# SCIONLAB: A Next-Generation Internet Testbed



**Author(s):**
Kwon, Jonghoon; García-Pardo, Juan A.; Legner, Markus; Wirz, François; Frei, Matthias; Hausheer, David; Perrig, Adrian

**Publication date:**
2020

**Permanent link:**
https://doi.org/10.3929/ethz-b-000453445



**Originally published in:**
https://doi.org/10.1109/ICNP49622.2020.9259355

# SCIONLAB: A Next-Generation Internet Testbed

Jonghoon Kwon*, Juan A. García-Pardo*, Markus Legner*, François Wirz*,
Matthias Frei*, David Hausheer†, Adrian Perrig*
*Department of Computer Science, ETH Zürich, Zürich, Switzerland
†Department of Computer Science, Universität Magdeburg, Magdeburg, Germany

***Abstract*—Network testbeds have empowered networking research and facilitated scientific progress. However, current testbeds focus mainly on experiments involving the current Internet. In this paper, we propose SCIONLAB, a novel global network testbed that enables exciting research opportunities and experimentation with the SCION next-generation Internet architecture. New users can join SCIONLAB as a full-fledged autonomous system with minimal effort and administrative overhead, and directly gain unfettered access to its inter-domain routing system. Based on a well-connected network topology consisting of globally distributed nodes, SCIONLAB enables new experiments, such as inter-domain multipath communication, path-aware networking, exploration of novel routing policies, and new approaches for DDoS defense. SCIONLAB has been operational since 2016 and has supported diverse research projects. We describe the design and implementation of SCIONLAB, and present use cases that illustrate exciting research opportunities.**

***Index Terms*—Global Network Testbed, Future Internet, Multipath Communication, Path-aware Networking, Secure Routing**

## I. INTRODUCTION

The rapid growth of the Internet is driving the adoption of various network services at a global scale, including content delivery networks (CDNs) [47], cloud storage systems [32], video conferencing [45], and software-defined networking in wide area networks [39]. All these services demand highly reliable, performant, and secure communication with flexible routing strategies. Network testbeds have facilitated experimentation and have thus contributed to the emergence of new network services. In particular, Emulab and PlanetLab [15], [44], [48], [57] were instrumental in supporting researchers with readily accessible testbeds.

Over the past two decades the majority of networking research was focused on intra-domain or data-center networking instead of inter-domain networking [18], [30], [61]. While the diameter of the Internet is indeed shrinking, inter-domain communication continues to be an important aspect of communication, as applications continue to exist that do not directly communicate with cloud, CDN, or hyperscaler data centers. As a result of the relative lack of progress in inter-domain research and innovation, problems continue to affect a diversity of applications. Fortunately, next-generation Internet architectures supporting new networking paradigms such as path-aware networking, multipath communication, and novel security approaches are promising to address these challenges and have the potential to drive the next generation of applications [10], [11].

Path-aware networking enables end-hosts to obtain information about network paths leading to the destination, and to select the path among a set of paths offered by the network [20], [52]. Path awareness offers exciting properties to applications, such as path transparency, fine-grained path control, fast failover or route optimization to alternate paths, or geofencing [50], [51], [63]. These properties enable the creation of new transport protocols and advanced application features. Path-aware networking also enables multipath communication: in case the end-host obtains multiple paths from the network, it can select the path on a per-packet basis. Multipath communication can provide increased bandwidth, improved reliability, and more efficient link utilization [46], [59]. Ongoing efforts at the IETF, e.g., TSVWG [12], MPTCP [17], and QUIC [21], [34], [54], have shown its feasibility and practicality.

Advanced security features built into many path-aware network architectures enable the creation of new applications and services, where the network provides built-in trust-establishment and key-distribution mechanisms, defenses against distributed denial-of-service (DDoS) attacks, and privacy-enhancing techniques. However, to tap the full potential of all these opportunities, further research is required and open questions need to be answered, such as the following: Which paths and which additional information should be disseminated to end-hosts? What is the API between network, transport, and application layer? How do the different layers work together to select the best paths with limited overhead? What congestion-control algorithms are suitable when end-hosts can switch paths or use multiple paths at the same time?

To enable researchers to explore path-aware networking architectures and support research trying to answer these questions, we propose SCIONLAB—a novel design for a flexible, scalable, and expandable global network testbed that is easy to use. SCIONLAB is based on the SCION Internet architecture, and thus inherits scalability, security, and efficiency properties. SCION improves security on various levels, e.g., by providing protection against malicious autonomous systems (ASes) and offering transparency and control over forwarding paths and trust roots. At the same time, the SCION approach ensures scalability and efficiency by placing forwarding information into packet headers to eliminate packet state in routers (see §II for an overview of SCION). The SCIONLAB network infrastructure is based on 35 ASes widely distributed across the world, and connects over 600 user ASes running on heterogeneous systems. The central coordination service, the SCIONLAB *Coordinator*, orchestrates the infrastructure and the user ASes to support seamless networking with a wide range of network topologies and user environments.

The core contribution of this paper is the design, implementation, and operation of SCIONLAB, a global next-generation network testbed, to enable research in path-aware networking, multipath communication, and security. We anticipate that these powerful concepts will fuel the next wave of innovation, efficiency, and security in inter-domain networking.

## II. BACKGROUND ON SCION

SCION is a clean-slate secure Internet architecture designed to provide high availability in the presence of adversaries, trust and path transparency, and inter-domain multipath routing [43], [62]. Although other promising next-generation Internet architectures have been proposed in the past, including NIRA [60], RINA [56], and Pathlets [27], SCION is a unique architecture that provides security, path-aware networking, multipath communication with the maturity of the system—as of today SCION went through 5 generations of code and an estimated 150 person-years of effort.

SCION organizes existing ASes into groups of independent routing planes, called *isolation domains* (*ISDs*), which interconnect to provide global connectivity. An ISD is administered by a set of ASes called the ISD core who define the ISD's trust roots and issue certificates for ASes in the ISD, and provide inter-ISD connectivity.

Besides high security, SCION also provides a scalable routing infrastructure and efficient packet forwarding. As a path-based architecture, SCION end-hosts obtain inter-domain network path segments, and combine them into end-to-end inter-domain paths that are carried in packet headers. Thanks to embedded cryptographic mechanisms, path construction is constrained to the route policies of Internet service providers (ISPs) and receivers, offering path choice to all parties: senders, receivers, and ISPs. This approach enables path-aware communication, a promising approach to networking [52]. Thanks to a multitude of path segments that can be combined in different ways, the end-hosts can typically select among over a dozen end-to-end paths. In concert, these features enable features such as rapid failover in case of network failures, dynamic traffic optimization, and facilitate DDoS defenses.

**Control Plane.** The main objectives of the SCION control plane are to explore the network topology (path exploration), choose paths according to each AS's routing policy (path registration), and provide end-hosts inter-domain network path segments that are cryptographically protected (path resolution).

• *Path Exploration.* To construct a path segment in SCION, the AS beacon service propagates a *path-segment construction beacon* (*PCB*) to neighboring ASes, which add AS information to the path segment and forward them according to their routing policy. This *beaconing* process results in a policy-constrained flood of a PCB, starting at a core AS and following AS relationships (provider–customer relationships within an ISD and peering relationships across ISDs). SCION path segments also encode identifiers of ingress/egress interfaces of two neighboring ASes, which enables fine-grained selection of the specific link, which is an important property to achieve path diversity even across a sequence of identical ASes.

• *Path Registration.* The beaconing process discovers different paths over time. ASes select a subset of the paths and register them with path servers in the ISD core and the local path servers. Each path server processes the path registration requests and stores the paths in the path database.

• *Path Resolution.* Upon request from an end-host, the local path service returns a number of path segments, which the end-host composes providing several possible end-to-end paths. All control-plane information is digitally signed and can be verified using certificates issued through the SCION control-plane public-key infrastructure that is operated by the certificate service in each AS.

**Data Plane.** SCION border routers perform inter-AS packet forwarding. Data packets contain the forwarding information in the packet header, encoded as a sequence of per-AS information consisting of an expiration time, ingress and egress link identifiers, and a message authentication code that enables the forwarding AS to cryptographically validate its own information. Upon arrival of a data packet, the border router verifies the correctness of the path information in the header and forwards the packet to the next border router. For forwarding the packet from the SCION ingress to the SCION egress router of an AS, any intra-domain communication technology can be used, such as MPLS, IP switching, etc., and an appropriate header is added by the ingress border router and removed by the egress border router. SCION border routers thus do not keep any inter-domain routing tables.

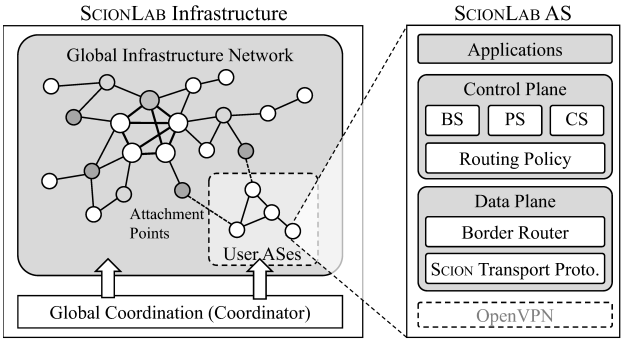



Figure 1: An overview of the SCIONLAB architecture. Experimenters obtain ASes and connect to the infrastructure network orchestrated by a global coordination service.

## III. SCIONLAB ARCHITECTURE

In this section, we first present a high-level overview of the SCIONLAB architecture, along with the core design principles, and then provide further details on the individual components.

### A. Architecture Overview

The SCIONLAB infrastructure provides a globally distributed SCION network infrastructure, to which user ASes connect for running experiments. A global coordination service, the *Coordinator*, provides seamless cooperation among all the entities. Figure 1 illustrates how the main components

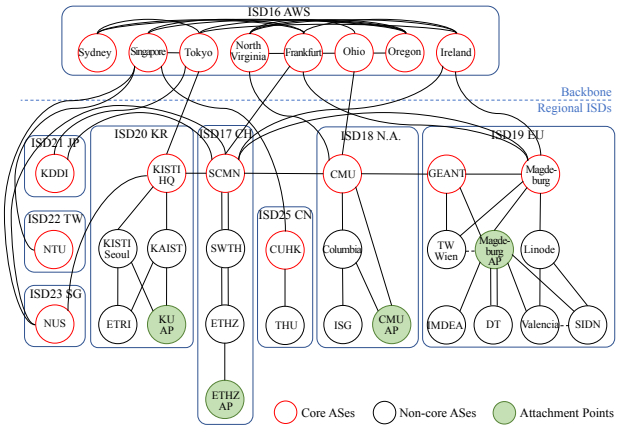



Figure 2: The SCIONLAB infrastructure topology. The wide distribution of infrastructure nodes provides researchers global connectivity and path diversity.

are organized. The SCIONLAB administrators operate the network infrastructure and the Coordinator. The global topology is created to support a variety of paths between any two ASes, which empowers multipath operation. The infrastructure offers several *attachment points*, to which user ASes connect to participate in the SCIONLAB network.

Depending on who owns them, two AS types exist: infrastructure AS (run by SCIONLAB administrators) and user AS (run by users). Both AS types are regular SCION ASes, with a beacon service (BS), certificate service (CS), and path service (PS), and one or multiple border routers. Some infrastructure ASes additionally feature an attachment point that provides connectivity to user ASes.

SCIONLAB operates with full cryptographic support. Each ISD defines its own roots of trust within a *trust root configuration*, where the core ASes control the root keys for the control plane. Each AS has a globally unique AS number (ASN) and a public/private key pair, which is certified through a certificate signed by a core AS. The certification infrastructure and trust validation are provided by the SCION architecture [43].

The life cycle of a user AS starts with the user opening up an account on the Coordinator hosted at https://www.scionlab.org. The user can select an attachment point to connect to, and the Coordinator provides the AS configuration which includes the cryptographic certificates required for operation. The user AS then starts receiving beacons from the AS of the attachment point, registers its down-paths in the core path service, and is then reachable from other SCIONLAB ASes. Initially, each user AS receives the same bandwidth allocation towards an attachment point. Later on, users may increase their allocation by operating an attachment point and thus providing connectivity to other user ASes.

### B. Infrastructure

**Global Topology.** The core principles behind constructing the SCIONLAB topology are i) to create a reliable global network infrastructure, and ii) to provide numerous diverse routing paths. Figure 2 depicts the global SCIONLAB AS topology, where each node represents an AS and each edge represents a connecting network link.

For a reliable infrastructure, it is essential to ensure that the infrastructure ASes and inter-AS links are stable. We achieve this through monitoring and close interaction with the local AS operators. Over time, the infrastructure topology continued to expand—as of early 2020 SCIONLAB consists of a backbone ISD and 8 regional ISDs built on over 35 infrastructure ASes, connecting a total of 622 user ASes; 271 ASes at ETHZ AP (Switzerland), 252 ASes at Magdeburg AP (EU), 54 ASes at KU AP (South Korea), and 45 ASes at CMU AP (North America). We note that 549 out of the 622 ASes are still active.

The global topology is structured to achieve high network capacity with high path diversity for multipath communication. When we add an infrastructure AS to SCIONLAB, we first attempt to provide a direct SCION connection, which is achieved through a direct layer-2 connection to a neighboring AS's SCION border router (through a connection that does not rely on BGP). In case the AS is not a direct neighbor of an existing SCIONLAB AS, we set up a connection via an IP overlay. We select the end points of the overlay connection based on latency, bandwidth, and number of physical hops. Finally, we initiate the new infrastructure AS with `Ansible` [1], installing SCIONLAB services, deploying network configuration, and updating the network topology of neighbor ASes.

In SCIONLAB, all inter-AS SCION traffic is encapsulated in UDP over IP packets by design, regardless of the physical link type used to connect those two ASes. The UDP encapsulation allows us to inexpensively deploy SCION border routers almost anywhere, even being co-located with normal IP routers or located behind firewalls. Other mechanisms that allow communication between two endpoints via IP—e.g., OpenVPN or GRE—can also be used for the IP overlay.

To achieve additional path diversity, we built a global backbone ISD (see §IV-B) using geographically distributed Amazon EC2 instances. As Figure 2 shows, the border routers running in the various EC2 data centers connect to nearby ASes. The low-latency interconnection across EC2 data centers provides attractive low-latency alternative paths compared to the overlay connections.

**Attachment Point.** We realize the topology reconfigurability with the notion of *attachment points*. SCIONLAB provides basic infrastructure ASes to which user ASes can attach, extending the network topology with the experimenters' own resources. They can grow these resources in their preferred way, while the SCIONLAB infrastructure provides global connectivity. If desired, a user AS can also be an attachment point, enabling a scalable extensibility of the topology.

When a new user AS selects an attachment point to connect to, a connectivity negotiation event called *JoinRequest* arises at the Coordinator, which includes information such as link type, target bandwidth and border router details. Then, a new link is added to the border router responsible for the inter-AS routing. Once the SCIONLAB control-plane packets, i.e., PCBs, are successfully forwarded through the new border router, the new AS becomes a part of the network.

**Selective Path Properties.** It is important for network experimenters to demonstrate their ideas in various network conditions. SCIONLAB infrastructure provides a link with three different degradation levels for a specific link property. In the provided configuration, the bandwidth is constrained to three different rates, latency is increased and loss is introduced. With the selective path properties, applications can be evaluated under deteriorated but predictable network conditions.

### C. Management

**Coordinator.** The Coordinator system orchestrates the creation and connection of user ASes by providing AS number, cryptographic keys and their associated certificates, and initializing the overlay connections to the attachment points.

SCIONLAB follows SCION AS numbering [2]: each AS number is 64 bits long, where the top 16 bits represent the ISD and the remaining 48 bits indicate the AS. As a central authority, the Coordinator issues a unique ASN to each AS in SCIONLAB, which is a combination of an ISD number of the selected attachment point and sequentially aligned AS number range in `ffaa:1:0/32` which is reserved for private use.

For a new user AS, the Coordinator generates an asymmetric key pair using the `Curve25519` algorithm [16], and issues a X.509-based [19] certificate signed by the appropriate core AS's private key. The cryptographic key information is stored in the internal Coordinator database along with AS details such as the owner of the AS, the organization and current status including network connectivity.

The Coordinator assists with the connectivity negotiation; when a user AS triggers a *JoinRequest* event, the Coordinator notifies the attachment point's operator of the negotiation request via email. When an AS receives the request, its network operator can approve or reject the request at the AS management interface on the Coordinator. Upon approval, the network configuration is updated and facilitated by the Coordinator. Note that for the infrastructure attachment points under our supervision, no email is needed and all *JoinRequests* are automatically approved.

**Enabling Computation Resources Scaling.** For a shared infrastructure open to any researcher, managing shared computation resources is a challenge. To overcome this, we design SCIONLAB with the notion of BYOC (Bring Your Own Computation), enabling researchers to provision their own computational capabilities that best suit their experiments. Since cloud hosting is broadly affordable, this concept enables anyone to participate and scale up their computation resource requirements as needed.

**Fairness on Network Resource Sharing.** To allocate network bandwidth fairly among users, SCIONLAB needs an allocation, monitoring, and enforcement mechanism. The coordinator performs resource allocation, as it has the complete view of the shared network topology. By setting up an attachment point, an AS could obtain additional bandwidth. Monitoring and enforcement are performed based on probabilistic detection of overuse, and bandwidth limitation by border routers. A non-compliant user AS could remove the bandwidth limitation in the border router, but SCIONLAB operators will detect overuse and can take mitigating measures—in the worst case revoke the AS certificate. In the intermediate term, monitoring and enforcement will be accomplished by a QoS system, that is introduced in SCIONLAB later this year. This system will enable fine-grained resource allocation policies for QoS-protected traffic, leaving best effort traffic to rely on regular congestion-control mechanisms for fair sharing.

Table I: Parameters of routing policies that users can specify.

| | | | |
|---|---|---|---|
| **Path Exploration** | | | |
| $TTL$ | Lifetime for PathSegment (in seconds) | | |
| $PT$ | Propagation time period for Beacon (in seconds) | | |
| $RT$ | Registration time period for PathSegment (in seconds) | | |
| **Path Resolution** | | | |
| $BSS$ | # of best paths | $DT$ | Delay time |
| $CSS$ | # of path candidates | $DJ$ | Disjointedness |
| $ABW$ | Available bandwidth | $ET$ | Expiration time |
| $GBW$ | Guaranteed bandwidth | $HL$ | Hop length |
| $TBW$ | Total bandwidth | $PL$ | Peer links |
| $LST_1$ | Last seen time | $UAS$ | Unwanted ASes |
| $LST_2$ | Last sent time | $UDT$ | Update time |

### D. Participant

**AS Initialization.** Researchers can instantiate a user AS through a simple and intuitive web interface provided by the Coordinator, where local information such as the installation type, and border router IP address is entered. The Coordinator then issues a unique AS number, cryptographic keys with public-key certificates, and depending on the installation type, a `Vagrant` file or instructions on how to install the Debian packages are provided. In all cases the AS is automatically instantiated, setting up the SCIONLAB services, corresponding packages and configuration. Although SCIONLAB runs on Linux, execution on other systems is supported through a virtual machine (VM), see §IV-A. The operator for a new user AS joining the network selects an attachment point from the list given by the Coordinator, which triggers a *JoinRequest* event. This mechanism will also be invoked to establish additional peering connections with other user ASes.

To achieve global connectivity for SCIONLAB, most links use an IP overlay to connect the border routers between neighboring ASes. Two issues that arise from this for the connection to the attachment point come from NAT devices and dynamic IP addresses. In case of NAT devices, UDP destination port `50000` needs to be forwarded to the SCION border router to provide connectivity. To handle dynamic IP addresses, the user can establish the link to the attachment point over an OpenVPN connection, which also enables traversal of multiple layers of NAT devices. We will discuss the implications of this decision in §IV and §V.

**Fully Provisioned AS.** Researchers set up a full-fledged AS participating without limitations in the SCIONLAB control plane. The user can configure the routing policy of the AS. Each AS can also set up native SCION end-hosts that can use the local AS infrastructure to communicate with other SCIONLAB hosts. The control plane enables user ASes to

interact with the rest of the network and implement their own fine-grained routing decisions. Through the beaconing to neighbor ASes and registration of paths in the core path servers, user ASes propagate connectivity information and thus announce their presence. Researchers can control their path-exploration policies through parameters shown in Table I—for instance, setting the parameters $PT = 300$ and $TTL = 3600$ implies that announcements are performed every five minutes and considered valid for one hour.

Besides the routing policy that constrains the set of available paths for an AS at the control-plane level, end-hosts can also define path policies defining which paths they want to use. From this, experimenters can lock end-host traffic to particular paths or avoid certain untrusted or inefficient paths. For example, by defining a path policy with $UAS = \{\text{ASes in Backbone ISD}\}$, an AS can avoid all paths that traverse the backbone ISD.

**SCION End-host.** A user AS supports SCION end-hosts running applications. By setting up the SCION network stack on a host, it can be configured to use the path and certificate services, as well as the border router of the user AS to communicate with any other host on SCIONLAB. For users who do not want to run AS services, installing only the SCION end-host is also supported. By configuring a host with the SCION daemon (`sciond`), the end-host can opt in to an existing AS, and communicates with the AS services to get paths and certificate information.

On an end-host, an application can use a UDP or a QUIC socket, and perform path control by selecting among a set of paths offered by the network. Each packet header contains the forwarding path information, indicating the AS-level forwarding path information. The granularity of the forwarding information is at the border router interface level, thereby fully qualifying the forwarding between ASes. SCIONLAB border routers therefore simply forward packets to the next hop without an inter-domain routing-table lookup. Consequently, the SCIONLAB data plane gives experimenters the ability to utilize a given path on a per-packet granularity.

## IV. IMPLEMENTATION AND DEPLOYMENT

SCIONLAB was started as a proof-of-concept testbed in 2016. The number of participants in SCIONLAB continuously increased, reaching 22 SCIONLAB ASes (4 core ASes) within a year, distributed over four continents (Europe, North America, Asia, and Oceania). In this initial stage, utilization of SCIONLAB was possible only for the participating ASes. To open SCIONLAB to the public, the Coordinator web service was developed, allowing new users to join and coordinate their requests. It has since received several updates, such as support for VM-based user ASes and heterogeneous systems, VPN connectivity, automatic SCION code updates, and the creation of several global attachment points. The Coordinator and all associated software are open source [6], and all ASes run the open-source SCION codebase [4].

### A. Implementation Details

**SCIONLAB Coordinator.** The Coordinator consists of a web interface, an API supported backend, and a database. The web interface is the public entry-point to SCIONLAB. A new user registers by simply signing up with a user name, organization and email address. After an email verification process, the user can initiate new user ASes. If a new request to join the network is made, the *Event Handler* populates the database with the user request, invokes the user AS creation procedure, and initiates updates for the infrastructure. The user AS creation is driven by the *User AS Module*. It issues a new ASN, cryptographic material, and services' configuration files. Finally, the configuration files are delivered to the requester in an appropriate form, depending on the installation type.

The *Infrastructure AS Handler* updates the topology and service configuration for the attachment points, and repopulates the database. The updated configuration is not instantly deployed to the infrastructure. Instead, we implement a push-based update procedure, where the Coordinator can trigger the machines of the attachment point to obtain an update. The attachment point then queries the Coordinator via the *API Module* to synchronize with the current topology view, and reconfigure its services. The database stores the state of the SCIONLAB infrastructure. This ensures that the infrastructure is always up-to-date with the simple *sync* API, even after system or network outages. A similar approach is used to update the user ASes, but without the push: the machines of the user ASes request information about their configuration to the Coordinator—e.g., every time there is a package update, or when the user requests it.

**AS Execution on Heterogeneous Systems.** We support various ways to deploy a SCIONLAB AS in an automated manner:

• *Virtual machine.* The easiest way to run a SCIONLAB AS is by downloading a VM configuration from the Coordinator. Based on this configuration, a full-fledged SCIONLAB AS is installed in an Ubuntu-18.04-based VM on any host using VirtualBox and Vagrant in a fully automated manner. This includes the download of an Ubuntu 18.04 base image, configuration of time synchronization, SCIONLAB repository and package installation, and the necessary configuration; in particular, the SCION configuration contains the "gen folder", which connects the VM-based SCION AS to one of the existing SCIONLAB attachment points that the user selected in the Coordinator. The VM configuration also automatically installs a number of SCION applications within the VM and forwards specific ports to the host system to, e.g., allow access to the local SCION visualization website from a web browser on the host or to access the SCIONLAB AS from an end-host outside the VM. Furthermore, the VM-based SCIONLAB AS is automatically updated (if needed) at 7:00 UTC.

• *Packages.* Users may choose to install a SCIONLAB AS on a dedicated host. At this stage, we support Debian-style packages, and provide simple instructions on how to both install the packages, and obtain the configuration, with the execution of a simple command. These packages can be

installed on a wide range of Linux systems, like Debian or Ubuntu, among others.

• *Developer Installation.* For the specific case where the user is building the SCION services from sources, the Coordinator can return a configuration that is developer friendly: this configuration consists of services' configuration and a set of *supervisord* [8] files. This way, developers can start and stop their locally built services much more easily than with *systemd*.

• *Android device.* Finally, it is also possible to run SCION on an Android device. Specifically, the SCION app [3] enables to run an entire SCION AS attached to the SCIONLAB network on an Android smartphone. The app is based on standalone executables compiled from the SCION Go and C sources using the Android NDK. The setup of the AS is done in fully automated manner based on the configuration received from the Coordinator as documented in the SCIONLAB tutorial [5].

### B. Global Deployment

Core ASes play an important role in SCION, and thus the core ASes in SCIONLAB are deployed in highly reliable network infrastructures, such as regional ISPs (e.g., Swisscom and SWITCH) and international research organizations (e.g., GÉANT). Most organizations participate in SCIONLAB as non-core ASes. These ASes also contribute to SCIONLAB as a basic component of the topology by constructing various forms of connectivity between ASes such as peering links, redundant links or multi-homing, providing various networking opportunities to themselves and other participants.

**Large ISPs.** For a large ISP, we place a core server responsible for the SCIONLAB control-plane services (i.e., beacon, path, and certificate services) and multiple software-based border routers that interconnect other ASes. We next describe the representative cases of Swisscom and SWITCH.

• *Swisscom.* For Swisscom, a top-tier ISP in Switzerland, we deployed 6 SCIONLAB border routers in two different points-of-presence (PoPs): datacenters in Zürich (e.g., Equinix ZH) and the CERN Internet eXchange Point (CIXP) in Geneva, Switzerland. We collocate them alongside the legacy IP border routers at the same peering point. For the neighbor ASes located overseas, which are not directly connected, the SCIONLAB routers are interconnected via an IP overlay. In case an AS is a direct neighbor of Swisscom AS, we built a native 10 Gbps layer-2 connectivity.

• *SWITCH.* For SWITCH (the Swiss national research and education network) network, we deployed 4 SCIONLAB border routers—2 for the upstream AS (i.e., Swisscom) and others for the downstream AS (i.e., ETH Zürich). For the upstream links, the interconnection consists of two distinct physical links; one in Equinix Zürich and another in CIXP. The links are built on native layer-2 connectivity, providing dedicated 10 Gbps bandwidth each. Similar to the upstream links, we also built two physical links for downstream connectivity with 10 Gbps and 1 Gbps bandwidth, respectively. Both links are dedicated to the SCION connection.

**Backbone Network.** In SCIONLAB, ISD 16 is the designated *backbone ISD*. It leverages the Amazon EC2 system which provides low-latency BGP-free routes, and contains several ASes running on globally distributed EC2 data centers (Germany, Ireland, North America [North Virginia, Ohio, Oregon State], Japan, Singapore, and Australia) connected through dedicated lines [13]. Thanks to the backbone ISD it is possible to use paths connecting different regional ISDs with lower latency than the current Internet (see §V-C).

**Redundant Links.** Using multipath communication and immediate path revocation in SCION, redundant links between any two ASes can be used with immediate path revocation. As shown in Figure 2, redundant links between SWITCH and Swisscom (SWTH ↔ SCMN), and ETH and SWITCH (ETHZ ↔ SWTH), respectively, were created for this purpose. Users in SCIONLAB can usually make full use of both links, utilizing idle resources on backup links, while simultaneously obtaining additional bandwidth and higher reliability.

## V. EVALUATION

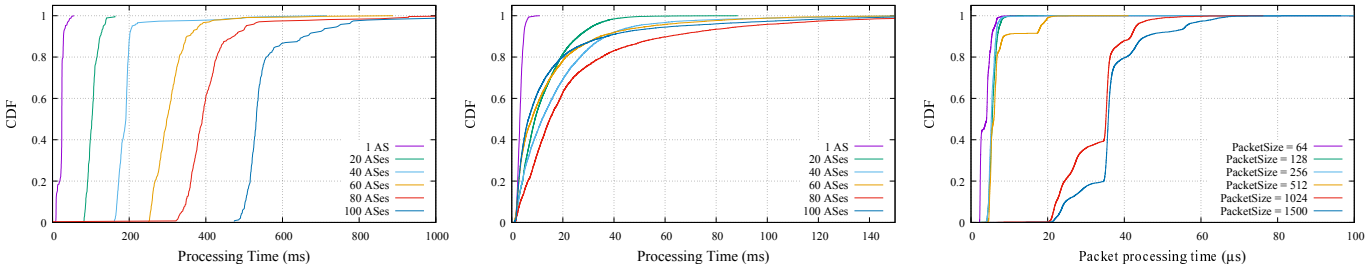
### A. Microbenchmarks

The microbenchmarks are conducted on commodity machines equipped with an Intel i7 2.9 GHz CPU, 16 GB memory, and a 1 GbE NIC. By default, we set $PT = 5$, $RT = 5$, and $BSS = 5$ (see Table I); no other routing-policy constraints are used in any AS.

**Beacon Propagation.** The number of ASes downstream of an attachment point affects the processing time of the path-exploration process; the beacon service updates the PCBs with their ingress and egress interface IDs according to which downstream AS the PCB is forwarded to, and computes a digital signature. We measure the processing time of the PCB propagation at the attachment point, as the other beacon services should not—and do not—exhibit any variation.

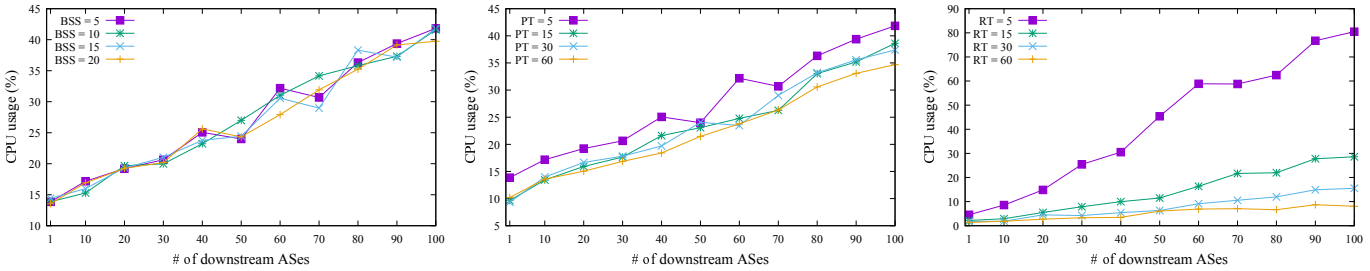
Figure 3a shows different cumulative-distribution-function (CDF) curves of the processing time for a single PCB propagation depending on the number of downstream ASes. With a single downstream AS, the average processing time required is 23 ms. The processing time increases as the number of AS increases and reaches 560 ms with 100 ASes, which is 20 times longer than the first measurement. Nonetheless, considering the fact that the average degree of ASes in the current Internet is approximately 6—the core network have closer to a hundred connections, while the edges only have a few [26], [36], [42]—the result indicates that the overall performance of the current beacon service is sufficient to support the path exploration on an Internet-like topology.

**Path Registration.** We observed a slight increase in the processing time for a path registration, from 3.3 ms to 15.8 ms between 1 and 100 ASes, on average (see Figure 3b). Indeed, the number of downstream ASes does not directly affect the processing time since the path server only checks the authenticity of the registration packet by verifying the signature using the AS certificate. According to our investigation, the additional delay can be attributed to the filesystem due to bursts of new path registration, and can be reduced in the real-world operation with a simple path-registration scheduling.




(a) CDF of beacon propagation time on BS. (b) CDF of path registration time on PS. (c) CDF of packet processing time for BR.

Figure 3: Microbenchmark results for the key operations in SCIONLAB.




(a) Beacon server for different values of $BSS$. (b) Beacon server for different values of $PT$. (c) Path server.

Figure 4: CPU usage in the control-plane services.

**Packet Processing.** To observe how much delay is introduced by our software-based border router, we measured the packet processing time $\Delta t = t_{\text{out}} - t_{\text{in}}$, where $t_{\text{in}}$ denotes the timestamp when the border router reads a packet from the raw input socket buffer and $t_{\text{out}}$ is when the router writes it to the output buffer. $\Delta t$ includes the processing time for packet validation, payload parsing, and routing the packet. We ensure that the border router is the only application running on the test machine, in order to obtain reproducible results.

Figure 3c shows a CDF for various packet sizes processed by the border router. Although the packet-processing times vary considerably depending on the packet size, according to iMIX distribution [40], over 90 % of packets are processed in less than 20 µs (10.17 µs on average). A speedup can be achieved through kernel bypassing, for instance using the DPDK system [22]. Furthermore, SCION end-to-end paths typically traverse fewer than 3 intermediate SCION ASes (besides the source and destination ASes), at most 6 SCION border routers are traversed, which translates into at most $6 \cdot 20\,\mu\text{s} = 120\,\mu\text{s}$. Since propagation latencies are in the order of tens to hundreds of milliseconds (depending on the distance), that additional overhead is negligible in practice.

### B. Control-Plane Scalability

To measure the scalability, we create up to 100 leaf ASes attached to an attachment point and investigate the computational resource consumption of the control-plane services at the aggregation points such as the core AS for the path service and the attachment point for the beacon service. Each service exposes resource metrics through `Prometheus` [24] to report observed resource consumption in real time. The experiment runs 10 times and the results are averaged.

**Beacon-Service Scalability.** According to the evaluation shown in Figure 4a, the CPU consumption increases linearly with the number of downstream ASes. The CPU usage reaches 41 % on average to serve 100 downstream ASes, which is only a three-fold increase compared to the 14 % baseline measured with a single downstream AS. In addition, no difference in CPU consumption for $BSS \in \{5, 10, 15, 20\}$ is observed; this indicates that the computational cost for beacon message construction does not increase much for larger PCB packets.

In contrast, an increase of the frequency of PCB propagation (i.e., a decrease of $PT$) also increases the workload (Figure 4b). While the propagation frequency decrements from twelve times to once per minute, the resource consumption decreases by 20–30 %. Although the results with $PT \in \{15, 30, 60\}$ show a similar CPU usage before they reach 30 downstream ASes, the gaps between them gradually increase afterwards. From this, we can extrapolate that the control plane scales well when reducing the propagation frequency.

Overall usage of memory is also reasonably small—only 50 MB of memory are needed for 100 downstream ASes—and there is no significant increase compared to our baseline with one leaf AS (i.e., 40 MB to 50 MB).

**Path-Service Scalability.** We evaluate the computational scalability on the core path server, an aggregation point of all the path registration, which serves the 100 child ASes. The parameter for the frequency of path registration, $RT$, ranges from 5 to 60 seconds. Figure 4c shows the evolution of CPU usage to support the path registration and resolution for up to 100 child ASes and different values for $RT$. Similar to the resource consumption in beacon servers, the results indicate that there is linear increase of the CPU utilization when scaling up the number of child ASes. Recall that the path registration

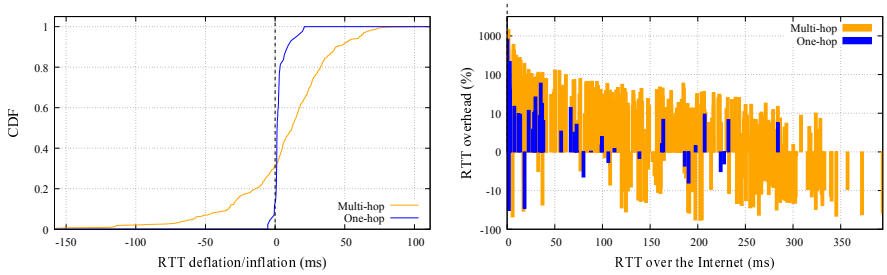



Figure 5: RTT deflation/inflation for all pairs of infrastructure ASes on SCIONLAB ($RTT_{\text{SCMP}} - RTT_{\text{ICMP}}$).

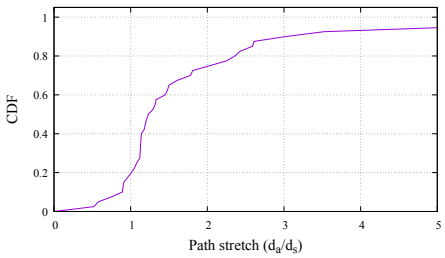



Figure 6: CDF curves of path stretch for all direct links.

frequency of once every 5 seconds is an aggressive approach and a value of $PT$ near 3600 seconds would start being reasonable. We conclude that only a small amount of resources is expended by the core path server in most cases. For example, the CPU consumption is bounded below 29 % of a single core for $RT = 15$. The memory consumption ranges $22 \sim 39$ MB.

Path resolution may cause a serious resource exhaustion on the core path servers, which are responsible for all the path requests within the corresponding ISD. Therefore, caching path segments in local path servers is essential to reduce the heavy load on the core path server, achieving the path-service scalability. Recall that each path server caches requested path segments and reuses them upon receiving the same requests until they expire; only the cache-missed requests are recursively forwarded to the core path server. To evaluate this, we assess the actual path requests collected from the SCIONLAB path servers during a hackathon event on October 12, 2019 [9].

From the results, we make the following observations: 1) The number of path requests observed at the core path server does not reflect the total number of path requests within the ISD. 2) From the local path servers, only a few path requests (27 % of cache-misses) are forwarded to their core. 3) With multiple core path servers, the load of path requests can be distributed. The observations suggest that the load balancing (i.e., the caching and distributed services) is the key to minimizing the overhead of path resolution, achieving path-service scalability.

### C. Data-Plane Performance

We now measure the latency stretch, compared to a direct Internet connection. Interestingly, SCIONLAB forwarding can achieve lower latency than the Internet, thanks to the possibility to optimize the path in a path-aware network architecture.

**Network Latency in SCIONLAB.** We start by measuring the round-trip time (RTT) between all the SCIONLAB node pairs—in total of 496 pairs for 32 ASes under our administration—using the SCION Control Message Protocol (SCMP)–based echo application, which is similar to ICMP echo for the IP network; we then compare it with the RTTs using the current Internet. Figure 5 plots the RTT differences between SCIONLAB and the current Internet, i.e., $RTT_{\text{diff}} = RTT_{\text{SCMP}} - RTT_{\text{ICMP}}$. If the $RTT_{\text{diff}}$ is negative, it means that SCIONLAB provides better latency than the IP network for the particular node pair. Note that the results do not contain $RTT_{\text{diff}}$ for all node pairs as nine institutions hosting SCIONLAB nodes filter ICMP packets for "security reasons". Nonetheless, the number of the institution pairs that filter ICMP from both sides is low (36 out of 496 pairs) and we consider the missing results to be negligible and not affecting the overall result.

Since the majority of links in the current SCIONLAB topology are overlay connections, it can be expected that most values of $RTT_{\text{diff}}$ are positive, as there is an additional overhead such as queuing delay and processing delay taken by the software-defined SCIONLAB applications. As we can see in Figure 5 (left), $RTT_{\text{diff}}$ is almost always positive between node pairs with a *one-hop* distance, because of this additional overhead, although over 83 % of node pairs show a $RTT_{\text{diff}}$ with a tolerable $RTT_{\text{diff}} \leq 5$ ms. The negative $RTT_{\text{diff}}$ for *one-hop* measurements is observed between the backbone ASes due to the ICMP deprioritization on AWS.

With *multi-hop* communication, SCIONLAB achieves better network latency for almost one third of the node pairs. This prompts the question: "*How are these improvements even possible despite the additional delays in one-hop communication?*" The improvement is possible by taking advantage of the path-aware networking provided by SCIONLAB. For example, consider end-to-end communication where one end-host is in Europe and another is in East Asia. In the current Internet, the packets typically travel over North America, which is a serious detour, but there is no other option the end-host can take because it does not control routing. Recall that BGP does not always select the latency-optimized path [49]. In contrast, SCIONLAB provides various possible forwarding path candidates to users, allowing them to choose a desired path and forward their packets according to their own criteria, achieving a better forwarding path minimizing latency. Figure 5 (right) shows that SCIONLAB achieves better latency for a majority of node pairs with high ICMP latency ($RTT_{\text{ICMP}} \geq 200$ ms). There are 94 pairs with a negative $RTT_{\text{diff}}$ and $RTT_{\text{ICMP}} \geq 200$ ms, and out of them 54 do not have an AS from the backbone ISD in their forwarding path. This implies that the improvement on the better latency comes from path-awareness in most cases.

Two thirds of the cases score $RTT_{\text{diff}} \geq 0$ ms; one third incurs more than 25 ms of additional delay and 8 % of node pairs have $RTT_{\text{diff}} \geq 50$ ms. The additional delays are mostly caused by inevitable detours in the current SCIONLAB

topology. For example, since there is no direct link between USA and China in SCIONLAB, the packets need to be diverted via Singapore or South Korea, which introduces unnecessary delays. Nonetheless, we believe that these additional delays will be reduced when the SCIONLAB topology expands and contains more direct inter-AS connections.

**Path Stretch.** We measured *path stretch*, the fraction $d_a/d_s$ where $d_s$ is the delay of the shortest path (fewest hops) and $d_a$ is the propagation delay on an alternative path (second shortest path) between two ASes. Figure 6 illustrates the CDF of the path stretch for all the direct links in SCIONLAB. Over 60 % of the direct links result in a path stretch $d_a/d_s \leq 1.4$, which is the threshold for an alternative to be considered useful [28]; this indicates that SCIONLAB users have alternative path choices that can be used without exorbitant delays. We also emphasize that approximately 20 % of the direct links score $d_a/d_s \leq 1.0$ meaning that an alternative path with a better propagation delay exists. This results from the ability to select low-latency paths in SCIONLAB instead of the bandwidth-optimized paths in the Internet.

## VI. Discussion and Outlook

We now describe ongoing research projects running on SCIONLAB and report on some operational insights and challenges we have gained as a platform operator for the last three years. Finally, we outline how SCIONLAB is long-term viable.

### A. Research Projects

**Path-quality Prediction.** In path-aware networking, end-hosts need a good understanding of the network topology to choose routing paths. Even more, sources can predict the quality of feasible paths to ensure efficient data transmission regardless of dynamic changes of the link condition. Research on path-quality prediction has been conducted on SCIONLAB by changing path conditions (e.g., artificial link failures), and evaluating the performance of the prediction model [31], [58]. Such experiments are difficult to realize on existing testbeds.

**Resource Monitoring.** Multipath communication leads to a better use of the available link capacity, but it can also exacerbate the exhaustion of the shared network resources, which in turn impacts all the network participants. This contingency motivates research on how to monitor the total amount of bandwidth usage over multiple paths, and achieve fairness of bandwidth sharing in multipath environment. With SCIONLAB, the concept of enforced resource fairness and monitoring called SpeedCam has been evaluated [7].

**Reliable Blockchain Platform.** The emergence of a booming cryptocurrency scene has given attackers new high-stake targets. Thanks to the multitude of security benefits offered by SCION, applications such as cryptocurrency infrastructure can achieve high availability and performance, even in cases when the Internet experiences outages. As an ongoing research project, SCIONLAB is being utilized as a blockchain network platform that guarantees reliable networking with multipath communication [55].

**Selective Reachability.** The unlimited reachability on today's Internet, allowing all entities to send packets to anyone without explicit permission, makes end-hosts potentially vulnerable to unwanted traffic. Given the path-awareness shown in SCIONLAB, each entity can express the destination-driven decision about their reachability to the network. They do so by announcing the path availability only to trusted parties, in order to explicitly limit untrusted sources from engaging in end-to-end communication [33].

### B. Platform Administration

**Update Procedure.** We have automated the update process of the SCIONLAB infrastructure. Since SCIONLAB is a research testbed, and we constantly apply new state-of-the-art features, some updates break the backward compatibility (breaking changes). In these cases, we inform the users days in advance of the changes and the rollout date. User ASes running inside VMs are updated automatically on that date or whenever they reboot. Users that run dedicated systems can also follow a simple process that we indicate in the communication.

**Failure Detection.** We run `Prometheus` on all the SCIONLAB infrastructure nodes; this allows us to detect issues with the SCION services and the connectivity. We expose Prometheus metrics in all our SCION services, and additionally run `node exporter` [23] on all nodes. This setup provides us with a clear image of the stability and performance of each service. For all these metrics, we have set up alerts to detect when a service goes down. The alerts are delivered via email and a notification service.

**Troubleshooting.** To ensure the stable operation of SCION services, we employ `systemd` [25], a service management system that monitors and controls linux daemons. It automatically restarts a crashed SCION application. Each SCION application logs the status of the service, which helps with further investigations when troubleshooting. There is also the possibility of enabling jaeger traces [29], as the SCION services are instrumented for them.

### C. Challenges

**Operation behind NAT.** We had to solve the problem of how to reach the large number of users, who are behind NAT devices and are not able/authorized to configure the required port-forwarding rules. Our OpenVPN-based mechanism, where we run an OpenVPN server at each attachment point, enables us to bypass this issue.

**Reliance on AWS.** A weakness of the current topology is our reliance on AWS for the backbone ISD. This raises questions about long term funding, since the costs incurred depend on the bandwidth usage. We plan to solve the issue by collaborating more closely with international research network partners such as GÉANT, KREONET2 and Internet2, replacing AWS with those large-scale, high-speed networks that reach across the world. The new backbone is available in late 2020.

**Single Point of Failure (Coordinator).** The Coordinator's role as a central authority can be perceived as a weakness of SCIONLAB's architecture. However, the Coordinator does

Table II: Comparison of SCIONLAB's capabilities with other network testbeds.

| Capabilities | PlanetLab | Mininet | EmuLab | VINI | GENI | RIPE Atlas | PEERING | SCIONLAB |
|---|---|---|---|---|---|---|---|---|
| Global distribution | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Expandable topology | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Run user codes | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ |
| Routing control | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ |
| Instant participant | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ |
| Multipath support | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Path awareness | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Embedded cryptography | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |

not necessarily have to be globally unique. As all the source code is publicly available [6], there could be, in principle, multiple instances managing different parts of the SCIONLAB network. This would increase decentralization, reduce the load on the currently single Coordinator instance, and eliminate the potential single point of failure. Moreover, the impact of a Coordinator failure is limited, as it only affects the ability to make coordinated changes to the SCIONLAB network topology but not operation of the SCIONLAB network itself.

### D. Long-term Viability

We plan for SCIONLAB to expand in several phases. In the short term (6–18 months), additional ASes will be established at various universities and (cloud-based) research environments (e.g., NRENs, research testbeds). Some of these ASes will serve as additional attachment points. Moreover, the number of native layer-2 links between nodes will be increased. In the intermediate term (12–24 months), we envision to connect SCIONLAB to the PEERING testbed. We will also send out small-scale PCs (PC-engines) running preconfigured SCIONLAB ASes to individual groups, in order to expand the SCIONLAB network. In the long term (18–36 months), we aim to establish a vibrant community that will include corporations (cloud and others) offering ASes for experimentation.

## VII. Related Work

We discuss related work in two categories: network measurement platforms and network testbeds. In the category of measurement platforms, RIPE Atlas [41] focuses on diagnosing and troubleshooting real-world network infrastructure [14]. Passive and active measurement systems related to BGP include RouteViews [53] and BGP Beacons [37].

In the category of network testbeds, experimentation platforms are provided based on dedicated connections, overlays, or emulation and allow researchers to test out new networking services. Among the most influential ones are PlanetLab [44], EmuLab/Netbed [57], VINI [15], and GENI [38], each of them with a particular focus and catering to a specific class of experiments. Mininet [35] provides the possibility to even run a complete virtual network on a single machine. All of these testbeds are mainly designed for *intra-domain* research and do not allow its users to affect the inter-domain routing system. PEERING [48] combines intra-domain routing based on VINI, EmuLab, or Mininet with a control over the inter-domain routing system.

Compared to all these systems, SCIONLAB provides a number of innovations and new opportunities as summarized in Table II. A first important difference, in particular compared to the measurement platforms, is that SCIONLAB is not intended for research on the current Internet architecture and ecosystem. Instead, SCIONLAB seeks to provide research opportunities on a future Internet architecture providing path-aware networking and security mechanisms. Another difference is that users receive a first-class AS, obtaining all cryptographic credentials to participate in the global SCIONLAB control plane, and the opportunity for user ASes to create additional network connections. Thanks to the BYOC approach, computation scalability is provided to the user, also facilitating operation of the network infrastructure.

## VIII. Conclusion

SCIONLAB provides a global research testbed to support next-generation inter-domain research and development. SCIONLAB introduces several innovations: users create their own full-fledged ASes with their own cryptographic credentials to participate in global inter-domain routing and achieve computation scalability by running these ASes on their own computation infrastructure and connecting to the SCIONLAB network via distributed attachment points. This infrastructure provides an environment that enables novel research such as path-aware and multipath-enabled transport protocols, global key agreement based on AS-level certificates for secure communication, next-generation routing policies, traffic engineering, and DDoS defense mechanisms. We anticipate that SCIONLAB will expand over time to provide an even more densely connected, global network providing highly available end-to-end communication, which in turn could empower and support research that relies on a next-generation communication fabric.

## IX. Acknowledgements

We would like to thank J. Boogman, A. Chapuis, and D. Watrin from Swisscom, K. Baumann, D. Bertolo, A. Gall, and S. Leinen from SWITCH, B. Cho and W. Lee from KISTI, and A. Radhakrishnan, M. Usman, B. Weber, and E. Capone from GÉANT for support with the global deployments. We are grateful to M. Farb, C. Hähni, D. Roos, and E. Ucan who dedicated for the SCIONLAB implementation and impovement, and S. Gorinsky who provided early feedback. We also thank our shepherd, M. Chiesa, and the anonymous reviewers for their insightful feedback and suggestions. We gratefully acknowledge support from ETH Zurich, and from the Zurich Information Security and Privacy Center (ZISC). This work was also supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01697, Development of Automated Vulnerability Discovery Technologies for Blockchain Platform Security).

## REFERENCES


[1] Ansible. https://github.com/ansible/ansible.

[2] ISD and AS numbering. https://github.com/scionproto/scion/wiki/ISD-and-AS-numbering.

[3] SCION App. https://play.google.com/store/apps/details?id=org.scionlab.scion.

[4] SCION implementation. https://github.com/scionproto/scion/.

[5] SCION Tutorials. https://docs.scionlab.org/.

[6] SCIONLab codebase. https://github.com/netsec-ethz/scionlab.

[7] SpeedCam. https://github.com/Meldanor/SCIONLab_SpeedCam/.

[8] Supervisor: A process control system. http://supervisord.org/index.html.

[9] Hackathon influx dataset. https://drive.google.com/open?id=1IpN_z1lEzUvDgJ0KDIs6SKAp24oVsfhV, 2019.

[10] ETSI GS NGP 001. Next generation protocols (NGP); scenario definitions, January 2019.

[11] ETSI GS NGP 005. Next generation protocols (NGP); next generation protocol requirements, April 2017.

[12] Hakim Adhari, Thomas Dreibholz, and Martin Becke. SCTP socket API extensions for concurrent multipath transfer. IETF Internet Draft, https://datatracker.ietf.org/doc/draft-dreibholz-tsvwg-sctpsocket-multipath/, September 2018.

[13] AWS. Building a scalable and secure multi-VPC AWS network infrastructure. https://aws.amazon.com/answers/networking/aws-multiple-region-multi-vpc-connectivity/, 2013.

[14] Vaibhav Bajpai and Jurgen Schonwalder. A survey on Internet performance measurement platforms and related standardization efforts. *IEEE Communications Surveys & Tutorials*, 17(3), 2015.

[15] Andy C. Bavier, Nick Feamster, Mark Huang, Larry L. Peterson, and Jennifer Rexford. In VINI veritas: realistic and controlled network experimentation. In *Proceedings of the ACM SIGCOMM*, 2006.

[16] Daniel J Bernstein. Curve25519: new diffie-hellman speed records. In *International Workshop on Public Key Cryptography*, 2006.

[17] O. Bonaventure, C. Paasch, and G. Detal. Use cases and operational experience with multipath TCP. RFC 8041, January 2017.

[18] Matt Calder, Ashley Flavel, Ethan Katz-Bassett, Ratul Mahajan, and Jitendra Padhye. Analyzing the performance of an anycast CDN. In *ACM Internet Measurement Conference (IMC)*, 2015.

[19] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. RFC 5280, May 2008.

[20] Spencer Dawkins. Path aware networking: A bestiary of roads not taken. IETF Internet Draft, https://datatracker.ietf.org/doc/draft-dawkins-panrg-what-not-to-do/, June 2018.

[21] Quentin De Coninck and Olivier Bonaventure. Multipath QUIC: Design and evaluation. In *Proceedings of ACM CoNEXT*, 2017.

[22] DPDK Project. Data Plane Development Kit. https://dpdk.org.

[23] Cloud Native Computing Foundation. Node exporter. https://github.com/prometheus/node_exporter.

[24] Cloud Native Computing Foundation. Prometheus. https://github.com/prometheus/prometheus/.

[25] freedesktop.org. systemd system and service manager. https://www.freedesktop.org/wiki/Software/systemd/.

[26] Agostino Funel. The graph structure of the Internet at the autonomous systems level during ten years. *Journal of Computer and Communications*, 7(8), 2019.

[27] P Brighten Godfrey, Igor Ganichev, Scott Shenker, and Ion Stoica. Pathlet routing. *ACM SIGCOMM Computer Communication Review*, 39(4):111–122, 2009.

[28] Nikola Gvozdiev, Stefano Vissicchio, Brad Karp, and Mark Handley. On low-latency-capable topologies, and their impact on the design of intra-domain routing. In *Proceedings of the ACM SIGCOMM*, 2018.

[29] jaegertracing.io. Jaeger: open source, end-to-end distributed tracing. https://www.jaegertracing.io/.

[30] Yuchen Jin, Sundararajan Renganathan, Ganesh Ananthanarayanan, Junchen Jiang, Venkata N. Padmanabhan, Manuel Schroder, Matt Calder, and Arvind Krishnamurthy. Zooming in on wide-area latencies to a global cloud provider. In *Proceedings of the ACM SIGCOMM*, 2019.

[31] Zeno Koller. Measuring loss and reordering with few bits, 2018. Master's thesis.

[32] Donald Kossmann, Tim Kraska, Simon Loesing, Stephan Merkli, Raman Mittal, and Flavio Pfaffhauser. Cloudy: A modular cloud storage system. *Proceedings of the VLDB Endowment*, 3(1-2), 2010.

[33] Jonghoon Kwon, Taeho Lee, Claude Hähni, and Adrian Perrig. SVLAN: Secure & scalable network virtualization. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS)*, 2020.

[34] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. The QUIC transport protocol: Design and Internet-scale deployment. In *Proceedings of the ACM SIGCOMM*, 2017.

[35] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, 2010.

[36] Priya Mahadevan, Dmitri Krioukov, Marina Fomenkov, Xenofontas Dimitropoulos, Amin Vahdat, et al. The Internet AS-level topology: three data sources and one definitive metric. *ACM SIGCOMM Computer Communication Review*, 36(1), 2006.

[37] Z. Morley Mao, Randy Bush, Timothy G. Griffin, and Matthew Roughan. BGP beacons. In *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2003.

[38] Rick McGeer, Mark Berman, Chip Elliott, and Robert Ricci, editors. *The GENI Book*. Springer International Publishing, 2016.

[39] Oliver Michel and Eric Keller. SDN in wide-area networks: A survey. In *Proceeding of Fourth International Conference on Software Defined Systems (SDS)*, 2017.

[40] A. Morton. IMIX Genome: Specification of Variable Packet Sizes for Additional Testing. RFC 6985 (Informational), July 2013.

[41] RIPE NCC. RIPE Atlas. https://atlas.ripe.net/.

[42] Aleksi Peltonen. A map of the Internet. Project report, https://csperkins.org/research/routing/2017-05-01-peltonen-project/report.pdf, March 2017.

[43] Adrian Perrig, Pawel Szalachowski, Raphael M. Reischuk, and Laurent Chuat. *SCION: A Secure Internet Architecture*. Number ISBN 978-3-319-67080-5, https://www.scion-architecture.net/pdf/SCION-book.pdf. Springer Verlag, 2017.

[44] Larry L. Peterson, Andy C. Bavier, Marc E. Fiuczynski, and Steve Muir. Experiences building PlanetLab. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI)*, 2006.

[45] Miroslav Ponec, Sudipta Sengupta, Minghua Chen, Jin Li, and Philip A Chou. Multi-rate peer-to-peer video conferencing: A distributed approach using scalable coding. In *IEEE International Conference on Multimedia and Expo*, 2009.

[46] Costin Raiciu, Christoph Paasch, Sebastien Barre, Alan Ford, Michio Honda, Fabien Duchene, Olivier Bonaventure, and Mark Handley. How hard can it be? designing and implementing a deployable multipath TCP. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (NSDI)*, 2012.

[47] Sara Retal, Miloud Bagaa, Tarik Taleb, and Hannu Flinck. Content delivery network slicing: Qoe and cost awareness. In *IEEE International Conference on Communications (ICC)*, 2017.

[48] Brandon Schlinker, Kyriakos Zarifis, Italo Cunha, Nick Feamster, and Ethan Katz-Bassett. PEERING: An AS for Us. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, 2014.

[49] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. Making middleboxes someone else's problem: network processing as a cloud service. *ACM SIGCOMM Computer Communication Review*, 42(4), 2012.

[50] Mourad Soliman, Biswajit Nandy, Ioannis Lambadaris, and Peter Ashwood-Smith. Source routed forwarding with software defined control, considerations and implications. In *Proceedings of ACM CoNEXT student workshop*, 2012.

[51] Carl A Sunshine. Source routing in computer networks. *ACM SIGCOMM Computer Communication Review*, 7(1), 1977.

[52] Brian Trammell, Jean-Pierre Smith, and Adrian Perrig. Adding path awareness to the Internet architecture. *IEEE Internet Computing*, 22(2), March 2018.

[53] University of Oregon. University of Oregon route views project. http://www.routeviews.org.

[54] Tobias Viernickel, Alexander Froemmgen, Amr Rizk, Boris Koldehofe, and Ralf Steinmetz. Multipath QUIC: A deployable multipath transport protocol. In *IEEE International Conference on Communications (ICC)*, 2018.

[55] Aleksandar Vorkapic. Secure blockchain network communication using SCION, 2018. Master's thesis.

[56] Yuefeng Wang, Flavio Esposito, Ibrahim Matta, and John Day. RINA: An architecture for policy-based dynamic service management. Technical Report BUCS-TR-2013-014, November 2013.

[57] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the 5th Symposium on Operating System Design and Implementation (OSDI)*, Dec. 2002.

[58] François Wirz. Network performance evaluation and prediction on SCION, 2018. Master's thesis.

[59] Wen Xu and Jennifer Rexford. Multi-path interdomain routing. In *Proceedings of the ACM SIGCOMM*, 2006.

[60] Xiaowei Yang, David Clark, and Arthur W Berger. Nira: a new interdomain routing architecture. *IEEE/ACM Transactions On Networking*, 15(4):775–788, 2007.

[61] Kok-Kiong Yap, Murtaza Motiwala, Jeremy Rahe, Steve Padgett, Matthew Holliman, Gary Baldus, Marcus Hines, Taeeun Kim, Ashok Narayanan, Ankur Jain, Victor Lin, Colin Rice, Brian Rogan, Arjun Singh, Bert Tanaka, Manish Verma, Puneet Sood, Mukarram Tariq, Matt Tierney, Dzevad Trumic, Vytautas Valancius, Calvin Ying, Mahesh Kallahalla, Bikash Koley, and Amin Vahdat. Taking the edge off with Espresso: Scale, reliability and programmability for global Internet peering. In *Proceedings of the ACM SIGCOMM*, 2017.

[62] Xin Zhang, Hsu-Chun Hsiao, Geoffrey Hasker, Haowen Chan, Adrian Perrig, and David Andersen. SCION: Scalability, control, and isolation on next-generation networks. In *Proceedings of IEEE Symposium on Security and Privacy*, May 2011.

[63] Yiming Zhang, Dongsheng Li, Zhigang Sun, Feng Zhao, Jinshu Su, and Xicheng Lu. CSR: Classified source routing in distributed networks. *IEEE Transactions on Cloud Computing*, 6(2), 2018.