The Internet today carries a great volume and variety of data to enable new waves of innovation. Network data has grown to an unforeseen scale and is seemingly too complex to manage. In response to the increasing security, availability, and performance demands, many efforts envision an ambitious self-driving network such that large-scale, diverse applications can make real-time decisions. For decision-making, these applications usually need to perform a task called network telemetry to analyze what is happening in the underlying network.

Traditionally, network telemetry is done via coarse-grain packet sampling and offline packet analysis. To achieve a self-driving network, I argue that network telemetry needs to be a first-class citizen to enable real-time applications. Thus, network telemetry remains a challenging task: (1) applications are growing in diversity, (2) scaling to large networks is difficult, and (3) devices and platforms are evolving. For instance, since attackers are getting smarter and new attacks are appearing, telemetry tools need to look into different measurements in order to detect them. Network operators expect to gain fine-grained visibility from the data while traffic volume keeps growing. New platforms such as FPGA, P4, SmartNIC (programmable network interface card), and virtual switch are emerging as vantage points for network telemetry and introduce new challenges on adapting to various resource constraints and programming paradigms.

My research vision is to make network telemetry be “future-proof” by supporting a variety of (possibly unforeseen) applications and measurements, future-scale data processing, and diverse deployment platforms and workloads, in order to enable real-time, closed-loop networked applications.

To realize this vision, we need principled, provable approaches with efficient algorithmic tools to achieve system generality and to handle the increasing data complexity. My approach is to bridge system problems and theoretical approaches such as sketching, sampling, and graph theory in a way that is both practical and has formal guarantees. Thus, I try to gain insights into practical systems, algorithms, and real-world workloads. Accordingly, my work spans building future-proof network telemetry with its applications and a broader class of data analytics systems inspired by my approaches.

Research impact: My research has appeared in top venues of networking [SIGCOMM’16, SIGCOMM’19, SIGMETRICS’20] and systems [OSDI’18, FAST’19]. My work received interdisciplinary recognition, including USENIX FAST’19 Best Paper Award, USENIX ATC “Best-of-Rest”, ACM STOC “Best-of-Theory” plenary talk, and AT&T research Best Poster Award. I closely collaborate with industry to adopt my research into production. For instance, by collaborating with Intel, NitroSketch is in the process of up-streaming to Intel DPDK packet sampling library and sample applications. During my collaboration with Barefoot Networks, my colleagues and I built a prototype of Jaqen, a major security application for their programmable switch chips.

Future-Proof Network Telemetry

Foundations of Future-Proof Network Telemetry

Network telemetry is the foundation for a range of tasks including traffic engineering, attack and anomaly detection, and forensic analysis. At a high level, there are two classes of techniques to estimate these metrics of interest. The first class of approaches relies on some form of packet sampling, e.g., NetFlow or sFlow. While packet sampling is sufficient for coarse-grained visibility, prior work has shown that it provides low accuracy for more fine-grained metrics such as traffic entropy and distinct flows. These well-known limitations of sampling motivated a second class of techniques based on sketching algorithms (sketches): Given an input sequence of items, the algorithms make a single or constant number of passes over a data stream to approximate some statistics while using sub-linear (usually poly-logarithmic) space in the size of the dataset. Custom data structures here are designed for specific metrics of interest and can yield provable resource-accuracy trade-offs.

Ideally, we want a telemetry framework that offers generality by delaying the binding to specific metrics of interest but simultaneously provides the required fidelity and performance for estimating statistics on diverse platforms and workloads. My overarching vision is to address this challenge by forcing a fundamental trade-off between generality and fidelity and developing a future-proof framework that can simultaneously achieve high generality, fidelity, and performance across a broad spectrum of telemetry tasks. I believe that such a network telemetry framework will pave the way for future self-driving networks.

Universal telemetry system: UnivMon Monitoring (UnivMon) [1] is the first monitoring framework showing that both high generality and fidelity can be achieved across a broad spectrum of telemetry statistics, such as flow size distribution, heavy hitters, and entropy estimates. Our insight is to draw an exciting connection between
the system problem and the theory: UnivMon leverages and extends recent theoretical advances on universal streaming to build a general and memory-efficient “universal sketch” on the data plane to provably preserve a wide class of functions. From the universal sketch, users can query any statistics in the range without tight binding to one or two specific applications in advance. For instance, when anomalies happen, users can still conduct an in-depth diagnosis with UnivMon without the need of “guessing” what statistics in the network might be of interest before the fact. UnivMon was one of the pioneer telemetry systems prototyped in P4.

**Telemetry system in software platforms:** NitroSketch [2] is a robust and general sketch-based telemetry platform for high-speed software switches. Sketching algorithms used in network telemetry were initially designed with a goal to optimize the space complexity. Nevertheless, this overly simplified goal in theory does not cover all the system requirements: These algorithms are not running fast enough in software systems and cause significant packet losses. In this case, algorithm design goals need to reflect real system needs. Thus, I approached the problem from first principles: (i) Identify the system bottlenecks and understand that software switches have more severe requirements on packet processing than memory efficiency. We can relax the space complexity for packet processing speedup but should maintain the robust accuracy guarantees from original algorithms. (ii) I then introduce the idea of counter-based sampling to yield a better controllable balance between accuracy, performance, and memory usage for software platforms. Finally, NitroSketch can accelerate UnivMon and other sketches to 40Gbps line-rate under min-sized packets with negligible CPU overhead and is a promising option for future software switch with >40Gbps line-rates.

**Network performance monitoring:** Network performance issues are notoriously hard to diagnose. Prior performance monitoring tools require per-flow information and cannot scale to large network data. I design lean algorithms with sketches to 40Gbps line-rate under min-sized packets with negligible CPU overhead and is a promising option for future software switch with >40Gbps line-rates.

**Applications of Future-Proof Network Telemetry**

My research on network telemetry enables and amplifies a range of applications, such as distributed storage and network security systems. The provided timely and accurate telemetry information is the fundamental enabler for these large-scale systems to make real-time, flexible decisions.

**Distributed storage system:** DistCache [4] is a novel distributed caching mechanism that leverages heavy hitter telemetry information to provide provable load balancing for large-scale storage systems. For user satisfaction, the storage systems are expected to meet strict service-level objectives (SLOs), regardless of the workload distribution. A key challenge for scaling out to many clusters of storage servers is load balancing. Traditional ways of using consistent hashing or using a big fast cache node for all storage servers cannot scale to large dynamic workloads. Instead, DistCache shows that a “one big cache” can be achieved by an ensemble of small cache nodes, enabling practical load balancing on a datacenter scale. In DistCache, I leverage independent hash functions and heavy hitter information to timely cache the hottest items on a set of cache nodes. The cache allocation mechanism co-designs with multi-layer cache topology and query routing, and maintains the latest query loads from the network. Our prototype using programmable switches demonstrates that DistCache balances the loads of up to 4096 servers and opens up the potential to serve billions of queries per second.

**DDoS defense system:** Jaqen [5] is a performant and flexible DDoS defense solution on commodity programmable switches leveraging sketch-based detection. Volumetric Distributed Denial of Service (DDoS) attacks (e.g., UDP flood, SYF flood, and DNS amplification) remain a major threat to the Internet today. Existing DDoS defense systems using software-based virtual appliances or proprietary hardware have undesirable trade-offs between cost, performance, and flexibility. I realized a new opportunity to revisit the limitations of past solutions with emerging programmable switches. The key challenges on this opportunity is that switch has very limited hardware resources to guarantee line-rate packet processing. Thus, prior efforts on this domain were only providing ad-hoc solutions that have significant security issues on one or more dimensions: (i) low coverage for handling attacks, (ii) blind spots in tackling dynamic attacks, and (iii) slow responses in adapting to changing attacks. Instead, Jaqen addresses the security problems by broad-spectrum, in-band detection and on-demand, responsive mitigation. In Jaqen, the broad-spectrum detection relies on the fast and accurate network telemetry with UnivMon [1], and the best-practice mitigation mechanisms can be enabled in real-time for Tbps-level DDoS defenses.
Other Research on Data Analytics System

My research on network telemetry has developed a set of techniques including sketching and sampling algorithms to handle network data. In the same spirit, I believe that processing other types of data would leverage a similar methodology to achieve the desired efficiency and accuracy.

**Graph pattern mining system:** ASAP [6] is a fast, approximate computation engine for large-scale graph pattern mining. While there have been tremendous interests in processing graph-structured data, existing distributed graph processing systems take several minutes or even hours to mine simple patterns on graphs. The key bottleneck is that the pattern mining algorithms running on distributed systems generate a vast amount of intermediate data and require a number of shuffling between the servers. Inspired by the graph sketching and sampling approaches, my colleagues and I design ASAP with an extended graph sampling algorithm to optimize the informational value of the data on the shuffling phase, where only an important subset of intermediate data is transmitted. The experimental results show that ASAP outperforms existing exact pattern mining solutions by up to $77\times$ with just 5% loss of accuracy.

Future Directions

My plan is to design, implement, and evaluate systems that are future-proof for evolving applications and scenarios. I am excited about contributing to the ambitious goal of letting networks run themselves for innovative network control and applications. With the growth of data-driven systems, I'm also cautious about what ML can possibly go wrong (before it is too late). In the longer term, my collaborations in other areas such as ML, security, and privacy have started to draw my interests into system approaches to protect private user data in future ubiquitous AI-enabled world.

- **Closing the loop with automated, trustworthy telemetry.** In the future-proof network telemetry, any closed-loop systems should (1) take as input a high-level goal about performance, coverage, or fidelity, and automatically deploy to a heterogeneous network in an optimized way, and (2) output trustworthy results. However, existing telemetry tools do not fully meet these goals and pose significant challenges on users: (a) How to optimize the deployment of the telemetry capabilities on heterogeneous devices to meet user requirements? (b) How to guarantee the trustworthiness of the results under adversarial scenarios?

  Resolving (a) requires careful orchestration between the telemetry capabilities and the network. First, we need efficient network performance profiling. By introducing *lean* algorithms [3], my colleagues and I demonstrate the feasibility of using memory-efficient sketching algorithms to measure a set of performance statistics. Based on lean algorithms, an interesting question is how to build a universal data structure that can preserve a range of popular performance statistics. Second, given the network performance, I’m interested in understanding how to deploy telemetry capabilities to gain a holistic view from a range of emerging reconfigurable devices such as software switches, SmartNICs, and FPGAs. Since these devices have diverse computation power and programmabilities, it is important to explore an accurate and general profiling model for them. Finally, I plan to develop a robust network-wide optimizer to automatically decide the deployment of telemetry algorithms for maximum coverage and minimum resource and performance overhead.

  For (b), I propose that network telemetry results need to be secure and trusted when an adversary is present. Unfortunately, existing telemetry tools are far away from being trustworthy. For instance, if the virtual machines and virtual switches in a datacenter network are compromised, the adversary can manipulate the measurement results in the switches by forging, delaying, or dropping the packets and counters. To address this security issue, I plan to explore various secure execution environments (e.g., Intel SGX and ARM TrustedZone) with UnivMon [1] and NitroSketch [2] to ensure that users can obtain reliable measurements about their network.

- **Making data analytics energy-efficient.** Wireless sensors, mobile phones, and wearable devices are just a few examples of distributed lower-power networks generating a wealth of data. The limited energy and communication capabilities in these devices make it inefficient or even infeasible to transmit sizable data to conduct real-time global analytics and control using ML. To achieve energy-efficient data analytics, we have several lessons learned: (1) Situational awareness is key to efficient systems. (2) Approximation techniques can make a useful trade-off between space and accuracy. (3) Hardware-based optimization is a moving target. I believe the resource-efficient algorithms designed for high-energy, datacenter networks (e.g., [1, 2]) can also be very useful in low-power devices. I plan to further explore a low-cost and general methodology without per-hardware optimization and with a (universal) range of preserved features that global analytical tasks may be
interested, i.e., support various feature combinations in multiple spatiotemporal aggregations such as locations, IPs, and user sessions in health data. If successful, energy cost (e.g., radio communication) can be reduced by several orders of magnitude such that the battery life of these devices can be significantly enhanced. I expect that the reduced energy cost will enable more low-power devices to participate in large-scale data analytics in a timely fashion.

- **Self-evolving, trusted, privacy-preserving learning at the edge.** ML has been successfully adopted in a variety of human-facing scenarios at the network edge, such as smart-home intrusion detection, speech recognition, and health monitoring. Nevertheless, existing ML inference and training tasks performed at the edge (e.g., in mobile devices, base stations, and edge clusters) have key security and privacy limitations: (1) Untrusted model inference. Pre-trained or local learnable ML models deployed at edge devices are responsible for many security-critical services, but they cannot be trusted when an adversary is present. For example, in a smart home, we want to ensure the deployed intrusion system can correctly infer a break-in incident via collected video, sound, and sensor data, and faithfully execute the emergency response plans without interruptions. In the presence of an evasion attacker, existing edge devices are unable to achieve this goal. (2) Privacy issues in learning. Privacy is a major concern in distributed learning. Even though federated learning has emerged as a promising tool to train statistical models at the edge without sharing raw user data, local model updates during the training are still readable by a third-party (e.g., a remote server) and leak private information.

**Trusted, performant model inference:** To resolve (1), I plan to explore the usage of trusted execution environments in ARM-based processors to secure the model inference. To optimize the inference and program execution speed, I am interested in extending the model compression techniques with various hardware accelerators (e.g., GPU) to secure enclaves in order to achieve trustworthy and performant model computation.

**Self-evolving, privacy-preserving distributed learning:** Ideally, we want to perform distributed machine learning tasks in a way that is both privacy-preserving and communication-efficient. My insight is that privacy preservation and communication reduction are naturally related. Both tasks share a similar aim to reduce, mask, or transform information that is shared across the network in a way that preserves the underlying properties of the data. As a way of “compressing” input data with randomness, sketching algorithms or other approximation techniques are promising options for jointly optimizing communication and privacy.

In this direction, I have started to collaborate with ML faculty on federated learning [7]. My overarching goal is to explore a closed-loop distributed learning system that is self-evolving with three major components: (a) Configurable Privacy: a communication channel to transmit learning data with configurable privacy guarantee and communication cost. (b) User-defined Detector: a telemetry system to detect privacy leakage based on user-defined privacy requirements. (c) Self-evolving Configuration: a learning-based configuration tool (e.g., reinforcement learner) to optimize the configurations for the next round of data transmission in order to fix the privacy leaks. I believe building this system will require interdisciplinary background and collaborative efforts from the communities of systems, ML, theory, security, and privacy. I will continue collaborating with researchers in these fields to fulfill this vision.

**REFERENCES**


