Title: An API for Network Services on the Internet | *This is an original proposal.*

Keywords: network services, middleboxes, programmable networks

Background: The explosive growth of Internet applications stemmed partially from the simplicity of the Internet's design. The Internet aimed to be agnostic to applications developed on top of it, providing only a common goal of best-effort delivery of packets. All applications share a simple interface to the network: specify a destination, a payload, and a few other details, and the network transports the payload to the desired destination. This simple functionality supports great diversity in Internet technologies, from the web, to streaming video, to 'Voice over IP' telephone service. Wildly different applications can take place over the same medium, due to the simple flexibility of its design.

However, the constant drive to improve performance and security of higher layer applications has led to deployment of new functionality in the Internet. Today's networks provide limited support for caches, firewalling, prioritized traffic, and a variety of other additions. Typical deployments of these features often involve on-the-path 'middleboxes' that perform specialized functionality.

This deployment model, despite providing new features, constrains higher layer applications because it does not provide end hosts with any interface to the new functionality. Without explicit direction from applications, middleboxes are forced to 'classify' traffic and guess how to apply their services. These inferences may not always be correct, particularly when new applications are completely unrecognized by the network. For example, a network firewall may provide a host security by blocking traffic destined to unknown ports. When a new application makes use of a port that the firewall doesn't recognize, the application may have it's inbound traffic blocked. Because there is no interface to communicate with the firewall, the application has no mechanism to modify the networks behavior. In short, there is no way to opt in, opt out, or configure any of the embedded network functionality.

Proposal: I propose to provide an programmable interface to end hosts to directly invoke and configure the network processing performed on their traffic. Under this framework, the network will expose to end hosts high level interfaces, which describe collections of related functionality. Each interface will consist of a set of related invocable behaviors. For example, a firewall interface might provide a function to set a rule to block traffic matching a certain pattern, as well as a function to remove existing rules. An application can invoke these functions to configure network functionality relevant to it. This abstraction balances the needs of both end hosts and service providers. For application programmers at end hosts, network functionality is readily exposed to them. Behavior of the network is modifiable, allowing developers to explicitly inform the network how to process their traffic. For network providers, the interface abstraction enables a network provider to implement the functionality however they wish. It doesn't matter if the functions are performed in a single middlebox, load-balanced across multiple middleboxes, processed by programmable routers, or even outsourced to another network provider who performs the functionality on their behalf. This ability to invoke functionality in inter-domain settings is very powerful; it means that network providers can collaborate to provide services which require hop-by-hop support across the Internet (e.g. Quality-of-Service) and that providers can enable functions for their users which they themselves do not have infrastructure for, but that neighboring networks do. Thus, a programmable interface abstraction allows network providers to provide complex functionality while keeping the services simple from the perspective of end hosts.

Intellectual Merit: This project is challenging because it involves re-considering every aspect of the Internet's design. In discovering available interfaces, I will have to design a mechanism for end hosts to communicate directly with their networks, and for networks to communicate directly with each other. I will have to reconsider routing, as certain applications may prefer to traverse networks that provide support for interfaces they wish to invoke, rather than the BGP-announced path. I will have to pay close attention to the fundamentals of inter-domain policy, and make sure my design allows networks to retain control over which customers, peers, and providers they provide functionalities to. In every step of my design, I will have to consider security, as malicious users may try to interfere with others' access to network functionalities, or use the new functionalities to amplify or anonymize their attacks.

A fundamental challenge to this design is trying to envision the needs of the future. Just as the original designers of the Internet never planned for the diversity of services we have today, it is impossible to fully envision the applications and network functionalities that will be deployed beyond what we have today.

Related Work: This is not the first proposal to address the issue of new, non-forwarding functionality in the network. Active Networks [1] envisioned snippets of code embedded with data that would be executed by any router processing a packet. The authors envisioned that this 'active' code could be used to implement services like those deployed in middleboxes today. For a variety of reasons outside the scope of this proposal, services today are instead deployed in networks through middleboxes, rather than active code. My proposal addresses this modern state of affairs.

Several projects [2, 3] have suggested that middleboxes should be directly invoked by traffic. However, these proposals require that the end user name the middlebox itself, rather than the functionality it performs. As a result, invocation becomes complex as end hosts are responsible for identifying appropriate middleboxes to perform desired functionalities, knowing how to invoke them, and how to compose functionalities (if at all). My proposal decouples functionality from how it is performed in the network. This allows for flexibility and control on behalf of the network provider, while keeping interaction simple at the end hosts.

Current work in making routing hardware more flexible [4] sets the stage for my proposal, providing mechanisms to easily make changes to support new network designs like this.

Qualification: My previous research experience has trained me in core skills for project leadership, development of solutions to unsolved problems, implementation and evaluation of my solutions, and publication of completed research. My background in Internet measurement has provided me a solid foundation of networking fundamentals, as have my graduate and undergraduate networking coursework. While this is my first project in Internet architecture, I am confident that under the supervision of my advisor Dr. Sylvia Ratnasamy, I will have proper resources and support to make contributions in this area.

Broader Impacts: My proposal identifies a threat to the future flexibility of the Internet: the lack of an interface to network functionalities will limit innovation in networked applications. My proposal provides a way to continue to provide new functionalities in a principled manner, keeping the door open for further innovation. As novel Internet applications have created entirely new industries while dramatically improving quality of life, it is important that the Internet remain an open, flexible playing field for new technologies as it has in the past.

[1] D. Wetherall, U. Legedza, and J. Guttag, "Introducing new Internet services: Why and how," in IEEE Network, May/June 1998.

[2] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker, "Middleboxes no longer considered harmful," in Proc. USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2004.

[3] L. Popa, N. Egi, I. Stoica, and S. Ratnasamy, "Building extensible networks with rule-based forwarding (RBF)," in Proc. USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2010.

[4] N. McKeown, T. Anderson, H. Balakrishnan, H. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turn, "OpenFlow," in *Computer Communication Review*, April 2008.