Internet Architecture Evolution: Found in Translation

Anonymous Author(s)

ABSTRACT

The success of the Internet is undeniable, but so are its limitations. Over the past two decades, the research community has responded with clean-slate redesigns, proposing innovative architectures focused on issues like security and information dissemination, among others. Unfortunately, these efforts have had limited impact on the commercial Internet, if any. The reason is that the Internet architecture is deeply entrenched, making a complete replacement elusive.

In this paper, we argue that a successful approach to evolving the Internet requires three key ingredients. It should (1) be backwards compatible with the current Internet, (2) evolve from the existing architecture, and (3) allow new architectures to reach their full potential. Recently, the community introduced an overlay-based approach for an Extensible Internet. We believe this is a clear step in the right direction: it is backwards-compatible, does not require replacing the current Internet infrastructure, and is deployable today. However, we contend that this approach is not fully adequate as it lacks the third requirement, which we deem crucial for new architectures to gain a foothold and grow. As an alternative, we advocate for a translation-based approach and present our rationale on how it may enable effective Internet evolution by meeting the three requirements above.

1 INTRODUCTION

The remarkable growth of the Internet over the past five decades and its establishment as the dominant global communications infrastructure demonstrate the wisdom of its original design principles [14]. However, in recent decades, some fundamental assumptions underlying these principles have changed, and it is widely acknowledged that the Internet architecture is lacking along several dimensions [21].

The most acknowledged architectural flaw is the lack of security in the original design, leaving the Internet vulnerable to various attacks, including DDoS [5] and route hijacks [19, 31]. Others [24, 49] have questioned whether point-to-point packet delivery is still the appropriate service model in a content-oriented world. The research community responded to this problem with clean slate redesigns of the Internet, including security approaches [50], information-centric architectures [24, 49], mobility-oriented solutions [46], architectures centred around evolvability [35], and entirely new conceptions of the Internet [43].

These clean-slate designs have had little to no impact on the commercial Internet. There are several reasons for this apparent lack of success. First, clean-slate architectures require a massive overhaul of the Internet infrastructure or its entire replacement. However, the Internet architecture is deeply embedded in its elements (routers and end-hosts), which means a total replacement or a significant overhaul remains elusive. Indeed, *any architecture should be backwards-compatible with the existing Internet*. The second problem is that each clean-slate solution addresses a specific issue with the current architecture. By elevating one problem above the others, other issues remain. Unfortunately, there is no one-size-fits-all architecture, and the future is hard to predict.

Over the past decades, we have also learned that different types of network changes can lead designers to question their initial assumptions. For instance, rapid changes in user behaviour, new infrastructure or operational methods, or even political and economic changes often unveil hidden architectural limitations. This assumption mismatch leads to a second requirement: *the architecture should support network evolution*. Clean-slate designs that consider this capability a first-class citizen [35, 43] suffer from a deployability problem by either requiring the replacement of the current Internet [43] or centring its evolution mechanism in the new architecture [35]. In other words, they *do not evolve from the existing architecture*, an aspect that is subtly — yet crucially — different than guaranteeing backwards compatibility (Section 2.2).

Recently, McCauley et al. proposed Trotsky [33], an architectural framework that provides a backwards-compatible path to an Extensible Internet[1]. Trotsky's key idea is to introduce a new layer (L3.5), which is an intrinsic overlay on L3. In addition, it decouples the tasks of interconnecting networks within a domain (left to L3) and interconnecting different domains with the new L3.5. Trotsky is a simple and elegant solution that not only eases the deployment of radical new architectures but also ensures compatibility with the legacy Internet, providing a promising path forward. The fact that it remains IP-centric (as we expect IP to be the *de facto* L3 underlay) is an essential advantage for deployment.

The IP-centrality is, however, both a blessing and a curse. On the one hand, IP is deployed everywhere, thus enabling the fast deployment of Trotsky and a myriad of multiple architectures on top, as L3.5 overlays. On the other hand—and, we argue, more fundamentally—the centrality of IP may stifle the potential for new architectures to blossom, for two fundamental reasons. First, as the new architectures run as a L3.5 layer on top of this L3 underlay, they inherit the intrinsic limitations of the L3 that is used as a "logical pipe". As a result,

[1] The EI design is further detailed in [8].

they hamper some of the essential services offered by the L3.5 architecture—often, the precise services that motivated it in the first place (Section 2.3)!

Second, an overlay-based approach offers misguided incentives. As in many cases the underlay precludes offering the full-service set of the L3.5 architecture, it limits the incentives for its deployment and use. In addition, it does not incentivize the replacement of the good old Internet architecture (with all its recognized limitations) with better alternatives. On the contrary, we believe it propels the current Internet to become even more entrenched!

Faced with this conundrum, in this paper we revive a decades-old solution [15, 44] and argue for ***translation*** as the approach to enable a multi-architecture Internet that is backwards-compatible, enables evolvability, and avoids the limitations of overlaying. Specifically, we propose *direct* translation between L3 architectures (Section 3) to enable the inter-operation of architectures, both present and future.

In contrast to overlay approaches [33], a framework for Internet evolution based on translation enables a new architecture domain to offer all its services, retaining all the benefits that motivated its design, *as it is not dependent on the intrinsics of an L3 underlay.* The challenge becomes the development of *effective translation mechanisms*, a new avenue for research. We believe that the time is ripe for the networking community to embrace this approach. Advances in fast programmable networking hardware (programmable ASICs [23], SmartNICs [30]) and host stacks [9, 47]) can enable architecture translators at production-grade performance and scale. Indeed, their packet processing capabilities have recently enabled the development of routing node prototypes for new network architectures, including for NDN [40] and SCION [16], which is indicative of the plausibility to construct effective and fast Internet translators.

After detailing the limitations of existing work (Section 2) and arguing for translation (Section 3), in this paper we also present our initial exploration of this approach. We present the design of two translators—IP to SCION and IP to NDN—to shed light on the practicality of their development (Section 4), as well as a discussion on open challenges (Section 5).

2 BACKGROUND AND MOTIVATION

We review problems with the Internet architecture (Section 2.1) that motivated the development of clean-slate designs over the past 15 years (Section 2.2). We then discuss Trotsky, a promising path for Internet evolution, and its intrinsic limitations that motivate our paper (Section 2.3).

2.1 Limitations of the Internet Architecture

The outstanding success of the Internet as a global communication infrastructure is a testament to the quality of its architectural design. However, the success that spurred its growth also revealed important flaws.

Lack of security. Security was not among the primary goals of the Internet's original design. As a result, the Internet infrastructure is prone to multiple attacks, including DDoS and route hijacks [19, 31]. Although modern cryptography has provided solutions to enable confidentiality and integrity in end-to-end communications, availability remains an issue, along with the lack of path control provided to end-hosts (e.g., to avoid compromised domains).

Host-centricity. The Internet was originally designed as a host-to-host communication network. Today, however, its use is primarily dominated by the consumption of multimedia content and other forms of information dissemination. CDNs have emerged to address this mismatch, enabling efficient content distribution on a global scale, overcoming the absence of content-oriented primitives in the Internet's design. However, their use leads to complex agreements with ISPs and other stakeholders, incurring significant operating, capital, and efficiency costs. More worryingly, as only a few large commercial players can afford to deploy and operate a CDN, this trend is leading to Internet flattening [7].

Fixed end-hosts. The original architecture provided unicast point-to-point communication between fixed end-hosts. That model starkly contrasts the massive presence of mobile devices, whose location is constantly changing. The mainstream communication abstractions for mobility rely on an indirection layer [37] that decouples sending and receiving hosts through application-specific and network-level solutions that led to security [36] and performance [22] issues.

Difficult to evolve. The Internet design did not consider the possibility of evolving its network layer. While the elegant minimality of the Internet's architectural waist has allowed for much innovation at layers above and below it, the current design lacks the abstractions to allow for incremental architectural improvements. The lack of principles of abstraction and modularity [28] for the evolution of the architecture has led to the "ossification" of the Internet [45].

2.2 Clean-slate architectures

Unlike the incremental patchwork of evolutionary approaches [38], clean-slate research gives the opportunity to rethink the Internet without being constrained by the actual realization [17]. Over more than a decade, clean-slate research spurred regular specific funding programs, e.g., NSF's FIND/FIA and FP7-ICT's FIRE. Several architectures emerged in this context, each typically focusing on a specific architectural issue.

Embracing security. SCION [50] proposes a path-aware internetworking approach to building a communication infrastructure that provides security and high availability by design, including preventing route hijacks and several forms of DDoS attacks. A path-aware network architecture provides information about paths to endpoints, which is useful for enforceable path control. The added transparency and control are fundamental to improving security. For example, disjoint network paths can be used to mitigate network failures, and the exclusion of specific routes can be used to resist surveillance or bypass a network under attack.

Embracing content. Information-centric networking [4] architectures treat content as a first-class citizen and implement communication models that decouple content consumption and production. NDN [49], arguably its most successful instance, builds the narrow waist around named content. To support named-based content retrieval, NDN implements network mechanisms such as name-based routing and forwarding, data-centric security, and in-network caching.

Embracing mobility. MobilityFirst [46] proposes mobility as the dominant communication pattern. Its key idea is the separation of names or identifiers from network addresses or locators, relying instead on a scalable, distributed, global name service to dynamically bind identifiers to network addresses. In addition, its design includes security, context awareness, and content retrieval aspects.

Embracing evolvability. The RINA Architecture [43] is based on the principle that networking is inter-process communication (IPC). It utilizes a recursive set of layers, with each layer performing the same functions but at different scopes and granularities. RINA supports gradual upgrades, promoting the evolution of the infrastructure. XIA [35] enables end hosts to express a range of delivery mechanisms and services through network packets carrying multiple forms of addresses simultaneously. To handle partial rollout and backward compatibility, XIA encodes a directed acyclic graph in the packet header. This allows the packet to fall back on alternative services that, when combined, provide the intended service.

Limitations. The first set of clean-slate designs [46, 49, 50] are effective in addressing a specific limitation of the Internet architecture. However, each, individually does not address *all* known Internet problems. And, paraphrasing [12], *an Internet architecture is a different effort than the simple union of these sub-architectures.*

By contrast, RINA and XIA target evolvability, but they have their own issues. RINA, on one hand, would require a significant infrastructure overhaul—a replacement of the existing IP-based infrastructure. XIA, on the other hand, enables partial deployment via translations between architectures. However, it assumes that the core translation mechanism is built around XIA. In other words, XIA is not designed to operate as an extension of IP but as a replacement. We argue that any deployable solution should consider a different starting assumption: that the IP Internet is widely deployed. This is one of the insights of Trotsky, described next.

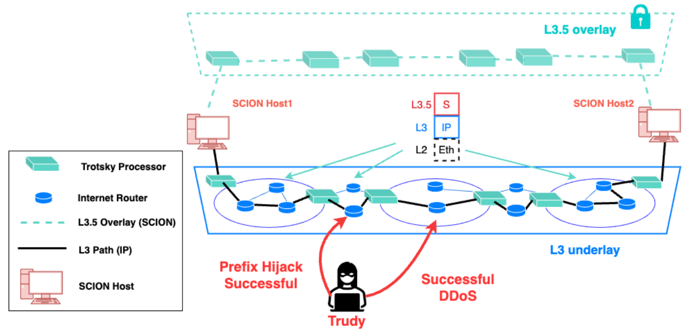

Figure 1: A SCION L3.5 overlay on top of an IP L3 loses its security benefits.

2.3 Evolution via overlaying

Trotsky [33] is an overlay-based framework for evolving the Internet architecture. It forms the basis of the proposal by the same authors for an Extensible Internet [8]. Trotsky departs from the assumption that IP is so deeply entrenched in existing networking hardware and software applications that moving away from it will hardly happen. *We agree with this assumption and believe it should be a foundational design principle.* The key idea is introducing a new layer, L3.5, on top of the traditional L3. New architectures are deployed on this L3.5 layer, which runs on top of "logical pipes", facilitating the seamless integration of new network architectures without necessitating a complete overhaul of the existing infrastructure. Trotsky has the potential to be deployed on a global scale, leveraging the reuse of IP for the L3 pipes, a key advantage. Trotsky's deployment strategy involves decoupling inter-domain (left to L3.5) and intra-domain networking (left to L3) and implementing Trotsky processors at domain edges, where L3.5 services would be supported (as well as in hosts).

Trotsky is a simple, backwards-compatible approach to Internet evolution, as it reuses existing tunnelling and forwarding mechanisms, treating inter-domain routing as an L3.5 overlay. One of Trotsky's features is its ability to enable the incremental deployment of new architectures, ensuring a smooth transition without disrupting the existing Internet.

While an elegant, practical solution to the evolution of the Internet architecture, we argue that an overlay-based approach such as Trotsky's has limitations that are detrimental to its effectiveness. We support our claim with two main arguments, in addition to a few secondary ones.

Overlaying limits the benefits of new architectures. The key problem is that deploying a new architecture as an overlay may preclude some of its advantages. Indeed, the underlay could potentially nullify the very reason for the architecture, as the new L3.5 will inevitably inherit limitations of the L3 underlay. We illustrate this problem with the example of running SCION as an L3.5 architecture on

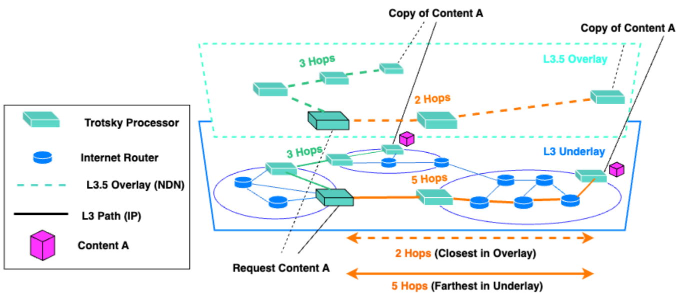

Figure 2: A NDN L3.5 overlay on top of an IP L3 sees its caching benefits reduced.

top of the current L3 IP-based infrastructure (Figure 1). As explained in Section 2, the current Internet is vulnerable to DDoS attacks and route hijacks. SCION, on the other hand, has intrinsic mechanisms to prevent both. For prefix and route hijacking, SCION includes cryptographic techniques and a secure path construction process that ensures each path segment is verified [29]. The cryptographic path protection enables path hiding even if an attacker knows the network topology, making the path impossible to DDoS [10]. As the L3 pipe used in Trotsky is vulnerable to these attacks, a SCION L3.5 running as an overlay would inherit this problem. Despite the robust mechanisms SCION incorporates to prevent the attacks, it would still be susceptible due to the underlay it uses as a communication pipe. *We argue this would fundamentally undermine the rationale for deploying this architecture as an overlay.*

Another example is an information-oriented architecture that includes in-network caching (e.g., [49]). Running NDN as an L3.5 layer using Trotsky would not guarantee the locality of content to an L3.5 interest request, as a neighbour in the overlay can be many hops away in the underlay (e.g., in Figure 2, the closest NDN node is two L3.5 hops away, while the nearest L3 node in the underlay is only three hops away). The caching benefits from the overlay can, therefore, be lost. While specific cross-layer mechanisms can tackle this problem, it is important to be mindful that layering violations often come with significant cost, increased complexity, scalability issues and security concerns.

Misguided incentives. While a new architecture running as an overlay can offer new functionalities, it may not address fundamental limitations in the underlying architecture, as we have shown above. This limitation may discourage investment and development in deploying and utilizing the envisioned L3.5 architecture, because the underlying L3 cannot fully support its potential. Instead of promoting the adoption of fundamentally better alternatives, overlay-based approaches might reinforce the dominance of the existing Internet, *entrenching the current architecture further.*

Other concerns. An overlay solution introduces inefficiencies due to the overhead imposed by the new layer it adds. This overhead can constrain both performance and scalability. In addition, while an end-to-end approach is valued for its simplicity and robustness (as noted in [39]), end-hosts must operate under the same architecture. While acknowledging the manifold benefits of an end-to-end approach, we recognize its limitations in accommodating emerging use cases with varied host architectures.

3 EVOLUTION VIA TRANSLATION

The approach we argue for in this paper is to have the network explicitly translate between architectures. Unlike an overlay, the idea is to deploy the new architectures "in series". We should start by noting that we are not the first to propose translation to address limitations of the Internet architecture or as a way to extend it. In 1993, Paul F. Tsuchiya and Tony Eng proposed the Network Address Translator (NAT) [44] to address the problems of IP address depletion and scaling in routing—a translator widely used today. One decade later, Jon Crowcroft et al. proposed Plutarch [15], an inter-networking solution that stitches together architectural contexts—sets of network elements that share the same architecture in terms of naming, addressing, packet formats and transport protocols. These contexts communicate through interstitial functions that translate different architectures.

In this paper, we revisit Plutarch, armed with the knowledge of two decades of attempts to evolve the Internet. Our goal is to articulate the arguments of why we think translation is the most effective approach to achieve a multi-architecture Internet that (1) enables architecture evolution, (2) is backwards-compatible, (3) offers the full benefit of the new architectures, (4) incentivizes their deployment, (5) enables end-hosts from different architectures to communicate end-to-end, and (6) guarantees the level of performance of today's Internet. In the following, we justify how direct translation may fulfil all these requirements.

The approach. We envision *direct translation* between different architectures. This approach refers to the process of converting the protocols or architectural principles from one Internet architecture to another, directly at L3, while preserving their essential functionalities and characteristics. Translation enables interoperability and seamless communication between architectures without the need for an additional L3.5 layer. The successful translator ensures that data packets, communication protocols, and network services retain their intended meaning and functionality across the transition between architectures. Like Trotsky, we make the pragmatic choice to use Autonomous Systems (or domains) as a starting point, and propose introducing translators between architecture domains.

We note that direct translation is similar in spirit to Plutarch but differs from XIA [35]. The latter requires deploying that

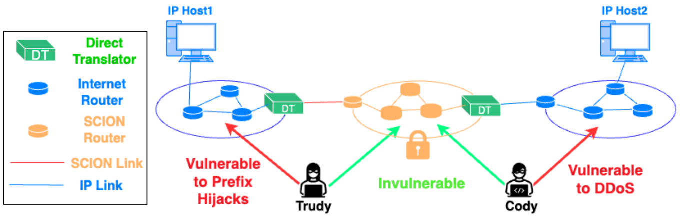

Figure 3: Translation allows the SCION domain to retain all its security properties.

new architecture and using their built-in translation mechanisms. Direct translation, much like the Trotsky overlay approach, is immediately deployable due to its reliance on the existing Internet as a starting point.

The advantage. By stitching together different architectures in "series", translation enables *(1) architecture evolution* and is *(2) backwards-compatible*. New architectures are added to the Internet by integrating translators between the new and some existing architecture, with the current Internet as a natural first target. Crucially, a new architecture domain is deployed at L3, so it does not inherit the limitations of an underlay connectivity layer. An effective translator will therefore *(3) enable retaining the full benefits of the architecture in that domain*. Figure 3 provides a visual representation of the integration of a SCION domain into the Internet architecture. In this example, two IP hosts communicate via a SCION network. The translator at the edge of the IP domain translates the IP packets sent by the host into SCION packets, which are then forwarded towards the destination[2]. Contrary to the packets traversing the IP domains, the packets in the SCION network are protected against several network attacks, including route hijacks and (several forms of) DDoS.

This improvement over overlay-based approaches *(4) incentivizes* the investment in and deployment of new architectures that add value (e.g., the security benefits of SCION), as in these new domains, users retain the full benefits of the architecture. As a result, we expect this approach to allow new network architectures to blossom while interacting seamlessly with the current infrastructure. The possibility to *(5) enable hosts from diverse architectures to communicate seamlessly* is also an added value of the approach.

We hold two reasons for trusting the feasibility of this approach concerning performance. First, translation avoids overlay overheads. However, it is crucial that the translation mechanism itself does not impact performance. Fortunately, modern hardware (programmable switches, SmartNICs/DPUs) and fast network stacks (DPDK, XDP) can assist in achieving this objective. For instance, programmable chips capable of Terabit speeds, such as the Intel Tofino [23], enable the implementation of fully customized packet processing logic directly within the switch ASIC using high-level languages like P4 [11]. Translation involves manipulating protocol data units (e.g., rewriting or repurposing packet fields) and bridging disparate network semantics (e.g., converting one address type to another), tasks ideally suited for high-speed packet processors. Seminal studies [18, 20, 34, 40, 41] also demonstrate the viability of unconventional forwarding mechanisms in high-speed programmable hardware, including the development of routing nodes for various clean-slate architectures [49, 50], offering further evidence of *(6) the feasibility of high-speed Internet translators*.

4 INITIAL EXPLORATION

We now address the question: is it possible to develop effective and high-performance translators? We present our initial exploration for two clean-slate architectures.

IP-SCION translation. We have built a prototype of an IP to SCION translator (the Direct Translator in Figure 3). Due to page restrictions, we focus on the most challenging aspect of its design: conversion from the IP destination address to its counterparts, the SCION address and Forwarding Path fields. The first thing to consider is that the control plane of the translator acts like a regular SCION *host*, as it is the ingress to the SCION network. As with any host, the translator control plane must contact the Address Resolution service (to obtain the SCION address) and the Path Servers (to get the Forwarding Path). The latter involves several steps: path lookup, path verification, and path combination. To obtain these fields, our translator uses the SCION-IP Gateway (SIG), as a proxy. The SIG service allows legacy IP hosts to communicate via the SCION network[3]. The SIG services return the SCION address and forwarding path, and our control plane installs the required translation rules in the switch tables. Note that the first packet of a flow triggers this process, but the subsequent packets remain entirely in the data plane.

We also developed a P4 program for the data plane to translate from IP to SCION for the Tofino 2 Native Architecture. Our program compiles in the Intel Tofino SDE, which guarantees it achieves 10+ Tbps throughput when running in our hardware switch. Our solution enables a maximum Forwarding Path of 20 hops (we highlight this aspect as particularly challenging). For context, the average AS Path Length on the Internet is currently around 4, and 16 is considered an extreme case by the SCION authors [13]. We also evaluated the translation functionality by testing our solution using the SCIONLab network [3], with a similar setup to that of Figure 3. We achieved successful communication (using ping and iperf) between two IP hosts located in different domains,

[2]Note that there are no tunnels involved; the packets are translated directly between architectures.

[3]Note that the SIG encapsulates IP packets into SCION packets (i.e., following an overlay approach).

with packets traversing several countries across the SCION domain.

IP-NDN translation. There are two main challenges in building IP-NDN translators: (i) NDN does not reflect the traditional network stack [1], and (ii) NDN uses different packet types for requests and responses, namely Interest and Data packets, whose structure is highly variable [2].

Addressing the first challenge requires spanning the different layers of the two network stacks. To translate network-level names from NDN to IP, we need to convert the transport and application-level semantics of NDN names for the TCP/IP counterparts. Conversely, translation from IP to NDN may require inspection of application-level data in the IP payload, which is difficult in networking hardware.

For the second challenge, NDN packets contain network-level names (similar to URLs) required for name-based forwarding. Parsing names with highly variable structures within the constraints of high-speed networking is also a challenging task [6, 25, 26, 32, 48]. We are investigating efficient data plane designs for these translation challenges.

5 DISCUSSION AND OPEN CHALLENGES

Our initial exploration offers confidence in the feasibility of building architecture translators in high-speed networking hardware. In this section, we discuss other challenges.

The right incentives. We argued that one advantage of translation over overlaying is a better alignment regarding incentives. Anyway, it is essential to ensure that partial deployments at one domain have built-in incentives to increase in size and entice other domains to follow suit. Partial or disconnected (island-like) deployments should offer some partial utility with the promise that extending and joining adjacent deployments can offer even more utility. It is also important to avoid situations where new deployments can be perceived as unduly burdening or operating unfairly towards existing infrastructure, or situations where partial deployments are inhibited by existing infrastructure such that they are only useful when deployed at scale. The first parties to benefit from the new architecture should have the power to deploy it on their own at least initially–even if only partially–and to then incentivize their suppliers and peers to follow suit. Finally, avoiding scenarios where the first movers do not immediately benefit is fundamental. If ISPs need to build or replace infrastructure in the hope that customers will want it, they put themselves at a competitive disadvantage to competitors who do not make that move.

Building efficient translators. Developing efficient translation operations involves the codification of generic and architecture-specific translation paths, which may span from simple modifications and re-purposing of packet fields to complex match-action computations. A related challenge is state management. Can the required state be stored in the switch hardware, or do we need to explore other mechanisms (e.g., RDMA-based)? What if it is impossible to maintain the required state or part of the transition computations in a single platform (e.g., in a network switch)? It may be necessary to investigate multi-platform approaches (involving a slow and a fast path, for instance) and develop mechanisms to divert traffic to the right platform. Another important aspect is related to configuration. Defining the runtime mechanisms for reconfiguration will be particularly challenging.

Do we need global translation services? This concern arises from the need for global resolution entities in several clean-slate architecture proposals, namely in ICN (e.g., DONA [27], PSIRP [42], NDN [49]). We conjecture that some global (DNS-like) mechanisms may be needed, and it seems to be an exciting avenue to explore. For instance, exploring source-routing mechanisms based on (scalable) gossip mechanisms (as [15] suggests) may be an option to consider.

How many translators do we need? In theory, to support n different network architecture we would need n^2 translators [33]. This may not be a serious concern, for two reasons. First, we expect the number n to be small [15], as evidenced by the number of clean-slate architectures developed so far. To be fair, we expect that an effective multi-architecture approach, namely the one we favour here, should incentivize the emergence of architectures. Anyway, the initial translators should be from the new architecture to IP (NEW <-> OLD), leading to a linear growth with n in the beginning. In case there is demand for translators between new architectures (say, NEW_A <-> NEW_B), we can also interconnect them by stitching together two (NEW <-> OLD) translators, as in (NEW_A <-> OLD <-> NEW_B).

6 CONCLUSION

We conclude with a biological analogy to clarify our main point. An overlay-based approach is similar to an *epiphyte*, a plant that grows on the surface of another plant. They differ from parasites in that they grow on other plants for physical support and do not negatively affect the host; quite the contrary. This is similar to an overlay-based approach to Internet evolution, enclosing many positives. *Still, the host plant can limit epiphyte diversity and abundance.*

Our alternative is similar to *mutualistic symbiosis*, a relationship where two species interact closely, benefiting each other and evolving together. In a translation-based Internet evolution, new architectures are introduced and interwoven with the existing system through translation. The old and new systems coexist, with each providing mutual benefits: the new architectures gain a foothold and can grow, while the old system continues to operate and support the transition. *Notably, mutualistic symbiosis can drive evolutionary changes, promoting diversification and the emergence of new species.*

REFERENCES

[1] [n. d.]. Named Data Networking: Architecture Overview. https://named-data.net/project/archoverview/ Retrieved 2024-06-28.

[2] [n. d.]. NDN Packet Format Specification v0.3. https://docs.named-data.net/NDN-packet-spec/current/ Retrieved 2024-06-28.

[3] [n. d.]. The SCIONLab global research network. https://www.scionlab.org/ Retrieved 2024-06-28.

[4] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Borje Ohlman. 2012. A survey of information-centric networking. *IEEE Communications Magazine* 50, 7 (2012), 26–36.

[5] Albert Gran Alcoz, Martin Strohmeier, Vincent Lenders, and Laurent Vanbever. 2022. Aggregate-based congestion control for pulse-wave DDoS defense. In *Proceedings of the ACM SIGCOMM 2022 Conference*. 693–706.

[6] Ali AlSabeh, Elie Kfoury, Jorge Crichigno, and Elias Bou-Harb. 2022. P4ddpi: Securing p4-programmable data plane networks via dns deep packet inspection. In *NDSS Symposium 2022*.

[7] Todd Arnold, Jia He, Weifan Jiang, Matt Calder, Italo Cunha, Vasileios Giotsas, and Ethan Katz-Bassett. 2020. Cloud provider connectivity in the flat internet. In *Proceedings of the ACM Internet Measurement Conference*. 230–246.

[8] Hari Balakrishnan, Sujata Banerjee, Israel Cidon, David Culler, Deborah Estrin, Ethan Katz-Bassett, Arvind Krishnamurthy, Murphy McCauley, Nick McKeown, Aurojit Panda, et al. 2021. Revitalizing the public internet by making it extensible. , 18–24 pages.

[9] Tom Barbette, Cyril Soldani, and Laurent Mathy. 2015. Fast userspace packet processing. In *2015 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*. IEEE, 5–16.

[10] Laure Blanchez. [n. d.]. SCION eliminates DDoS like a boss - This is How. https://www.anapaya.net/blog/scion-eliminates-ddos-like-a-boss Retrieved 2024-06-28.

[11] Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, et al. 2014. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 87–95.

[12] Robert Braden, David Clark, Scott Shenker, and John Wroclawski. 2000. Developing a next-generation Internet architecture. *White paper, DARPA* (2000).

[13] Laurent Chuat, Markus Legner, David Basin, David Hausheer, Samuel Hitz, Peter Müller, and Adrian Perrig. 2022. The Complete Guide to SCION. *Information Security and Cryptography* (2022).

[14] David Clark. 1988. The design philosophy of the DARPA Internet protocols. In *Symposium proceedings on Communications architectures and protocols*. 106–114.

[15] Jon Crowcroft, Steven Hand, Richard Mortier, Timothy Roscoe, and Andrew Warfield. 2003. Plutarch: an argument for network pluralism. *ACM SIGCOMM Computer Communication Review* 33, 4 (2003), 258–266.

[16] Joeri de Ruiter and Caspar Schutijser. 2021. Next-generation internet at terabit speed: SCION in P4. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. 119–125.

[17] Anja Feldmann. 2007. Internet clean-slate design: what and why? *ACM SIGCOMM Computer Communication Review* 37, 3 (2007), 59–64.

[18] Sergio Gimenez, Eduard Grasa, and Steve Bunch. 2020. A Proof of Concept implementation of a RINA interior router using P4-enabled software targets. In *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. IEEE, 57–62.

[19] Sharon Goldberg, Michael Schapira, Peter Hummon, and Jennifer Rexford. 2010. How secure are secure interdomain routing protocols. In *Proceedings of the ACM SIGCOMM 2010 Conference (SIGCOMM '10)*.

[20] Diogo Gonçalves, Salvatore Signorello, Fernando MV Ramos, and Muriel Médard. 2019. Random linear network coding on programmable switches. In *2019 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*. IEEE, 1–6.

[21] Mark Handley. 2006. Why the Internet only just works. *BT Technology Journal* 24, 3 (2006), 119–129.

[22] Robert Hsieh and Aruna Seneviratne. 2003. A comparison of mechanisms for improving mobile IP handoff latency for end-to-end TCP. In *Proceedings of the 9th annual international conference on Mobile computing and networking*. 29–41.

[23] Intel. [n. d.]. The Intel® Tofino™ series of P4-programmable Ethernet switch ASICs. https://www.intel.com/content/www/us/en/products/details/network-io/programmable-ethernet-switch/tofino-series.html Retrieved 2024-06-28.

[24] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. 2009. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. 1–12.

[25] Alexander Kaplan and Shir Landau Feibish. 2022. Practical handling of DNS in the data plane. In *Proceedings of the Symposium on SDN Research*. 59–66.

[26] Jason Kim, Hyojoon Kim, and Jennifer Rexford. 2021. Analyzing traffic by domain name in the data plane. In *Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR)*. 1–12.

[27] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. 2007. A data-oriented (and beyond) network architecture. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*. 181–192.

[28] Teemu Koponen, Scott Shenker, Hari Balakrishnan, Nick Feamster, Igor Ganichev, Ali Ghodsi, P Brighten Godfrey, Nick McKeown, Guru Parulkar, Barath Raghavan, et al. 2011. Architecting for innovation. *ACM SIGCOMM Computer Communication Review* 41, 3 (2011), 24–36.

[29] Markus Legner, Tobias Klenze, Marc Wyss, Christoph Sprenger, and Adrian Perrig. 2020. {EPIC}: every packet is checked in the data plane of a {Path-Aware} Internet. In *29th USENIX Security Symposium (USENIX Security 20)*. 541–558.

[30] Jianshen Liu, Carlos Maltzahn, Craig Ulmer, and Matthew Leon Curry. 2021. Performance characteristics of the bluefield-2 smartnic. *arXiv preprint arXiv:2105.06619* (2021).

[31] Robert Lychev, Sharon Goldberg, and Michael Schapira. 2013. BGP security in partial deployment: is the juice worth the squeeze?. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM (SIGCOMM '13)*.

[32] Aniss Maghsoudlou, Oliver Gasser, Ingmar Poese, and Anja Feldmann. 2022. FlowDNS: correlating Netflow and DNS streams at scale. In *Proceedings of the 18th International Conference on Emerging Networking EXperiments and Technologies*. 187–195.

[33] James McCauley, Yotam Harchol, Aurojit Panda, Barath Raghavan, and Scott Shenker. 2019. Enabling a permanent revolution in internet architecture. In *Proceedings of the ACM Special Interest Group on Data Communication*. 1–14.

[34] Rui Miguel, Salvatore Signorello, and Fernando MV Ramos. 2018. Named data networking with programmable switches. In *2018 IEEE 26th International Conference on Network Protocols (ICNP)*. IEEE, 400–405.

[35] David Naylor, Matthew K Mukerjee, Patrick Agyapong, Robert Grandl, Ruogu Kang, Michel Machado, Stephanie Brown, Cody Doucette, Hsu-Chun Hsiao, Dongsu Han, et al. 2014. XIA: architecting a more trustworthy and evolvable internet. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 50–57.

[36] Charlie Perkins. 1999. Mobile IP and security issue: an overview. In *First IEEE/POPOV Workshop on Internet Technologies and Services. Proceedings (Cat. No. 99EX391)*. IEEE, 131–148.

[37] Charles E Perkins. 1997. Mobile ip. *IEEE communications Magazine* 35, 5 (1997), 84–99.

[38] Jennifer Rexford and Constantine Dovrolis. 2010. Future Internet architecture: clean-slate versus evolutionary research. *Commun. ACM* 53, 9 (2010), 36–40.

[39] Jerome H Saltzer, David P Reed, and David D Clark. 1984. End-to-end arguments in system design. *ACM Transactions on Computer Systems (TOCS)* 2, 4 (1984), 277–288.

[40] Salvatore Signorello, Radu State, Jérôme François, and Olivier Festor. 2016. Ndn. p4: Programming information-centric data-planes. In *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. IEEE, 384–389.

[41] Kamila Soucková. 2019. *FPGA-based line-rate packet forwarding for the SCION future Internet architecture*. Master's thesis. ETH Zurich.

[42] Sasu Tarkoma, Mark Ain, and Kari Visala. 2009. The publish/subscribe internet routing paradigm (psirp): Designing the future internet architecture. In *Towards the Future Internet*. IOS press, 102–111.

[43] Joe Touch, Yu-Shun Wang, and Venkata Pingali. 2006. A recursive network architecture. *ISI, Tech. Rep* 626 (2006).

[44] Paul F Tsuchiya and Tony Eng. 1993. Extending the IP Internet through address reuse. *ACM SIGCOMM Computer Communication Review* 23, 1 (1993), 16–33.

[45] Jonathan S Turner and David E Taylor. 2005. Diversifying the internet. In *GLOBECOM'05. IEEE Global Telecommunications Conference, 2005.*, Vol. 2. IEEE, 6–pp.

[46] Arun Venkataramani, James F Kurose, Dipankar Raychaudhuri, Kiran Nagaraja, Morley Mao, and Suman Banerjee. 2014. Mobilityfirst: A mobility-centric and trustworthy internet architecture. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 74–80.

[47] Marcos AM Vieira, Matheus S Castanho, Racyus DG Pacífico, Elerson RS Santos, Eduardo PM Câmara Júnior, and Luiz FM Vieira. 2020. Fast packet processing with ebpf and xdp: Concepts, code, challenges, and applications. *ACM Computing Surveys (CSUR)* 53, 1 (2020), 1–36.

[48] Jackson Woodruff, Murali Ramanujam, and Noa Zilberman. 2019. P4dns: In-network dns. In *2019 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*. IEEE, 1–6.

[49] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, KC Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. 2014. Named data networking. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 66–73.

[50] Xin Zhang, Hsu-Chun Hsiao, Geoffrey Hasker, Haowen Chan, Adrian Perrig, and David G Andersen. 2011. SCION: Scalability, control, and isolation on next-generation networks. In *2011 IEEE Symposium on Security and Privacy*. IEEE, 212–227.