Supporting Network Evolution and Incremental Deployment with XIA

Robert Grandl*, Dongsu Han, Suk-Bok Lee, Hyeontaek Lim, Michel Machado†, Matthew Mukerjee, David Naylor

Carnegie Mellon University * University of Wisconsin-Madison † Boston University

ABSTRACT

eXpressive Internet Architecture (XIA) [1] is an architecture that natively supports multiple communication types and allows networks to evolve their abstractions and functionality to accommodate new styles of communication over time. XIA embeds an elegant mechanism for handling unforeseen communication types for legacy routers.

In this demonstration, we show that XIA overcomes three key barriers in network evolution (outlined below) by (1) allowing end-hosts and applications to start using new communication types (e.g., service and content) before the network supports them, (2) ensuring that upgrading a subset of routers to support new functionalities immediately benefits applications, and (3) using the same mechanisms we employ for 1 and 2 to incrementally deploy XIA in IP networks.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms

Design

Keywords

Internet architecture, evolution, multiple communication styles

1 Introduction

What makes network evolution so hard? The "narrow waist" design of the Internet has been tremendously successful. However, the Internet does not facilitate a clean, incremental path for the adoption of new capabilities at the narrow waist. Two main barriers against evolution are:

B1 It is not possible for an end-host or an application to use new functionality until the network supports it.

B2 There is no built-in mechanism to support incremental deployment of new functionality. For example, enabling a single router in a IP network to support content-centric networking is fruitless if that traffic will simply be tunneled through the network using IP.

XIA [1] is a clean-state architecture that solves these issues. However, a clean-state design has its own problems:

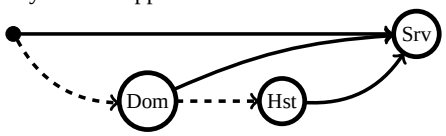
SIGCOMM'12, August 13–17, 2012, Helsinki, Finland.
ACM 978-1-4503-1419-0/12/08.

B3 Incrementally deploying a new architecture over IP is non-trivial and often requires careful configuration.

In this demo, we show that XIA effectively overcomes these three barriers. The rest of this section provides a brief summary of XIA's core mechanism for evolution.

How does XIA solve this? The key architectural element that XIA adds to provide evolvability is the notion of an extensible set of *principal* communication types (e.g., host-to-host, service-oriented, and content-oriented). By selecting the appropriate principal for a given communication, applications can directly express the *intent* of that communication.

XIA's addresses can simultaneously express both identifiers for a "new" principal type (e.g., a content or service identifier) and one or more backwards compatible identifiers (e.g., host identifiers) called *fallbacks*. XIA allows for this flexibility by specifying addresses as *directed acyclic graphs* (DAGs). For example, the following DAG specifies that the intent is to reach a service ("Srv"), but routers may instead forward the packet to host "Hst" in destination network "Dom" if they do not support services.



Thus, by using fallbacks, two hosts that have been upgraded to understand the service principal can begin to use it even if no routers support the principal type, overcoming **B1**. As some routers are upgraded to support services, applications (without any modification) will begin to see benefits (**B2**).

How can XIA be deployed? To address **B3**, we introduce a new principal type: IPv4 addresses, or "4IDs." 4IDs leverage the power of the XIA addressing scheme (namely DAGs and fallbacks) to support old communication styles (IPv4). We do this by encoding IPv4 addresses within these 4ID identifiers, which are included in DAGs as fallbacks.

Using 4IDs, we can effectively connect an XIA island to the XIA backbone. A 4ID specifies an IPv4 entry point (a dual-stack router) of an XIA network, and is obtained through name resolution. Thus, we eliminate the burden of manually configuring tunnels between XIA islands. When a packet with a 4ID fallback can no longer progress though the XIA network, we use the IPv4 address and dynamically tunnel the packet to the XIA network whose IPv4 entry address is specified by the 4ID.

This begs the question: where do 4IDs come from? The sender must know of an IPv4 address that serves as an ingress point to the destination XIA network. We handle this by having each domain register a set of IPv4 addresses that it is willing to use as ingress

points. These addresses are returned as 4IDs during the name resolution process.

2 System Overview

We have implemented support for host-, service- and content-oriented communication over XIA. This includes an XIA router with principal-specific processing for each of these principals as well as each type's end-host stack and socket API.

Router: We have implemented a prototype router in Click that processes XIA packets. All routers support host-based communication. Additionally, routers may support the content, service, and/or 4ID principal types. If content support is enabled, our routers can act as in-network caches. The router forwards packets that contain content to the content module, which stores content and inserts a routing entry for the cached content to direct future requests to itself. If services are supported, anycast and failover support are implemented.

End-host: We have implemented end-host support for service- and content-oriented communication and exposed this functionality to application developers through a sockets API resembling standard Berkeley sockets. To invoke a particular transport type (and thereby choose either service- or content-oriented communication), developers pass an extra parameter when creating a socket.

Applications: See below.

3 Demo Scenarios

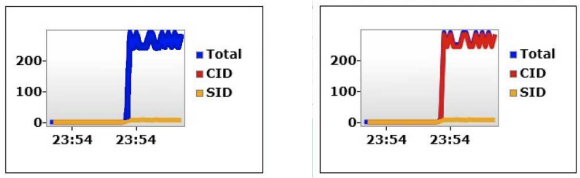
We will demonstrate our solutions to the three barriers discussed in §1, using both the service and content principal types as examples of new functionality we wish to add to the network incrementally. To do so, we will use two small XIA networks we have deployed on GENI and at Carnegie Mellon University (CMU).

Applications We have developed some simple applications that run over XIA and are in the process of developing more. Some subset of these will be used to demo XIA:

- Web server (and proxy)
- Video streaming server (uses content support)
- "Stock Ticker" server (uses service support)
- "VoIP" (VoXIA, really) — in development
- A fun, interactive, application that you too can try (see "Attendee Participation" below) — in development

Demonstrating Solutions to Evolution Barriers We will first demonstrate that XIA allows applications to begin using new principal types without network support (**B1**) by running applications that use the service and content principals over a network of XIA routers in GENI that support only the host principal. The applications run without problems because the routers use fallbacks to satisfy the application's intent. The audience will be able to observe the amount of traffic (by principal type) flowing on each link through our real-time visualization tool that monitors activity on GENI. Figure 1(a) illustrates this example; none of the packets are being processed using CID- or SID-specific processing, meaning the "legacy" routers are using the host ID fallbacks.

We will then demonstrate that applications can benefit by upgrading even just one router in the network to support a new principal type (**B2**). By enabling support for the content principal type at one of the routers in our demo network, our video streaming application can immediately benefit from in-network content caching. The audience will be able to observe that the video is being served



(a) Routers forward based on fallback host IDs

(b) Upgraded routers now forward packets based on CIDs

Figure 1: Video streaming over XIA networks with and without support for the content principal.

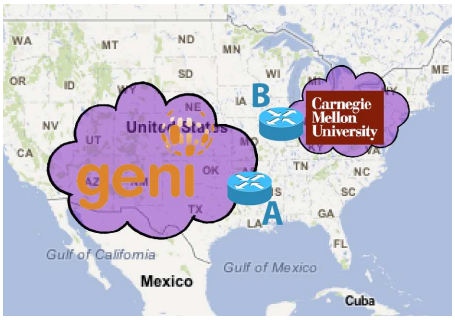
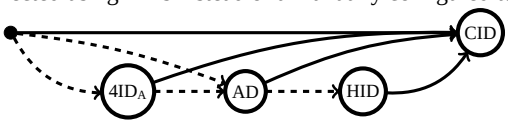


Figure 2: We use the IPv4 principal type to communicate between XIA networks without explicit tunnels.

from a nearby cache instead of the server we fell back to previously. On the upgraded router the traffic will look like Figure 1(b): the same packets are passing through the link, but our router now understands that we are using content IDs (CIDs).

Finally, we can show how XIA might be incrementally deployed in today's IP Internet (**B3**). By adding 4IDs as fallback nodes in our DAG addresses (see below), we will allow XIA-enabled dual-stack machines at CMU to communicate with our XIA network on GENI (Figure 2). The GENI XIA backbone and the CMU XIA island will be connected using 4IDs instead of a manually configured tunnel.



Attendee Participation We are developing an interactive application to run over XIA that will allow us to demonstrate 4IDs on a larger scale. The application will be added to the XIA source code (already available[1]) and, environment permitting, we hope to allow conference attendees to download our VM and join the fun. Each XIA-enabled VM would function as its own singleton XIA network; client applications connecting to our servers would use 4IDs as fallbacks to traverse the IPv4 Internet. The demo will provide entertainment as well as first-hand experience with how two disjoint XIA networks can be connected using 4IDs.

4 Acknowledgments

This research was supported in part by the National Science Foundation under awards CNS-1040757, CNS-1040800, and CNS-1040801.

5 References

[1] D. Han, A. Anand, F. Dogar, B. Li, H. Lim, M. Machado, A. Mukundan, W. Wu, A. Akella, D. G. Andersen, J. W. Byers, S. Seshan, and P. Steenkiste. XIA: Efficient support for evolvable internetworking. In *Proc. 9th USENIX NSDI*, San Jose, CA, Apr. 2012.

[1] `http://www.cs.cmu.edu/~xia/`