opsec
Internet-Draft
Intended status: Informational
Expires: 4 December 2023

F. Gont
G. Gont
SI6 Networks

2 June 2023

Implications of IPv6 Addressing on Security Operations
draft-ietf-opsec-ipv6-addressing-00

Abstract

The increased address availability provided by IPv6 has concrete implications on security operations. This document discusses such implications, and sheds some light on how existing security operations techniques and procedures might need to be modified accommodate the increased IPv6 address availability.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 4 December 2023.

Table of Contents

1. Introduction

The main driver for the adoption of the IPv6 protocol suite is its increased address space, which can provide a vast number of public addresses for every device attached to the public Internet.

IPv6 addresses [RFC4291] can differ in a number of properties, such as address scope (e.g. link-local vs. global), stability (e.g. stable addresses vs. temporary addresses), and intended usage type (outgoing communications vs. incoming communications).

IPv6 hosts may configure and use multiple addresses with different combinations of the aforementioned properties, depending on the local host policy and the local network policy. For example, in networks where Stateless Address Auto-configuration (SLAAC) is employed for address configuration, host will typically configure one stable address and one (or more) temporary addresses per network interface, for each prefix advertised advertised for address configuration. On the other hand, in networks that employ Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC8415] for stateful configuration, it is quite common for hosts to configure a single stable address per network interface.

Section 2 discusses the semantics of IPv6 addresses in terms of the entity or entities the identify, according to the deployed Internet. Section 3 discusses the semantics of IPv6 addresses in terms of the entity or entities the identify, according to the deployed Internet.

Section 4 discusses the usage of IP addresses in security operations. Section 5 discusses the implications of IPv6 addressing on security operations. Finally, Section 6 provides guidance on the usage of IPv6 addresses for security operations.

2. The Semantics of IPv4 Addresses and IPv4 Prefixes

While the original Internet architecture defined IP addresses to identify a network interface, the deployed Internet has embraced Network Address Translation (NAT) over time. Initially, NATs were deployed on customers premises (e.g., in Customer Edge routers). However, as a result of the imminent IPv4 address space exhaustion, Internet Service Providers (ISPs) have resorted to deploying the so-called Carrier-Grade NATs (CGNs).

As a result, in scenarios where an ISP has *not* deployed CGN, an IP address will tyipically identify one single customer, whereas in scenarios where an ISP has deployed CGN, a single IPv4 address may identify multiple customers. Depending on the type of "customer" (e.g., a home user vs. an educational institution), one or multiple actors might be associated with the "customer" in question.

In the IPv4 Internet, one may assume that an attacker will have control over a single IPv4 address (an IPv4 /32). However, an attacker might be able to leverage DHCP RELEASE messages to switch over different IPv4 addresses (/32s), and hence control more than a single IPv4 address.

3. The Semantics of IPv6 Addresses and IPv6 Prefixes

As noted in Section 1, IPv6 hosts typically configure multiple addresses with different properties. One of the most common deployment scenarios is that in which the subnet employs SLAAC [RFC4862] for address configuration, and where hosts configure both stable [RFC8064] [RFC7217] and temporary [RFC8981] IPv6 addresses. From this perspective, it is clear that multiple addresses may correspond to the same IPv6 host.

While rather uncommon in legitimate use cases, an IPv6 host may configure addresses from a larger address block. For example, it is common for Internet Service Providers (ISPs) to lease a /56 or a /48 to each subscriber, and thus a skilled user could readily employ the leased prefix in a single or multiple IPv6 hosts (whether virtual or not).

On the other hand, while one might assume that an IPv6 address would correspond to at most one host (strictly speaking, to one network interface of a host), this is not necessarily the case in the

deployed Internet. For example, deployments that employ "Network Address Port Translation + Protocol Translation" (NAPT-PT) [RFC2766] for IPv6 are not uncommon, whether along with technologies such as Kubernetes, or in IPv6-enabled VPNs. Thus, a single IPv6 address may actually identify multiple IPv6 hosts.

4. Security Operations

There is a whole range of security processes and operations that involve the usage of IP addresses. This document discusses the implications of IPv6 addressesing on security operations via two examples:

* Enforcement of Access Control Lists (ACLs)

* Network Activity Correlation

The following subsections discuss these two examples in detail.

NOTE: [RFC9099] provides an overview of the most important aspects of IPv6 security operations, whereas this document elaborates on the implications of IPv6 addressing on security operations.

4.1. Enforcement of Access Control Lists (ACLs)

It is common for network deployments to implement any of these types of Access Control Lists (ACLs):

* Allow-lists

* Block-lists

Allow-lists are typically employed as part of a defense-in-depth [NSA] strategy, where access to specific resources may be allowed only when requests originate from specific IP addresses or prefixes. For example, an organization may employ a Virtual Private Network (VPN), and require that certain resources be accessed only via the VPN, by enforcing that requests originate from the IP address (or addresses) of the VPN concentrator.

On the other hand, block-lists are typically implemented to mitigate threats. For example, a network firewall might be fed with an IP reputation block-list that is dynamically updated to reflect the IP address (or addresses) of known or suspected attackers.

Both types of ACLs have a similar challenge in common: identifying the minimum set of addresses that should be employed in the ACLs definitions such that the ACLs can successfully enforce the controls

they are expected to enforce while minimizing collateral damage. For example, in the case of allow-lists, the corresponding ACLs should encompass possible legitimate changes in the set of legitimate addresses, thus avoiding false negatives (i.e., incorrectly preventing access to legitimate users). On the other hand, in the case of block-lists, the ACLs should encompass the attacker's ability to use different addresses (or vantage points), while minimizing false positives (i.e., incorrectly blocking legitimate users).

4.2. Network Activity Correlation

Another fundamental aspect of security operations is that of network activity correlation (at times with the goal of attribution). That is, a security analyst may want or need to infer the relationship among different network activities, and possibly assess whether they can be attributed to the same actor. This may be necessary for security investigations, but also to e.g. subsequently mitigate a threat by enforcing ACLs that block the alleged attacker.

5. Implications of IPv6 Addressing on Security Operations

5.1. Access-Control Lists

A key question when implementing ACLs is deciding which granularity to use for the ACL specification. If one were to follow IPv4 practices, one would be tempted to specify ACLs with a /128 granularity (i.e., the equivalent to a /32 in the IPv4 world). However, as noted in Section 3, most IPv6 host implementations employ IPv6 temporary addresses [RFC8981], and thus an allow-list specified as a /128 would eventually fail. Thus, one might be tempted to specify an allow-list as a /64 -- that is, an entire /64 might need to be "allowed", to accommodate the usage of IPv6 temporary addresses [RFC8981]). However, since such IPv6 prefix might be shared by other hosts in the same subnet, this would likely result in false-positives (i.e., all hosts in the target /64 would be allows access) -- which is probably unacceptable in most cases.

In some scenarios, a network administrator might be able to disable the use of temporary addresses [RFC8981] via e.g. group policies [GPO], or by enforcing the use of DHCPv6 [RFC8415], thus having more control on the addresses employed by local hosts. In these specific cases, it might be possible to implement an allow-list for a host by specifying a single IPv6 address (i.e., a /128).

NOTE: Some IPv6 host implementations, notably the one in the Android operating system, do not support DHCPv6. Therefore, the option to enforce DHCPv6 usage might be unfeasible.

On the other hand, implementing block-lists may also be tricky. For example, IP reputation lists (whether commercial or not) are commonly employed in the deployed Internet, and used to e.g. dynamically configure ACLs on devices such as firewalls. However, these IP reputation lists generally specify offending addresses as /128. This means that an attacker could simply regularly change his/her IPv6 address, thus reducing the effectiveness of these lists. Additionally, an attacker regularly changing his/her address might (whether intentionlly or inadvertently) cause the block-list to grow to such an extent that the proper functioning of the associated filtering devices might be affected -- and thus the filtering device may have to resort to trimming the block-list.

Similarly, tools of the kind of [fail2ban] are commonly employed by system administrators to mitigate e.g. brute-force authentication attacks by banning IP addresses after a certain number of failed authentication attempts. These tools might ban IPv6 addresses on a /128 granularity, thus meaning that an attacker could easily circumvent these controls by changing the IPv6 source address every few attempts (e.g. before an address becomes banned). Additionally, as with the IP reputation lists previously discussed, an attacker performing a brute force attack *and* regularly changing his/her addresses could cause the block-list grow to an extent where it might negatively affect the system enforcing the block-list, or might cause other legitimate entries to be discarded in favor of the transient IPv6 addresses.

One might envision that IPv6 reputation lists might aggregate a large number of offending IPv6 addresses into a prefix that encompasses them. However, this practice is not really widespread, and it might also increase the number of false positives. Thus, this is a topic that may warrant further research.

5.2. Network Activity Correlation

Performing IPv6 network activity correlation can be very tricky, since the semantics of an IPv6 address in terms of what an address may identify (see Section 3) can be complex. As discussed before, a single IPv6 address could correspond to either a single host, or multiple hosts behind an IPv6 NAPT-PT device -- this being similar to IPv4 scenarios.

However, multiple IPv6 addresses might or might not identify multiple different actors. In some cases, some heuristics might help infer whether a group of addresses belonging to a /64 correspond to the same host. However, as the attacking addresses become more sparse (e.g., an attacker leverages a /48), this may be more challenging. And, while some heuristics could be employed to perform network

activity correlation across multiple addresses, most tools commonly used in the deployed Internet do not implement these kind of features.

NOTE: Section "2.6.2.3. Correlation" of [RFC9099] discusses network activity correlation for local nodes, whereas [IPv6-Scanning] discusses the challenges of network activity correlation when detecting IPv6 scanning attacks.

6. Advice on Security Operations

6.1. Access-Control Lists

This section provides advice on the usage of IPv6 ACLs, whether as allow-lists or block-lists.

6.1.1. Allow-lists for the Destination Address of Incoming Packets

This type of ACLs are typically enforced when a network firewall is meant to allow incoming packets to subset of nodes on the local subnet.

The feasible granularity of such allow-lists will depend on the address configuration method employed in the local network. ACLs with a granularity of /128 will only be feasible if:

* DHCPv6 IA_NA is employed to lease stable addresses to local hosts, and IA_TA (DHCPv6 temporary addresses) is disabled, or,

* SLAAC is employed for host address configuration, and use of temporary addresses [RFC8981] is disabled, or,

* Manual configuration is employed for the local hosts.

It should be noted that, as a result of Neighbor Cache Exhaustion (NCE) attacks [RFC6583], it might be desirable to limit the allowed destination address ranges to the IPv6 addresses or prefixes that are actually in use in the target network. For example, in scenarios where DHCPv6 is employed, allow-lists for the destination address of incoming packets could be specified with the same granularity as the DHCPv6 addresss pool -- e.g. in a /64 subnet where a DHCPv6 server leases addresses from a a /112, a /112 prefix coould be used to specify an allow-list for such group of DHCPv6 hosts. However, when SLAAC is employed on the local subnet, and IPv6 temporary addresses [RFC8981] are enabled, the entire /64 would need to be allowed when specifying an allow-list for the Destination Address of incomming packets.

NOTE: The only alternative to specifying a /64 allow list would be to configure (stateless) ACLs for the stable addresses of the IPv6 hosts, and allow for the dynamic creation of stateful rules for packets that originate from the local network.

6.1.2. Allow-lists for the Source Address of Incoming Packets

These type of ACLs are typically employed to allow incoming packets only when they originate from a specific IP addresses or prefix.

In a similar vein as the allow-lists from Section 6.1.1, the granularity of these allow-lists will depend on the address configuration method employed at the network where packets originate.

Allow-lists with a granularity of /128 will only be feasible if:

* DHCPv6 employed for address configuration, and IA_TA (DHCPv6 temporary addresses) is disabled, or,

* SLAAC is employed for host address configuration, and temporary addresses [RFC8981] are disabled, or,

* Manual configuration is employed for the local hosts.

NOTE: SLAAC [RFC4862] does not provide a mechanism to convey a policy as to whether temporary addresses [RFC8981] should be configured. This policy is typically a local host policy, which may be overriden via out-of-band mechanisms such as GPOs [GPO]. Since temporary address are typically preferred over stable addresses, a granularity of /128 will only be feasible if temporary addresses are disabled.

In scenarios where DHCPv6 is employed at the remote network, allow-lists for the source address of incomming packets could be specified with the same granularity as the DHCPv6 addresss pool of the remote network. For example, in a /64 subnet where a DHCPv6 server leases addresses from a a /112, a /112 prefix coould be used to specify an allow-list for the group of DHCPv6 hosts.

In all other cases, it would be unfeasible to specify an allow-list for the source address of incoming packets with a granulary other than /64, since their addresses would be ramdomly selected from the /64 prefix.

6.1.3. Block-lists for the Source Address of Incoming Packets

As noted in Section 3, attackers may have control over large IPv6 address blocks, and might be able to change their IPv6 addresses within such address blocks, rendering /128 IPv6 block-lists ineffective.

Security technologies meaning to enforce IPv6 block-lists should be able to infer when an attacker has control over an IPv6 address block, such that the granularity of the associated block-lists can be dynamically adapted to effectively enforce the intended controls. This subsection describes one possible way to implement this.

The following table specifies one possible set of parameters to be employed with this implementation:

LEVEL	PREF_LEN	AGGR_THRES	ACL_LIFETIME
1	/128	10	1 hour
2	/64	10	45 min
3	/56	10	30 min
4	/48	N/A	15 min

Table 1: ACL Granularities

The meaning of each of the parameters is as follows:

LEVEL:
: ACLs may be enforced with different granularity levels, ranging from 1 to N, where 1 corresponds to the finest granularity, and N corresponds to the coarsest granularity.

PREF_LEN:
: Prefix Length corresponding to each granularity level. In our table, the finest granularity is a /128, whereas the coarsest granularity is a /48.

AGGR_THRES:
: A threshold specifying the number of ACLs of this level that, if/ when possible, should be aggregated into an ACL of level (n+1).

ACL_TIME:
: The maximum lifetime of an ACL for this level.

The algorithm would work as follows:

1. If/when malicious activity is detected for an IPv6 address, create LEVEL=1 ACL (i.e., an ACL with a /128 granularity, and a lifetime of ACL_LIFETIME(1).

2. If/when possible, aggregate at least AGGR_THRES(n) LEVEL(n) ACLs into a single LEVEL(n+1) ACL (with a ACL_LIFETIME(n+1) lifetime).

3. Once ACL_LIFETIME(n) has elapsed, eliminate the associated LEVEL(n) ACL.

As an example, if e.g. offending activity were detected for the IPv6 address 2001:2b8:0:1::1, a 2001:2b8:0:1::1/128 ACL would be created. If offending activity was subsequently detected for the IPv6 address 2001:2b8:0:1::2, a 2001:2b8:0:1::2/128 ACL would be created. If a total of 10 (AGGR_THRES(1)) offending IPv6 addresses were detected, the associated LEVEL(1) ACLs would be aggregated into a single 2001:2b8:0:1::/64 (LEVEL(2))ACL, with a lifetime of 1 hour (ACL_LIFETIME(2)). If offending acitvities were detected for IPv6 addresses in the 2001:2b8:0:2::/64 prefix, individual /128 ACLs would be created for each IPv6 address, which would eventually be aggregated into a single 2001:db8:0:2::/64 ACL. Then, if 10 (AGGR_THRES(2) /64 (AGGR_LEVEL(2)) ACLs were eventually created in the 2001:2b8::/56 (AGGR_LEVEL(3)) prefix, these ACLs would be aggregated into a 2001:2b8::/56 (AGGR_LEVEL(3)) ACL, etc.

6.2. Network Activity Correlation

As discussed in Section 3, performing IPv6 network activity correlation can be tricky. As the bare minimum, security tools should allow security analysts to select the granularity to be employed when performing network activity correlation. For example, security tools should allow security analysts to specify that all activities withina given /128, /64, /56, or /48 correspond to the same actor.

7. Security Considerations

This entire document is about the implications of IPv6 addressing on security operations. It analyzes the impact of IPv6 addressing on a number of security operations areas, raising awareness about the associated challenges, and providing guidance on how IPv4 security operation practices should be dapted to embrace IPv6.

8. Acknowledgements

The authors of this document would like to thank (in alphabetical order) Henri Alves de Godoy, Nick Buraglio, Rich Compton, Oliver Gasser, William Herrin, Henrik Kramselund, Ted Lemon, Jen Linkova, Markus Reschke, Andrew Ruthven, Eduard Vasilenko, Eric Vyncke, and Andrew Walding, for providing valuable comments on earlier versions of this document.

This document borrows some text and analysis from [I-D.gont-v6ops-ipv6-addressing-considerations], authored by Fernando Gont and Guillermo Gont.

Fernando would also like to thank Nelida Garcia and Jorge Oscar Gont for their love and support.

9. References

9.1. Normative References

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <https://www.rfc-editor.org/info/rfc7217>.

[RFC8064] Gont, F., Cooper, A., Thaler, D., and W. Liu, "Recommendation on Stable IPv6 Interface Identifiers", RFC 8064, DOI 10.17487/RFC8064, February 2017, <https://www.rfc-editor.org/info/rfc8064>.

[RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <https://www.rfc-editor.org/info/rfc8415>.

[RFC8981] Gont, F., Krishnan, S., Narten, T., and R. Draves, "Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6", RFC 8981, DOI 10.17487/RFC8981, February 2021, <https://www.rfc-editor.org/info/rfc8981>.

[RFC9099] Vyncke, É., Chittimaneni, K., Kaeo, M., and E. Rey, "Operational Security Considerations for IPv6 Networks", RFC 9099, DOI 10.17487/RFC9099, August 2021, <https://www.rfc-editor.org/info/rfc9099>.

9.2. Informative References

[fail2ban] fail2ban, "fail2ban project", <https://www.fail2ban.org/>.

[GPO] Microsoft, "Windows Server 2012 R2 and Windows Server 2012", 2016, <https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/hh831791(v=ws.11)>.

[I-D.gont-v6ops-ipv6-addressing-considerations] Gont, F. and G. Gont, "IPv6 Addressing Considerations", Work in Progress, Internet-Draft, draft-gont-v6ops-ipv6-addressing-considerations-02, 1 June 2022, <https://datatracker.ietf.org/doc/html/draft-gont-v6ops-ipv6-addressing-considerations-02>.

[IPv6-Scanning] Richter, P., Gasser, O., and A. Berger, "Illuminating Large-Scale IPv6 Scanning in the Internet", IMC '22: Proceedings of the 22nd ACM Internet Measurement Conference, Pages 410-418, <https://doi.org/10.1145/3517745.3561452>, October 2022, <https://olivergasser.net/papers/richter2022illuminating.pdf>.

[NSA] NSA, "Defense in Depth: A practical strategy for achieving Information Assurance in todays highly networked environments", <https://web.archive.org/web/20121002051613/https://www.nsa.gov/ia/_files/support/defenseindepth.pdf>.

[RFC2766] Tsirtsis, G. and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, DOI 10.17487/RFC2766, February 2000, <https://www.rfc-editor.org/info/rfc2766>.

[RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, DOI 10.17487/RFC6583, March 2012, <https://www.rfc-editor.org/info/rfc6583>.

Authors' Addresses

Fernando Gont
SI6 Networks
Evaristo Carriego 2644
1706 Haedo
Provincia de Buenos Aires
Argentina
Email: fgont@si6networks.com
URI: https://www.si6networks.com

Guillermo Gont
SI6 Networks
Evaristo Carriego 2644
1706 Haedo
Provincia de Buenos Aires
Argentina
Email: ggont@si6networks.com
URI: https://www.si6networks.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 10 January 2024

C. Liu, Ed.
Q. Wu
L. Xia
Huawei Technologies

9 July 2023

On Network Path Validation
draft-liu-on-network-path-validation-00

Abstract

Network path validation refers to a technology that ensures data packets to strictly travel along a chosen network path. It aims to enforce data to travel only on the assigned network path and provide evidence that the data has indeed followed this path. While existing efforts primarily focus on the control plane, path validation protects and monitors routing security in the data plane. This document provides a technical definition of the Network Path Validation problem, briefly overviews past efforts, models its ideal solution and design goals, and lists out different use case across various layers of the Internet.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 10 January 2024.

Copyright Notice

Table of Contents

1. Introduction

In the current Internet architecture, the network layer provides best-effort service to the endpoints using it [RFC9217]. This means that the endpoints are unaware, unable to visualize, and unable to control the network path between them, and thus the traffic inside the path too. This deficiency not only undermines Internet routing security but also hampers the development of new concepts like path-aware networking [RFC9217][PAIA]. Exploiting this vulnerability, various routing attacks have emerged, including:

* Routing Hijack / Prefix Hijack: AS (Autonomous System) incorrectly announces prefix ownership, diverting normal traffic to the wrong AS.

* Route Injection / Traffic Detour: Attacker injects additional hops into a path, redirecting traffic to locations where it can be monitored, analyzed, or even manipulated before being sent back to the original destination.

* Route leak: Propagation of routing announcements beyond their intended scope [RFC7908], causing unintended ASes to receive traffic.

* Denial of Service (DOS): Adversary overwhelms important routers with interfering traffic, preventing them from receiving and processing valid traffic.

These attacks exploit the trusting and flexible nature of the Internet, resulting in unreliability in both path establishment and actual data forwarding. To address this issue, several works are proposed focusing on securing network path in the control plane. Resource Public Key Infrastructure (RPKI) [RFC6810] consider IP prefixes as resources, and their ownership must be proven by signed statements called Route Origin Authorizations (ROAs), issued by the root CA or authorized CAs of the Internet Routing Registry -- similar to how certificates work in regular PKI. Through a chain of ROAs, BGPSec [RFC8205] can secure an AS path.

While these measures provide necessary authentication services and enhance routing security in the control plane, they have limitations. Securing a path in the control plane does not necessarily mean we can control and track the actual forwarding of traffic within these paths. To put it simply, even though we have secured highways to connect correct locations so that cars can reach their intended destinations, controlling how cars actually travel on the highways and reliably logging their movements is a separate challenge. In order to achieve this goal, an effective path validation mechanism should enable data packets to carry both mandatory routing directives and cryptographically secure transit proofs in their headers. This mechanism should serve as an orthogonal complement to existing techniques that primarily focus on the control plane. Cisco made an exploratory attempt by designing a Proof of Transit scheme using modified Shamir Secret Sharing [I-D.ietf-sfc-proof-of-transit-08]. Although they did not provide a rigorous security proof and the work regretfully discontinued but the question they asked is too significant to be left undiscussed.

2. Use Cases

We have compiled a list of use cases that highlight the importance of path validation. We invite discussions to add more cases, aiming to cover as many scenarios as possible.

2.1. Use Case 1: Proof of Service Level Assurance

Internet Service Providers (ISPs) often have different levels of routing nodes with varying service qualities. When customers like Alice subscribe to premium plans with higher prices, it is reasonable for them to expect superior connection quality, including higher bandwidth and lower latency. Therefore, it would be beneficial to have a mechanism that ensures Alice’s traffic exclusively traverses

premium routing nodes. Additionally, it is important to provide Alice with verifiable proof that such premium services are indeed being delivered.

2.2. Use Case 2: Proof of Security Service Processing

Service Function Chaining enables the abstraction of services such as firewall filtering and intrusion prevention systems. Enterprises need to demonstrate to others or verify internally that their outbound and inbound traffic has passed through trusted security service functions. In this context, the service function acts as the node that must be transited. After the processing and endorsing of these security service functions, traffic becomes verifiably more reliable and more traceable, making it possible to reduce spamming and mitigate Distributed Denial-of-Service (DDoS) attacks.

2.3. Use Case 3: Security-sensitive Communication

Routing security is a critical concern not only on the Internet but also within private networks. End-to-end encryption alone may not be sufficient since bad cryptographic implementations could lead to statistical information leak, and bad cryptographic implementation or API misuse is not uncommon [BADCRYPTOIMPL1][BADCRYPTOIMPL2]. If a flow of traffic is maliciously detoured to the opposing AS and secretly stored for cryptanalysis, useful information (such as pattern of plaintexts) could be extracted by the adversary. Thus, when given a specific path or connection, it is crucial to ensure that data packets have solely traveled along that designated route without exceeding any limits. Ultimately, it would be advantageous to provide customers with verifiable evidence of this fact.

3. Design Goals

As the name suggests, the Network Path Validation mechanism aims to achieve two main goals:

1. Enforcement: Committing to a given network path and enforcing traffic to traverse the designated nodes in the specified order.

2. Validation: Verify the traffic indeed transited the designated nodes in exact order specified for this path.

The enforcement and validation to the traffic forwarding are two sides of a coin. In order to achieve these goals, two additional pieces of information must be added to the data header.

1. Routing Directive: A routing directive commands the exact forwarding of the data packet hop-by-hop, disobeying which will cause failure and/or undeniable misconduct records.

2. Transit Proof: A transit proof is a cryptographic proof that securely logs the exact nodes transited by this data packet.

4. Modelling the Ideal Solution

4.1. Roles:

The path validation mechanism should include three roles:

* The network operator chooses or be given a routing path P and commit to it. P = (n_1, n_2, ..., n_i, ..., n_N) is an ordered vector consists of N nodes. The network operator also does the setup and pre-distribution of the public parameters.

* The forwarding "node" is a physical network device or a virtual service that processes and forwards the data traffic. Within that path, this node is the minimal atomic transit unit meaning there are no other perceptible inferior nodes between these regular nodes.

* The observer is an abstract role that represents public knowledge. Any publicized information is known to the observer. Any person or device who is interested in examining the trustworthiness of this routing path could be an instance of observer. An observer can verify publicized information such as node identity or transit proof with an unbiased property.

4.2. Required Functionality:

The path validation mechanism consists of the following algorithms:

1. Configure: Setup control plane parameters based on a security parameter.

    * Input: Security parameter

    * Output: Control plane parameter distributed

2. Commit: Generates a commitment proof for the chosen path using public parameters and the path itself.

    * Input: public parameters, path P

    * Output: Commitment Proof C of the path P

3. CreateTransitProof (in-situ / altogether): Generates transit proofs for individual nodes or sets of nodes, either during data processing or when transmission finishes.

    * Input: public parameters, index i of node n_i or indices I of a set of node n_I, identity information of node n_i or set of nodes n_I.

    * Output: Transit proof p_i or batch transit proof p_I

4. VerifyTransitProof (in-situ / altogether): Verifies transit proofs for individual nodes or sets of nodes, either in-step or all at once.

    * Input: public parameters, transit proof p_i/p_I, index i of node n_i or indices I of a set of node n_I, identity information of node n_i or set of nodes n_I.

    * Output: success = 1, fail = 0

The Network Operator performs the Configure and Commit steps. The CreateTransitProof step could be done by either each node n_i during he is processing the data, or the end node n_N when the transmission finishes altogether. That being said, the VerifyTransitProof step can also be executed in an in-situ (for step-by-step control and visibility) or one-time fashion. Usually the VerifyTransitProof step is executed by the observer, but it can also be executed by the next-hop node for origin verification.

5. Security

As we can see, the creation and verification of the transit proof is the critical part of the mechanism. Therefore, we define the security of the Network Path Validation mechanism around the security of the transit proof:

We say a Network Path Validation mechanism is secure if the transit proof is correct, unforgeable and binding.

* *Correctness:* Transit proof created by the right node n_i at the position i must pass the verification. (probability of a correct proof not passing verification is smaller than a negligible function)

* *Unforgeability:* Transit proof at position i must only be created by the node n_i. (probability of a forged proof passing verification is smaller than a negligible function)

* *Binding:* An identity value at position i different than what is committed created by polynomial adversary cannot pass a verification check.

Other security discussions like replay attack resistance are discussed separately. Since transit proof is added to the header, the compactness of proof, short proof creation and verification time is also critical. Ideally:

* *Efficiency:* The creation time, verification time and size of the transit proof is sublinear to the number of total nodes on a path.

6. IANA Considerations

This document has no IANA actions.

7. References

7.1. Normative References

[RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <https://www.rfc-editor.org/rfc/rfc8205>.

[RFC6810] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol", RFC 6810, DOI 10.17487/RFC6810, January 2013, <https://www.rfc-editor.org/rfc/rfc6810>.

7.2. Informative References

[RFC9217] Trammell, B., "Current Open Questions in Path-Aware Networking", RFC 9217, DOI 10.17487/RFC9217, March 2022, <https://www.rfc-editor.org/rfc/rfc9217>.

[RFC7908] Sriram, K., Montgomery, D., McPherson, D., Osterweil, E., and B. Dickson, "Problem Definition and Classification of BGP Route Leaks", RFC 7908, DOI 10.17487/RFC7908, June 2016, <https://www.rfc-editor.org/rfc/rfc7908>.

[I-D.ietf-sfc-proof-of-transit-08] Brockners, F., Bhandari, S., Mizrahi, T., Dara, S., and S. Youell, "Proof of Transit", Work in Progress, Internet-Draft, draft-ietf-sfc-proof-of-transit-08, 1 November 2020, <https://datatracker.ietf.org/doc/html/draft-ietf-sfc-proof-of-transit-08>.

[PAIA] "Adding Path Awareness to the Internet Architecture", April 2018, <https://ieeexplore.ieee.org/document/8345560>.

[BADCRYPTOIMPL1] "Secure coding practices in Java":" challenges and vulnerabilities", May 2018, <https://dl.acm.org/doi/10.1145/3180155.3180201>.

[BADCRYPTOIMPL2] "Mining Your Ps and Qs":" Detection of Widespread Weak Keys in Network Devices", August 2012, <https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/heninger>.

Authors' Addresses

Chunchi Liu (editor)
Huawei Technologies
101 Ruanjian Ave
Nanjing
210012
China
Email: liuchunchi@huawei.com

Qin Wu
Huawei Technologies
China
Email: bill.wu@huawei.com

Liang Xia
Huawei Technologies
China
Email: frank.xialiang@huawei.com