MEMOCO: Memory-Efficient 100 Gbps Traffic Replay with Sequence Guarantee on Multi-core Servers

Ranzheng Cao*
BISTU
China

Wendi Feng*†
BISTU
China

ABSTRACT

Network has become the most important infrastructure in today's information society, in which traffic generation is essential in aiding network system development and testings. However, high-performance traffic replay, say 100 Gbps, is challenging. **i)** Achieving 100 Gbps packet sending requires using multiple cores to simultaneously send packets, but the sequence is hard guarantee. Moreover, **ii)** replaying real-world traffic at 100 Gbps and beyond for one second needs to load at least 12.5 GB into the memory, which incurs a significant burden to the memory system. To this end, we present MEMOCO, a software-based 100 Gbps traffic generator to generate sequence guaranteed packets and replay real-world traces using fixed memory footage. At the heart of MEMOCO is a judiciously designed inter-core scheduler and a machine learning-based traffic pattern recognizer. We present the key design of MEMOCO to call for broad comments to the system.

CCS CONCEPTS

• **Networks** → **Programmable networks**.

ACM Reference Format:
Ranzheng Cao and Wendi Feng. 2022. MEMOCO: Memory-Efficient 100 Gbps Traffic Replay with Sequence Guarantee on Multi-core Servers. In *Asia-Pacific Workshop on Networking (APNet '22), July 1–2, 2022, Fuzhou, China.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3542637.3543710

1 INTRODUCTION

Along with the evolution of network systems, traffic generation plays an important role in developing and testing network systems. And the sequence of the generated packets is critical. For example, testing deep packet inspection (DPI) systems may require a specific order of flow packets to update the state machine of its underlying protocol. Besides, replaying real-world traffic using the traffic trace files (*e.g.*, PCAP files) captured from the real-world are commonly used for testing network systems' performance in the real-world and is a key functionality of traffic generator.

Traditional traffic generators are vendor-specific and often shipped in a dedicated hardware box. It may achieve notable performance, but the cost can be prohibitive. Network function virtualization (NFV) introduces the idea of running network functions (NFs) software on general-purpose commodity servers, which lights up the hope to realize software-based high-performance traffic generator with acceptable costs [3].

However, achieving high-performance traffic generation on commodity server, say 100 Gbps and beyond, is challenging as the platform is not optimized for network packet processing. A single CPU core is insufficient in conducting 100 Gbps packet sending [1] due to limited L1/L2 cache resources. However, using multiple cores to send packet without caution may fail to comply with the packet sequence constraint. Although a simple lock-based synchronization technique can sequentialize the packets, but it also sequentializes the generation tasks across different cores, which loses the benefits of parallelism and may result in failing to achieve 100 Gbps traffic generation [1, 3]. Moreover, when replaying traffic from trace files, theoretical calculation indicates that a 12.5 GB file must be loaded to the memory for only a second packet replay. The huge amount of data can significantly burden the memory system.

To this end, we present a software-based traffic generator, MEMOCO. It sends packets at 100 Gbps line-rate speed without compromising the packet sequence. And it can also replay real-world traffic from trace files with minimum memory usage overhead using fixed-size memory footage. The magic behind is an intelligent yet efficient inter-core scheduler to coordinate the packet sending across cores with little overheads and a machine learning-based (ML-based) traffic pattern recognizer that identifies the pattern from a trace file (regardless of its size) and represents the pattern in a fixed-sized memory footage. We present the key design of MEMOCO in this paper, and its contribution is two-fold, listed as follows.

- We identify the challenges in high-performance (*i.e.*, 100 Gbps) traffic replay on general purpose commodity servers, and
- we present MEMOCO and introduces its key design.

2 MOTIVATION

Recent studies reveal that traffic generators face significant hurdles in generating high line-rate traffic on commodity server platforms. **i)** A single CPU core cannot support traffic sending at 100 Gbps line-rate speed. Sending 64 B small packets at 100 Gbps line-rate speed results in an average packet processing delay of 6.7ns or less. A typical multi-core server's L1/L2 cache (exclusive to each core) has an access latency of less than 5 ns. The LLC/L3 cache (shared between cores) has an access latency of around 13-20 ns. So The key to generating packet traffic at 100 Gbps line-rate speed is to ensuring that both the packets and the state are "bound" to L1/L2

*Both authors contribute equally to the paper.
†Corresponding author. Email: wendifeng@bistu.edu.cn

APNet '22, July 1–2, 2022, Fuzhou, China

ACM ISBN 978-1-4503-9748-3/22/07…$15.00
https://doi.org/10.1145/3542637.3543710

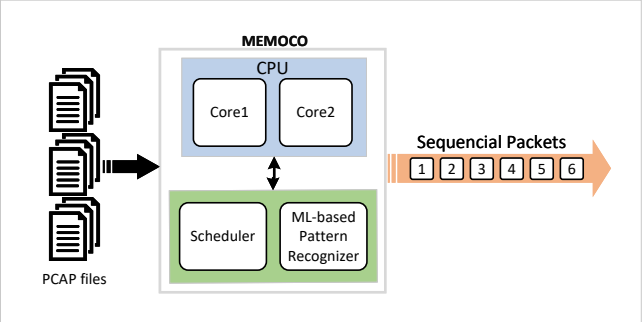

Figure 1: MEMOCO design overview.

caches. Packet grouping requires data to pass through L1/L2. However, the L1/L2 cache capacity of a single CPU core is minimal, even with many optimization techniques such as TSO, aRFS, and Jumbo Frame. Therefore, it's necessary to distribute traffic generation and send tasks to multiple cores to make full use of the processing capacity of the multi-core processor. Nevertheless, if we do not restrict the consistency of packets and cores, **ii)** It may not guarantee the holistic sequence in which each core generates and sends packets independently. If a flow is divided into multiple parts and each part sent by different CPU cores, it may cause the holistic sequence in chaos. This might have extremely catastrophic consequences. For example, NFs can be easily attacked (such as DDoS) under this circumstance. In addition, **iii)** Traffic replay considerably overwhelms the memory system. Traffic replay is a crucial feature of the traffic generator because it reads PCAP files into memory for processing and then extracts vital information to recompose data packets for sending. The amount of PCAP data to read is up to 750GB for keeping replay the traffic at 100 Gbps line rate speed only in one minute. If we load all of the files into memory to process them, the traditional server won't be able to replay the high-performance traffic over a long time. The performance is also non-extensible. Based on the analysis above, it is necessary to rethink and reconstruct the high-performance traffic generation system fundamentally.

3 MEMOCO DESIGN

To address the aforementioned issues, we present MEMOCO. At the heart of the intelligence is the two key components, namely, a *Sequence Scheduler* and a *ML-based Traffic Pattern Recognizer*, as shown in Figure 1. We demonstrate their main concepts in the remainder of this section.

3.1 Packet Sequence Guarantee

Guaranteeing the sequence of the generated packets is not an easy task, as multiple CPU cores are needed and synchronizations are required to ensure the sequence. Moreover, the process should be fast enough to comply with the processing latency requirement in 100 Gbps line-rate speed packet sending. MEMOCO employs a memory reserving strategy borrowed from RDMA, in which each core reserve its own block of memory space in the TX queue, and packets generated by the core are thus in the proper order the traffic/flow. The key here is to determine the size of the reserved space for the core. Meanwhile, MEMOCO employs an efficient scheduler to adjust the space based on history sending rates, with which, the overall traffic generation task can be split into sub-tasks that the generated packets can be fit into the reserved space. So that each packet generated by the core can appear in the proper relative order. If a flow needs to be sent continuously, the TX will be fully allocated to the CPU core. If a core is allocated a flow without sending tasks, the core will not occupy the TX space, thus avoiding the phenomenon of TX "empty space" and reducing performance loss. For example, Core 1 sends 1,2,3 and core 2 sends 4,5,6. Note that each flow must be generated and sent on the same core, specially bound on this core, which avoids the out-of-order of packets. The scheduler makes a reasonable schedule to determines whether the packet should be placed on TX. Therefore, as shown in Figure 1, MEMOCO ensure the sequential packets 1,2,3,4,5,6 in this manner.

3.2 ML-Based Fixed-Size-Memory Usage

Another design goal is to use a fixed-size-memory ML-model to represent the patterns in the PCAP files. We employ linear regression and taking flow identifier (*e.g.*, 5-tuple information) as the key feature. And we use Caida dataset [2] as the training set, time series as the input, and the number of packets that appear as the output. The flow identifiers are unique, so the output from the ML-model is consistent each time. It reduces the strain on the server memory system caused by reading PCAP files with enormous data volumes, resulting in a significant increase in traffic generation efficiency. The harsh packet generation requirement does not allow deep neural network models due to their complexity. The ML- models should be kept as minimal as feasible since the memory system is not expected to be overburdened. Furthermore, it is to be predicted that the time spent preparing the PCAP file will be shorter than the time spent training the model with the PCAP file. When PCAP files are received from the network or copied from other machines, the data is transferred to a specialized training server for "online" discovery of traffic pattern characteristics, which minimizes the preprocessing wait time. MEMOCO achieves the actual high-performance PCAP traffic replay in this way.

4 CONCLUSION

In this paper, we have presented the key design of MEMOCO, a software-based 100 Gbps sequence guarantee packet generator. It employs a judiciously designed scheduler and ML-based traffic pattern recognizer to guarantee the packet sequences and reduce memory overheads introduced by reading bulk of PCAP files for replaying at 100 Gbps, respectively. We present the primary design here to broadly call for comments of our design.

ACKNOWLEDGMENTS

This work is supported by BISTU Research Fund Under Grant 2022XJJ19.

REFERENCES

[1] Qizhe Cai, Shubham Chaudhary, and et al. 2021. Understanding host network stack overheads. In *SIGCOMM*. 65–77.
[2] Caida. [n.d.]. Caida dataset. https://www.caida.org, accessed: may 20, 2022.
[3] Peng Zheng, Wendi Feng, and et al. 2020. NFV Performance Profiling on Multi-core Servers. In *IFIP Networking*. 91–99.