Research Statement

Jiarong Xing

Modern applications are characterized by ever-increasing demands for *security*, *reliability*, and *performance*. This has in turn imposed stringent requirements for the underlying infrastructure. My research aims to address these requirements by designing the next generation of systems and networking infrastructure.

My work capitalizes on the trend that the infrastructure is being enriched with emerging hardware accelerators, such as SmartNICs, Remote Direct Memory Access (RDMA) NICs, programmable ASICs, and GPUs. These accelerators provide raw materials for innovative systems, but at the same time, their full potential will only be unleashed via systematic designs of both their software and hardware layers. Therefore, I take a hardware/software co-design approach, which not only leads to new systems but also establishes synergy between software and hardware trends. By examining hardware trends, I identify better ways to design software systems; by understanding software requirements, I further identify gaps in today's hardware ecosystem and propose new device requirements. As a concrete example, my work leverages programmable ASICs to mitigate fast-changing network attacks ([1] in §1), and this, in turn, uncovers the limitation of current hardware in supporting reliable runtime updates. Following this realization, my subsequent work designs runtime programmable ASIC support ([2] in §2), which further opens new opportunities for enhancing software efficiency with real-time infrastructure optimization for dynamic workloads ([3] in §3).

My work has addressed problems in networking, systems, and security, employing an interdisciplinary approach that integrates insights from programming languages, program analysis, and machine learning. Thus far, my research has resulted in 11 conference publications at SIGCOMM, NSDI, SOSP, ASPLOS, MobiCom, USENIX Security, MLSys, and NeurlPS; seven workshop papers; and two papers under preparation. My work has been recognized by the USENIX Security Distinguished Paper Award, a Google PhD Fellowship, a Meta PhD Fellowship Finalist, and three university-wide fellowships at Rice. In terms of service, I have served on program committee for five conferences/workshops, including IEEE Symposium on Security and Privacy, a top-tier venue. My research outcome has been deployed in production systems by TikTok/Bytedance, and featured by Nvidia and Microsoft. Below, I describe my research in three dimensions.

- Infrastructure-native security. This work aims to architect a spectrum of defenses, normally performed external to the infrastructure, back into its substrate [4]. It transforms the infrastructure into a defense backbone, secure against various attacks targeting end-hosts [5, 6, 7], NICs [8] and the network [1, 9, 10].
- Reliable infrastructure management. My goal is to drastically increase the reliability and manageability of the infrastructure, by redesigning hardware [2, 11], developing programming abstractions [12, 11, 13], and optimizing resource provisioning [14, 11]. I consider this in the contexts of individual hardware and software devices [2, 14], datacenter networks [12, 11], and the cloud infrastructure [13].
- Performance clarity and optimization. I build efficient systems using a combination of performance prediction [15], runtime profiling [3, 16], and template-based profiling [17], accelerating the performance of SmartNICs [15, 3], GPUs [17], and the network stack [16].

1 Infrastructure-native security

The escalating cyberattacks have made security an imperative requirement. However, today's infrastructure has not played an active role in security defense. This is especially true for the network, which forwards normal and malicious traffic with minimum distinction, because traditional network devices cannot be programmed to perform needed defenses. Recent programmable network devices have opened up new opportunities to design security support inside the network with unique advantages, e.g., per-packet check at line rate. Leveraging this trend, my research proposes a new vision—Programmable In-Network Security, which architects security defenses into the network, complementing host-based defenses for attacks that CPUs either fail to perceive or cannot effectively handle. I pursue this vision in four steps.

Single-switch defenses [5, 6]. I start with transforming a programmable switch into a defense platform—while the switch forwards traffic, it also dynamically activates security logic as needed for defense. In this direction, I have developed a high-performance covert channel defense called NetWarden [5, 6]. Covert channels leak confidential data by modifying packet header fields and/or packet timing. Their defenses require examining, modifying, and/or time-modulating every single packet at terabits per second, which is extremely inefficient with software. NetWarden adopts a hardware/software co-design strategy where it partitions per-packet operations to the switch ASICs and offloads ASIC-unsupported operations (e.g., statistical tests) to the switch CPUs. This design enables NetWarden to detect and mitigate various types of covert channels at line rate. Its detection accuracy can be further improved with time-aware monitoring [18].

Network-wide defenses [1, 4]. Next, I transform a network into a defense fleet by coordinating individual switches—while the network routes traffic end to end, it also synchronizes security signals and mitigates attacks in real time [4]. I applied this strategy to link-flooding defenses in Ripple [1]. Link-flooding attacks can cut off a victim from the Internet by flooding the critical links that carry most of its traffic. The attackers can rapidly change their attacking vectors, e.g., target links, to launch dynamic attacks. This renders existing centralized solutions ineffective due to the prolonged latency of their feedback loops. Ripple addresses this problem through a decentralized design, where each programmable switch monitors local attack signals and synchronizes with other switches to construct a global view. The local and global views are programmable in our domain-specific language, and Ripple's compiler generates distributed switch programs automatically.

Protecting in-network defenses [10, 19, 9]. Furthermore, I safeguard in-network defenses themselves by exploring potential attack vectors and defense strategies. Programmable switches exhibit different behaviors based on input packets, which can be easily manipulated by adversaries. To understand the potential attack vectors, my colleagues and I developed P4wn [10, 19], which can perform probabilistic profiling of switch programs, and further identify edge-case behaviors for adversarial testing. It uses symbolic execution to analyze the behaviors of stateful P4 programs over a packet sequence, characterizes the probability for each processing behavior, and generates test traces for validation. My subsequent work P4Sync [9] expands the protection to decentralized network-wide defenses, protecting the integrity of synchronization probes.

From the network to hosts [7, 8]. Finally, I expand the protection to end hosts. My colleagues and I designed RDMI [7], which efficiently introspects remote machines' memory to detect kernel attacks (e.g., kernel rootkits) using Remote Direct Memory Access (RDMA). It implements an introspection machine on the switch ASICs to read and introspect memory by generating RDMA requests at line rate. This machine, furthermore, is reconfigurable at runtime using our domain-specific language and compiler to enforce varied introspection policies. On the other hand, RDMA is vulnerable to various attacks in clouds, because it lacks thorough security considerations in its original design. Traditional CPU-based defenses are inherently ineffective due to the violation of CPU bypassing, the key design feature of RDMA. In Bedrock [8], I solve this challenge by co-designing host kernels and programmable switches to provide transparent, in-network security support, including source authentication, fine-grained access control, monitoring, and logging.

2 Reliable infrastructure management

The demand for high reliability requires the infrastructure to maintain uninterrupted service during management events, such as function upgrades and failure handling. However, today's infrastructure lacks efficient and cost-effective reliability support for management events—current practices often involve costly service backup, intricate workload migration, or error-prone manual efforts. For instance, upgrading the in-network defenses on a switch (discussed in §1) requires complicated traffic engineering to drain traffic from this switch. Otherwise, it will incur significant service downtime because the program on switch hardware cannot be changed at runtime. My work enables reliable infrastructure management by building runtime systems for individual hardware and software devices and the entire network.

Runtime programmable hardware [2]. Today's networking hardware (e.g., switches, NICs) are only programmable at compile-time, but they effectively become fixed functions at runtime. Reprogramming a device incurs significant downtime due to the need to reflush the hardware. To avoid downtime, traffic rerouting is usually employed, which is intricate and impossible in many cases. My work FlexCore [2] solves this challenge by enabling runtime, in-place ASIC program reprogramming with zero downtime. It achieves this through a whole-stack design including hardware enhancement, reconfiguration primitives, and consistency guarantees. FlexCore enables a plethora of new opportunities such as just-in-time network function optimization ([3] in §3) and real-time attack mitigation. The novel hardware design has been implemented by Nvidia on their Spectrum switches and BlueField SmartNICs.

Resilient 5G software [14]. Next, I expand my research to 5G virtualized radio access networks (vRANs) at the far edge. One key missing feature in today's 5G vRANs is system resilience, i.e., maintenance and failover without long service disruptions. The rigorous real-time processing requirement of 5G precludes the use of existing resilience techniques like VM migration. My work Atlas [14] enables resilient 5G vRANs by repurposing existing cellular mechanisms (i.e., handovers and cell reselection) through a transparent runtime. For maintenance, it simultaneously serves both the old and new cells via the same radio by a novel radio sharing technique, and uses handovers between these cells to migrate user devices. For failover, it identifies and overcomes deficiencies in existing RAN protocols that disrupt cell reselection after failure.

Reliable network management [12]. Managing networks at scale is even more challenging. State-of-the-art network management systems implement workflows with operational steps in arbitrary scripts, posing substantial challenges to reliability. My ongoing work proposes a systematic design for reliable network management. It leverages the fact that modern network management systems are backed with a source-of-truth database which makes customized database techniques effective for network management. Thus, I expose a programming model for network operators to convey the key management logic, by which they are shielded from reliability concerns, such as operational conflicts, task atomicity, and failures. Instead, these concerns are handled by the runtime system using customized database techniques. Later, my colleagues and I have further expanded reliable management to the entire cloud infrastructure [13].

3 Performance clarity and optimization

The evolving hardware landscape offers a unique opportunity for accelerated data processing. Yet, the inherent hardware diversity, dynamic workloads, and obscured hardware details present significant challenges in maximizing their performance benefits. My approach to addressing these challenges is to achieve performance clarity through accurate, efficient modeling and profiling, which provides valuable insights for software optimization. I have applied this strategy to optimize SmartNIC offloading, both at compile time and runtime, as well as machine learning (ML) inference on GPUs.

Heterogeneous SmartNIC offloading insights [15]. SmartNICs are an important offloading platform for network function (NF) acceleration. They usually have heterogeneous architectures, which makes it challenging to understand the expected performance of offloaded NFs and determine the optimal offloading strategy. Today's developers perform time-consuming and error-prone manual analyses through cross-platform porting. To solve this problem, my colleagues and I have designed Clara [15], an automated tool that generates offloading insights for SmartNIC NFs. Clara analyzes an unported NF program, predicts its key performance parameters when offloaded to a SmartNIC, and suggests optimal porting strategies by augmented program analysis techniques with machine learning prediction.

Just-in-time SmartNIC optimization [3]. I have further extended SmartNIC optimization from compile-time to runtime, capitalizing on the hardware runtime programmability enabled by my FlexCore work [2]. The constantly varying workloads often render compile-time optimization ineffective in achieving desired performance across all scenarios. To address this challenge, I have developed Pipeleon [3], the first profiled-based, just-in-time performance optimization framework for SmartNICs. Pipeleon can adapt the implementation based on dynamic workloads, ensuring optimal performance in all scenarios. Pipeleon constructs a performance model and performs lightweight runtime profiling to identify hotspots in the program. It then computes runtime optimization plans to specialize the program layout based on the latest profile.

Hardware-aware ML interference optimization [17]. Finally, I expand my research to designing efficient ML systems. To achieve faster inference, auto-tuners (e.g., AutoTVM) are commonly used to search for efficient model implementations. However, today's auto-tuners search with opaque hardware details, so their performance could fall behind that of hardware-native libraries, which are hand-optimized by device vendors to extract high performance. Conversely, these vendor libraries lack automation support afforded by auto-tuners. My work, Bolt [17], achieves the best of both worlds using hardware-native template-based profiling and search, enabled by the emerging modularized and reconfigurable vendor libraries (e.g., CUTLASS). Bolt provides end-to-end optimizations at the graph, operator, and model levels. It significantly improves the inference speed and reduces the search time. Moreover, my other work improves system efficiency using ML-based congestion control [16].

4 Future work

Looking ahead, I am particularly excited about developing infrastructure support in several directions.

End-to-end runtime programmable networks. I plan to design an end-to-end runtime programmable network (FlexNet), vertically from host kernels to NICs, and horizontally extending across switches to the other end of the network, as an integral platform. Resources in the network are fungible across the stack, so functions can freely migrate from one location to another to minimize the overhead and achieve optimal performance. This will advance the programmable network to the next level, but it requires solving a host of new challenges. For example, today's network devices use different programming models/toolchains, making it hard to write network-wide programs running on heterogeneous devices. Moreover, there lacks a systematic approach for incrementally programming and compiling runtime changes. I plan to propose a unified programming model that allows incremental runtime modifications to the whole network in a modular manner. Additionally, the evolving role of the network requires a more intelligent network controller. Thus, I aim to design a new controller with a set of novel management APIs capable of managing all fungible resources and offloaded applications, and dynamically optimizing the network at runtime for better performance.

Disaggregated systems. I envision future data centers to be built with disaggregated resources (e.g., GPUs, SmartNICs, DRAM) interconnected by high-speed runtime programmable networks. Tasks will be partitioned and executed on suitable resources, leveraging network assistance for acceleration and security. This new paradigm promises enhanced data center scalability and cost efficiency, while also raising a range of new design challenges. The first research question is to identify the best way to disaggregate different types of resources. While full disaggregation could maximize resource utilization, it may hurt the performance of data-intensive applications like machine learning and database systems. I plan to explore a hybrid strategy for optimal balance, e.g., equipping GPUs with a small local DRAM as a cache and using the larger remote DRAM as a backup. I am also interested in designing new OSes, compilers, schedulers, and programming abstractions for disaggregated systems, e.g., compilers that can intelligently partition programs to execute on heterogeneous resources, and schedulers that can optimally schedule compute and I/O tasks to mitigate the increased latency of network-based resource access. Moreover, it creates new security challenges due to the dispersion of resources and the need for data transfer across networks, which increases the attack surface and makes conventional monolithic mechanisms inadequate. I plan to explore the possibilities of providing in-network security support by leveraging the opportunities offered by FlexNet.

5G security. Beyond data centers, I am also excited about 5G systems at the far edge. 5G features a distinctive system architecture and deployment approach. For instance, it disaggregates and virtualizes RAN functions, and introduces a new Near Realtime RAN Intelligence Controller (Near-RT-RIC). While enabling new features, this also significantly expands the attack surface. It is imperative to comprehensively evaluate and understand vulnerabilities across the entire 5G stack, considering its importance as national civil infrastructure. Recently, the 5G ecosystem has become open and standardized (e.g., O-RAN), making it more accessible to academia. I plan to start by looking at two aspects. First, the 5G fronthaul network between the Radio Unit (RU) and Distributed Unit (DU) presents unique security challenges. Due to the rigorous low latency requirements, integrity protections like MACSec are not currently employed, so attackers can easily explore integrity attacks. I plan to first understand the vulnerabilities and their impacts, and then devise defenses that can meet the stringent performance requirements [20]. Next, I plan to study the security of the Near-RT-RIC where a variety of xApps manage and optimize RAN functions. xApps that are malicious, misconfigured, or conflicted could disrupt the radio services or leak confidential data. I plan to enhance the controller security by taking inspiration from prior studies on SDN controller security, such as configuration verification and privilege isolation.

Overall, I am excited to continue working on diverse topics in systems and networking and their associated security challenges. I am also interested in providing system and security support for emerging large deep neural networks. In the future, I am eager to collaborate with colleagues in related fields, such as systems and networking, security, machine learning, computer architecture, and programming languages.

References

[1] **Jiarong Xing**, Wenqing Wu, and Ang Chen, "Ripple: A Programmable, Decentralized Link-Flooding Defense Against Adaptive Adversaries," in *USENIX Security*, 2021.

[2] **Jiarong Xing**, Kuo-Feng Hsu, Matty Kadosh, Alan Lo, Yonatan Piasetzky, Arvind Krishnamurthy, and Ang Chen, "Runtime Programmable Switches," in *USENIX NSDI*, 2022.

[3] **Jiarong Xing**, Yiming Qiu, Kuo-Feng Hsu, Songyuan Sui, Khalid Manaa, Omer Shabtai, Yonatan Piasetzky, Matty Kadosh, Arvind Krishnamurthy, T. S. Eugene Ng, and Ang Chen, "Unleashing SmartNIC Packet Processing Performance in P4," in *ACM SIGCOMM*, 2023.

[4] **Jiarong Xing**, Wenqing Wu, and Ang Chen, "Architecting Programmable Data Plane Defenses into the Network with FastFlex," in *ACM HotNets*, 2019.

[5] **Jiarong Xing**, Qiao Kang, and Ang Chen, "Mitigating Network Covert Channels while Preserving Performance," in *USENIX Security*, 2020.

[6] **Jiarong Xing**, Adam Morrison, and Ang Chen, "NetWarden: Mitigating Network Covert Channels without Performance Loss," in *USENIX HotCloud*, 2019.

[7] Hongyi Liu, **Jiarong Xing**, Yibo Huang, Danyang Zhuo, Srinivas Devadas, and Ang Chen, "Remote Direct Memory Introspection," in *USENIX Security*, 2023.

[8] **Jiarong Xing**, Kuo-Feng Hsu, Yiming Qiu, Ziyang Yang, Hongyi Liu, and Ang Chen, "Bedrock: Programmable Network Support for Secure RDMA Systems," in *USENIX Security*, 2022.

[9] **Jiarong Xing**, Ang Chen, and T.S. Eugene Ng, "Secure State Migration in the Data Plane," in *ACM SIGCOMM SPIN*, 2020.

[10] Qiao Kang, **Jiarong Xing**, Yiming Qiu, and Ang Chen, "Probabilistic Profiling of Stateful Data Planes for Adversarial Testing," in *ACM ASPLOS*, 2021.

[11] **Jiarong Xing**, Yiming Qiu, Kuo-Feng Hsu, Hongyi Liu, Matty Kadosh, Alan Lo, Aditya Akella, Thomas Anderson, Arvind Krishnamurthy, T. S. Eugene Ng, and Ang Chen, "A Vision for Runtime Programmable Networks," in *ACM HotNets*, 2021.

[12] **Jiarong Xing**, Kuo-Feng Hsu, Yiting Xia, Yan Cai, Yanping Li, Ying Zhang, and Ang Chen, "Reliable Network Management with a Principled Programming System,"

[13] Yiming Qiu, Patrick Tser Jern Kon, **Jiarong Xing**, Yibo Huang, Hongyi Liu, Xinyu Wang, Peng Huang, Mosharaf Chowdhury, and Ang Chen, "Simplifying Cloud Management with Cloudless Computing," in *ACM HotNets*, 2023.

[14] **Jiarong Xing**, Junzhi Gong, Xenofon Foukas, Anuj Kalia, Daehyeok Kim, and Manikanta Kotaru, "Enabling Resilience in Virtualized RANs with Atlas," in *ACM MobiCom*, 2023.

[15] Yiming Qiu, **Jiarong Xing**, Kuo-Feng Hsu, Qiao Kang, Ming Liu, Srinivas Narayana, and Ang Chen, "Automating SmartNIC Offloading Insights for Network Functions," in *ACM SOSP*, 2021.

[16] S P Sharan, Wenqing Zheng, Kuo-Feng Hsu, **Jiarong Xing**, Ang Chen, and Zhangyang Wang, "Symbolic Distillation for Learned TCP Congestion Control," in *NeurIPS*, 2022.

[17] **Jiarong Xing**, Leyuan Wang, Shang Zhang, Jack Chen, Ang Chen, and Yibo Zhu, "Bolt: Bridging the Gap between Auto-tuners and Hardware-native Performance," in *MLSys*, 2022.

[18] Yiming Qiu, Kuo-Feng Hsu, **Jiarong Xing**, and Ang Chen, "A Feasibility Study on Time-aware Monitoring with Commodity Switches," in *ACM SIGCOMM SPIN*, 2020.

[19] Qiao Kang, **Jiarong Xing**, and Ang Chen, "Automated Attack Discovery in Data Plane Systems," in *USENIX CSET*, 2019.

[20] **Jiarong Xing**, Sophia Yoo, Xenofon Foukas, Daehyeok Kim, and Michael K. Reiter, "Understanding the Security Vulnerabilities in Open 5G Fronthaul Networks,"