Yi-Xi Chen

SOFTWARE ENGINEER · NETWORK PROFESSION

Tainan, Taiwan

(+886) 910635509 | chen.ishi@gmail.com | https://chenishi.github.io/ | chenIshi | yi-xi chen

Education

Tsinghua University
Master of Science in Computer Science
Beijing, China
Sep. 2019 - June 2022

- GPA: 3.65/4.0.
- Built a Rust interpreter, NFD2Rust, for NFD, a domain-specific language for cross-platform network functions within two months. Interpreting a Cpp-like language (NFD) to Rust executable, requiring plenty of additional optimization to satisfy Rust syntax rules, including lifetimes and ownerships.
- Maintained 8 servers connecting with both programmable and commodity switches in a server room, including setting up Ethernet connections and configurations, gatekeepers between intranet and internet, and reverse proxy for access to private servers.
- Cooperated with Hwawei engineers on a collaborative design of network gateways between network switches and servers. Offloaded most elephant flows, at most 83% traffic, to high-throughput switches by traffic prediction.
- Courses Taken:
  - Formal Methods and Protocol Engineering in Computer Network. Following my previous research during my Bachelor's degree on applying formal verification on smart contracts. Systematically learned verification tools such as PITSv3 and apply them to my master's research on custom protocols.
  - Distributed System. Implemented the Raft consensus algorithm to provide a fault-tolerant key/value service as the first project in Golang.
  - Network Security. Experimented with applying up-to-date protocols (e.g., ESNI on DNS) on different websites to achieve censorship evasion in mainland China.
  - Computer System Performance Testing. Motivated by my previous working experience on tracing code in the QEMU project with a huge code base. Systemically learned different evaluation approaches including instrumentation or profiling tools like gprof.

National Cheng Kung University
Bachelor of Engineering in Electrical Engineering
Tainan, Taiwan
Sep. 2015 - June 2019

- GPA: 3.64/4.3.
- Persistently contributed to writing the book expected to be released this year, ”Demystifying the Linux CPU Scheduler”, with professors and undergraduate students. Elaborate on evaluating scheduler designs like CFS and corresponding implementations.
- Engaged in open-source communities abroad including making presentations at FOSSASIA and HKOSCon.

Work Experience

National University of Singapore
Research Assistant
Singapore
Jul 2023 - Now

- Automated fine-grained network monitoring for a self-deployed research-focused open-5G network stack with Apache Ansible.
- Captured and analyzed traffic offline to troubleshoot bottlenecks in the network stack with P4 and Golang.
- Implemented a framework that adapts CPU core configuration based on network loads, reducing energy consumption while maintaining bandwidth requirements with eBPF/XDP.

Industrial Technology Research Institute
Research Intern
Hsinchu, Taiwan
July 2018 - Aug. 2018

- Elaborated with two full-time engineers on an open-source project, Cuju, an extensive virtualization-based fault tolerance implementation for migrating QEMU virtual machines.
- Located and troubleshot 47 integrity failures within a week using official unit tests like *qtest*.

Publications

Conference Paper

- Permal, S.K., **Chen, Y.**, Khooi, X.Z., Qiu, B., and Chan, M.C., 2024, Dec. ”Towards Continuous Latency Monitoring through Open RAN Interfaces.” In *2024 IEEE/ACM 4th 5G-MeMU Workshop* (in submission).
- ”Revisiting CPU Performance Scaling for Energy-Efficient Packet Processing” In *2024 IEEE/ACM 23th HotNets Workshop* (in submission).
- **Chen, Y.**, Wu, W., Shen, S.H. and Zhang, Y., 2022, June. ”NQ/ATP: Architectural Support for Massive Aggregate Queries in Data Center Networks.” In *2022 IEEE/ACM 30th International Symposium on Quality of Service (IWQoS)* (pp. 1-10). IEEE.
  - Resolved incast bandwidth bottlenecks in scaling data-center networks by traffic aggregation through programmable switches.
  - Reduced 96.82% link usage even with dynamic routing, boosting performance even further as the network size scales.
- Lao, C., Le, Y., Mahajan, K., **Chen, Y.**, Wu, W., Akella, A. and Swift, M.M., 2021, April. ”ATP: In-network Aggregation for Multi-tenant Learning.” In *NSDI* (Vol. 21, pp. 741-761).
  - Mitigated incast traffic towards parameter servers distributed machine learning with multiple tenants sharing the limited memory space on switches.
  - Guaranteed performance and correctness with multiple tenants competing for limited memory usage.
  - Accerlated training processes ranging from 1.24 to 8.7 with different models. **Best Paper Award**.

Open-source Conference Sessions

- Chen, Y., 2022, July. ”Why should we attend conferences oversea ? (translated from Mandarin).” Tapei. In *COSCUP*.
- Chen, Y., 2019, June. ”Formal Verification on Smart Contract.” Hong Kong. In *HKOSCon*.
- Chen, Y., 2018, March. ”Only Empty Lip Service ? Free Software Development in Taiwan.” Singapore. In *FOSSASIA*.

Skills

Programming	**Proficient:** P4, **Experienced:** C, Golang, Rust, Python
CS Interests	Programmable networks, virtual networks, network protocols, distributed system, data-center networking
Languages	Mandarin Chinese (Native), English (Business)

Appendix

Research intern: Cuju

https://github.com/Cuju-ft/Cuju

- Frequent VM state backup required in system migration led to high performance overhead and RTD(Round-Trip Delay) by about 40 times.
- Cuju reduced the processing overhead of fault tolerance with optimizations like dirty state tracking, boosting performance using fake ACK to unlock traffic rate control within VMs, reducing latency by 97%
- Connected VMs with virtual bridge networks on tap interfaces and also with the host machine with NIC configuration.
- Troubleshot integrity failures using official tests like *qtest*, *iotest*, and *kvm tests*, where failures include functionality not implemented and invalid function call paths.
- Led to further master study on networking at Tsinghua University.

ATP: In-network Aggregation for Multi-tenant Learning.

https://www.usenix.org/conference/nsdi21/presentation/lao

- Incast traffic from workers to parameter servers led to scalability bottlenecks in distributed machine learning, impacting even more on communication-heavy models like VGGs.
- With a coherent design between network switches and end servers, ATP accelerates the training process by up to 38% - 66%.
- Built custom traffic rate and congestion control in a multi-tenant scenario with variable background traffic patterns.
- Optimizations including converting floating-point operations to fixed-pointed integers due to the lack of P4 semantic support and packet segmentation due to restricted header parsing ability.
- Integrated ATP with different tools like PyTorch, TensorFlow ,and MXNet.

NQ/ATP: Massive Aggregate Queries in Data Center Networks.

https://ieeexplore.ieee.org/document/9812906/

- Extensive research based on ATP, making in-network aggregation adapted to the nature of dynamic routing in scalable data-center networks.
- Existing solutions with in-network computation failed under a non-deterministic routing path.
- Binding the routing protocol along with the multi-layered aggregation, NQ/ATP resolved the congested traffic by 87.5% and saves up at most 97.6% of switch memory usage.
- Based on a revised version of source routing, NQ/ATP constructs the necessary routing fields on-the-fly, where routing headers are shared by all logically-equivalent switches in dynamic routing.

Collaborative project with Hwawei: Gateway offloading

- Software gateways on commodity servers came with large capacity but low throughput. Thus, we offloaded the elephant flows to network switches with high throughputs.
- Implemented a coherent gateway between network switches and servers, servers predicting flow patterns and located elephant flows with the top-K ”predicted” traffic.
- Proposed a distributed gateway design to achieve higher bandwidth, splitting the flow space evenly to each software gateway based on the hashed result, evaluated with its fairness index.

NFD2Rust: Interpret NFD with Rust

https://github.com/chenlshi/NFD2Rust

- Interpret NFD, a Cpp-like language to describe network virtual functions, in addition to describing network terms like TCP SYN/ACK fields when parsing.
- As the first project in my master;s degree, the Rust language, and network virtualization functions, I actively cooperate with my senior colleague, the inventor of NFD, to clarify the syntax and behavior of NFD.

Demystifying the Linux CPU Scheduler

https://github.com/sysprog21/linux-kernel-scheduler-internals

- Elaborate on the Linux scheduler design iteration, each with hands-on experiments to identify the physical behaviors within the Linux kernel using profiling tools.