Workshop on Self-Driving Networks — Workshop Report

Nick Feamster, *Princeton University*
Jennifer Rexford, *Princeton University*

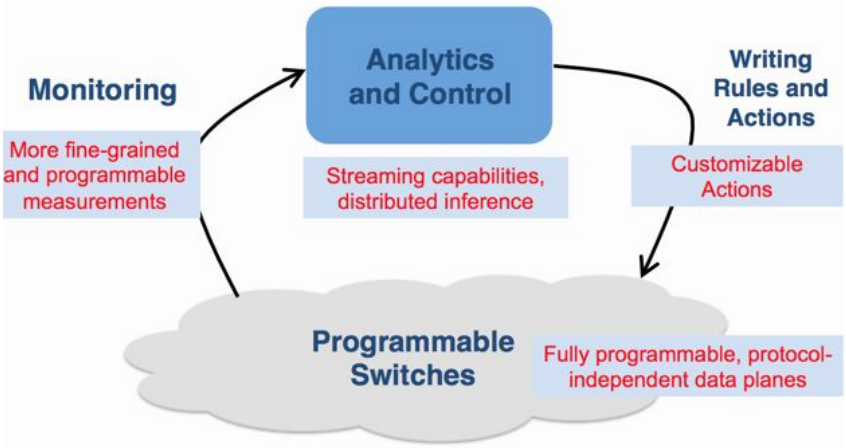
Table of Contents

Motivation

Due to the proliferation of applications and services that run over the Internet—ranging from video streaming to Internet-connected smart-home devices to augmented reality—the expectations for the performance, reliability, and security of our communications networks are greater than ever. To meet these expectations, network operators work tirelessly to: (1) continuously collect troves of heterogeneous data from the network; (2) analyze this data to infer characteristics about the network; and (3) decide whether and how to adapt the network's configuration in response to changing network conditions (e.g., a shift in traffic demand, an attack). Today, these three steps are decoupled: operators perform them separately, on different timescales, often in a slow or manual fashion that relies on intuition, as opposed to data, analysis, inference, and optimization.

As the Internet has evolved and matured over the past 40 years, this mode of operation has remained largely unchanged. The tools that network operators use to gather data from the network have not changed appreciably in decades, even as both demands on the network and traffic volumes have increased. If computer networks are to achieve high performance, reliability, and security going forward, the research community must rethink how networks are managed, from the ground up. Specifically, new security and performance requirements create a growing need for new approaches to real-time network management that exploit the growing capabilities in programmable networks and systems that support the analysis of real-time streaming data. Although there is much previous work on algorithmic support for streaming data, the set of queries that network management requires is far more rich than can be supported by these methods alone.

Our vision is that networks might one day be able to largely manage themselves, through a combination of query-driven network measurement, automated inference techniques, and programmatic control. The figure below roughly summarizes this vision.



We briefly expand on the three classes of capabilities shown above:

- *Query-driven network measurement.* Today's network measurement facilities (e.g., packet capture, IPFIX) are often either too fine-grained or too coarse for a particular network operations task. For example, packet timing information that could help infer characteristics of a gaming application of a video stream is readily available from packet capture, but widespread raw packet capture is prohibitively expensive for most large networks, given today's traffic volumes. On the other hand, measurements such as IPFIX records are too coarse to provide information on packet timing. Instead, we envision technology where the data that the network devices collect is driven by the queries that network operators express through a common, familiar abstraction, such as MapReduce.
- *Automated inference techniques.* In many scenarios, network-management tasks range from trial-and-error to a black art. Twenty years ago, it was possible to derive analytic, closed-form solutions for the performance of an application, because the protocols and systems could be cleanly modeled. Today, many applications and systems are too complicated to analyze in closed-form. Prediction problems such as determining how search query response time would vary in response to the placement of a cache cannot be analyzed with closed-form models, and are much more well-suited to statistical inference and machine learning based on models learned from data. The past ten years has demonstrated significant promise in using machine learning to both detect and predict network attacks. The next steps should be to build on the increasing amount of work in automated inference in network management, to integrate it into a control loop that can enable more automated decision-making.
- *Programmatic control.* The last ten years has seen a paradigm shift in how networks are controlled, with the advent of Software-Defined Networking (SDN). Entire networks can now be controlled from a single, high-level control program that is written in a high-level language—often, the control program can even have provable correctness properties, or respond automatically to changing events in the network. While the capabilities for such control exist, they have not (yet) been coupled with measurement and inference to enable more automated decision-making with respect to network management. We believe that this is a logical and important next step.

Application pull. A wide range of network-management applications, including network security, application performance troubleshooting, and network provisioning demand better network monitoring capabilities and technologies. Existing network monitoring technologies (e.g., packet capture, IPFIX/Netflow) are often not well-suited to the corresponding network management tasks, providing measurement data that is too coarse or too fine for the task at hand. Network-management applications need measurement technologies that are driven by queries and applications themselves. Software control and programmable network data planes can ensure that the only data that is collected is that which addresses a specific query or task. Software control can also help close the control loop to help better automate network management tasks such as traffic

engineering or quarantining, in response to changes in network conditions (e.g., traffic load shifts, security events) that the network measurement and inference observes.

Technology push. Significant technology trends over the past few years have created the opportunity for advances:

- Fully, programmable, protocol-independent data planes, and mechanisms to program them (e.g., P4)
- More fine-grained, programmable network measurements (e.g., in-band network telemetry)
- Scale-out, distributed streaming capabilities (e.g., Spark, Storm)
- Customizable actions in the forwarding plane (e.g., with P4)
- Software and systems that support inference and prediction over large datasets
- The broad availability of statistical analysis and machine learning tools and software (e.g., MLLib, SciKitLearn) that can be readily applied to streaming data

The rest of this report summarizes discussions that were held at Princeton University on February 15–16, 2018. The report appendix includes a summary of workshop attendees and agenda.

Use Cases

In this section, we explore two different motivating use cases for self-driving Networks. The first involves automatic inference of application quality from network traffic, and possible mechanisms for network response to quality inference. The second use case involves automatic detection and remediation of a wide range of security related events.

Application Quality of Experience (QoE)

Networks must continually adapt to provide good application performance in the face of changing network conditions, ranging from congestion to failures.

- *Network adaptation* may include taking simple actions from re-routing around congestion or failure to more complex actions such as initiating redundant in-network transmissions (e.g., duplication, network coding) or sending explicit signals or feedback to applications (e.g., indicating to a video application that re-encoding may be necessary due to a persistent congestion event).

- *Application adaptation* may include taking specific steps, such as adding redundancy or adjusting the bitrate of a transmission in response to changing network conditions.

A network operator may want the network and application to adapt to specific objectives. They might want the network to adapt automatically, and they may also want to help an application provider determine whether the adaptation that the application is performing is improving the performance of the application. Application providers have models for quality of experience (QoE), and they make assessments of the effectiveness of their adaptations based on these metrics. Currently, network operators do not have visibility into application QoE. The network needs better models for measuring network metrics and inferring quality of experience, but if the application provider could provide these metrics to the network directly, the network could optimize for these metrics—perhaps automatically.

More generally, many applications can provide a model of how they define goodness and utility, but there is no standard interface for exposing these metrics to the network layer. With better knowledge of these metrics, the transport layer could offer additional functionality beyond TCP to help deliver high-quality video traffic to end users.

Such optimization depends on having the appropriate models for user utility. Many of the existing models provide some quality score, given a sequence of video frames, but they do not directly take into account network effects such as rebuffering or join time. Historically, these metrics have focused on the effects of lossy encoding and compression on video quality. The next generation of models should take into account

network effects, such as rebuffering, as well. One challenge involves combining existing metrics to devise one single QoE "score". Some application providers have people watch different sequences of frames and rate and perform qualitative analysis of video quality; many application providers use a metric known as Video Multi-Assessment Fusion (VMAF).

Inferring these types of video QoE metrics naturally require gathering raw measurements from the network. An open challenge involves gathering application-specific metrics from the video stream, such as frame rate, using network hardware, likely with cooperation from the application. One possibility is for the application to define a few quality or performance metrics that it could communicate back to enable the network to optimize performance. Gathering measurements from multiple vantage points—both at the edge of the network and along an end-to-end path—presents new challenges, particularly if these distributed measurements are needed to drive real-time inference.

When end-to-end paths traverse multiple domains and independently operated networks, coordinating these measurements becomes even more challenging. Doing so, however, is important. For example, suppose that the network attempts to assert some level of control over a video-streaming application (e.g., throttling, rate limiting). In such cases, the application may react by retrieving content from an alternate content distribution network (CDN), or it may simultaneously retrieve video segments from multiple CDNs. More generally, applications may simultaneously react to network adaptations, amplifying these effects. The interactions between the network and adaptive video streaming applications may thus result in conflicting control loops.

In some cases, the application may benefit from having better information about activities taking place in the network. For example, a video streaming client may typically start streaming at a low bitrate and gradually increase the bitrate it asks for according to network conditions. However, network caching can create inversions whereby cached versions of the video at a higher bitrate exist in caches closer to the client than the streaming server. In such situations, it can make sense to *increase* the bitrate of the streamed video to reduce the load on the network. Networked systems must thus incorporate information that not only includes network traffic load but also auxiliary information such as the location of video content.

Research Challenges. Video QoE poses a variety of important research questions relevant to self-driving networks. Specifically, there are questions involving inference of video quality from passive traffic monitoring, as well as learning the appropriate action to take in response to different network scenarios. Open questions include:

- What is the relationship of network utilization to video quality of experience, and application quality of experience more generally?
- How accurately can application quality of experience (QoE) be diagnosed from passive network traffic monitoring? What features are most useful in diagnosing and predicting application QoE, for different applications? Can such inference be performed at high traffic rates? What techniques are applicable at different points along the end-to-end network path (e.g., in the home network, in the access network, at interconnection points, at the server)?
- In the event of degraded network conditions (e.g., congestion), to what extent should adaptation entail *application* adaptation (e.g., changes in video bitrate quality) vs. *network* adaptation (e.g., selection of an alternate route between the content and the user)?
- When network (re)action is required, how should reactive approaches be specified? For example, should network changes be automatically determined from optimization? To what extent should the operator be in the loop when executing these changes (both on longer, planning timescales and on shorter, operational timescales)?

Security

Network security is another area of network operations that could benefit from self-driving network functionality. There is an abundance of research that applies data-driven analysis and machine learning to

detection of network attacks. Examples of applications of machine learning to network security problems include:

- Detection of spam email from network traffic patterns
- Detection of bulletproof hosting sites from changes to interdomain connectivity
- Detection of botnets from coordinated DNS lookup activity
- Prediction of phishing attacks from DNS domain registration
- Prediction of denial-of-service attacks from analysis of social media

Despite continual and rapid advances in data-driven analysis and detection in network security, existing work largely suggests the *potential* of these techniques to be applied in operational settings, as they operate on offline network traces. Training often requires large quantities of data and is often difficult to perform in real time. Similarly, classification and prediction of network attacks often requires data collection and analysis that are expensive and complex—sometimes too complex for the underlying network hardware to collect, especially in real time.

The tradeoff between privacy and data-driven security remains a concern, as well. Specifically, many data-driven classification and prediction problems require collection of network traffic, some of which may contain information that puts user privacy at risk. DNS queries and responses are a quintessential example of this tradeoff: On the one hand, unusual DNS lookups can indicate anomalous behavior, such as a compromised device, a network attack, or exfiltration of private data to untrusted third parties. On the other hand, DNS queries also reveal significant information about user behavior, such as the devices that a user owns, the websites that a user visits, and so forth.

In light of this tradeoff, automated detection algorithms should practice data minimization, where only the data that is necessary to perform classification or prediction is collected. Implementing such a standard is, of course, complicated, because it requires determining the importance of individual network features for accurately detecting or predicting an attack. Thus, an important research challenge involves analyzing the effectiveness of various network-level features in detecting and predicting attacks, as well as both the *privacy cost* and *performance cost* of collecting these features. Optimization in machine learning is a longstanding problem area, but it has not been considered in the context of privacy and network performance. Further, multi-objective optimizations based on both privacy and performance present new challenges.

Additionally, existing research has typically not explored how the outcomes of various prediction algorithms could be used to drive network decision-making. Decisions about actions to take in light of a detected attack could include blocking, throttling, and quarantining. How the network responds to different types of attacks may depend on a variety of circumstances. Determining how to encode this decision process, as well as how various aspects of this process should be automated, will be an important research challenge.

Research Challenges. Most previous applications of machine learning to network security involve offline analysis. A major step forward towards self-driving networks will involve "closing the loop", through both (1) enabling real-time detection; and (2) using the outcomes of detection and prediction to better automate decision-making processes. Specific research challenges include:

- Developing cost-sensitive machine learning algorithms that capture only the features necessary to achieve prediction or detection with a certain level of accuracy.
- Developing new machine-learning techniques that can detect network attacks while optimizing for cost and privacy.
- Re-examining many of the existing offline prediction and detection algorithms to determine which are well-suited for deployment in network hardware and real-time analysis.
- Exploring how a broader range of signals (e.g., posts to social media) might ultimately allow operators to predict attacks before they occur.

- Designing a system whereby a provider could provide service-level agreements for security, and whereby customers could enforce them. What would an "SLA for security" even look like? What types of features would be specified in such an SLA, for example?

Closing the Control Loop

One important aspect of self-driving networks is closing the control loop that we described earlier. This broader research challenge entails several smaller subproblems: (1) Determining what aspects of network management should be automated; (2) Specifying objectives to satisfy or optimize, actions to automate, and when they should be automated; and (3) striking a balance between sophisticated, precise, accurate prediction and the limitations that arise due to the need to run at high speed.

Deciding how to automate. A wide variety of interesting research questions involve exploring the role different machine-learning techniques can play in enabling automation:

- Training a machine-learning algorithm on historical data (e.g., of cyber attacks or performance problems) can enable better detection techniques to use in the live network.
- Reinforcement learning can drive decision-making in complex settings where it is hard (if not impossible) to formulate or solve an optimization problem.
- Natural language processing techniques can be used to analyze (say) Twitter data or Web blog postings as an early warning about performance and security problems facing the network (KarDo).
- Learning techniques can drive protocols that automatically adapt to changing network conditions (e.g., PCC for congestion control).
- Learning techniques can help in building more scalable systems that try to mimic the behavior of more fine-grained solutions that require more detailed analysis of the data (e.g., inferring classifiers for IDS-like functionality on more limited data (e.g., Netflow, P4-compatible telemetry).
- Active learning can drive adaptive decisions about data collection, by weighing the cost of collecting additional data with the extra accuracy that the data would bring to the inference process.
- Machine learning can help uncover the relationship between lower-level network metrics (e.g., loss, delay, throughput) and quality-of-experience metrics (e.g., mean opinion score for video streams).

Deciding what to automate. One challenge in closing the control loop for self-driving networks is determining which aspects of network management can be automated. Ideally, an operator may want to automate any aspect of the network that could be automated, effectively removing the human from the loop of low-level decision-making tasks. It is reasonable to expect that there will always be a human in the loop to *some* extent, and the most interesting research question may thus be not whether the network should be fully automated or not, but rather what role a human should play in a self-driving network, and where to draw the "line" between automation and human intervention for various tasks. One extreme outcome is not that far from the scenario in today's networks, where operators are overwhelmed with alerts; the other extreme outcome is complete automation, with no humans ever taking action in response to network events. The realistic and reasonable design points likely lie at different points along this spectrum, for various tasks. In the case of network performance, some traffic engineering and management tasks may be fully automatable; in other cases (e.g., in security incidents where the decision may involve quarantine or disconnection), a human may need to be in the loop.

Specifying objectives, measurements, and actions. A self-driving network needs to know what to optimize. Thus, an important facet of a self-driving network involves developing ways to specify objectives (e.g., objective functions, minimum performance guarantees, detection thresholds), as well as the actions that the network should take in various circumstances. Such decision-making processes also require having up-to-date data about the network; in these cases, an operator may need to specify what information to measure, from where, to enable the appropriate automated decisions. In the case of satisfying particular network objectives, the network may need to expose to the operator an interface that allows the operator to specify (for example) the service-level agreements or performance constraints that the network should try to meet.

The role of automation vs. the role of humans. Self-driving networks move the human operator out of the real-time "control loop" for measuring, analyzing, and controlling the network. Self-driving networks would still engage with human experts, albeit on a longer timescale. The network administrators would still write the specifications of higher-level policies, such as deciding which performance of security metrics to optimize for particular traffic flows or applications. Also, the software for self-driving networks would (presumably) be written by human programmers, and these programmers may be the ones most equipped to reason about the system when it doesn't behave as expected. Plus, human administrators play an important role in identifying new features that might be helpful to the system (e.g., realizing that the temperature sensors in a point-of-presence may be helpful in predicting equipment failures, or noting that Twitter feeds give a good indication of customer dissatisfaction with the network). Finally, the human administrators would play an important role in created labeled data for training the machine learning algorithms. So, while network operators would get "out of the loop", a group of network experts would still provide the "outer loop" of configuring, training, and troubleshooting the self-driving networks.

Ensuring that operators can interpret automated decisions. To support these human operators, future self-driving networks need to offer better ways for these operators to understand how the network is behaving, and why. This includes better visualization techniques to give a more intuitive, high-level understanding of the network. Interpretable machine-learning algorithms will be important for self-driving networks, both to build confidence in moving to greater automation and to allow network administrators to recognize whether (and why) the system is behaving differently than expected.

Understanding performance tradeoffs. Prediction and classification accuracy generally improve as machine-learning algorithms have access to more data and features. Naturally, there is a tradeoff between the desire for improved accuracy and the cost of gathering these additional features. For example, in some cases, it may be helpful to get detailed information from packet payloads, or to see both directions of traffic in a traffic flow, which can be problematic given that many Internet routes are asymmetric. Additional limitations in gathering certain features may arise due to the cost of capturing traffic at high speed, the presence of a large number of devices or flows, and so forth. An important research area will involve understanding the nature of the tradeoffs between the accuracy of a model and the cost and complexity of gathering various features to support accurate decision-making.

The role of encryption. Network management commonly faces a tension between the need to gather data concerning network operations or service delivery (e.g., security, application performance) and the increasing pervasiveness of end-to-end encryption, which makes it more difficult for the network to infer certain properties concerning security or application performance. One specific example relates to the original video QoE case study: HTTPS encryption prevents any on-network device from seeing the contents of a client's requests for video segments, which makes it more difficult to infer application performance metrics such as the video resolution or bitrate. New algorithms will thus be needed—for a wide range of applications—to help the network determine when application quality has degraded to a point where the network itself should take steps to improve application performance.

Moving Beyond a Single Protocol and Administrative Domain

Most practical network-management problems go beyond a single protocol, network, or institution, leading to interesting challenges in creating a complete, coherent system.

Multiple layers. Networks often use multiple protocols or mechanisms simultaneously, leading to interacting control loops. For example, *congestion control* determines the sending rate for each transport connection (based on observations of packet loss and delay), and *traffic engineering* determines the paths between endpoints (based on the network topology and the offered load). Designing and operating each protocol layer in isolation can lead to unexpected interactions, because each protocol may "learn" the wrong information from its past observations of the network. For example, traffic-engineering decisions based on recent load may lead to more efficient paths with lower link utilizations, only to lead the congestion-control algorithm need to (re)learn the path-level conditions and ultimately send traffic at a higher rate, only to lead the traffic-engineering algorithm to need to (re)learn the offered load. New research is needed to understand how to design control loops that interact well to achieve a larger goal, by selecting the right combinations of "features" for the protocols to measure, the right timescales for them to adapt, and lightweight ways for them to share information.

Multiple domains. Large organizations often divide their networks into multiple domains, to achieve better scalability and to deploy different kinds of solutions in different parts of the network. For example, a large cloud provider might have multiple data centers (each with their own network configuration) interconnected by a wide-area backbone network. Similarly, an Internet service provider may divide its backbone network into multiple regions that each form an OSPF area or BGP autonomous system. In the future, each of these constituent networks may be "self driving", but they need to work effectively together as a single end-to-end system. In some cases, finding the best way to divide the network into parts is itself a challenge. Today, network operators must identify good ways to configure protocols to aggregate or hide information (e.g., configuring BGP route filters, OSPF area boundaries, etc.) to strike a good balance between scalability and efficiency, e.g., to use reasonable computation, bandwidth, and storage resources to compute routes, while still selecting short paths. Future self-driving networks should learn how to make these trade-offs automatically.

Multiple institutions. Networks often consist of multiple organizations that must coordinate at domain boundaries (e.g., interdomain routing) or share information (e.g., about attack signatures). These institutions may be part of a competitive cooperation, where they must cooperate with each other to reach other parts of the Internet while simultaneously competing with each other for customers. As a result, these institutions may not trust each other to participate honestly in protocols or share accurate information about their parts of the network. Yet, cooperating can be mutually beneficial, e.g., by enabling all participating institutions to detect cyber attacks faster and block them closer to the senders. A self-driving internetwork needs standard protocols for sharing data, as well as models that capture the incentives for the participants to share data honestly.

Multiple systems. Often, the network is just one part of a large information technology infrastructure that includes computation and storage resources—as in data centers, for example, where a cloud provider often owns and manages the servers, the storage systems, and the network. Even when different institutions manage different parts of the infrastructure, coordinating across these components can lead to greater performance, efficiency, and security. A focus on the entire system broadens the set of possible control actions. For example, a data center could alleviate network congestion by migrating a virtual machine from one location to another, or make better routing decisions by understanding which traffic flows are part of the same higher-level application service. Similarly, a self-driving system can naturally focus on end-to-end metrics, such as application performance (e.g., job completion time), rather than lower-level network metrics (e.g., packet loss and delay), to address the real concerns of end users.

The Ethics and Policy of Automated Decision Making

In addition to the technical challenges, self-driving networks also raise new and interesting questions at the intersection of technology, law, policy, privacy, and ethics.

Legal and Policy Challenges:

- *Contracts/SLAs.* What should SLAs and other contractual relationships between networks look like in the context of self-driving networks? The ability to make complex inference could enable more complex contracts, but they could also make contracts more difficult to enforce and validate.
- *Explainability of algorithms.* Explainable algorithms are naturally important to help operators debug their networks, but they are equally important for ensuring that a network conforms to regulatory guidelines. Take, for example, a restriction that would prohibit paid prioritization. If traffic prioritization is explicitly enabled through configuration, then ensuring that it does not take place on a given link or path may be more straightforward than if prioritization and routing decisions are being made in part by an inscrutable decision-making process. Even with the repeal of the Open Internet Order, various transparency requirements remain in place. From a policy perspective, it is worth considering whether transparency is even possible if operators cannot understand how various network management decisions are being made. In this vein, policies and laws may need to be cognizant of trends in automation to ensure that transparency requirements can be satisfied.
- *Fairness.* "Fair" resource allocation has a long history in the context of transport protocols (e.g., TCP fairness). In the context of application quality, new definitions of fairness may be necessary or appropriate, as fair sharing of throughput or bandwidth may not be necessary or sufficient to guarantee an equitable allocation of resources, either across applications or across subscribers. Given new constructs for fairness, it may then behoove us to ask whether a particular automated decision-making algorithm results in a "fair" outcome, akin to how previous generations of research studied the fairness behavior of transport protocols.
- *Transparency.* self-driving networks may result in situations where the network operates differently for different users, neighboring networks, or regions. Today, researchers gain insights into the network's behavior by measuring it from the edge of the network. For example, a significant amount of research has shed light on prioritization and blocking practices of Internet service providers by measuring the behavior of the network from the edge. Yet, if automation results in a situation where the network behaves differently for each user, then even simple questions like testing for traffic differentiation may become exceedingly difficult.
- *Accountability when interdomain routing decisions start to interact with one another.* Even seemingly "simple" interdomain routing protocols like the Border Gateway Protocol (BGP) are notoriously difficult to control and understand when multiple independent networks interact. Imagine, then, a network when each network is making an autonomous decision about a variety of network management practices (including, but not limited to, routing), where the decisions in one network depend on observations of behavior in neighboring networks. In such an environment, where two (or more) closed loop systems interact and each system is by itself difficult to understand, the entire system may become very difficult to understand or explain, let alone stabilize. The potential for instability and cascading failures is high.
- *Resilience of the infrastructure.* The potential for cascading failures raises the need for "stop gap" protective measures—or even kill-switch types of mechanisms—that might prevent the types of cascading failures that we see in other automated decision-making environments (e.g., algorithmic trading). There may be an important role for humans to play in ensuring stability and reducing the likelihood of cascading failures.
- *National borders and automated routing decisions.* The paths that data takes through the network—and where data resides when it is "at rest"—is gaining increasing attention with the rise in data requests from domestic and foreign governments. self-driving networks could be either a threat or solution in this context. On the one hand, automatic optimization may result in network management decisions that may run counter to political, social, economic, or national security goals (e.g., routing traffic through a certain country, or not). On the other hand, with the appropriate

auxiliary data, such information could be incorporated into an automated decision-making process as part of an optimization, potentially giving operators *more* control over how data is routed than they have today.

Privacy Challenges:

- The rise of programmable networks raises the possibility to collecting information that is more tailored to a specific network management problem, rather than collecting packet traces wholesale In this regard, designing new ways to collect network traffic in privacy-preserving ways may align well with the need to perform certain types of collection and analysis at high traffic rates and volumes.
- Some privacy policies and data collection disclosures state both data that is being collected *and* the purpose for collecting that data. If privacy disclosures included the purpose of collection, it might be possible to design the measurement infrastructure for a self-driving network to more finely tailor collection of data to the purpose for its collection?
- Once machine learning is integrated into the network, the networks themselves can essentially perform inference, label data, and link one dataset to another. It may become increasingly easy for a broader set of parties to build profiles about users based on the data that it will soon become possible to collect on a large scale, and the sophisticated inference that may be possible from this data.
- New laws such as the General Data Protection Requirement (GDPR) in Europe require the anonymization of vast amounts of data, including IP addresses. When such data is used for decision making but cannot be stored, it may become impossible to replicate (or understand) how the network made past decisions. How can we reconcile the need for privacy and minimization of data retention with the potential requirement that a network may need to explain how a past decision was made?
- Does automated decision-making (and the data collection that supports it) imply any new considerations for user consent? Along these lines, how does an ISP write a meaningful privacy policy when collection decisions are automated, and one doesn't totally understand the conditions under which a certain dataset might get collected? Can that even be explained to a user?

Ethics Challenges:

- *Automated security decisions creating unintended side effects in cyberphysical systems.* self-driving networks vastly increase the potential for unintended consequences and side effects. For example, a network may make an automated decision to firewall an IoT device in a "smart home" due to a perceived attack, but doing so might cause the device to malfunction, potentially putting the occupants of the home at physical risk or inconvenience.
- *Collateral damage due to automated filtering.* There is an increasing call for networks to automatically moderate the content and speech that appears on their networks and platforms. Yet, language and norms shift over time. Automated filtering may thus censor or block content, traffic, or speech that does not conform to a past model of "normal" traffic. A classic example is a spam filter that has an unacceptably high false positive rate that results in filtered emails. Such a problem is likely to get a lot larger as more decisions about allowing or blocking traffic flows are based on automated decisions, rather than fixed rules.
- *Prioritization and disenfranchisement.* When the network starts learning information including (1) which subscribers are doing mission-critical applications; (2) which subscribers complain more, there may be a tendency for networks to optimize business processes to maximize profits, thus giving some users better service than others based on various features. There is a strong possibility that if prioritization decisions are automated, some sets of users and subscribers may be marginalized, likely even without the operator's awareness.
- *Unknown effects on the network edge.* More generally, the effects of control decisions in the middle of the network may not have always obvious impacts on end hosts, which may make reasoning about ethics difficult in some circumstances, if the effects of decisions are not known.
- *Kill switches.* In the event that the network completely spins out of control (e.g., due to attack, instability, failure), would there ever be a need for a "kill switch"? If so, what should such a kill switch do? Should it return us to an operating mode of "today's Internet", or should it result in some other

behavior? Who should have the authority to make a decision to exercise a "kill switch"?

Infrastructure/Community Support

To propel the field of self-driving networks forward, researchers need concrete use cases, real datasets, and platforms for evaluating their ideas.

Access to operational networks: Working closely with network administrators provides a great way for researchers to learn more about practical challenges, acquire much-needed data to evaluate their ideas, and have a "tech transfer" path for their solutions. Collaborations with large cloud companies (e.g., Google, Microsoft, and Facebook) and carriers (e.g., AT&T and Comcast) are relatively common in the computer networking research community, but even there we have a sizable gap between the "haves" and the "have nots". Perhaps more importantly, the community often lacks the broader perspective that could come from interactions with a wider range of institutions (e.g., hospitals, schools, libraries, homes, factories, etc.). Engaging more broadly with networking practitioners across a range of environments can help the community avoid "blind spots" while also creating new opportunities for broad societal impact. To help foster these connections, funding agencies like the NSF could offer opportunities for researchers to embed with network administrators (in a program like I-Corp, but focused on network operators rather than entrepreneurship) or attend network operator conferences (e.g., LISA, NANOG, and M3AAWG).

Fostering open platforms: The creation of open interfaces and open-source software can allow researchers to build on top of each other's work.. In the area of Software-Defined Networking (SDN), we have seen significant progress through open interfaces (e.g., OpenFlow and, more recently, P4) and software (e.g., Mininet, various SDN controller platforms, and more). In self-driving networks, too, we should create incentives for researchers to make their systems available to others, through awards for open-source contributions, funding programs dedicated to creating reusable software and data, and so on. Also, government agencies can partner with industry to lower the barriers to getting state-of-the-art commercial platforms (e.g., programmable switches, big-data systems, and machine-learning libraries) in the hands of academic researchers—or even deployed on their campuses.

Creating realistic data and evaluation scenarios: Network control and management are elusive topics for researchers, due to the paucity of public data about the structure and behavior of operational networks. The community needs to think creatively about how to acquire (labelled) data from real networks. For example, crowdsourcing may be an option, where network operators are asked, and paid, to generate example device configurations that realize a specified high-level policy. Similarly, black-hat practitioners could be enlisted to attack prototype self-driving networks to test the effectiveness of techniques for detecting and stopping cyber attacks. In addition, open competitions, like the DARPA grand challenge for autonomous vehicles, can propel the field forward by drawing attention to important topics and allowing multiple teams to demonstrate the progress they have made. Having events where multiple solutions for self-driving networks compete for prizes could help motivate students and draw attention to the technical advances that work well in practice.

Conclusion

Self-driving networks can improve network security, reliability, and performance, by adapting automatically to changing network conditions. We believe the time is right for significant progress on this challenge, due to technical advances on several fronts (e.g., programmable networking devices, machine learning, big-data platforms) and society's increasing reliance on network infrastructure. Making progress will rely on interdisciplinary collaboration between computer networking and other areas (e.g., machine learning, distributed systems, security, and programming languages), as well as stronger connections with operational networks to acquire data, identify important use cases, and evaluate solutions.

Appendix: Schedule and Attendees

Schedule

Thursday, February 15, 2018	
10:00 - 10:30 a.m.	Continental Breakfast / Pastries
10:30 - 10:50 a.m.	Introductory Talk (Nick and Jen)
10:50 - 11:30 a.m.	**Industry Plenary Talks (Part 1).** Opening talks that frame the needs of network operators and modern services ("application pull") and opportunities from advances in machine learning and programmable network devices ("technology push") . - John Leddy/Gulrukh Ahangar - Ken Duell - Walter Willinger
11:30 - 12:30 p.m.	**Breakout Session #1: Industry Problems/Use Cases.** Discussions with our industry experts on the challenges faced by industry, and the opportunities that technology can and should bring.
12:30 - 1:30 p.m.	**Working Lunch.** Discussion and Agenda Bashing.
1:30 - 2:00 p.m.	**Plenary Talks (Part 2).** Research Plenary Talks: - Tudor Dumitras - Balaji Prabhakar
2:00 - 2:30 p.m.	**Breakout Session #1 (Part 2): Use Cases.** Breakout groups comprising experts from a cross-section of computer science disciplines to address problems in the application areas, including: - Security - Performance - Network, application, and service provisioning - Troubleshooting and diagnosis We expect that some broader application topics may also emerge from the morning plenary talks. (3–4 Groups)
2:30 - 2:45 p.m.	**Readout from Session #1**
2:45 - 3:00 p.m.	Coffee Break
3:00 - 3:40 p.m.	**Breakout Session #2: Cross-Cutting Themes.** The second set of breakout sessions should address cross-cutting themes. Suggested working topics for three breakouts (preceded by agenda bashing

	as necessary): ● The unification of compute, storage, and networking infrastructure ● The relationship of interconnection and interdomain relationships to application performance, as well as other cross-domain measurement issues. ● Legal, policy, and privacy problems and questions that could be affected by or informed by improvements to network measurement infrastructure.
3:40 - 4:30 p.m.	**Readout from Session #2**
4:30 - 5:15 p.m.	**Reconvene Breakout Session #1.** The first breakout session reconvenes to incorporate lessons or insights from the cross-cutting themes that could inform the problems discussed from the first breakout. This time should also be dedicated to writing a breakout summary and writeup. More likely, we will reconvene as one group to identify themes and takeaways.
5:30 - 7:30 p.m.	**Working Dinner for Cross-Disciplinary Collaboration Building** Palmer House, Princeton University 1 Bayard Lane, Princeton, NJ
Friday, February 16, 2018	
8:30 a.m. - 9:00 a.m.	Breakfast
9:00 a.m. - 10:00 a.m.	**Breakout Session #3: The Relationship Between Measurement and Control.** Previous research on software-defined networking has largely focused on how programmable software controllers can better manage network traffic. This breakout session will consider the role that next-generation measurement technology can play in this control loop. Specifically, we will focus on two questions: (1) **Enabling Better Measurement:** How programmatic control, coupled with programmable data planes, can guide more scalable, efficient measurements; (2) **Using Measurement to Drive Control:** How inference of higher-level network properties can better guide network control and automation to help realize the goals of "self-driving networks".
10:00 a.m. - 10:30 a.m.	**Session #3 Readout**

10:30 a.m. - 10:45 a.m.	**Coffee Break**
10:45 a.m. - 12:00 p.m.	**Reconvene Session #3.** Further discussion and writeup.
12:00 p.m. - 1:30 p.m.	**Working Lunch**. Discussion over themes and recommendations to convey in final report. Other topics to discuss: - How should we make our ideas "real"? - What infrastructure will we need? - Where will such networks be deployed? - What will self-driving networks look like in 10 years? - What role will NSF, DARPA, industry, etc. play?
1:30 p.m. - 2:30 p.m.	**Reconvene Breakout #2:** Further discussion and writeup on the cross-cutting themes above.
2:30 p.m. - 3:00 p.m.	**Wrap-Up**

Attendees

Academia:

- Mohammad Alizadeh (MIT)
- Aditya Akella (U. Wisconsin)
- Theo Benson (Brown)
- Tudor Dumitras (U. Maryland)
- Zakir Durumeric (Stanford)
- Nick Feamster (Princeton)
- Nate Foster (Cornell)
- Philippa Gill (UMass-Amherst)
- Alex Halderman (U. Michigan)
- Xin Jin (Johns Hopkins)
- Ethan Katz-Bassett (Columbia)
- Shriram Krishnamurthi (Brown)
- Balaji Prabhakar (Stanford)
- Jennifer Rexford (Princeton)
- David Walker (Princeton)

Industry:

- Gulrukh Ahanger (Comcast)
- Ken Duell (AT&T)
- John Leddy (Comcast)
- Walter Willinger (Niksun)

NSF :

- Jack Brassil (NSF)
- Darleen Fisher (NSF)
- Ann von Lehman (NSF)

Student scribes:

- Noah Apthorpe (Princeton)
- Yilong Geng (Stanford)
- Arpit Gupta (Princeton)
- Mina Tahmasbi Arashloo (Princeton)

- Rob Harrison (Princeton)