GEN: A GPU-Accelerated Elastic Framework for NFV

Zhilong Zheng  Jun Bi  Chen Sun  Heng Yu  Hongxin Hu
Zili Meng  Shuhe Wang  Kai Gao  Jianping Wu
Network Function Virtualization (NFV)

**Virtualization Techniques**

- Low cost
- Elasticity control
- Service provisioning flexibility
CPU-based NFV

- OpenNetVM (HotMiddlebox’16)
- NetBricks (OSDI’16)
- NFP (SIGCOMM’17)
- Metron (NSDI’18)

General-purpose Multi-core Servers

• Problems
  – Low performance with negative improvement expectation
  – Coarse-grained scaling
Problems of CPU-based NFV

• Low performance with negative improvement expectation
  – Hard to achieve high performance (e.g., 40~100Gbps) for a wide range of NFs
  – The slow/end of Moore’s Law

<table>
<thead>
<tr>
<th>IPSec</th>
<th>NIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>(AES &amp; SHA1)</td>
<td>(Aho-Corasick)</td>
</tr>
<tr>
<td>2.6 ~ 7.7 Gbps</td>
<td>4.2 ~ 10.4 Gbps</td>
</tr>
</tbody>
</table>


• Coarse-grained scaling

<table>
<thead>
<tr>
<th>9 Mpps</th>
<th>10 Mpps</th>
<th>11 Mpps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 CPU core</td>
<td></td>
<td>2 CPU cores</td>
</tr>
</tbody>
</table>

E5-2650 v2 (8 Cores, 2.6 GHz)

10 Mpps

4.2 ~ 10.4 Gbps

Underutilized
GPU as An Accelerator for NFV

• Benefits of GPU
  – Massive processing cores
  – Fine-grained computing units

• Existing work
  – Router (*PacketShader*, SIGCOMM’10)
  – SSL proxy (*SSLShader*, NSDI’11)
  – NIDS (*Kargus*, CCS’12)
  – IPSec (*NBA*, EuroSys’15)
  – NFV framework (*G-NET*, NSDI’18)
GEN exploits GPU to support high-performance SFCs with fine-grained scaling
GEN Framework Overview
Infrastructure Design

CPU (User Space)

- SFC Manager
- Output Queuing
- Packet Forwarder
- Packet Dropper

SFC Controller #1

- Adaptive Batcher
- SFC Starter

SFC Controller #n

- Chain Classifier

GPU (2k~3k physical cores)

- Global Memory

- Chain #1 NF #1
- Chain #1 NF #2
- Chain #1 NF #3
- Chain #n NF #1
- Chain #n NF #2
- Chain #n NF #m

NIC

10/40/100 GbE Ports

- Tx
- Rx

SFC Stater

Adaptive Batcher

Packet Forwarder

Packet Dropper

High Performance

Elastic Scaling
Problem #1: SFC Model Selection

Pipelining

Run-to-completion (RTC)
SFC Model Selection: Pipelining

• Two potential ways to support pipelining in GPU

**Sequenced invocations**

**CPU**
- Packet batch
- 1. Packet copying
- Worker-NF1
- Worker-NF2
- 5. Next NF
- Out

**GPU**
- Packet Buffer
- 3. Reading
- NF1
- 4. Synchronization
- 6. Kernel invocation
- NF2
- 7. Reading

**Persistent kernels**

**CPU**
- Packet batch
- 1. Packet copying
- Worker-SFC
- Out

**GPU**
- Packet Buffer
- 2. Reading
- NF1 (persistent)
- 3. Next NF
- NF2 (persistent)
- 4. Reading

*High overhead from frequent kernel invocations (~5us per invocation)*

*Hard and costly scaling*
SFC Model Selection: RTC

• RTC-based Model

- NFs are integrated into a specific **SFC Agent**
- SFC Agent (in GPU) is launched by **SFC Starter (in CPU)**

**Figure:**

- Worker-SFC: Packet batch → Out
- CPU: Packet Buffer
- GPU: NF1, NF2 → Packet
- RTC Model: Less kernel invocations (once per SFC), Easier scaling (not persistent)
Problem #2: Elastic Scaling

• Avoid monitoring NF load for scaling
  – Avoid deciding *when* to scale
  – Avoid deciding to what *extent* an NF should be scaled
  – Avoid considering how to quickly carry out NF scaling

• Avoid state management caused by scale out / in
  – Intuition: Use scale up / down to avoid state management

• *Adaptive Batcher*
Elastic Scaling – Adaptive Batcher

- Design of the adaptive batcher
  - Keeping the buffer occupancy at a low level
  - Scaling up/in GPU resource provisioning
Preliminary Evaluation

• **Hardware**
  – CPU: Two Intel Xeon E5-2650 v4 (10 physical cores)
  – GPU: NVIDIA TITAN Xp
  – NIC: Two Intel X520 (40 Gbps in total)

• **Software**
  – DPDK 17.11 for networking IO
  – CUDA 8.0 for GPU programming

• **NFs & SFCs**
  – IPv4Router (1k entries) → NIDS (3k rules) → IPSec (SHA1 & AES-128-CBC)
Performance of RTC vs. Pipelining

- Throughput (Gbps) comparison:
  - Pipeline vs. RTC for different packet sizes (Byte).
  - 95th percentile latency improvement: 33.7%.
  - 29.2% and 28.1% for certain packet sizes.
Fast Elastic Scaling

![Graph showing throughput and timeline](image)

Throughput (Gbps) vs. Timeline (second)

- **Fast converging** (<100ms)
Conclusion and Future Work

• **Gen**: a GPU-accelerated elastic framework for NFV
  – High-performance SFC
  – Elastic scaling

• **Future work**
  – More SFC performance enhancement in GPU
  – Coordination between CPU and GPU
  – Impact of dynamic traffic load
Thank You

http://netarchlab.tsinghua.edu.cn