Path-aware Access Control: Granting Access with Transit Network Attributes

Jonghoon Kwon
ETH Zürich
Zürich, Switzerland
jong.kwon@inf.ethz.ch

Julian Modanese
ETH Zürich
Zürich, Switzerland
jmodanese@student.ethz.ch

Jordi Subirà Nieto
ETH Zürich
Zürich, Switzerland
jordi.subiranieto@inf.ethz.ch

Adrian Perrig
ETH Zürich
Zürich, Switzerland
adrian.perrig@inf.ethz.ch

***Abstract*—Access control systems typically evaluate the security aspects of communication endpoints to determine access permission, but they often overlook network-level threats. We argue that access control decisions for remote entities should be made with explicit consideration of the transit network environment. In this study, we propose a novel concept called PAAC, path-aware attribute-based access control, which extends existing endpoint-oriented access decisions by considering transit network information. By incorporating network metrics, the access control system enables finer-grained access control and mitigates network-level threats such as BGP hijacking. Our experiments demonstrate that PAAC achieves comparable performance and scalability to existing attribute-based access control approaches.**

***Index Terms*—attribute-based access control, path awareness**

I. Introduction

Access control serves as the front line of security measures, allowing only authorized users to access resources or perform commands [5], [10], [12], [19], [23] Typically, access control systems evaluate the threat level of communication endpoints to determine access permission. They assess the security aspects of subjects (e.g., user roles, hardware, and protocols) in relation to the objects' security postures and conduct correlation analysis to determine access authorization.

Yet, network threats are not considered in the access control decision-making process, apart from the information provided by the endpoints (see Figure 1). Unless remote systems use a completely isolated private network that links to the local network, an attacker can impersonate legitimate access requests or, in severe cases, send multiple access requests to exploit vulnerabilities in the access control system, breaching the system. Even with the use of secure stream protocols such as Transport Layer Security (TLS) to establish secure tunnels for end-to-end security, vulnerabilities of man-in-the-middle attacks still exist [16], [22], [27].

In this paper, we propose a new concept called path-aware access control (PAAC). It extends the existing endpoint-oriented access decisions by considering the information about the transit networks used for communication. In particular, the path-based network information represents various network metrics such as ingress/egress interfaces, number of hops, and trustworthiness of autonomous systems (AS) on the path.

We gratefully acknowledge support from ETH Zurich, and from the Zurich Information Security and Privacy Center (ZISC).

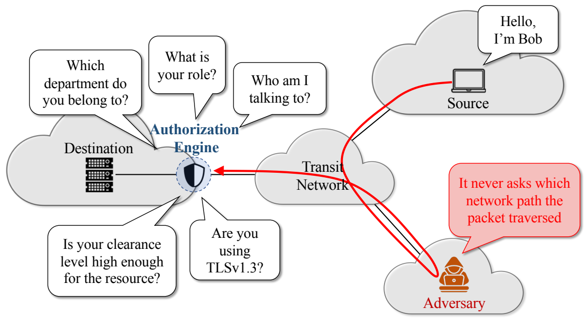
Which department do you belong to?
What is your role?
Who am I talking to?
Hello, I'm Bob
Source
Authorization Engine
Destination
Transit Network
It never asks which network path the packet traversed
Is your clearance level high enough for the resource?
Are you using TLSv1.3?
Adversary

Fig. 1. A case the modern access-control solutions cannot protect. The lack of consideration for transit networks introduces vulnerabilities to network attackers.

By incorporating the additional network metrics, the access control system enables finer-grained access control and can address—uncovered yet but existing—network-level threats such as Border Gateway Protocol (BGP) hijacking or insecure AS traversal [2], [7].

We developed a PAAC module based on Casbin [1], a well-established open-source access control library that supports traditional Attribute-Based Access Control (ABAC). To extend ABAC with transit network attributes, we integrated it into the SCION path-based routing architecture [6], leveraging underlying forwarding path information from packets in transit. Performance evaluations show that PAAC requires an average of 50 μs for path extraction from packet headers and rule matching, a negligible overhead given that typical Internet latency is on the order of hundreds of ms. This results in less than a 4% reduction in throughput compared to Casbin's standard ABAC approach.

In summary, this paper makes the following contributions:

- We first point to the lack of consideration over network attributes in the common access control systems.
- We propose a new notion of path-ware access control, arguing that the access control should consider network path, risk assessment of intermediaries, and path properties as the access decision metrics.
- We implement and evaluate PAAC, a new path-aware attribute-based access control system, and
- We describe ideas on how path-based attributes can be further extended and utilized.

II. Related Work

A. Access Control Policy Models

The access control policy establishes a set of rules to determine the appropriate action, such as granting or denying a request, copying and forwarding it, or requiring additional authentication, in response to each access request. Different access control models have been proposed to address various requirements. The Discretionary Access Control (DAC) model [12] is commonly used when users define permissions for their resources, often represented as an Access Control List (ACL) specifying acceptable access patterns. The Mandatory Access Control (MAC) model [19] involves a security operator assigning access levels to users and resources. Role-based Access Control (RBAC) [23] relies on role assignments to users and defines resource access permissions accordingly. The ABAC model [10], [11] is a more comprehensive framework that considers attributes of the subject, object, operation, and environment to define permissions. DAC, MAC, and RBAC can be incorporated into the ABAC framework. Risk-Adaptive Access Control (RAdAC) [5] explicitly assesses the trade-offs between risk and operational demands to make access control decisions.

B. Path-aware Networking

Path-aware networking [28] refers to communication techniques where the end-to-end path packets traverse is perceived by all the network entities, including sender, receiver, and intermediate forwarding devices. Unlike conventional BGP-based networking, where network entities independently perform routing decisions and the data plane only has information about the next hop for the destination address, path-aware networking typically includes complete forwarding path information in the packet header, ensuring transparency regarding path information.

Source Routing. A prominent concept in path-aware networking is source routing [26], which extends the notion of path-based routing to the endpoints, enabling senders to specify the forwarding path. The network control plane explores available forwarding paths for all sender-receiver pairs, and disseminates the path information to the endpoints. The data plane ensures that packets are transmitted along the sender-specified forwarding path [30].

Segment Routing. Segment routing [8] is another form of path-aware networking that divides the entire network into segments, encoding the desired path directly into the packet header using a stack of segment identifiers. An example implementation of segment routing is SR-MPLS [29], which involves stacking MPLS labels in the layer-2.5 MPLS header field, enabling network devices to forward packets based on the prescribed label sequence. Another notable technique is Segment Routing IPv6 (SRv6) [15]. It is based on the IPv6 forwarding plane, implementing the segment routing capability with the IPv6 routing extension header field (layer-3.5) describing a sequence of network nodes.

Future Internet Architecture. SCION [6], a future internet technology, represents a more advanced form of path-aware networking. SCION partitions the entire network into up, down, and core segments, with its control plane regularly broadcasting path exploration beacons to gather and distribute all available path segments. This approach empowers users to compose the path segments and construct end-to-end forwarding paths, enabling a considerable degree of programmability in path construction.

C. Path-based Access Control

Path-Aware Risk Scores for Access Control (PARSAC) [24] evaluates the network path taken by access request, assigning risk scores considering the number of hops and geographical locations. Nonetheless, it focuses on risk assessment rather than explicit access control and enforcement. Another pioneer in this domain is [4]. Each application in an enterprise network modifies the request metadata by appending its own access control information. As the request traverses the network, it carries an accumulated path history, allowing downstream applications to evaluate the path through which the packet went. However, the system is tailored for service-oriented architecture (SOA) networks, considering inter-application paths.

III. Path-aware Access Control

This section presents PAAC, a new ABAC solution that considers the attributes of transit networks for access control.

A. Technical Background

ABAC solutions currently combine resource, environmental, and user attributes to perform authorization decisions: i) resource attributes primarily describe the access target, ii) user attributes pertain to the individuals seeking access, and iii) environmental attributes encompass various properties related to the context of access attempts. Upon access request, these attributes are evaluated based on predefined access policies. Policy Enforcement Point (PEP) extracts attributes from a packet, generates an authorization request, and forwards it to Policy Decision Point (PDP). PDP evaluates the request by comparing it to the pre-defined access policies, and determines whether to grant or deny the request. The access decision is enforced at PEP.

For example, consider a scenario where Bob, a developer working at a European branch office, wants to pull the latest codebase from a development server at the US headquarters datacenter. In this case, we may have the following attributes:

User Attributes {Name="Bob", Role="Developer", Department="R&D", Clearance="7"}
Resource Attributes {Type="Code", Owner="R&D", Updated="06/30/23", SensitivityLevel="5"}
Environmental Attributes {Time="9:36 AM PDT", Location="CH", Protocol="TLS v1.3"}.

With an access policy stated, "*A developer can read codes if they are the owner, their clearance is higher than the sensitivity level of the codes, and it is working hours*," Bob would be able to download the codebase from the development server.

Yet, what if we need to consider third-party entities? For instance, assume that there is a network with national censorship between the European branch and the US headquarter. All

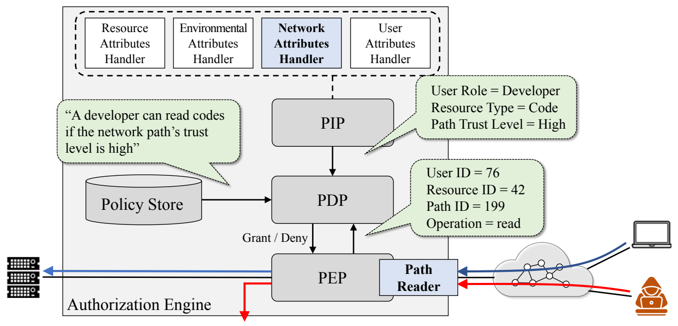

Fig. 2. A path-aware access control architecture.

traffic passing through that network is subject to censorship, potentially leading to the exposure of valuable business assets, e.g., a new product's source code. In this case, the network administrator would want to restrict access to sensitive assets when traffic passes through that network. However, the current ABAC techniques do not support establishing and projecting policies considering such network-level constraints. The only solution might involve setting up an expensive MPLS network to ensure trusted network paths. However, deploying a private network for all distributed branches worldwide is not practical. So, are there any more realistic and efficient alternatives?

In addition to the subject, object, and contextual attributes in ABAC solutions, we propose considering network attributes as a significant factor in authorization decision-making. This is the first argument that emphasizes the need to incorporate the attributes of the network connecting the two endpoints into the access control decision process.

B. System Architecture

PAAC incorporates two new modules into traditional ABAC solutions: *Path Reader* and *Network Attributes Handler*. Figure 2 illustrates the PAAC architecture.

Path Reader. When a packet arrives, the forwarding path information conveyed in the packet header is extracted to obtain the corresponding network attributes. This extraction is performed by Path Reader, which is located in the data plane's last mile and equipped with the network stack of the path-based routing infrastructure. The extracted path information—along with other attributes such as user ID, target resource, and request time extracted from other modules like a web proxy—is then forwarded to the Policy Decision Point (PDP).

Similar to other ABAC approaches, here we do not specify the physical configuration of the Path Reader, meaning that it can be integrated with other modules of the PEP or implemented as a standalone module. This lack of structural specificity allows for various configurations depending on the header design of the underlying path-aware network architecture, network setup, and implementation of the ABAC system. For instance, consider a scenario where a PEP is implemented behind a web proxy server behind an SR-MPLS switch. The switch, serving as the last endpoint of the SR-MPLS data plane, removes the MPLS header field from the packet before handing it over to the server [29]. In this case, the Path Reader implemented in the PEP would not have access to the path information, and therefore, it would need to be implemented within the switch. Similarly, SRv6 typically removes the IPv6 Segment Routing Header (SRH) at the last hop, but it can retain the SRH until it reaches the end host through options [15]. Additionally, in the case of SCION, the Path Reader can be implemented at the end host through a software-based SCION-IP-Gateway, enabling the forwarding path to be visible to the end host [6].

Network Attributes Handler. The network path serves as a significant indicator of information pertaining to transit networks through which packets are transmitted; however, it possesses limited inherent meaning. Consequently, the Network Attributes Handler (NAH) assumes a vital role in collecting information concerning the targeted network paths and supplying it to the Policy Decision Point (PDP). NAH stores attributes of interest related to prominent networks, which can be defined from the network administrator's organizational perspective. For instance, reliable full-path information to branch offices, reliability metrics for core networks, and presets for normal routing paths are pre-collected and organized within a database. Furthermore, Policy Information Point (PIP) can establish a secure channel with the control plane of the underlying path-aware infrastructure, facilitating the collection and storage of network changes and metadata associated with each path. Consequently, the collected information is conveyed in the form of attributes specific to each network path upon request by the PDP.

C. Network Attribute Properties

Network attributes encompass various properties depending on the underlying path-aware networking architecture, and here we discuss two main properties (see §III-D for their expressiveness).

Trust. This property is to determine the trustworthiness of the network path used for packet transmission. Similar to other attributes, trust of the ASes, routers, and switches involved in the path—and even the entire path—can be pre-evaluated and utilized in decision-making [14]. For example, assigning low trust to ASes located in countries with censorship or to routers with reported vulnerabilities, and using policies, e.g., "a user can view documents if the path does not include untrusted ASes," to make grant/deny decisions. Note that we do not need to keep this property for every network entity; by storing only the entities of interest, PAAC becomes scalable regardless of the size of the network.

Performance. Some path-aware networking architectures disseminate link-state information for each path [9]. One of the key strengths of path-aware networking is the ability to select and switch paths as needed. Based on service-specific routing metrics of the application, the available properties of each path (e.g., latency, bandwidth, and jitter) between the endpoints can be reviewed, and the most suitable path that meets the performance requirements can be chosen. For example, a teleconferencing application may selectively use a low-latency path, while a bulk transfer application may prioritize a high-

```
<?xml version="1.0" encoding="UTF-8"?>
 <Name>Example Policy</Name>
 <Rules>
  <Rule>
   <Name>Rule 1</Name>
   <Condition>
    <SourceIP>192.168.0.0/24</SourceIP>
    <PathID>80A3FE</PathID>
   </Condition>
   <Action>Allow</Action>
  </Rule>
  <Rule>
   <Name>Rule 2</Name>
   <Condition>
    <TransitASID>66666</TransitASID>
   </Condition>
   <Action>Deny</Action>
  </Rule>
  <Rule>
   <Name>Rule 3</Name>
   <Condition>
    <SourceIP>172.16.0.0/16</SourceIP>
    <Latency>50</Latency>
   </Condition>
   <Action>Allow</Action>
  </Rule>
 </Rules>
</Policy>
```

Listing 1. An example Policy.xml code considering network attributes.

bandwidth path. Additionally, if a link fails (e.g., congestion or failure), a quick transition to an alternate path is possible. To maximize the flexibility of path selection, the network control plane continuously probes and disseminates various status information for each link to network entities.

The performance metrics corresponding to the status of each path can be utilized in access control decision-making. For applications that require real-time interactions between server and client, upper limits of tolerable latency can be enforced as a factor in access control decisions. It is important to note that we do not perform real-time performance evaluations for each access request packet. Rather, assuming that we have prior knowledge of the performance metrics for each path derived from the underlying network architecture, we assess whether the path through which the packet was transmitted satisfies the minimum performance requirements defined for that access.

D. Path-aware Access Policy Expressiveness

The standard markup languages used in traditional ABAC systems, such as eXtensible Access Control Markup Language (XACML) [25], enable network administrators to establish various access control policies with high policy expressiveness. Network administrators define declarative, fine-grained access control policies that describe how access requests should be evaluated based on their respective goals and attribute-based criteria. Each access control goal is represented in the form of rules. Rules comprise conditions (i.e., sets of attributes to be satisfied) that serve as factors in determining whether to grant or deny an access request. Furthermore, complex rules that combine grant and deny rules using logical operators (e.g., AND, OR, NOT) can be implemented to achieve more granular control over access decisions.

PAAC can leverage the standard markup languages as well. As shown in Listing 1, network attributes can be used in access control within each rule by simply grouping them with other contextual attributes. Depending on the network information the underlying path-aware network architecture provides, access control policies can be established and applied based on full-path-based control (Rule 1), AS-based control (Rule 2), and even certain routers or switches. If path-specific metadata and properties—retrieved from the network control-plane or telemetry systems—are available, more precise access decisions can be made. For example, access will be denied for a path that passes through the untrusted transit AS (Rule 2) among various paths that satisfy $latency < 50\ ms$ from an AS with an IP prefix of 172.16.0.0/16 (Rule 3). Regarding conflicting rules, explicit priority levels are often assigned.

IV. Evaluation

We conducted benchmarks using a proof-of-concept (PoC) implementation to evaluate the feasibility of PAAC. This section outlines our implementation and summarizes the results.

A. Implementation and Experimental Setup

We have implemented a PAAC module as an extension of Casbin [1], an open-source authorization library that supports various access control models such as ACL, RBAC, and ABAC. Leveraging Casbin as the underlying access control system enables us to incorporate best practices in performance evaluation, facilitating a precise comparison between PAAC and traditional ABAC. To implement path-awareness, we integrated the SCION network stack [6]. The PAAC module listens for connections from remote endhosts, extracts path information, and synthesizes it to generate a path fingerprint. PAAC's NAH interfaces with both PIP and the SCION control plane, mapping path fingerprints to the corresponding path information retrieved from the control plane. Additionally, the NAH functions as a management interface for supplementary path metadata, such as performance metrics gathered by telemetry services. Our implementation is composed of modular components that operate concurrently and communicate through buffered Go channels, making it easily extendable to support other (path-aware) data-plane packet formats. The PoC implementation is publicly available [20].

All benchmarks were conducted on an AMD FX-8350 8-core processor with stock settings, running Ubuntu 22.04.4 LTS. Attribute maps were populated with n entries, evenly distributed across `int`, `float64`, and `string` types. The policy sizes ranged from 1 to 1,000 rules, with each rule containing 2 to 20 attribute evaluations. Within each policy, the sub-rules consist of attribute evaluations chained by OR operators. For simplicity and consistency, attribute evaluations are performed against constants of the corresponding types, rather than against other attributes. Each benchmark was executed 30 times.

B. Performance

To quantify the impact of the newly introduced PAAC module, we measured the processing time per packet and overall throughput of PAAC in comparison to Casbin's ABAC.

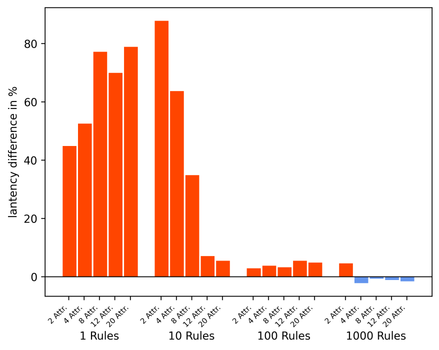

Fig. 3. Relative latency difference.

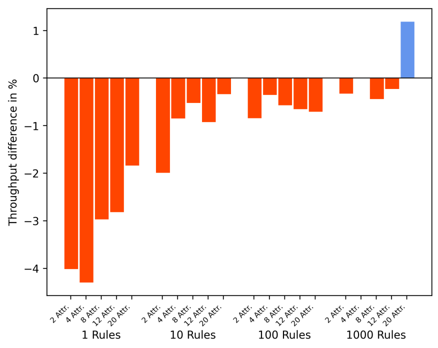

Fig. 4. Performance comparison in throughput.

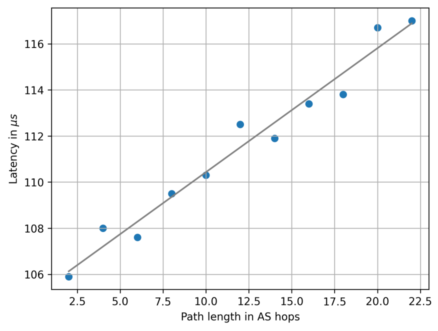

Fig. 5. Latency inflation per AS hop.

As shown in Figure 3, the PAAC module introduces noticeable overhead in per-packet processing time. This overhead initially increases as policy complexity grows but diminishes for larger policy sets, stabilizing at around 3-5% latency inflation. For very large policy sets, we even observe a slight reduction in latency, as PAAC's overhead becomes insignificant compared to the overall processing time, which averages around 50 μs.

Interestingly, the overall performance impact of PAAC, even under full load, is relatively low. Figure 4 shows the relative throughput of PAAC compared to ABAC, showing a 4% reduction in throughput for the smallest policy set. This overhead gradually decreases as policy size increases. Pipelining requests through buffered channels in our implementation significantly boosts performance, allowing individual goroutines to process requests without being blocked by access decisions.

It is important to note that the performance evaluations were conducted with path caching enabled, allowing PAAC to bypass retrieving path metadata from the SCION control plane, which incurs a constant delay of 1.2 ms, regardless of policy size. Nevertheless, this overhead is negligible, as caching is common practice in modern network systems, and the cache's TTL can inherit the path's expiration time, typically ranging from an hour to a day. In a nutshell, our PAAC PoC implementation demonstrates that incorporating path-awareness to ABAC is feasible with a minimal performance cost, resulting in less than a 4% reduction in throughput.

C. Scalability

PAAC demands more processing time as the network size grows. To evaluate the impact of network size on PAAC's performance, we tested various path lengths for end-to-end communication, ranging from two AS hops (where source and destination hosts are in adjacent networks) to 22 AS hops, representing typical Internet paths. Figure 5 shows the additional per-packet processing time, along with a trend line that highlights the linear increase in PAAC's runtime with longer paths. The added latency per hop is about 0.5 μs, resulting in a total increase of 11 μs for the longest path, which is negligible compared to the average Internet communication latency (order of hundreds of ms). Furthermore, PAAC's processing time can be further improved using Data Plane Development Kit (DPDK) optimizations.

V. Discussion

Path Validation. When making path-aware access control decisions, trust in the extracted network path information is a critical prerequisite. Therefore, path validation is imperative. It necessitates: 1) the ability to authenticate the sender, 2) ensuring the integrity of conveyed routing information, and 3) enabling each network entity to verify whether the intended routing decision is being executed.

Fortunately, research on path authentication and verification has been actively conducted to enforce path compliance in path-aware networking environments. ICING [21] uses cryptographic computation to verify paths by generating Message Authentication Codes (MAC) for each intermediate router. Each on-path router performs symmetric cryptographic operations for all routers on the path, ensuring both path verification and routing policy enforcement. OPT [13] focuses on verifying the path at each hop instead of calculating MAC for all routers, while EPIC [18] prevents the reuse and injection of valid path information by dynamically changing the per-hop MAC for each packet. Thanks to the effort, we can safely assume that such path verification techniques are used when collecting network attributes for access control.

Path Consent. Legitimate users requesting remote access to a target network may, for a seamless user experience, require an implicit understanding of the access control policies enforced by the target network. Path-aware networking architectures typically follow the source routing paradigm, enabling the sender to select the desired path for packet transmission based on their path selection policy (e.g., regarding trust, performance, and subscription model). It implies that the recipient has no control over the sender's path selection. However, If the path violates access policies, packets are discarded without notifying the sender. To ensure seamless access, prior consent for specific paths may be needed.

Network administrators can adopt different approaches based on the type of network access, which can be categorized into private and public. Private access involves users with proper privileges (e.g., employees) from known source networks, where trusted paths or access policies can be shared in advance, similar to conventional practices, e.g., enforcing specific VPN applications to access enterprise or school networks from external. Public access, however, involves less trustworthy or temporary users (e.g., web service users or contractors), where determining the source in advance or sharing policies is impractical. In such cases, a path negotiation system [3] can be used, allowing the recipient network to implicitly communicate acceptable communication paths to the sender before the exchange begins.

Limitations and Future Work. The granularity of path awareness varies depending on the underlying networking architectures; MPLS-SR operates on layer-2 switches, SRv6 relies on layer-3 routers, and SCION provides path awareness on AS border routers. To implement PAAC compatible with various path-aware networking architectures, multi-layer network attributes must be considered. Another challenge is to define path policies in loosely specified path-aware networks. For example, in MPLS-SR, non-MPLS switches using *MPLS-over-GRE* [17] may have incorrect segment information. Similarly, SRv6 operates with legacy routers via SRv6 encapsulation [15], allowing routing paths to be arbitrarily determined. Although such loose path awareness is an essential design decision for deployability and backward compatibility, it requires additional information for each network segment, hampering precise access control policy establishment.

In future work, we aim to conduct a comprehensive analysis of the efficacy and limitations of PAAC for various path-aware networking architectures. We also intend to explore details such as establishing an elastic interface with the network control planes through PIP and implementing real-time update techniques for network attributes.

VI. Conclusion

In this paper, we proposed a new notion of "path-aware access control" that leverages attributes of transit networks to consider not only "*who has access to whom*" but also "*through whom*" in access decisions. The intentionally under-specified design enables flexible system implementation tailored to the underlying network architectures. Our prototype implementation demonstrates that it is possible to perform network attribute extraction and rule matching with negligible overhead, even in Internet-scale path-aware network environments. We believe that PAAC conveys a hopeful message of achieving more secure network systems from network threats in current and future Internet landscapes. We anticipate that our study will lay the groundwork for follow-up research in this domain.

References

[1] Casbin Go package. https://github.com/casbin/casbin, v2.87.1.

[2] H. Birge-Lee, L. Wang, J. Rexford, and P. Mittal. SICO: Surgical Interception Attacks by Manipulating BGP Communities. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2019.

[3] M. Blarer, J. Kwon, V. Graf, and A. Perrig. Consent Routing: Towards Bilaterally Trusted Communication Paths. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2022.

[4] M. Burnside and A. D. Keromytis. Path-based Access Control for Enterprise Networks. In *International Conference on Information Security*. Springer, 2008.

[5] P.-C. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger. Fuzzy Multi-level Security: An Experiment on Quantified Risk-adaptive Access Control. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2007.

[6] L. Chuat, M. Legner, D. Basin, D. Hausheer, S. Hitz, P. Müller, and A. Perrig. *The Complete Guide to SCION*. Springer, 2022.

[7] T. Dai, P. Jeitner, H. Shulman, and M. Waidner. The Hijackers Guide To The Galaxy: Off-Path Taking Over Internet Resources. In *Proceedings of the USENIX Security Symposium*, 2021.

[8] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir. Segment Routing Architecture. *RFC 8402*, 2018.

[9] M. Frei. Path Metadata in Beacons. https://github.com/scionproto/scion/blob/master/doc/beacon-metadata.rst (last visited Apr 19, 2021), 2021.

[10] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, K. Scarfone, et al. Guide to Attribute-Based Access Control (ABAC) Sefinition and Considerations (draft). *NIST special publication*, 800(162):1–54, 2013.

[11] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas. Attribute-based access control. *Computer*, 48(2):85–88, 2015.

[12] C. S. Jordan. *Guide to Understanding Discretionary Access Control in Trusted Systems*. DIANE Publishing, 1995.

[13] T. H.-J. Kim, C. Basescu, L. Jia, S. B. Lee, Y.-C. Hu, and A. Perrig. Lightweight Source Authentication and Path Validation. In *Proceedings of the ACM SIGCOMM*, 2014.

[14] C. Krähenbühl, M. Wyss, D. Basin, V. Lenders, A. Perrig, and M. Strohmeier. FABRID: Flexible Attestation-Based Routing for Inter-Domain Networks. In *Proceedings of the USENIX Security Symposium*, 2023.

[15] J. Leddy, D. Voyer, S. Matsushima, and Z. Li. Segment Routing over IPv6 (SRv6) Network Programming. *RFC 8986*, 2021.

[16] S. Lee, Y. Shin, and J. Hur. Return of Version Downgrade Attack in the Era of TLS 1.3. In *Proceedings of the ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2020.

[17] W. Lee, R. Bhagavathula, N. Thanthry, and R. Pendse. MPLS-over-GRE based VPN Architecture: A Performance Comparison. In *Proceedings of the Midwest Symposium on Circuits and Systems*, 2002.

[18] M. Legner, T. Klenze, M. Wyss, C. Sprenger, and A. Perrig. EPIC: Every Packet is Checked in the Data Plane of a Path-aware Internet. In *Proceedings of the USENIX Security Symposium*, 2020.

[19] J. M. McCune, T. Jaeger, S. Berger, R. Caceres, and R. Sailer. Shamon: A System for Distributed Mandatory Access Control. In *Proceedings of the IEEE Annual Computer Security Applications Conference (ACSAC)*, 2006.

[20] J. Modanese and J. Kwon. PAAC Codebase. https://github.com/netsec-ethz/PAAC.

[21] J. Naous, M. Walfish, A. Nicolosi, D. Mazieres, M. Miller, and A. Seehra. Verifying and Enforcing Network Paths with ICING. In *Proceedings of the ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2011.

[22] E. Ronen, R. Gillham, D. Genkin, A. Shamir, D. Wong, and Y. Yarom. The 9 lives of bleichenbacher's cat: new cache attacks on tls implementations. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2019.

[23] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[24] J. H. Seaton, S. Hounsinou, T. Wood, S. Xu, P. N. Brown, and G. Bloom. Poster: Toward Zero-Trust Path-Aware Access Control. In *Proceedings of the ACM on Symposium on Access Control Models and Technologies (SACMAT)*, 2022.

[25] O. Standard. eXtensible Access Control Markup Language (XACML) Version 3.0. *A:(22)*, 2013. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html.

[26] C. A. Sunshine. Source Routing in Computer Networks. *ACM SIGCOMM Computer Communication Review*, 7(1):29–33, 1977.

[27] W. J. Tolley, B. Kujath, M. T. Khan, N. Vallina-Rodriguez, and J. R. Crandall. Blind In/On-Path Attacks and Applications to VPNs. In *Proceedings of the USENIX Security Symposium*, 2021.

[28] B. Trammell, J.-P. Smith, and A. Perrig. Adding Path Awareness to the Internet Architecture. *IEEE Internet Computing*, 22(2):96–102, 2018.

[29] X. Xu, S. Bryant, A. Farrel, S. Hassan, W. Henderickx, and Z. Li. MPLS Segment Routing over IP. *RFC 8663*, 2019.

[30] Y. Zhang, D. Li, Z. Sun, F. Zhao, J. Su, and X. Lu. CSR: Classified Source Routing in Distributed Networks. *IEEE Transactions on Cloud Computing*, 6(2), 2018.