XIA: Architecting a More Trustworthy and Evolvable Internet

David Naylor
Carnegie Mellon University
dnaylor@cs.cmu.edu

Matthew K. Mukerjee
Carnegie Mellon University
mukerjee@cs.cmu.edu

Patrick Agyapong
Carnegie Mellon University
pagyapong@cmu.edu

Robert Grandl
University of Wisconsin
rgrandl@cs.wisc.edu

Ruogu Kang
Carnegie Mellon University
ruoguk@cs.cmu.edu

Michel Machado[*]
Boston University
michel@bu.edu

ABSTRACT

Motivated by limitations in today's host-centric IP network, recent studies have proposed clean-slate network architectures centered around alternate first-class principals, such as content, services, or users. However, much like the host-centric IP design, elevating one principal type above others hinders communication between other principals and inhibits the network's capability to evolve. This paper presents the eXpressive Internet Architecture (XIA), an architecture with native support for multiple principals and the ability to evolve its functionality to accommodate new, as yet unforeseen, principals over time. We present the results of our ongoing research motivated by and building on the XIA architecture, ranging from topics at the physical level ("how fast can XIA go") up through to the user level.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols; K.4.1 [**Computers and Society**]: Public Policy Issues—*privacy*

Keywords

Internet architecture, network addressing

1. INTRODUCTION

The "narrow waist" design of the Internet has been exceptionally successful, helping to create an flourishing ecosystem of applications and protocols above the waist provided by the IP protocol and supporting diverse physical layers and access technologies below. However, the Internet, almost by design, does not provide a clean path for the incremental deployment of new capabilities. This shortcoming is clearly illustrated by the 15+ year deployment history of IPv6 and the difficulty of deploying primitives needed to secure the Internet. We argue that these problems are fundamentally tied to the difficulty of modifying in any deep way the information shared between edge and transit domains, which we will informally call the "contract", that informs core routers how to act upon packets. As a result, new functionality added to the network, such as caches and other middleboxes, must be transparent to protocols and applications running on hosts, adding complexity, increasing fragility, and hindering adoption of new technologies.

We make the case for a new Internet architecture, called the eXpressive Internet Architecture, or XIA, to address these problems from the ground up. XIA maintains several features of the current Internet such as a narrow waist and default-on communication, but modifies and extends the network layer in order to achieve the following goals:

Be trustworthy: Security, broadly defined, is the most significant challenge facing the Internet today.

Support long-term evolution of usage models: The primary use of the original Internet was host-based communication. With the introduction of the Web, communication has shifted to content retrieval and service access, and other usage models are likely to emerge in the future. The future Internet should support communication both between today's popular entities and future new entities.

Support long-term technology evolution: Advances in link technologies as well as storage and compute capabilities at both end-points and network devices have been dramatic. While the Internet does allow easy integration of new link technologies, it must also enable the evolution of functionality on all end-point and network devices in response to technology improvements and economic realities.

Support explicit interfaces between network actors: The Internet encompasses a diverse set of actors playing different roles with different goals and incentives. The architecture must support well-defined interfaces that allow these actors to function effectively.

In the rest of the paper, we first introduce the three principles that form the basis of XIA: expressiveness, intrinsic security, and flexible addressing. We also describe a specific instantiation of the architecture and our prototype realizations (Section 2). Sections 3 through 9 discuss research results that show how the architecture can be implemented and used to address a variety of network challenges. We conclude in Section 10.

2. XIA DATAPLANE ARCHITECTURE

We start by describing XIA at three levels of abstraction: principles, concrete specification, and implementation.

2.1 Architectural Principles and Invariants

XIA retains several concepts that have proven effective in today's Internet, e.g., packets, address-based forwarding, and the notion of layering with a narrow waist at the internetwork layer. XIA also introduces several new ideas that are based on three design principles, as articulated in [3].

[*] Additional authors can be found in Section 11 of the paper.

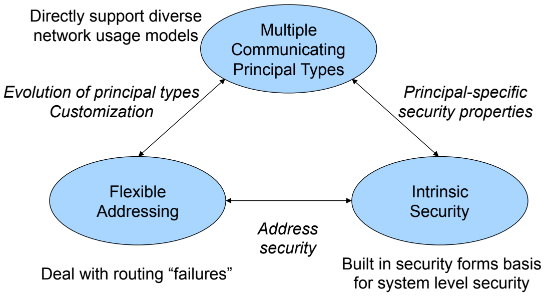
Directly support diverse network usage models
Multiple Communicating Principal Types
Evolution of principal types Customization
Principal-specific security properties
Flexible Addressing
Intrinsic Security
Address security
Deal with routing "failures"
Built in security forms basis for system level security

Figure 1: XIA Principles

The first principle is that the architecture must explicitly support **communication between diverse entities** that we refer to as *principals*. Examples include hosts, services and content, and additional entities motivated by future usage models. Each principal type is associated with a different contract between domains in the network and, therefore, enables a different communication style. This design principle directly addresses our evolvability goals: principal types allow users to express their intent to the network, creating many opportunities for in-network optimization and innovation in support of both today's and future usage models. For example, it enables XIA to provide many of the goals of alternative architectures such as content-centric networks [28, 27] that better support various forms of content retrieval, and service-centric networks [24] that provide powerful primitives such as service-level anycast.

The second principle is that each communication operation should have **intrinsic security** properties associated with it that follow as a direct result of the system design. Intrinsic security properties are principal-specific and allow each entity to validate that it is communicating with its intended counterpart without needing access to external information or configuration. XIA achieves this goal through the use of self-certifying identifiers for all principal types, providing a useful set of integrity and accountability properties. Secure identifiers can also be used to bootstrap systematic mechanisms for trust management, leading to a more secure Internet in which trust relationships are exposed to users. This principle directly addresses the first part of the vision.

The final principle is that the architecture must support **flexible addressing** that, at the very least, supports fallback addressing. An XIA *fallback* is addressing information that can be used by routers to deliver a packet even if the router does not support all of the destination information included in the packet, e.g., because the information includes a newly-introduced principal type. Combined with the first principle, flexible addressing directly supports evolvability (goals two and three). It also contributes to the last aspect of our vision because it gives users some control over how communication operations are performed.

Figure 1 summarizes the three XIA design principles and their benefits. While the principles are largely orthogonal, using them in combination offers additional benefits, as illustrated by the edges in the figure. For example, the combination of multiple principal types and flexible addressing supports evolvability. We describe some concrete examples of these benefits later in the paper.

2.2 A Concrete Architectural Specification

The second level of abstraction of XIA is a more concrete "IETF-level" specification that can be implemented by different parties and will, at a minimum, guarantee interoperability between networks [13]. This specification includes both the syntax (e.g., header formats) and semantics (e.g., forwarding behaviors). The specification of an XIA principal type includes the semantics of communication with principals of this type, the processing that is required to forward packets with addresses of this type, and the intrinsic security properties of the prinicpal. The initial XIA architecture defines four basic XIA identifier (XID) types and their intrinsic security properties:

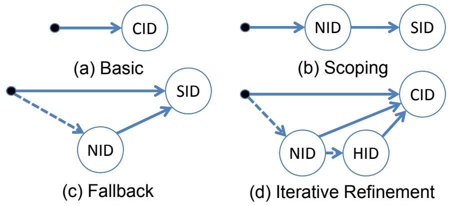

Figure 2: Examples of address DAGs

- **Host XIDs**: HIDs support unicast host-based communication semantics similar to IP. HIDs are a hash of the host's public key, supporting AIP-style accountability [4]. HIDs define *who* you communicate with.
- **Service XIDs**: SIDs support communication with (typically replicated) services and realize anycast forwarding semantics. SIDs allow clients to verify the identity of the service. SIDs define *what entities do*.
- **Content XIDs**: CIDs allow hosts to retrieve content from "anywhere" in the network, e.g., content owners, CDNs, caches, etc. CIDs are defined as the hash of the content, so the client can verify the correctness of the received content. CIDs define *what it is*.
- **Network XIDs**: NIDs specify a network, i.e., an autonomous domain, and they are primarily used for scoping. They allow an entity to verify that it is communicating with the intended network [4].

We have introduced numerous experimental XID types to date, notably: 4IDs, representing IPv4 addresses, and ScIDs, representing SCION paths. See Sections 5 and 7 for details.

The third XIA principle, flexible addressing, is realized by using Directed Acyclic Graphs (DAGs) of XIDs as addresses. DAGs are highly flexible and allow packets to express fallbacks as well as scoping to realize user intent. The simplest eXpressive Internet Protocol (XIP) address has only a "dummy" source •, representing the conceptual source of the packet, and the intent as the sink, e.g., a CID in Figure 2(a). This type of flat addressing is useful in some environments, e.g., LANs, but will not necessarily scale to the public Internet. DAGs can also be used to implement scoping, as shown in Figure 2(b). Starting with the source •, routers will first deliver the packet to a network NID. Once the NID is reached, routers will use the SID to deliver the packet to the intended service. This DAG is similar to the `[network_prefix, endpoint_id]` format of IP addresses. Figure 2(c) also shows a fallback NID and a fallback edge, shown as a dotted line. The fallback edge is used by a router if the intent SID is not recognized or available. Considering that SIDs have anycast semantics, packets using this DAG will be delivered to the closest service for a globally routable SID, but will be delivered to the service instance in network NID if global SID routing fails. Finally, the address DAG in Figure 2(d) combines the two mech-

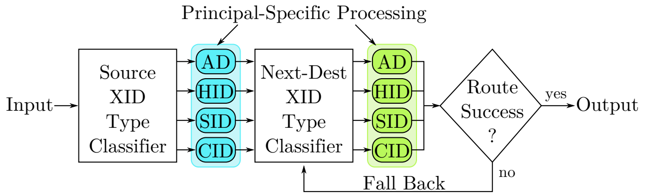
Principal-Specific Processing
Input
Source XID Type Classifier
AD
HID
SID
CID
Next-Dest XID Type Classifier
AD
HID
SID
CID
Route Success ?
yes
Output
no
Fall Back

Figure 3: XIA Forwarding Engine

anisms. Note that any router along the fallback path can satisfy the intent directly (solid edges to CID).

Figure 3 shows a simplified diagram of how packets are processed in an XIA router. The edges represent the flow of packets among processing components. Shaded elements are principal-type specific, whereas other elements are common to all principals. Our design isolates principal-type specific logic to make it easier to add support for new principals. The XIA packet header includes a pointer to a node within the destination address to indicate the current forwarding progress within the DAG. Using that node as a starting point, the router examines the outgoing edges in priority order, invoking type-specific forwarding logic. If the XID is recognized, the packet is forwarded; otherwise it is returned to the classifier, which will try the next edge. More details, including a performance evaluation of the forwarding engine, can be found in [13].

The XIA interfaces enable trade-offs and "tussles" [7] of control between actors. For example, compared to IP, XIA's XID types and address DAGs offer users more control over how communication occurs. SCION path selection also offers user control over what path is used to reach a destination (details in Section 7).

2.3 Prototyping Efforts

Finally, the lowest-level specification of XIA consists of two XIA prototypes that support network-level experiments. Our first prototype is based on the Click modular router [21]. As shown in Figure 4, it includes the XIA network protocol stack, a socket API, support for caching, and network bootstrap and support services (XIA versions of ARP, DHCP, name service, etc.). The socket API supports datagram and streaming using calls similar to POSIX calls, and the retrieval of content "chunks" identified by CIDs. The XIA prototype is available as Apache-licensed open source on GitHub (`http://www.github.com/xia-project/xia-core`). We also provide instructions and scripts for running experiments on GENI and in virtual machine environments.

More recently, we have pursued a native implementation of XIA in the Linux kernel in order to explore the feasibility of using an XIA stack in production environments (`https://github.com/AltraMayor/XIA-for-Linux`). We are also starting to use the Linux XIA implementation as a vehicle for demonstrating XIA's ability to evolve. Specifically, we are porting designs from alternative future Internet architectures into XIA while retaining the benefits of those alien architectures. While implementing new, medium-size principals in the kernel requires more coding effort than implementing them in Click, Linux XIA better suits complex designs that have already been realized in Linux. For example, we have successfully ported the Linux implementation of Serval [24] to XIA and leveraged XIA's features to improve Serval. In its original form, Serval gives up intrinsic security in its ServiceID identifiers in order to support global reachability; a demand that stems from its being a standalone design. Thanks to DAG-based addresses, the XIA version of Serval delegates the global reachability problem to other principals while making its ServiceID identifiers intrinsically secure. Not only does this change preserve Serval's original behavior, it also provides a way to protect XIA Serval from on-path attacks, thereby hardening the original design.

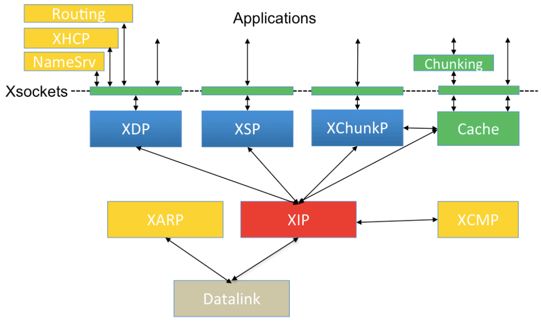

Figure 4: XIA Prototype Protocol Stack

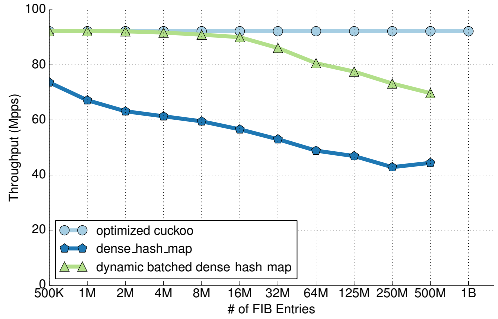

Figure 5: Packet forwarding throughput in Mpackets per second as the size of the FIB increases in Cuckooswitch.

In the rest of the paper, we summarize several research efforts on the implementation (Section 3) and use (Sections 4 and 5) of the XIA principles shown in Figure 1. We present research on extending XIA's flexiblity to other components of the XIA protocol stack (Sections 6 and 7). Finally, we look at how XIA may impact various actors in the Internet, including operators, CDNs, and users (Sections 8 and 9).

3. FORWARDING AT LINE SPEED

An important question is whether XIA's flexible DAG addressing can support high speed forwarding on a variety of router and switch platforms. This question arises in large part because multiple candidate forwarding edges may need to be evaluated at each routing hop in order to determine which next-hop to use. Can this be done at speeds comparable to IP? We are addressing this question through four avenues, three of which have promising results thus far: our Click prototype, the native Linux kernel implementation, a high-speed software router implementation using the Intel Data Plane Development Kit, and future plans to evaluate XIA in the context of TCAM-based forwarding such as is found in many hardware switches and OpenFlow.

Regarding XIA implementations, our Click implementation showed that XIA forwarding adds only marginal overhead compared to traditional IP forwarding [13], at least within the context of Click. For the native Linux implementation, we are exploring a route caching-based approach,

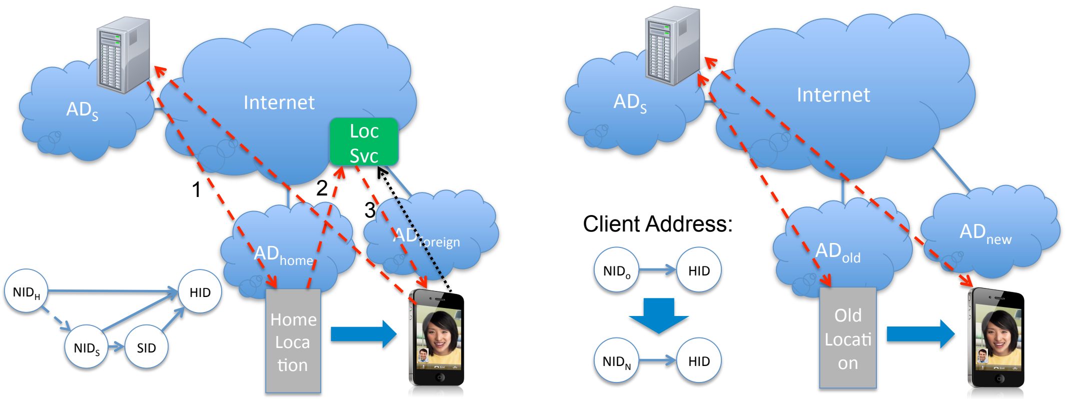

Figure 6: Supporting mobility: first contact (left) and during a session (right)

reflecting the higher likelihood of common destinations for packet flows received by a single host. Our software solution leverages fast data structures to support efficient lookups and routing decisions indexed by sequences of XIDs, mirroring the prioritized nature of next-hop candidate edges within XIP addresses. A technically intricate aspect of these data structures is their dynamic aspect: they preserve efficiency and correctness even when routing daemons perform frequent table updates. Our methods are also self-tuning, as frequently looked-up sequences are cached on a fast path. With promising performance results running Linux XIA on a single host, we are currently building a testbed to evaluate the performance of our solution against Linux IP at high speeds and larger scale.

To achieve higher speeds than the several million packets-per-second achievable using Click or Linux, we turned to the Intel DPDK ("Data Plane Development Kit"). We designed and implemented a flat address forwarding table ("FIB") that can efficiently scale to *billions* of entries while saturating 80 gigabits per second with 192-byte or larger packets (Figure 5). While this is not yet a full XIA implementation, it is an extremely promising first step for pushing the limits of extremely large scale flat host (HID), service (SID), or content (CID) forwarding tables [31].

Our current FIB implementation clearly shows the importance of batching and parallel memory lookups for high-speed operation. By manually implementing a parallel batch lookup scheme that prefetches FIB entries for many packets at a time to mask memory latency, the table achieves extremely high lookup rates—nearly 300 million per second, in comparison to the roughly 90 million packets per second that the machine's PCI buses can sustain. As a result, we are optimistic that a semi-parallel, heavily batched XIA lookup mechanism will sustain full forwarding speeds.

4. SUPPORTING MOBILITY

We have explored how XIA's DAG-based addresses, service identifiers, and intrinsic security can be used to support both establishing and maintaining a connection with a mobile end-point. There has been a great deal of work on supporting mobility in the Internet, e.g., [29, 12], but two concepts have emerged as the basis for most solutions. The first is the use of a rendezvous point that tracks the mobile device's current address (locator) and is used to reach a mobile host. The second is the observation that, once a connection is established, the mobile host can inform its stationary peer of any change in location, without the need for a rendezvous point. Both of these concepts are fairly easy to implement in XIA, in part because XIA distinguishes between the identifier for an endpoint (HID or SID) and its locator (an address DAG). We now sketch a possible implementation of each concept, using HIDs as an example.

To establish a connection with a mobile device, a communicating client will first do a name lookup. As shown in Figure 6(a), the address DAG that is returned could include the device's "home" locator (NID_H : HID) as the intent, and the location of a rendezvous service (NID_S : SID)), If the device is attached to its home network, it is delivered by following solid arrows in the DAG. If not, it is automatically delivered to the rendezvous service using the fallback. The rendezvous service could be hosted by the home network, or could be a distributed commercial service. The service forwards the packet to the mobile device, which then establishes the connection with the communicating client, as shown by the dashed arrows in Figure 6(a). It also provides its new address DAG, signing the address change using the private key associated with its HID. Clearly, many other solutions are possible. For example, the mobile device could register the DAG of the rendezvous point with its name service when it leaves its home network.

To maintain a session during mobility, the mobile host keeps the stationary host informed of its new address [13], as shown in Figure 6(b). Similar to the methods above, use of intrinsically secure identifiers simplifies securing this process since the mobile host can sign the address change using the private key associated with its HID or SID.

The mechanisms described above only support basic connectivity, but optimizations are critical in practice. Our previous work on Tapa [9] explores how content and service-centric support in the network can help optimize data transfers for mobile clients. Examples include using a content cache to stage content from remote servers and breaking the end-to-end transport connection into segments.

5. INCREMENTAL DEPLOYMENT

One of the bigest hurdles facing any New Network Architecture (NNA), such as IPv6, IP Multicast, and XIA, is incremental deployment over the Old Network Architecture's (ONA) infrastructure. To better understand the issues involved, we have studied incremental deployment techniques to create a design space based on how different approaches addressed four challenges: selecting an egress (resp. ingress)

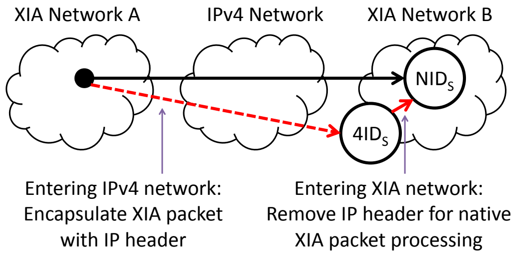
XIA Network A
IPv4 Network
XIA Network B
NID$_S$
4ID$_S$
Entering IPv4 network: Encapsulate XIA packet with IP header
Entering XIA network: Remove IP header for native XIA packet processing

Figure 7: Incremental deployment of XIA using 4IDs

gateway from the NNA source (resp. destination) network, reaching the egress gateway of the source NNA network from the source host, and reaching the ingress gateway of the NNA destination network from the NNA source network.

Based on the above design space, we were able to map all existing designs into two broad classes of solutions and we identified two new classes of designs:

1 **Static tunneling** approaches maintain connectivity between specific disjoint NNA network clouds by explicitly maintaining "static tunnels", e.g., 6in4 [23];
2 **Address mapping** mechanisms send packets that encode an ONA address of an ingress router within their NNA address, e.g, Teredo [17];
3 **Flexible addressing (new)** has packets store the ingress ONA (e.g., IPv4) address explicitly and separately from the NNA destination address.
4 **Smart control planes (new)** improve upon flexible addressing approaches by leveraging recent centralized control techniques to select the egress/ingress pair to use for any source/destination NNA addresses.

We have created a flexible addressing design for XIA that introduces a new XID type called a 4ID (named for IPv4), which exploits XID-based fallback within XIA addresses. Consider Figure 7, depicting tunnelling between two XIA-enabled networks, A and B. Both networks have at at least one node that operates as a dual-stack (XIA and IPv4) router. In order for a host in A to transmit to a destination in B (AD_S), the destination DAG address has a fallback hop that contains a 4ID encoding ($4ID_S$) of the IPv4 address of the dual-stack router in B. The egress router in network A will forward using the fallback address, since it does not have a forwarding entry for the primary intent, NID_S. This router then encapsulates the XIA packet into an IPv4 packet using the IPv4 address encoded in $4ID_S$, and then transmits it. On arrival at $4ID_S$, the packet is decapsulated and forwarded to the destination using XIP. Once the IPv4 network is upgraded to XIA, the same DAG can still be used to reach the destination without fallback.

We have also created a new design that we called "Smart 4ID" as an example of the smart control plane class of solutions. The Smart 4ID mechanism extends the 4ID design by adopting an SDN-style control plane to intelligently pick ingress/egress pairs based on a wider view of the local network. Our results show that the new Smart 4ID-based approach outperforms previous approaches while simultaneously providing better failure semantics [22].

6. DIVERSITY IN CONGESTION CONTROL

XIA achieves flexibility and evolvability by offering an interface (DAGs and diverse XID types) that gives both endpoints and the network control over how communication operations are performed. We decided to explore whether similar flexibility could be extended to congestion control, one of the most challenging functions implemented in transport protocols today. Congestion control is hard because it entails coordination among all participants (network and endpoints) on resource management. The key requirements for congestion control are 1) the flexibility to handle new requirements, such as new applications, and 2) ensuring coexistence between different behaviors without negatively impacting efficiency and fairness. For example, two different styles of congestion control often cannot easily coexist, as they interfere with each other's resource allocation.

TCP-style congestion control can support a wide range of congestion control techniques to meet different application requirements. Two key aspects of TCP's congestion control enable diversity: 1) its purely endpoint-based nature enables each end-point the *flexibility* to employ different algorithms, and 2) the notion of TCP-friendliness [11] provides a mechanism for *coexistence* between different algorithms and behaviors. On the other hand, router-assisted explicit congestion control, such as RCP and XCP, can be far more efficient [19, 10]. However, router-assisted congestion control is less flexible because network feedback strictly defines end-point behaviors.

We have designed FCP (Flexible Control Protocol) [14], a novel congestion control framework that provides the best of both worlds: it is as efficient as explicit congestion control, but retains (or even expands) the flexibility of end-point based congestion control. FCP incorporates two key ideas. First, to maximize end-point flexibility without sacrificing efficiency, FCP effectively combines fine-grained end-point control and explicit router feedback. To our knowledge, FCP is the first congestion control algorithm that incorporates the two. Second, to ensure safe coexistence of different behaviors within the same network, FCP introduces a simple invariant for fairness.

In FCP, each domain can allocate resources (budget) to a host, and explicitly signal a congestion price to the host, similar to classical ideas on economics-based congestion control [20]. Such *aggregation* balances endpoint and network control over resource use, since endpoints can freely assign resources to their flows within their budget while giving the network flexibilty to assign different pricing schemes to different classes of flows. To ensure safe coexistence of different endpoint resource allocation strategies, the system maintains a key invariant that the amount of traffic a sender can generate is limited by its budget, which is the maximum amount it can spend per unit time.

To ensure robust operation in practice, FCP accommodates high variability and rapid shifts in workload using *preloading*, a feature that allows senders to announce the amount of resources they want to spend ahead of time. FCP also provides a practical mechanism for cost-fairness [6] that is decoupled from actual billing, while enforcing that each sender operates within its own budget. Our evaluation shows that FCP is locally stable, achieves faster and graceful convergence, and outperforms existing explicit congestion control algorithms in many cases, without imposing unreasonable overhead. FCP's endpoint flexibility and coexistence mechanisms allow endpoints to easily implement diverse strategies, benefiting many types of applications.

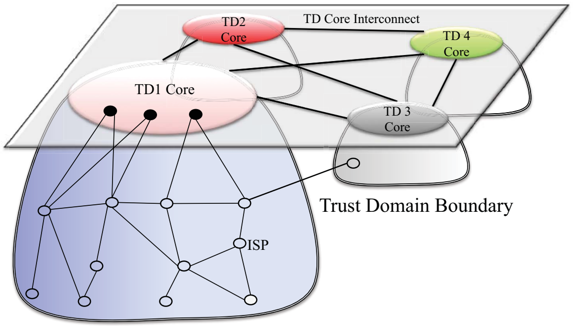

Figure 8: The SCION architecture

7. SCION

XIA provides intrinsically secure identifiers that ensure the validity of various principal types, such as improving the trustworthiness of end-to-end communication, service access, and content retrieval, as well as enabling inter-domain routing that is resistant against route hijacking attacks. Forwarding for various XID types can be based on existing routing algorithms, but it is interesting to explore routing techniques that provide even stronger security guarantees as follows: (1) Guaranteeing high availability under numerous network attacks for interdomain settings involving malicious domains, compromised routers, and large botnets; (2) Providing topological anonymity and location privacy without perceivable performance reduction.

The SCION architecture, illustrated in Figure 8, features scalability, control, and isolation for secure and highly available host-to-host communication [30]. The essential concept in SCION is network isolation that divides Autonomous Systems (ASes) into Trust Domains (TDs) that separate mutually distrustful entities, reducing the overhead of security mechanisms. Each TD selects a TD Core, a set of designated ASes that interface with other TDs. SCION offers balanced control over the forwarding paths: the TD Core initiates path construction; ISPs disseminate policy-compliant paths; destinations announce their preferred paths; and sources splice one of their paths with a destination path to compose an end-to-end path, with cryptographic methods used to ensure the authenticity of end-to-end paths.

To further secure SCION under DDoS attacks, we propose a mechanism to establish paths with guaranteed bandwidth [16]. Specifically, a TD Core initiates the tree-based bandwidth allocation process along with path construction, such that available bandwidth is allocated as paths are disseminated from the TD Core to its domains. Sources can request a bandwidth-guaranteed half path and combine it with the destination's half path to get an end-to-end path with certain properties. SCION's packet-carried forwarding state in turn provides a natural mechanism to encode network capabilities and the bandwidth allocations.

To provide topological anonymity and location privacy for SCION hosts, we propose that each ISP encrypt the packet-carried forwarding state [15]. This approach prevents the adversary from retrieving the sender's (or the receiver's) origin address from the packet. On-path routers decrypt forwarding state on the fly, which for symmetric-key encryption can be performed within nanoseconds without memory lookups. Given the isolation architecture and deterministic paths of SCION, a source node can determine the level of privacy achievable in this framework.

Both our security analysis and evaluation results show that SCION naturally prevents numerous attacks based on SCION's intrinsic security properties. Based on a formal analysis and large-scale simulation results, we confirm that SCION's DDoS defense mechanism mitigates emerging DDoS threats such as Denial-of-Capability (DoC) [5] and Coremelt [26] attacks that none of the existing DDoS defense mechanisms can achieve. Our implementation results show that SCION's topological anonymity mechanism improves anonymity with a negligible overhead and is more efficient than Tor [8].

We are integrating SCION with the XIA dataplane to do path selection for host-to-host communication. SCION uses a new principal type, named ScID, so SCION can coexist with other routing and forwarding alternatives. A host or gateway router of the source network constructs an end-to-end SCION path by composing two half-paths, as described above, and embeds it in an ScID that is included in the destination DAG. Routers have a ScID-specific forwarding logic (Figure 3) that uses the SCION path to forward the packet after successful verification of the forwarding information. Our proof of concept integration was demonstrated in Spring 2013 and is now being integrated into a more comprehensive routing framework for XIA.

8. ECONOMIC INCENTIVES FOR CACHING

Similar to other architecture proposals, XIA supports a content retrieval mechanisms, based on CIDs, that allows content to be delivered efficiently from caches distributed throughout the network. While this is attractive from a technical perspective, it does raise the question of appropriate incentives for network operators to deploy caches, and how widespread caching will impact Content Delivery Networks (CDNs), which today deliver a lot of content from locations close to the requestor.

We have developed a model that quantifies the costs and benefits for operators or CDNs to deploy caches [2]. Economic modeling and welfare analysis shows that the bandwidth cost savings to ISPs are insufficient to justify the level of caching that optimizes economic welfare. To achieve this optimum, there must be payment flows between publishers and caching providers. More generally, caching can be provided by transit networks, edge ("eyeball") networks, or CDNs. We find that edge networks, because of their terminating access monopoly, are in a position to dominate other providers of caching services in the competition for publisher payments, either by providing these services themselves or extracting any rents through inter-carrier payments.

Figure 9 shows how the producer surplus from the provision of caching can be allocated between networks, CDNs and publishers. For example, when an eyeball network performs commercial caching (i.e., gets payments from publisher), then the surplus in all four regions is realized between the eyeball network and publishers. However, when a CDN does commercial caching, then only the values in regions I and II are realized. When an eyeball network performs only transparent caching (i.e., opportunistically caches content that it forwards), then only files with popularity greater than ϕ are cached with the surplus in regions II and IV; in this case the gain to the eyeball network is limited to a reduction in bandwidth to its provider [2, 1].

Today's CDNs provide not only storage services, but a brokerage function minimizing transaction costs between the many publishers and eyeball networks. Future CCNs will

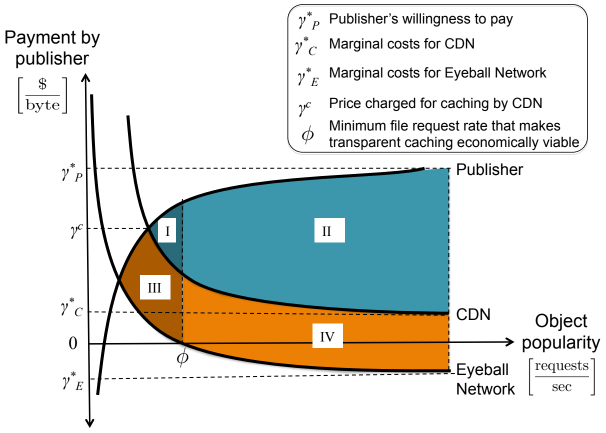

Figure 9: Viability of caching by CDNs and eyeball networks

similarly need a brokerage function to minimize costs when charging publishers for network-provided caching.

Implications for Chunk Design - We have also analyzed the functionalities provided by URLs, `http`, and TCP/IP in today's information retrieval architectures, and have shown how to provide equivalent functionality in XIA through the construction of cacheable information chunks with appropriate headers, and a new design for URIs with CID references. Based on our economic analysis, we are led to include support for billing information, e.g. the name of a publisher's content broker, in chunk headers as an essential feature to realize optimal cache deployments.

9. PRIVACY

Privacy is a major concern for Internet users. The architecture of the network has a direct impact on user privacy, since it defines the content of the network header, which, since it used to forward the packet, is visible to observers in the network. XIA research considers the issue of privacy both from a network and user perspective.

9.1 Privacy in XIA Addresses and Identifiers

One aspect of our research agenda is to explore how XIA features (Figure 1) may impact user privacy. XIA supports multiple types of identifiers, allowing users and applications to signal their intent to the network. While this has advantages, as discussed in Section 2, it may also have drawbacks. For example, the use of a CID or SID, as is common in content and service centric architectures, reveals more information than the HID-style addresses used in the current Internet. However, the flexibility of XIA allows end users to use an HID as the destination, thereby only revealing who the destination is, but allowing the actual intent of the packet to be concealed through encryption.

XIA identifiers (XIDs) of hosts and services are *self-certifying* in that every host or service is associated with a public-private key pair, and the XID is the hash of the public key. This provides a level of accountability (as in Section 2), but giving the end device this much control over its IDs also has privacy advantages. Devices may use one XID for email, and another for web searches, thereby making it difficult for an application service provider or transit network provider to determine that the same user is engaged in both activities, a practice that greatly undermines user privacy. This is not possible in a typical IP-based system. Conversely, the end device may choose to keep the same identifier even as it moves from one ISP to another. This increases the risk of tracking, but it allows the end device to maintain trust relationships with application service providers as it moves.

9.2 Public Perception of Privacy

In order to be able to help users address privacy concerns in XIA, or any network architecture, we have to gain a better understanding of public concerns about privacy [18, 25]. We collaborated with the Pew Internet Center to conduct a national representative sample of Internet users (775 persons). We found that 86% of adult Internet users have taken steps to avoid surveillance by other people or organizations when they were using the internet. Despite their precautions, 21% of online adults have had an email or social media account hijacked and 11% have had vital information like Social Security numbers, bank account data, or credit cards stolen - and growing numbers worry about the amount of personal information about them that is available online.

In further analyses, we explored which types of people are more or less likely to hide their information or identity in different circumstances and how various groups use different strategies and tactics to manage their digital footprints. We found for example that age is the only demographic factor that strongly predicts anonymity-seeking. Users also saw a significant difference between hiding one's content and hiding one's identity: people hide their content to protect and manage relationships, but hide their identity to handle security risks and threats. Finally, negative Internet experiences and social media use increase people's awareness of risk and their overall attempts to cover their digital footprints.

These results clearly indicate that we need to develop tools and interfaces that will allow users with privacy concerns to enable the solutions that address those concerns. For a typical user in today's Internet, this is a near impossible task, and it becomes even harder in networks that offer richer functionality and interfaces. Our ongoing research is on the development of such tools, specifically targeting users with no networking background. The results from the user studies help in identifying the most critical user concerns and provide guidance in the design of the user interface.

10. CONCLUSION

This paper motivates and describes three ideas that can complement IP's narrow waist design: multiple principal types, intrinsic security, and flexible addressing. We also presented results that highlight the practicality of the architecture, the benefits of the design, and extensions of the architecture's support of flexibility and evolution.

While these results highlight the potential of XIA, many challenges remain. First, we plan to look at the question of how to architect a control plane that can evolve and support incremental deployment of new protocols. Building on our experience with SCION, we will consider protocols for routing different XID types, monitoring, and diagnostics. Second, we will develop strategies for evaluating network architectures, including XIA and others. The idea is to combine more formal approaches with concrete use cases of the architecture. Finally, we will continue to refine the XIA architecture as we gain experience in deploying and using XIA. More information on ongoing research can be found on the project web site at `www.cs.cmu.edu/~xia`.

Acknowledgments

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

11. ADDITIONAL AUTHORS

Stephanie Brown, Cody Doucette (Boston University), Hsu-Chun Hsiao, Dongsu Han, Tiffany Hyun-Jin Kim, Hyeontaek Lim, Carol Ovon, Dong Zhou, Soo Bum Lee, Yue-Hsun Lin, Colleen Stuart, Dan Barrett, Aditya Akella (University of Wisconsin), David Andersen, John Byers (Boston University), Laura Dabbish, Michael Kaminsky (Intel), Sara Kiesler, Jon Peha, Adrian Perrig, Srini Seshan, Marvin Sirbu, Peter Steenkiste. Authors are from Carnegie Mellon University, unless otherwise noted.

12. REFERENCES

[1] P. Agyapong. *Economic Incentives in Content-Centric Networking: Implications for Protocol Design and Public Policy*. PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 2013.

[2] P. Agyapong and M. Sirbu. Economic Incentives in Content-Centric Networking: Implications for Protocol Design and Public Policy. *IEEE Communications Magazine, Special Issue on CCNs*, December 2012.

[3] A. Anand, F. Dogar, D. Han, B. Li, H. Lim, M. Machado, W. Wu, A. Akella, D. Andersen, J. Byers, S. Seshan, and P. Steenkiste. XIA: An Architecture for an Evolvable and Trustworthy Internet. In *The Tenth ACM Workshop on Hot Topics in Networks (HotNets-X)*, Cambridge, MA, November 2011. ACM.

[4] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker. Accountable Internet Protocol (AIP). In *Proc. ACM SIGCOMM*, Seattle, WA, Aug. 2008.

[5] K. Argyraki and D. R. Cheriton. Network capabilities: The good, the bad and the ugly. In *Proceedings of ACM HotNets*, 2005.

[6] B. Briscoe. Flow rate fairness: dismantling a religion. *ACM SIGCOMM CCR*, 37(2), March 2007.

[7] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden. Tussle in cyberspace: defining tomorrow's internet. In *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '02, pages 347–356, New York, NY, USA, 2002. ACM.

[8] R. Dingledine, N. Mathewson, and P. Syverson. Tor: the second-generation onion router. In *Proceedings of conference on USENIX Security Symposium*, 2004.

[9] F. Dogar and P. Steenkiste. Architecting for Edge Diversity: Supporting Rich Services over an Unbundled Transport. In *8th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT 2012)*. ACM, December 2012.

[10] N. Dukkipati. Rate Control Protocol (RCP): Congestion control to make flows complete quickly. In *Ph.D. Thesis, Department of Electrical Engineering, Stanford University*, October 2007.

[11] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *SIGCOMM 2000*, August 2000.

[12] A. Gladisch, R. Daher, and D. Tavangarian. Survey on Mobility and Multihoming in the Future Internet. *Wireless Personal Communication*, October 2012.

[13] 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 *Proceedings of the 9th Usenix Symposium on Networked Systems Design and Implementation*, NSDI 2012, 2012.

[14] D. Han, R. Grandl, A. Akella, and S. Seshan. FCP: A Flexible Transport Framework for Accommodating Diversity. In *Proceedings of Sigcomm 2013*, New York, NY, USA, 2013. ACM.

[15] H.-C. Hsiao, T. H.-J. Kim, A. Perrig, A. Yamada, S. C. Nelson, M. Gruteser, and W. Meng. LAP: Lightweight anonymity and privacy. In *Proceedings of IEEE Symposium on Security and Privacy*, May 2012.

[16] H.-C. Hsiao, T. H.-J. Kim, S. Yoo, X. Zhang, S. B. Lee, V. Gligor, and A. Perrig. STRIDE : Sanctuary Trail – Refuge from Internet DDoS Entrapment. In *Proceedings of ACM ASIACCS*, 2013.

[17] C. Huitema. Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs). RFC 4380 (Proposed Standard), February 2006. Updated by RFCs 5991, 6081.

[18] R. Kang, S. Brown, and S. Kiesler. Users' Perception about Internet Anonymity. In *ACM Conference on Human Factors in Computing Systems (CHI13)*, April 2013.

[19] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *SIGCOMM '02*, August 2002.

[20] F. Kelly and G. Raina. Explicit congestion control: charging, fairness and admission management. In B. Ramamurthy, G. N. Rouskas, and K. M. Sivalingam, editors, *Next-Generation Internet Architectures and Protocols*, pages 257–274. Cambridge University Press, 2010.

[21] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.

[22] M. Mukerjee, D. Han, S. Seshan, and P. Steenkiste. Understanding tradeoffs in incremental deployment of new network architectures. In *Proceedings of the 9th international conference on Emerging networking experiments and technologies*, CoNEXT '13, New York, NY, USA, December 2013. ACM.

[23] E. Nordmark and R. Gilligan. Basic Transition Mechanisms for IPv6 Hosts and Routers. RFC 4213 (Proposed Standard), October 2005.

[24] E. Nordstrom, D. Shue, P. Gopalan, R. Kiefer, M. Arye, S. Ko, J. Rexford, and M. J. Freedman. Serval: An end-host stack for service-centric networking. In *Proc. 9th Symposium on Networked Systems Design and Implementation*, NSDI '12, 2012.

[25] L. Rainie, S. Kiesler, R. Kang, and M. Madden. Anonymity, Privacy, and Security Online, September 2013. Pew Research Internet Project, `http://pewinternet.org/Reports/2013/Anonymity-online.aspx`.

[26] A. Studer and A. Perrig. The Coremelt Attack. In *Proceedings of ESORICS*, 2009.

[27] D. Trossen, M. Sarela, and K. Sollins. Arguments for an information-centric internetworking architecture. *ACM Computer Communication Review*, 40:26–33, April 2010.

[28] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking Named Content. In *Proc. CoNEXT*, 2009.

[29] Z. Zhu and R. Wakikawa and L. Zhang. A Survey of Mobility Support in the Internet, July 2011. IETF RFC 6301.

[30] X. Zhang, H.-C. Hsiao, G. Hasker, H. Chan, A. Perrig, and D. G. Andersen. Scion: Scalability, control, and isolation on next-generation networks. In *IEEE Symposium on Security and Privacy (Oakland)*, Oakland, CA, May 2011.

[31] D. Zhou, B. Fan, H. Lim, M. Kaminsky, and D. G. Andersen. Scalable, High Performance Ethernet Forwarding with CuckooSwitch. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 97–108. ACM, 2013.