) Re-balance flows by moving per-flow state
5) Divide flows evenly
Requires fine-grain control over middlebox state

Manipulating State

Determine where state resides
Create & update state

Broad Operations Interface
get (Filter, Key, Action)
add (Key, Action, Offset)
remove (Filter)

Expose State-Related Events
Triggers
1) Created/updated state
2) Require state to complete operation

Contains
- Key
- Packet
- State

Representing State

Key
Field1 = Value1
…
FieldN = ValueN

Action
Offset1 \rightarrow Const1
…
OffsetN \rightarrow ConstN

Supporting
Binary Blob

Structure depends on deep internal middlebox logic

Classification of Middlebox State

<table>
<thead>
<tr>
<th>Class</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Defines operations to apply to packets/flows</td>
</tr>
<tr>
<td>Supporting</td>
<td>Helps decide between possible actions</td>
</tr>
<tr>
<td>Tuning</td>
<td>Tunes algorithms for performance/efficiency</td>
</tr>
</tbody>
</table>

How is the logic divided?
Where is state manipulated?
What interfaces are exposed?

Key Issues

Controller
App
App

Middlebox
Middlebox
Middlebox

Open Questions

1) Representing shared (i.e., not per-flow) state
2) Encoding supporting state & other action state
3) Preventing invalid state manipulations
4) Maintaining operation during state changes
5) Design of control logics for a variety of scenarios