2005-10

A Clean Slate 4D Approach to Network Control and Management

Greenberg, A.

http://hdl.handle.net/10945/34771

A Clean Slate 4D Approach to Network Control and Management [*]

Albert Greenberg, Gisli Hjalmtysson, David A. Maltz, Andy Myers,
Jennifer Rexford, Geoffrey Xie, Hong Yan, Jibin Zhan, Hui Zhang
{dmaltz,acm,yh,jibin,hzhang}@cs.cmu.edu
gisli@ru.is jrex@cs.princeton.edu albert@research.att.com xie@nps.edu

ABSTRACT

Today's data networks are surprisingly fragile and difficult to manage. We argue that the root of these problems lies in the complexity of the control and management planes—the software and protocols coordinating network elements—and particularly the way the decision logic and the distributed-systems issues are inexorably intertwined. We advocate a complete refactoring of the functionality and propose three key principles—network-level objectives, network-wide views, and direct control—that we believe should underlie a new architecture. Following these principles, we identify an extreme design point that we call "4D," after the architecture's four planes: decision, dissemination, discovery, and data. The 4D architecture completely separates an AS's decision logic from protocols that govern the interaction among network elements. The AS-level objectives are specified in the decision plane, and enforced through direct configuration of the state that drives how the data plane forwards packets. In the 4D architecture, the routers and switches simply forward packets at the behest of the decision plane, and collect measurement data to aid the decision plane in controlling the network. Although 4D would involve substantial changes to today's control and management planes, the format of data packets does not need to change; this eases the deployment path for the 4D architecture, while still enabling substantial innovation in network control and management. We hope that exploring an extreme design point will help focus the attention of the research and industrial communities on this crucially important and intellectually challenging area.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Packet Switching Networks; C.2.2 [**Network Protocols**]: Routing Protocols; C.2.3 [**Network Operations**]: Network Management

General Terms

Measurement, Control, Performance, Reliability

Keywords

Network management, robustness, control

1. INTRODUCTION

Although IP networking has been wildly successful, there are serious problems lurking "under the hood." IP networks exhibit a defining characteristic of unstable complex systems—a small local event (e.g., misconfiguration of a routing protocol on a single interface) can have severe, global impact in the form of a cascading meltdown. In addition, individual Autonomous Systems (ASes) must devote significant resources to "working around" the constraints imposed by today's protocols and mechanisms to achieve their goals for traffic engineering, survivability, security, and policy enforcement. We believe the root cause of these problems lies in the control plane running on the network elements and the management plane that monitors and configures them. In this paper, we argue for revisiting the division of functionality and advocate an extreme design point that completely separates a *network's decision logic* from the *the protocols that govern interaction of network elements* We initially focus our attention on the operation of a single Autonomous System (AS), though we also discuss how multiple ASes can coordinate their actions.

The Internet architecture bundles control logic and packet handling into the individual routers and switches distributed throughout an AS. As a result, each router/switch[1] participates in distributed protocols that implicitly *embed* the decision logic. For example, in IP networks, the path-computation logic is governed by distributed protocols such as OSPF, IS-IS, and EIGRP. The routing protocols dictate not only how the routers learn about the topology, but also how they select paths. Similarly, in Ethernet networks, the path-computation logic is embedded in the Spanning Tree protocol [1]. However, today's data networks, operated by numerous institutions and deployed in diverse environments, must support network-level objectives and capabilities far more sophisticated than best-effort packet delivery. These ever-evolving requirements have led to incremental changes in the control-plane protocols, as well as complex management-plane software that tries to "coax" the control plane into satisfying the network objectives. The resulting complexity is responsible for the increasing fragility of IP networks and the tremendous difficulties facing people trying to understand and manage their networks.

Continuing on the path of incremental evolution would lead to additional point solutions that exacerbate the underlying problem of an overly-complex control plane. Instead, we advocate redesigning the control and management functions from the ground up. We believe that a clean-slate approach based on sound principles will, at the minimum, provide an alternative perspective and shed light on fundamental trade-offs in the design of network control and management functions. More strongly, we believe that such an

[*] This research was sponsored by the NSF under ITR Awards ANI-0085920 and ANI-0331653. Views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of AT&T, NSF, or the U.S. government.

[1] We use the terms "network element" and "router/switch" interchangeably throughout the paper.

approach is *necessary* to avoid perpetuating the substantial complexity of today's control plane. Fortunately, we can make significant, fundamental changes in the control and management of IP networks *without changing the format of the data packets*. This enables network evolution and provides a key lever for substantial innovation in the Internet architecture. A good example of this principle is the Ethernet technology, which has successfully evolved from a shared-medium network to a switched network with new control-plane protocols based on learning and spanning trees, all while leaving the packet format unchanged.

This paper presents an initial effort for a clean slate design approach to data-network control and management. To guide our design, we start from a small set of principles: *network-level objectives, network-wide views*, and *direct control*. These principles lead us to the 4D architecture that refactors functionality into four components: the *data, discovery, dissemination, and decision planes*. The decision plane for an AS creates a network configuration that satisfies AS-level objectives. The decision plane has a network-wide view of the topology and traffic, and exerts direct control over the operation of the data plane. No decision logic is hardwired in protocols distributed among the network elements. The output of the decision logic is communicated to routers/switches by the dissemination plane. Our study investigates an *extreme design point* where the decision logic is completely separated from distributed protocols. By pulling all of the decision logic out of the network elements, we enable both simpler protocols and more sophisticated algorithms for driving the operation of the data plane. In addition, we believe that the technology trends toward ever-more powerful, reliable, and inexpensive computing platforms make our design point attractive in practice.

Our goal for this paper is not to prove that 4D is the *best* approach. In fact, our research is still at an early stage and there are many unanswered questions about the architecture. Rather, by presenting a specific design alternative that is radically different from today's approach, and more reminiscent of early alternatives to IP such as SNA, we want to highlight the issues that need to be considered in a clean slate design of network control and management. We hope this work will help focus the attention of the Internet research community and industry on this crucially important and intellectually challenging area. In the next section, we present examples of the problems that face network designers today, and explain why conventional techniques are inadequate. We then step back and identify three principles that we argue should underlie the architecture for controlling and managing data networks. Next, we outline our results from a clean-slate redesign of the control and management architecture based on these principles. We set out the potential benefits and drawbacks of the architecture, and we articulate a research agenda with the challenges that must be met to realize the architecture. Finally, we explain how the architecture differs from previous approaches and present examples of how such research might be conducted.

2. CONTROL & MANAGEMENT TODAY

In today's data networks, the functionality that controls the network is split into three main planes: (i) the *data plane* that handles the individual data packets; (ii) the *control plane* that implements the distributed routing algorithms across the network elements; and (iii) the *management plane* that monitors the network and configures the data-plane mechanisms and control-plane protocols.

While the original IP control plane was designed to have a *single* distributed algorithm to maintain the *forwarding* table in the data plane, today's IP data, control and management planes are far more complex. The data plane needs to implement, in addition to next-hop forwarding, functions such as tunneling, access control, address translation, and queuing. The states used to implement these functions are governed by multiple entities and have to be configured through a rich set of individual, interacting commands. Even for the forwarding state, there are usually multiple routing processes running on the same router/switch.

While there are many dependencies among the states and the logic updating the states, most of the dependencies are *not* maintained automatically. For example, controlling routing and reachability today requires complex arrangements of commands to tag routes, filter routes, and configure multiple interacting routing processes, all the while ensuring that no router is asked to handle more routes and packet filters than it has resources to cope with. A change to any one part of the configuration can easily break other parts.

The problem is exacerbated as packet delivery cannot commence until the routing protocols create the necessary forwarding tables, and the management plane cannot reach the control plane until the routing protocols are configured. Resolving this catch-22 requires installing a significant amount of configuration information on IP routers before deployment.[2] Studies of production networks show them requiring hundreds of thousands of lines of low-level configuration commands distributed across all the routers in the network [2]. These configurations and the dynamic forwarding state they generate require a myriad of ad hoc scripts and systems in the management plane to validate, monitor, and update. The result is a complex and failure-prone network.

We present two examples that illustrate the network fragility caused by today's complex and unwieldy control and management infrastructure. The examples illustrate how the lack of coordination between routing and security mechanisms can result in a fragile network, and how today's control and management infrastructure makes it difficult to properly coordinate the mechanisms.

2.1 Reachability Control in Enterprises

Today, many enterprise networks attempt to control which hosts and services on their network can communicate (i.e., reach each other) as part of their security strategy [2]. They implement their strategies using a combination of routing policy and packet filters, but this approach is fraught with peril even in simple networks.

Consider the example enterprise network in Figure 1. The company has two locations, A and B. Each location has a number of "front office" computers used by the sales agents (AF1-2 and BF1-2). Each location also has a data center where servers are kept (AD1-2 and BD1-2). Initially, the two locations are connected by a link between the front office routers, R2 and R4, over which inter-office communications flow. The Interior Gateway Protocol (IGP) metric for each link is shown in italics. The company's security policy is for front-office computers to be able to communicate with other locations' front office computers and the local data center's servers, but not the data center of the other location. Such policies are common in industries like insurance, where the sales agents of each location are effectively competing against each other even though they work for the same company. The security policy is implemented using packet filters on the routers controlling entrance

[2] This problem is so profound that, whenever possible, remote routers/switches are plugged into telephone modems so that the Public Switched Telephone Network provides a management communication path of last resort. Before making configuration changes to the router over the Internet via Telnet or ssh, operators often double check that the modem connection is still functioning, lest an unfortunate configuration mistake leave them with no other way to contact the router, short of physical access to the console.

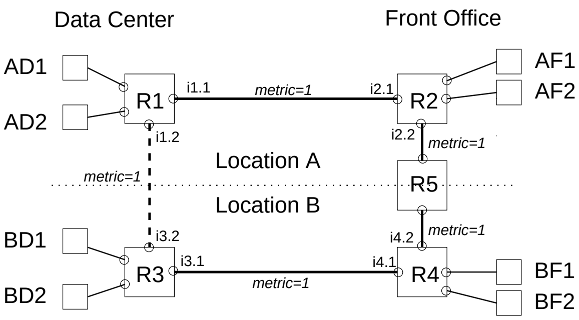

Figure 1: Enterprise network with two locations, each location with a front office and a data-center.

to the data centers to drop packets that violate the policy. Interface i1.1 is configured with a packet filter that drops all packets from the BF subnet, and interface i3.1 drops all packets from the AF subnet.

The network functions as desired, until the day when the datacenter staff decides to add a new, high-capacity dedicated link between the data centers (shown as a dashed line between R1 and R3—perhaps they have decided to use each other as remote backup locations). It seems reasonable that with packet filters protecting the entrances to the data centers, the new link between data centers should not compromise the security policy. However, the new link changes the routing such that packets sent from AF to BD will travel from R2 to R1 to R3 to BD—completely avoiding the packet filter installed on interface i3.1 and violating the security policy. When the designers eventually discover the security hole, probably due to an attack exploiting the hole, they would typically respond by copying the packet filter from i3.1 to i3.2, so it now also drops packets from AF. This filter design does plug the security hole, but it means that if the front office link from R2 to R4 fails, AF will be unable to reach BF. Even though the links from R2 to R1 to R3 to R4 are all working, the packet filter on interface i3.2 will drop the packets from subnet AF.

In this example, the problems arise because the ability of a network to carry packets depends on the routing protocols and the packet filters working in concert. While routing automatically adapts to topology changes, there is no corresponding way to automatically adapt packet filters or other state. It could be argued that a more “optimal” placement of packet filters, or the use of multi-dimensional packet filters (i.e., filters that test both source and destination address of a packet) would fix the problems shown in this example. However, as networks grow in size and complexity from the trivial example used here for illustrative purposes, finding these optimal placements and maintaining the many multi-dimensional packet filters they generate requires developing and integrating entirely new sets of tools into the network’s management systems. Since these tools will be separate from the protocols that control routing in real time, they will perpetually be attempting to remain synchronized with routing protocols by trying to model and guess the protocols’ behavior.

In contrast, the 4D architecture simply and directly eliminates this entire class of problems. The 4D architecture allows the direct specification of a “reachability matrix” and automated mechanisms for simultaneously setting the forwarding-table entries and packet filters on the routers based on the current network state.

2.2 Peering Policies in Transit Networks

Routing policy is based on the premise that a router that does not announce a route to a destination to a peer will not be sent packets for that destination by that peer. However, the routing system does nothing to prevent an unscrupulous peer from sending packets to that destination anyway. Enforcing routing policy is nearly impossible with today’s control and management planes.

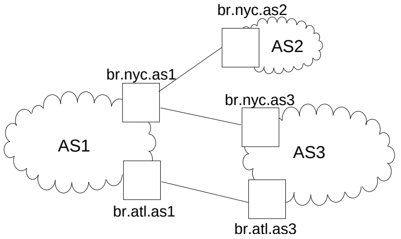

Figure 2: Autonomous Systems (ASes) peering with each other via external BGP (eBGP) sessions. AS1 must place packet filters on its ingress links to prevent AS3 from sending packets to destinations for which AS1 has not agreed to provide transit.

Figure 2 shows an example of three Autonomous Systems (ASes) peering with each other via three external BGP sessions (one eBGP session along each of the links shown in the figure). Assume that AS1 is a major transit network, and it announces a route to destination d in its eBGP session with AS2. If AS1’s policy is to not provide AS3 with transit service for d, it does not announce d in its eBGP sessions with AS3. However, if AS3 wishes to be unscrupulous (e.g., use AS1 for transit service without paying), it can assume AS1 does know a way to d (e.g., so AS1’s own customers can reach d). If AS3 sends packets for d to br.nyc.as1, they will definitely be delivered, as br.nyc.as1 must have a route to d in order to handle legitimate traffic from AS2.

Enforcing routing policy requires installing packet filters to drop packets to destinations which have not been announced as reachable. As the announcements received by an AS, and the AS’s own topology, change over time, the announcements sent by the AS will change and the packet filters must be moved correspondingly. Implementing such functionality by adding another ad hoc script to the management plane is essentially impossible today. Even if it were possible to write a script that snoops on the eBGP announcements sent to each neighboring border router and installs packet filters on the ingress interface as appropriate, the script would be extremely dangerous as it would not properly order the packet filter installation/removal with the BGP announcements. For example, it would be bad to announce to a neighbor border router a route to a destination before removing the packet filters that drop the packets sent to the destination.

Beyond ordering issues, transit networks handle a large number of destinations, and each packet filter applied to an interface consumes forwarding resources and reduces the effective capacity of the interface. It might be desirable to move packet filters into the network whenever possible, away from the ingress interfaces, so that one packet filter can enforce the BGP policy for multiple ingress interfaces.

Enforcing routing policy requires dynamically placing packet filters to respond to the continually changing routes selected by that policy. Correctly and optimally placing the filters requires that the placement be synchronized with the announcement of routing decisions and that the placement algorithms have access to the complete routing topology of the network. The 4D architecture provides the primitives and abstractions needed to implement correct placement strategies and support placement optimization algorithms.

2.3 Same Problems, Many Guises

There are many data networks, designed and managed by different organizations with different goals. Individual networks serve radically different purposes; in addition to the familiar backbone networks, there are access, metro, enterprise and data-center networks. In each of these settings, the network administrators struggle to "program" their networks, integrating a diverse set of technologies and protocols, and artfully setting the configurable parameters that determine the network's functionality and dynamics.

While the specific context, technology, and mechanisms may change from network to network, there is commonality among the problems. For example, while Ethernet was initially designed to run on a shared medium, it has since evolved into a networking technology with a full package of data plane, control plane, and management plane to rival IP. Just as IP has many routing protocols to compute the forwarding table, Ethernet has many variations of the spanning tree protocol [3]. Just as IP networks have mechanisms like MPLS to control the paths that packets take, Ethernet has virtual LANs (and VLANs-in-VLANs). Just as IP networks have needed to implement sophisticated functionality like traffic engineering, security policies and fast restoration, these same needs are being required of Ethernet in many contexts, such as enterprises, data centers [4], and metro/access networks [5]. Just as ad hoc management capabilities need to be overlaid on top of the IP control plane, achieving advanced functionality in Ethernet networks has led to increasingly ad hoc and complex management systems. The current architecture forces these systems to operate outside Ethernet's control plane, where they often come into conflict with it.

2.4 Moving Forward

We argue the key to solving the problems illustrated in this section is creating a way for the architectural intent and operational constraints governing the network to be expressed directly, and then automatically enforced by setting data-plane states on the individual routers/switches. Until this occurs, we expect the design and operation of robust networks to remain a difficult challenge, and the state of the art to remain a losing battle against a trend where ever richer and more complex state and logic are embedded in distributed protocols or exposed through box-level interfaces.

3. THE 4D ARCHITECTURE

Rather than exploring incremental extensions to today's control and management planes, we propose a *clean-slate* repartitioning of functionality. We believe that a green-field approach based on sound principles is necessary to avoid perpetuating the substantial complexity in today's design. We have developed the 4D architecture as an *extreme design point* that completely separates the decision logic from the underlying protocols. We deliberately chose an extreme design as we believe that it crystallizes the issues, so that exploring the strengths and weaknesses of this architecture will lead to important network-level abstractions and a deeper understanding of the essential functionality needed in the underlying routers and switches.

3.1 Design Principles

The rich literature on the complexity of today's control and management planes has led us to the following three principles that we believe are essential to dividing the responsibility for controlling and managing a data network:

Network-level objectives: Each network should be configured via specification of the requirements and goals for its performance. Running a robust data network depends on satisfying objectives for performance, reliability, and policy that can (and should) be expressed separately from the network elements. For example, a traffic-engineering objective could be stated as "keep all links below 70% utilization, even under single-link failures." A reachability policy objective could be stated as "do not allow hosts in subnet B to access the accounting servers in subnet A." Today's networks require these goals to be expressed in low-level configuration commands on the individual routers, increasing the likelihood that the objectives are violated due to semantic mistakes in translating the network-level objectives into specific protocols and mechanisms.

Network-wide views: Our notion of a network-wide view is borrowed from the database community and means having assembled a coherent snapshot of the state of each network component. Timely, accurate, network-wide views of topology, traffic, and events are crucial for running a robust network. The network-wide view must accurately reflect the current state of the data plane, including information about each device, including its name, resource limitations, and physical attributes. However, today's control plane was *not* designed to provide these network-wide views, forcing substantial retro-fitting to obtain them. Instead of adding measurement support to the system as an afterthought, we believe that providing the information necessary to construct a complete, consistent, network-wide view should be one of the primary functions of the routers and switches.

Direct control: Direct control means that the control and management system should have both the ability and the sole responsibility for setting all the state in the data plane that directs packet forwarding. The decision logic should not be hardwired in protocols distributed among routers/switches. Rather, only the output of the decision logic should be communicated to the network elements. Satisfying network-level objectives is much easier with direct control over the configuration of the data plane. IP and Ethernet originally embedded the path-computation logic in simple distributed protocols that incrementally grew more complicated, as discussed earlier in Section 1. Because of the difficulty of extending the distributed control protocols to support sophisticated network-level objectives such as traffic engineering or reachability control, the management plane is typically used to implement these additional capabilities. With only indirect influence over the network, today's management plane must replicate the state and logic of the control plane and perform a complex "inversion" of the functionality. The problem would be much easier to solve if the management plane could compute the forwarding tables and install them in the routers. For direct control to be meaningful, it must be complete. If configuration commands or multiple entities can affect the state in the network elements, then yet more entities are required for auditing (and correcting) the settings [6, 7, 8] to ensure the network-level objectives are met.

In addition to these three principles, any design must also consider traditional systems requirements, such as scalability, reliability, and consistency. Our three principles attempt to capture the issues specific to the control and management of networks. By separating the network-specific issues from the traditional systems requirements, we can apply existing techniques from other areas of distributed computing research to the traditional systems problems while exposing for closer scrutiny the network-specific ones.

3.2 New 4D Network Architecture

Although the three principles could be satisfied in many ways, we have deliberately made the 4D architecture an extreme design point where all control and management decisions are made in a logically centralized fashion by servers that have complete control over the network elements. The routers and switches only have the ability to run network discovery protocols and accept explicit

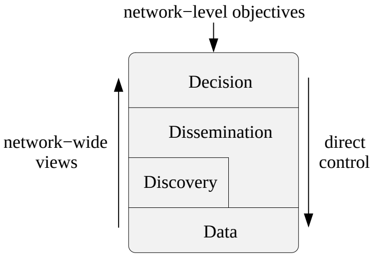

Figure 3: New 4D architecture with network-level objectives, network-wide views, and direct control

instructions that control the behavior of the data plane, resulting in network devices that are auto-configurable. Our architecture has the following four components, as illustrated in Figure 3:

Decision plane: The decision plane makes *all* decisions driving network control, including reachability, load balancing, access control, security, and interface configuration. Replacing today's management plane, the decision plane operates in *real time* on a network-wide view of the topology, the traffic, and the capabilities and resource limitations of the routers/switches. The decision plane uses algorithms to turn network-level objectives (e.g., reachability matrix, load-balancing goals, and survivability requirements) directly into the packet-handling state that must be configured into the data plane (e.g., forwarding table entries, packet filters, queuing parameters). The decision plane consists of multiple servers called decision elements that connect directly to the network.

Dissemination plane: The dissemination plane provides a robust and efficient communication substrate that connects routers/-switches with decision elements. While control information may traverse the same set of physical links as the data packets, the dissemination paths are maintained separately from the data paths so they can be operational without requiring configuration or successful establishment of paths in the data plane. In contrast, in today's networks, control and management data are carried over the data paths, which need to be established by routing protocols before use. The dissemination plane moves management information created by the decision plane to the data plane and state identified by the discovery plane to the decision plane, but does not create state itself.

Discovery plane: The discovery plane is responsible for discovering the physical components in the network and creating logical identifiers to represent them. The discovery plane defines the scope and persistence of the identifiers, and carries out the automatic discovery and management of the relationships between them. This includes box-level discovery (e.g., what interfaces are on this router? How many FIB entries can it hold?), neighbor discovery (e.g., what other routers does this interface connect to?), and discovery of lower-layer link characteristics (e.g., what is the capacity of the interface?). The decision plane uses the information learned from the discovery plane to construct network-wide views. In contrast, in today's IP networks, the only automatic mechanism is neighbor discovery between two preconfigured and adjacent IP interfaces; physical device discovery and associations between entities are driven by configuration commands and external inventory databases.

Data plane: The data plane handles individual packets based on the state that is *output* by the decision plane. This state includes the forwarding table, packet filters, link-scheduling weights, and queue-management parameters, as well as tunnels and network address translation mappings. The data plane may also have fine-grain support for collecting measurements [9] on behalf of the discovery plane.

The 4D architecture embodies our three principles. The decision-plane logic operates on a network-wide view of the topology and traffic, with the help of the discovery plane in collecting the measurement data, to satisfy network-level objectives. The decision plane has direct control over the operation of the data plane, obviating the need to model and invert the actions of the control plane. Pulling much of the control state and logic out of the routers enables both simpler protocols, which do not have to embed decision-making logic, and more powerful decision algorithms for implementing sophisticated goals.

3.3 Advantages of the 4D Architecture

Our 4D architecture offers several important advantages over today's division of functionality:

Separate networking logic from distributed systems issues: The 4D architecture does not and cannot eliminate all distributed protocols, as networks fundamentally involve routers/switches distributed in space. Rather, the 4D proposes separating the logic that controls the network, such as route computation, from the protocols that move information around the network. This separation creates an architectural force opposing the box-centric nature of protocol design and device configuration that causes so much complexity today. The 4D tries to find the interfaces and functionality we need to manage complexity—those that factor out issues that are not unique to networking and enable the use of existing distributed systems techniques and protocols to solve those problems.

Higher robustness: By simplifying the state and logic for network control, and ensuring the internal consistency of the state, our architecture greatly reduces the fragility of the network. The 4D architecture raises the level of abstraction for managing the network, allowing network administrators to focus on specifying network-level objectives rather than configuring specific protocols and mechanisms on individual routers and switches. Network-wide views provide a conceptually-appealing way for people and systems to reason about the network without regard for complex protocol interactions among a group of routers/switches. Moving the state and logic out of the network elements also facilitates the creation of new, more sophisticated algorithms for computing the data-plane state that are easier to maintain and extend.

Better security: Security objectives are inherently network-level goals. For example, the decision plane can secure the network perimeter by installing packet filters on all border routers. Managing network-level objectives, rather than the configuration of individual routers, reduces the likelihood of configuration mistakes that can compromise security.

Accommodating heterogeneity: The same 4D architecture can be applied to different networking environments but with customized solutions. For example, in an ISP backbone with many optimization criteria and high reliability requirements, the decision plane may consist of several high-end servers deployed in geographically distributed locations. A data-center environment with Ethernet switches may require only a few inexpensive PCs, and still achieve far more sophisticated capabilities (e.g., traffic engineering with resilience) than what spanning tree or static VLAN configuration can provide today.

Enabling of innovation and network evolution: Separating the network control from the routers/switches and protocols is a significant enabler for innovation and network evolution. The decision plane can incorporate new algorithms and abstractions for computing the data-plane state to satisfy a variety of network-level objectives, *without* requiring the change of either *data* packet formats or *control protocols* (dissemination and discovery plane protocols in the case of 4D). In addition, moving the control functionality out of the router/switch software enables new players (e.g., the research community and third-party software developers) to contribute to the creation of these algorithms.

3.4 Challenges for the 4D Architecture

While the 4D holds the promise of the advantages above, and initial implementation efforts show these benefits can be achieved [10, 11, 12], there are clear risks its design must avoid:

Complexity apocalypse: A major drawback of today's architecture is that it has enormous complexity distributed horizontally across the network elements and vertically across many layers. The 4D architecture must achieve the same functionality as today's systems, while also centralizing the decision logic and introducing new capabilities, such as a network-wide reachability policies and zero pre-configuration of routers/switches. Does the refactoring proposed by the 4D architecture dramatically simplify the overall system, or merely exchange one kind of complexity for another?

Stability failures: Since the network is distributed in space, there are unavoidable delays in informing the decision elements of events. For the global-scale enterprise and transit networks that companies want to create, is it possible to create a network-wide view stable and accurate enough for controlling such networks?

Scalability problems: The largest networks today have thousands of routers/switches and tens of thousands of devices and the default-free zone of today's Internet handles routes hundreds of thousands of destination prefixes. Is it possible for conventional servers to manage so many devices and respond to events fast enough to meet the network's goals? Will the amount of management information being moved by the dissemination plane overwhelm the network's ability to carry data?

Response time: With the unavoidable speed-of-light delays and the large quantity of control/management information to process, is it possible to respond to network failures and restore data flow within an acceptable period of time?

Security vulnerabilities: An attacker who compromises a decision element in a 4D network could control the entire network, similar to the power afforded an adversary that breaks into the today's management plane or the routers themselves. The security of a 4D system depends primarily on securing the dissemination plane that forms the communication channels between the routers/switches and the decision plane, and securing the decision plane itself. Is a 4D network more or less vulnerable to attack than routers running distributed routing protocols?

3.5 Routing Control Platform

There has been substantial work on problems of controlling and managing networks, and many different paradigms have been explored as outlined in Section 6 on related work. The Routing Control Platform (RCP) [11, 12] is especially consistent with our philosophy and objectives, and serves to show how substantial change in the management of IP networks is possible. RCP is a backwards compatible system designed to give the operators of transit networks more control over how BGP routing decisions are made in their Autonomous System (AS). We see RCP as an implementation of a specific point that lies inside the design space of the 4D architecture, where RCP makes its design decisions to emphasize scalability and deployability with conventional routers. For its decision elements, RCP uses Routing Control Servers, which do not need a coordination protocol because of the properties of the underlying discovery plane. For a dissemination plane, RCP uses iBGP sessions to tell the routers which BGP routes to use. For a discovery plane, RCP snoops on the flooding of OSPF link-state advertisements, and learns external BGP routes via the iBGP sessions with the operational routers.

This paper and the 4D architecture focus on questions unaddressed by the work on the RCP. Rather than focusing on BGP decision logic, we consider how a wide variety of network objectives could be expressed to the control/management system, and what new coordination protocols are required to achieve those objectives. RCP only considers BGP routes—a single part of the total state used by the data-plane to direct packets through the network. This paper asks how to control *all* the data-plane forwarding mechanisms (e.g., FIB entries, packet filters, NATs, tunnels, packet scheduling, and buffer management) in a coordinated fashion to achieve the network's objectives, and what protocols are needed to achieve this coordination. RCP assumes routers are already correctly configured with significant amounts of state, such as IP addresses and an Interior Gateway Protocol (IGP). This paper examines how *zero* pre-configuration of routers/switches can be achieved and how a clean slate design of device identifiers and the relationships among them can significantly simplify network control/management. Beyond considering only IP networks, this paper also examines how a single management architecture could control different types of networks such as Ethernet (with or without VLAN) IPv4, and IPv6 (with or without MPLS).

4. RESEARCH AGENDA

At this stage of our research, we do not yet know whether the advantages of the 4D architecture will outweigh the challenges. In the following sections, we will decompose these high-level questions into individual topics that constitute the research agenda that we are pursuing.

We recognize our vision for the 4D architecture is broader than what can be accomplished by us alone. By outlining the research agenda for the 4D architecture, we hope to start a discussion inside the larger research community on the clean slate design of network control and management.

4.1 Decision Plane

In the 4D architecture, the decision plane is responsible for direct control over the data plane based on a network-wide view, subject to network-level objectives. Designing the algorithms for the decision plane, and demonstrating their superiority over today's control plane, is an essential part of the 4D research agenda; finding effective ways to exploit the network structure and react in real time to network events is especially challenging and important. To avoid having a single point of failure, the decision-plane algorithms should run on multiple servers spread throughout the network, leading to questions about whether, and how, to coordinate the actions of the replicated decision elements (DEs). Ultimately, administrative boundaries and scalability concerns lead to an architecture with separate decision planes for different ASes or institutions. It is important to design protocols for DEs in one network to exchange information with DEs in other networks.

4.1.1 Algorithms Satisfying Network-Level Objectives

The decision plane implements logic that converts network-wide views and network-level objectives into directives for the data plane. For example, the decision plane should, given a network topology and traffic matrix, generate packet filters and forwarding-table entries that satisfy traffic-engineering goals and reachability constraints. Ultimately, an ambitious goal is to create a language or notation for expressing these network-level objectives. Below are examples of research areas that lead to that goal.

Traffic engineering: Given a network topology and traffic matrix, compute a forwarding graph—a forwarding-table entry for each destination prefix at each router—that minimizes an objective function, such as the maximum link utilization. This optimization problem has been widely studied in the context of existing IP routing protocols, such as OSPF and BGP, where the output is a set of OSPF weights and BGP policies that indirectly determine the forwarding-table entries [13, 14]. An interesting research direction is to explore whether the flexibility that results from having direct control over the forwarding tables allows us to move beyond the computationally intractable optimization problems that result from today's routing protocols [14].

Reachability policies: Given a network topology, a traffic matrix, and a reachability matrix, compute a forwarding graph and packet filters that minimize an objective function, while satisfying the reachability constraints. In the simplest case, every edge link could be configured with packet filters to impose the reachability restrictions, with the forwarding-table entries permitting all pairs of end-points to communicate. However, routers typically have resource limitations that restrict the number of packet filters on each link, which substantially changes the nature of the problem.

Planned maintenance: Given a network topology, a traffic matrix, and a planned event to disable certain equipment, compute a sequence of changes to the forwarding graph to avoid using the routers and links undergoing maintenance. (The same schedule of forwarding-table changes could be applied, in reverse, to reintroduce the equipment into the network after the maintenance completes.) Each step should avoid introducing forwarding anomalies (such as loops and blackholes) or link congestion. The goal is to allow maintenance to proceed without disrupting applications, such as voice-over-IP (VoIP) and online gaming, that are sensitive to transient packet losses during routing-protocol convergence.

In addition to these and other related algorithmic questions in isolation, there are also several larger issues that arise in the design of the decision plane:

Leveraging network structure: For each of these algorithmic questions, there are scenarios where the decision plane can exploit knowledge of the network structure. For example, the algorithms for imposing reachability constraints would become simpler in an access network with a tree-like structure. As another example, the computational complexity of the algorithms could be reduced by modeling a backbone network's topology at a coarser level—where each node is a Point-of-Presence (PoP) rather than a router. In each case, the knowledge of the network structure could reduce the computational overhead of the algorithms and facilitate better solutions.

Specification of network-level objectives: In addition to creating algorithms that solve specific optimization problems, we need to design a decision plane that can satisfy multiple constraints and optimize across multiple objectives simultaneously. An important first step of this research is a deeper understanding of how to specify network-level objectives, including a configuration language for the decision plane. The proposed configuration language should be evaluated along two dimensions: complexity and expressiveness. It should have a lower complexity than that of configuring individual routers today. In addition, it should be able to express the network-level objectives that arise in existing networks [2].

Finding the right separation of timescales: The decision plane must react in real time to network events, such as equipment failures and routing-protocol messages from neighboring domains, without invoking a complex optimization algorithm. Identifying the right abstractions to support rapid reactions to unplanned events, while still supporting optimization based on network-wide objectives, is an important and challenging research problem. In designing the decision plane, there is the opportunity to create new algorithms that compute quick answers for the data plane, while offering tunable parameters that can be optimized to satisfy network-level goals. In contrast, today's routing protocols (e.g., OSPF and BGP) were not designed with optimization in mind, which leads to computationally intractable optimization problems [14].

4.1.2 *Coordination Between Decision Elements*

Having a reliable decision plane is crucial to the robust operation of the network. To avoid having a single point of failure, multiple Decision Elements (DEs) should connect to the network at different locations. Yet, the presence of multiple DEs should not compromise the stable and consistent operation of the network. There are several approaches for coordinating the decisions of the DEs, with different underlying assumptions about the decision algorithms and consistency requirements, including:

Distributed election algorithms: In one class of solutions, the multiple DEs run a standard distributed-election algorithm, where only the elected leader sends instructions to the data plane. This approach avoids scenarios where different DEs send inconsistent directives and obviates the need for the routers/switches to determine which state to use. However, the election algorithm introduces additional complexity and overhead, as well as delay for the network to recover when the current leader fails.

Independent DEs: A second class of solutions allows the DEs to operate independently, without any explicit coordination. Each DE executes decision algorithms and contacts the network elements based only on information provided by the dissemination plane. A network element resolves commands from different DEs based on static priorities and/or a timeout mechanism. This approach has faster failover time and eliminates the need for the DEs to coordinate, at the expense of more control traffic and the need for stronger assumptions about the consistency of the information provided by the dissemination plane. Initial studies in the context of BGP routing suggest that this approach is viable [12], though we need to investigate how well (and whether) the approach applies to other kinds of network state.

It is also possible to have hybrid schemes where each network element receives instructions from a small subset of the DEs, using priority and timeout mechanisms to resolve conflicts.

4.1.3 *Introducing Hierarchy in the Decision Plane*

In the simplest case, each DE has a complete view of the network and makes decisions on behalf of each router. It is important to enable hierarchical control of large networks over multiple (sets of) decision elements. Consider the following two scenarios:

Large network managed by a single institution: Today, the main techniques for scaling a large network include segmenting the topology into multiple ASes, grouping nearby routers into a single OSPF area, and aggregating destination prefixes at area and AS boundaries. However, existing routing protocols lack the basic abstractions common in hierarchical network designs, such as routing complexes (or central offices), Points-of-Presence (PoPs), and geographic regions, and largely ignore the roles the routers play in the network (e.g., access, hub, backbone, and peering). There is a great opportunity for novel research that explores using these design abstractions to support the management of hierarchical networks, including effective ways to divide responsibility across DEs and to coordinate their actions.

Multiple networks managed by different institutions: Ultimately, the decision plane for one network will need to communicate with the decision planes in other institutions, such as customer, peer, and provider ASes. If two neighboring ASes each have a decision plane, their DEs can communicate directly to exchange interdomain routing information, and perhaps to coordinate in other ways (e.g., traffic engineering and network troubleshooting) [11, 15]. In this setting, neighboring ASes may be business competitors that are reluctant to share information and are wary of cooperation.

4.2 Dissemination Plane

To establish the feasibility of the 4D architecture, we must design a dissemination plane that provides robust communication paths between decision elements and the routers/switches of the network. Our vision for the dissemination plane is that it will expose an interface that enables independent innovation of the decision elements above and independent evolution of routers/switches below [16, 17]. As a first step towards a dissemination plane that can serve as a universal kernel, we are designing a single dissemination plane that can be used in both Ethernet and IP networks.

Connecting decision elements with routers/switches: It is important to create robust and efficient dissemination paths to carry management information between routers/switches and decision elements, without necessarily requiring successful establishment or convergence of data plane paths. We propose to achieve this via distinct protocols and forwarding tables for management information. This approach has several advantages: (1) unlike data paths, which must be optimized for a variety of objectives like traffic engineering or security, dissemination paths can be optimized solely for robustness of connectivity under failures; (2) management information can be communicated to and from routers before the data channel is up or converges; (3) the dissemination paths are agnostic to data plane technology or policies; and (4) management information can be carried across data links as well as any extra physical links created specifically for management robustness (e.g., modem lines, or the supervisory channel on SONET and optical links).

There are at least three classes of solutions: flooding schemes, spanning-tree protocols, and source routing. Flooding scales well with the number of decision elements (by robustly multicasting data from all routers/switches to all DEs), but scales poorly with the number of router/switches. Spanning-tree protocols scale well with both the number of decision elements and the number of router/-switches, but exhibit poor reconvergence properties [18]. In source routing schemes, beacons can assist in creating source routes from each router/switch to the decision elements, or the decision elements can use their network-wide views to choose source routes that load-balance dissemination data across the network.

Achieving direct control: Choosing the right transport and session-layer semantics for the dissemination plane is critical for achieving our principle of direct control, and there is a broad design space to explore. Packets carrying management information through the dissemination plane may be lost, but retransmission of lost packets may not be the best policy. Instead, it might be better for the decision elements to calculate new state updates for the remaining routers/switches that can be reached without losses, where these new state updates cause data packets to circumvent the network elements that the decision plane can no longer reach.

Most state changes ordered by decision elements will involve updating state on multiple routers/switches. There is a wide spectrum of session layer semantics to explore, from the weak semantics of "each router independently applies an update as soon as it is received," to network-wide commit semantics that apply all received updates at a particular time, to full transactional distributed-commit semantics. It is also possible to introduce various optimization techniques, such as means of grouping related state updates into a single session "transaction" and methods for allowing multiple decision elements to send updates to overlapping sets of routers. Another interesting idea is to exploit good time synchronization (e.g., through NTP or a GPS receiver at each router or PoP) to instruct the routers/switches to change from one configuration to another at a specific time, resulting in infinitesimal convergence delay.

4.3 Discovery Plane

Controlling and managing a network requires creating a network-wide view of all the devices that comprise the network, and the physical and logical relationships between those devices. Today, information about devices, their identities, and the relationships between them, is encoded in the static configuration files present on the devices themselves and/or in management databases. For example, router/switch interfaces are often configured with IP subnets, and chaos ensues if cables are accidentally swapped such that interfaces with different subnets end up plugged together. Similarly, IP-level interfaces connected by ATM or Frame Relay services must be configured with the correct circuit ID used by the lower layer or the interfaces will be unable to exchange packets. Maintaining consistency between the inventory databases, configuration files, and physical reality is a major headache and creates some of the thorniest problems faced in existing networks. These problems could be eliminated by research to create a discovery plane that operates from the ground up: automatically discovering the identities of devices and components and the logical and physical relationships between them. Some particularly interesting problems include the following.

Support for decision-plane algorithms: An interesting research direction is to design discovery services that support the decision-plane algorithms described in Section 4.1.1, study the set of physical and logical entities and the corresponding set of relationships that need to be managed, and explore how the persistence properties of the identities and relationships should be defined and enforced. As an example of the issues to be considered, a router interface may be associated with a hardware port, a layer-2 logical port, an index for SNMP polling, an association with an optical circuit, and more. With today's architecture, most statistics, such as utilization and interface failure rates, are retrieved and tracked using the identity of the interface card. If the old card is moved to another router and a new card installed in its place, the correct adjustment (to have the traffic statistics stay with the new card and the history of interface failures move with the old card) is difficult to realize in today's systems. Yet, maintaining correct semantics during low-level network change is extremely important to many high-level network functions. For example, tracking transient failures is important for predicting whether an interface card needs to be replaced, and an accurate history of traffic load between each pair of routers is important for traffic engineering and capacity planning (whether or not the specific cards have changed).

Bootstrapping with zero pre-configuration beyond a secure key: In contrast to today's networks, which require extensive configuration before a router/switch can communicate, it is possible to automatically bootstrap a 4D network assuming only that each network element has a credential installed via a flashcard or USB key. For example, upon booting, the router/switch will first automatically generate an identity for itself and discover all its physical components and attributes. Then, the router/switch will discover its neighbors by exchanging identifiers and credentials with them. The credentials help to establish the boundary of a network: two adjacent routers/switches will continue with discovery only if they have compatible credentials. Once neighbor discovery completes, the router/switch can participate in the dissemination plane, allowing it to send information about its physical components and attributes (including the relationships between identifiers) to the decision plane. Compared with today's networks where identities and relationships are established via a manual, open-loop configuration process, within the 4D architecture the identities and relationships are either discovered based on physical relationships or assigned based on policies.

Supporting cross-layer auto-discovery: Two switches may not be directly connected, but instead connected by a lower layer network, such as a SONET network. There are two alternative architectures to achieve cross-layer auto-discovery: peer-to-peer and overlay. In the peer-to-peer architecture, directly connected devices exchange discovery information, within and across network layers (e.g., routers at layer 3 and SONET ADMs at layer 1). In the overlay architecture, discovery happens only between adjacent devices at the same logical layer. A generic interface needs to be defined between the two layers to allow the automatic establishment of the associations between routers and switches. Two types of lower layer networks, Ethernet and SONET, are good candidates to explore these issues.

4.4 Data Plane

In the 4D architecture, the data plane handles data packets under the direct control of the decision plane. This is in sharp contrast to today's architecture, where the responsibility for configuring the data plane is split between the control plane (which combines information from different routing protocols to generate a forwarding table) and the management plane (which configures access-control lists, link-scheduling weights, and queue-management policies). Although our framework is applicable to a wide variety of data-plane technologies, we believe that the capabilities of the data plane have a direct influence on the simplicity and flexibility of the logic in the decision plane:

Packet-forwarding paradigms: Data networks employ a wide variety of techniques for forwarding packets, ranging from the longest-prefix match paradigm (IPv4 and IPv6), exact-match forwarding (Ethernet), and label switching (MPLS, ATM, and Frame Relay). A forwarding-table entry may direct traffic to a single outgoing link or multiple links, with either equal splitting of traffic or more general support for weighted splitting. We plan to explore how our decision-plane algorithms would vary depending on the forwarding-paradigm supported in the data plane. For example, if the data plane performs weighted splitting of traffic over multiple outgoing links, the decision plane could apply multi-commodity flow algorithms that assume that traffic is infinitely divisible. In contrast, if each router directs all traffic for a destination to a single outgoing link, the decision plane would be forced to construct a sink tree for each destination prefix, requiring more complex algorithms for optimizing the construction of the forwarding tables. Studying these trade-offs will shed light on the tension between the packet-forwarding capabilities of the data plane and the effectiveness of the decision plane.

Advanced data-plane features: The data plane could incorporate new features that support the direct, network-wide control of the decision plane. For example, the data plane could provide an integrated mechanism that combines packet forwarding, filtering, and transformation (e.g., packet forwarding based on the five-tuple of the source and destination addresses, port numbers, and protocol, and efficient support for policy-based routing) to give the decision plane direct control over reachability through a single mechanism. To allay concerns over the response time of a 4D network, the data plane could use preconfigured tunnels to support immediate local reactions to unexpected network events, such as failures. For example, the data plane could have a table that indicates how to adapt packet forwarding after a particular link or path fails [19, 20] to allow the data plane to react to network events before receiving further instruction from the decision plane. Finally, the data plane can assist in constructing an accurate, network-wide view of the traffic by keeping fine-grain counters of the number of packets and bytes that match in certain attributes [9] or providing configurable support for packet sampling [21].

Throughout this part of our study, our goal is to understand how enhancements to the data plane would help support the decision plane, rather than to propose entirely new data-plane technologies.

5. EVALUATING NEW ARCHITECTURES

A major frustration for the research community has been the difficulty in conducting realistic network control and management experiments to validate or experiment with alternate designs. Thankfully, this is now changing. There are multiple platforms on which early research can be conducted and more opportunities than ever before for deployment experience in production networks.

5.1 Experimental Platforms

For clean-slate designs that desire the maximum flexibility to explore the space of network control and management, there are now experimental platforms that allow the creation of complete networks. For example, Emulab [22] allows experimenters to construct a network using PCs with multiple Ethernet interfaces as routers. The operating system on the PCs can be modified to implement new packet forwarding paradigms, signaling protocols, or control protocols. Large networks with almost arbitrary topologies can be configured using the several hundred PCs available.

Several other experimental platforms even makes it possible to specify and test the data plane. Jon Turner's Open Network Lab (ONL) [23] allows remote users to modify both the software running on the routers' embedded processors and the actual packet forwarding hardware, which is implemented using Field Programmable Gate Arrays (FPGAs). Nick McKeown's NetFPGA project [24] similarly allows experimenters to modify both the router software and hardware. NetFPGA also makes it easy to create networks that mix physical router nodes with virtual nodes and to pipe Internet traffic through a test network.

The GENI (Global Environment for Network Investigations) [25] initiative at the U.S. National Science Foundation provides an ideal environment for large-scale evaluation of the 4D architecture with real user traffic. The GENI facility would provide an experimental infrastructure with programmable network elements that can support new control and management architectures, while allowing multiple researchers to evaluate different designs at the same time in different "slices" of the infrastructure. As part of future work, we hope to create a software infrastructure that would allow a wide range of researchers to build and evaluate new designs for each of the "planes" of the 4D architecture on GENI.

5.2 Opportunities for Field Deployment

Gaining field experience with new ideas for control and management requires both finding networks willing to deploy the new ideas and adapting the ideas to those networks.

While there is a small and shrinking number of global transit networks, municipal networks provide the potential for real deployment experience in a carrier network—and their numbers are growing rapidly. As of 2004 there were over 40 efforts by municipalities and non-government organizations in the United States to construct networks in their regions that connect some combination of businesses, public institutions, and residential users [26]. Often these networks aim to provide a triple-play of voice, video, and data services, and so are an excellent challenge for any control/management architecture. As municipal activities, they are often open to collaborations with area universities and researchers.

Beyond the carrier network space, there is growing recognition of the complexity of enterprise networks. Enterprise networks are frequently called on to implement a wide variety of complex services, but they are often run by operators who are not networking experts, which increases the need for new ideas for controlling and managing these networks. There are hundreds of thousands of enterprises of all different sizes and with many different requirements, greatly expanding the number of potential deployment opportunities. We have found the IT departments of our own organizations to be excellent places to begin our research.

Deploying new ideas for control and management into production networks likely requires implementing those ideas using conventional routers, which have closed software architectures and speak only pre-existing protocols. However, there are many techniques for overcoming this challenge and crafting hybrids that pair current hardware/software with new ideas. For example, the Routing Control Platform (RCP) [11] work shows how a discovery plane can be built by passive monitoring of the intradomain routing protocol (OSPF) [27, 28] and learning interdomain routes through internal BGP (iBGP) sessions with the conventional routers. Similarly, dissemination can be implemented by using iBGP sessions to force the desired routes into each router, or by using translator proxies to convert the commands of the control/management system into the configuration language of the routers.

6. RELATED WORK

The importance of network control and management in creating robust networks has been recognized by both the research and network operator communities for many years. Many different paradigms for this area have been explored, including Signaling System 7 and the Intelligent Network, active networks, and policy-based networking, and there is increasing attention in the Internet research community [29]. This section explains the relationship between the 4D architecture and some of the many existing efforts.

Traditional telecommunications networks: The concept of centralization is heavily used in many management paradigms for telecommunication networks, usually based on circuit-switched technology [30]. In contrast, 4D focuses on packet-switching data networks that have more complex data-plane primitives (e.g., packet forwarding based on longest-prefix matching, access control, NAT, and tunnels) and higher network dynamics. Like Signaling System 7 (SS7) [31, 32], the 4D architecture keeps communication channels for management information isolated from the paths used by user data. However, SS7 takes the approach of a hard separation between management and user data onto separate links or channels, while the 4D architecture explores a softer logical separation appropriate for links like Ethernet. The Intelligent Network (IN) architecture [33] supports extension of network functionality by enabling user data (call placements) to trigger Detection Points that invoke per-service code. Because the terminals in data networks are fully-programmable devices, the 4D architecture deliberately does not provide a way for a user-originated message carried by the data plane to invoke functionality in the decision plane in order to avoid a class of Denial of Service attacks to which the IN is vulnerable.

Active networks: The active networks community sought to create networks with extensible functionality, and pursued several approaches. Some, such as code-carrying packets, are quite different from the 4D approach, but others, such as creating a minimal kernel of functionality implemented on each router/switch to be invoked from another location [34], share the same goals as the 4D.

Management tools for a distributed control plane: Many tools have been developed to ease the configuration of the existing architecture for control and management, which depends on individually configured switches/routers running a distributed control plane. Some approaches, like those adopted by Cplane and Orchestream, developed frameworks to solve the problems inherent in configuring large numbers of distributed switches/routers that may use different command languages. Other tools focus on specific operational tasks, such as traffic engineering or mitigation of Denial-of-Service (DoS) attacks. For example, Cariden's MATE [35] and OpNet's SP Guru [36] products can tune OSPF costs or MPLS Label Switched Paths to the prevailing traffic, and ArborNetwork's PeakFlow DoS [37] product detects DoS attacks and generates filters to block the offending traffic. The general approach of Policy-based networking (PBN) has been studied to automate provisioning and network management in applications such as QoS [38].

While very useful for specific tasks, network-management tools and PBN approaches usually assume the existing control-plane protocols, focus on a small portion of the configuration state (e.g., packet filters, but not routing), and do not consider the interactions among multiple mechanisms. In contrast, in the 4D architecture the network is directly controlled by decision elements using network-wide views to manage all network state—it explicitly establishes the decision plane as the place in the architecture for coordinating *all* of the data-plane mechanisms and provides the decision plane with the information it needs to operate.

Router configuration: A 2002 study estimates that half of network outages stem from human configuration error [39]; similar results have been found in studies of Internet services [40]. Analysis focusing specifically on BGP routing suggests that configuration errors are responsible for many network anomalies [41, 8]. Several tools provide analysis across configuration files to reverse engineer the router topology and summarize the status of the network [6, 7, 8, 42, 43]. However, despite their wide usage, these tools have not eliminated configuration problems. In the 4D architecture, we *eliminate the router/switch configuration files entirely* and along with them the need to verify their consistency or reverse engineer their actions. Rather than retrofitting analysis tools on top of a shifting pile of management tools, we propose an architectural change so the network itself generates a view of its status and topology.

Separating forwarding and control: Driven by the desire to separate router forwarding from protocols and network services, significant prior work attempted to define an open router interface analogous to OS interfaces at end-systems [44, 45, 46, 47]. Recent standardization efforts within the IETF reflect this desire [48, 49]. Efforts in the OpenArch and OPENSIG communities succeeded in provisioning QoS in multi-service networks [50, 51, 52, 53]. Whereas these efforts attempt to modularize the architecture and the functionality of *individual* routers, we propose to move the logic (e.g., path computation) currently in the control plane *out* of the routers and control plane altogether into a separate decision plane equipped with network-wide views. Several recent proposals [11, 54, 55] argue for separating the computation of routes from the individual routers. We also argue for placing the key functionality outside of the network but go further in two respects. First, we believe that the architecture should explicitly provide a robust dissemination means to *directly* control the data plane plane, rather than driving the control plane by sending BGP or MPLS messages to routers, as extensive configuration is required before the BGP or MPLS messages can even be delivered. Second, we believe that the management plane should dictate other aspects of network operations *beyond routing* (e.g., packet filtering and quality of service).

Discovery: Techniques for auto-discovery between neighbors have been proposed in ATM with Integrated Local Management Interface (ILMI) [56] and optical networks with Generalized Multi-Label Switching (GMPLS) [57] and Link Management Protocol (LMP) [58]. ATM discovery assumes homogeneous link technology (SONET), OSI control protocol stack, and requires NSAP addresses to be configured first. GMPLS discovery assumes IP control protocols at each switch controller and requires the protocols to be configured first. In the 4D architecture, we aim to design discovery service applicable to multiple network types that requires *zero* pre-configuration. In addition, the discovery service will provide interfaces to the decision plane to enable consistent and explicit management of physical and logical identities, their scopes, their persistence, and their relationships.

7. CONCLUDING REMARKS

There are fundamental questions to be answered in redesigning control and management functions for data networks: How to go from networks that blend decision logic with specific protocols and mechanisms to an architecture that abstracts and isolates the decision logic and admits a range of efficient implementations? How to go from networks that consist of numerous uncoordinated, error-prone mechanisms, to ones where the low-level mechanisms are driven in a consistent manner by network-level objectives? How to go from networks where operators tune parameters, hoping to coax the system to reach a desired state, to one where network *designers* can directly express controls that automatically steer the system toward the desired state? How to go from networks where human operators leverage network-wide views and box-level capabilities at slow timescales in decision-support systems, to one where the network itself leverages this information in real time?

We believe there are huge opportunities for the research community to pursue a more revolutionary clean-slate approach to the problem of network control and management. If successful, the line of research could create an entire landscape of possibilities for networking researchers to deploy their ideas on real networks. Previously closed and proprietary control plane protocols will be replaced by software running on conventional servers. New algorithms and logic for network control can be developed *and deployed*. Ultimately, data networks, equipped with new control and management protocols and software, could be simpler, more robust, more evolvable, and less prone to security breaches.

8. ACKNOWLEDGMENTS

This paper benefited greatly from the extensive comments provided by the anonymous reviewers, and we would like to thank them for their efforts and ideas.

9. REFERENCES

[1] LAN/MAN Standards Committee of the IEEE Computer Society, *IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Common specifications Part 3: Media Access Control (MAC) Bridges*, 1998.

[2] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjalmtysson, and A. Greenberg, "Routing design in operational networks: A look from the inside," in *Proc. ACM SIGCOMM*, August 2004.

[3] LAN/MAN Standards Committee of the IEEE Computer Society, *802.1Q IEEE Standards for Local and metropolitan area networks Virtual Bridged Local Area Networks*, 2003.

[4] S. Sharma, K. Gopalan, S. Nanda, and T. Chiueh, "Viking: A multi-spanning-tree Ethernet architecture for metropolitan area and cluster networks," in *Proc. IEEE INFOCOM*, March 2004.

[5] "Yipes." http://www.yipes.com.

[6] A. Feldmann and J. Rexford, "IP network configuration for intradomain traffic engineering," *IEEE Network Magazine*, pp. 46–57, September/October 2001.

[7] D. Caldwell, A. Gilbert, J. Gottlieb, A. Greenberg, G. Hjalmtysson, and J. Rexford, "The cutting EDGE of IP router configuration," in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*, November 2003.

[8] N. Feamster and H. Balakrishnan, "Detecting BGP configuration faults with static analysis," in *Proc. Networked Systems Design and Implementation*, May 2005.

[9] G. Varghese and C. Estan, "The measurement manifesto," in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*, November 2003.

[10] A. Greenberg, G. Hjalmtysson, D. A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang, "Refactoring network control and management: A case for the 4D architecture," tech. rep., Computer Science Department, Carnegie Mellon University, 2005. Available as http://www.cs.cmu.edu/∼4D/papers/casefor4D-2005.pdf.

[11] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and J. van der Merwe, "The case for separating routing from routers," in *Proc. ACM SIGCOMM Workshop on Future Directions in Network Architecture*, August 2004.

[12] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and Jacobus van der Merwe, "Design and implementation of a Routing Control Platform," in *Proc. Networked Systems Design and Implementation*, May 2005.

[13] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communication Magazine*, October 2002.

[14] J. Rexford, "Route optimization in IP networks," in *Handbook of Optimization in Telecommunications*, Kluwer Academic Publishers, 2005. To appear.

[15] R. Mahajan, D. Wetherall, and T. Anderson, "Towards coordinated interdomain traffic engineering," in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*, November 2004.

[16] D. D. Clark, J. Wroclawski, K. Sollins, and R. Braden, "Tussle in cyberspace: Defining tomorrow's Internet," in *Proc. ACM SIGCOMM*, August 2002.

[17] D. Clark, K. Sollins, J. Wroclawski, D. Katabi, J. Kulik, X. Yang, R. Braden, T. Faber, A. Falk, V. Pingali, M. Handley, and N. Chiappa, "FINAL TECHNICAL REPORT New Arch: Future generation Internet architecture." Available from http://www.isi.edu/newarch/.

[18] A. Myers, E. Ng, and H. Zhang, "Rethinking the service model: Scaling Ethernet to a million nodes," in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*, November 2004.

[19] M. Shand and S. Bryant, "IP fast reroute framework." Internet Draft draft-ietf-rtgwg-ipfrr-framework-03.txt, work in progress, June 2005.

[20] P. Pan, G. Swallow, and A. Atlas, "Fast reroute extensions to RSVP-TE for LSP tunnels," May 2005. RFC 4090.

[21] N. Duffield, "A framework for packet selection and reporting," January 2005. Internet Draft draft-ietf-psamp-framework-10.txt, work in progress.

[22] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. Operating Systems Design and Implementation*, pp. 255–270, December 2002.

[23] J. Turner, "Open Network Laboratory." http://onl.arl.wustl.edu/. Last visited 7/2005.

[24] N. McKeown, "The NetFPGA project." http://yuba.stanford.edu/NetFPGA/. Last visited 7/2005.

[25] NSF CISE, "The GENI initiative." http://www.nsf.gov/cise/geni/.

[26] M. Sirbu, A. Greenberg, H. Zhang, and D. A. Maltz, "Municipal networks: Catalysts for change." Presented to NSF, March 2004. Available as http://www.100x100network.org/talks/2004-03-05-nsf-muninet-pitch-public.ppt.

[27] A. Shaikh, M. Goyal, A. Greenberg, R. Rajan, and K. K. Ramakrishnan, "An OSPF topology server: Design and evaluation," *IEEE J. Selected Areas in Communications*, May 2002.

[28] A. Shaikh and A. Greenberg, "OSPF monitoring: Architecture, design, and deployment experience," in *Proc. Networked Systems Design and Implementation*, March 2004.

[29] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the Internet," in *Proc. ACM SIGCOMM*, pp. 3–10, 2003.

[30] A. Chiu and J. Strand, "Control plane considerations for all-optical and multi-domain optical networks and their status in OIF and IETF," *Optical Networks Magazine*, vol. 4, no. 1, pp. 26–35, 2003.

[31] T. Russell, *Signaling System #7*. McGraw-Hill, 2nd ed., 1998.

[32] "Introduction to CCITT signalling system no. 7." ITU-T Standard Q.700.

[33] "Introduction to intelligent network (IN) capability set 1." ITU-T Standard Q.1211.

[34] J. M. Smith and S. M. Nettles, "Active networking: One view of the past, present and future," *IEEE Transactions On Systems, Man and Cybernetics*, vol. 34, pp. 4–18, Feb 2004.

[35] "Cariden MATE framework." http://www.cariden.com/products/. Last visited 9/2005.

[36] "OpNet SP Guru." http://www.opnet.com/products/spguru/home.html. Last visited 9/2005.

[37] "Arbor Networks Peakflow." http://www.arbornetworks.com/products_sp.php. Last visited 9/2005.

[38] R. Chadha, G. Lapiotis, and S. Wright, "Policy-based networking," *IEEE Network Magazine*, vol. 16, pp. 8–9, 2002.

[39] Z. Kerravala, "Configuration management delivers business resiliency." The Yankee Group, Nov 2002.

[40] D. Oppenheimer, A. Ganapathi, and D. A. Patterson, "Why do Internet services fail, and what can be done about it?," in *Proc. USENIX Symposium on Internet Technologies and Systems*, 2003.

[41] R. Mahajan, D.Wetherall, and T. Anderson, "Understanding BGP misconfiguration," in *Proc. ACM SIGCOMM*, August 2002.

[42] "WANDL IP analysis tools." http://www.wandl.com/html/ipat/IPAT_new.cfm. Last visited 1/2005.

[43] "OPNET NetDoctor." http://www.opnet.com/products/modules/netdoctor.html. Last visited 1/2005.

[44] G. Hjalmtysson, "The Pronto platform - a flexible toolkit for programming networks using a commodity operating system," in *Proc. International Conference on Open Architectures and Network Programming (OPENARCH)*, March 2000.

[45] L. Peterson, Y. Gottlieb, M. Hibler, P. Tullmann, J. Lepreau, S. Schwab, H. Dandekar, A. Purtell, and J. Hartman, "A NodeOS interface for active networks," *IEEE J. Selected Areas in Communications*, March 2001.

[46] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The Click modular router," *ACM Trans. Computer Systems*, August 2000.

[47] T. V. Lakshman, T. Nandagopal, R. Ramjee, K. Sabnani, and T. Woo, "The SoftRouter architecture," in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*, November 2004.

[48] A. Doria, F. Hellstrand, K. Sundell, and T. Worster, *General Switch Management Protocol (GSMP) V3*. Internet Engineering Task Force, 2002. RFC 3292.

[49] "Forwarding and Control Element Separation Charter." http://www.ietf.org/html.charters/forces-charter.html.

[50] A. T. Campbell, I. Katzela, K. Miki, and J. Vicente, "Open signaling for ATM, Internet and mobile networks (OPENSIG'98)," *SIGCOMM Computer Communications Review*, vol. 29, no. 1, pp. 97–108, 1999.

[51] A. Lazar, S. Bhonsle, and K. Lim, "A binding architecture for multimedia networks," *Journal of Parallel and Distributed Systems*, vol. 30, pp. 204–216, November 1995.

[52] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. C. Liaw, T. Lyon, and G. Minshall, "Ipsilon's general switch management protocol specification version 1.1." RFC 1987, August 1996.

[53] A. Banerjea, "The XBONE: Building overlay networks," in *Proc. Workshop on Open Signaling for ATM, Internet and Mobile Networks*, 1998.

[54] O. Bonaventure, S. Uhlig, and B. Quoitin, "The case for more versatile BGP route reflectors," July 2004. Internet Draft draft-bonaventure-bgp-route-reflectors-00.txt, work in progress.

[55] A. Farrel, J.-P. Vasseur, and J. Ash, "Path computation element (PCE) architecture." Internet Draft draft-ash-pce-architecture-01.txt, July 2005.

[56] ATM Forum Technical Committee, *Integrated Local Management Interface (ILMI) Specification Version 4.0*, 1996.

[57] L. Berger, *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description*, 2003. RFC 3471.

[58] J. Lang, *Link Management Protocol (LMP), draft-ietf-ccamp-lmp-10.txt*, October 2003.