Workgroup: v6ops
Updates: 1122, 4291 (if approved)
Published: 29 March 2022
Intended Status: Informational
Expires: 30 September 2022
Authors: P. Thubert, Ed.
Cisco Systems

Yet Another Double Address and Translation Technique

Abstract

This document provides a mechanism named YADA to extend the current IPv4 Internet by interconnecting IPv4 realms via a common footprint called the shaft. YADA extends [INT-ARCHI] with the support of an IP-in-IP format used to tunnel packets across the shaft. This document also provides a bump-in-the-stack method to enable YADA on a legacy stack, e.g., to enable virtual machines without changing them. This document also provides a stateless address and IP header translation between YADA and IPv6 [IPv6] called YATT and extends [IPv6-ADDRESSING] for the YATT format. YATT can take place as a bump in the stack at either end, or within the network and enables an IPv6-only stack to dialog with an IPv4-only stack across a network that can be IPv6, IPv4, or mixed. YATT requires that the IPv6 stack owns a prefix that derives from a YADA address and the IPv4 stack is capable of YADA, so it does not replace a generic 4 to 6 translation mechanism for any v6 to any v4.

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 30 September 2022.

Table of Contents

1. Introduction

This document provides a mechanism called Yet Another Double Address (YADA) to grow the Internet beyond the current IPv4 [IPv4] realm that limits its capacity to form public addresses. This is achieved by interconnecting IPv4 realms via a common footprint called the shaft.

In the analogy of a building, the ground floor would be the Internet, and each additional floor would be another IPv4 realm. The same surface of floor is available in each level, analog to the full IPv4 addressing that is available in each realm. The same footprint is dedicated across the building levels for the elevator shaft. The elevator shaft enables a third dimension that spans across the levels and allows to traverse from any level to any other level. The elevator shaft cannot be used for living or office space.

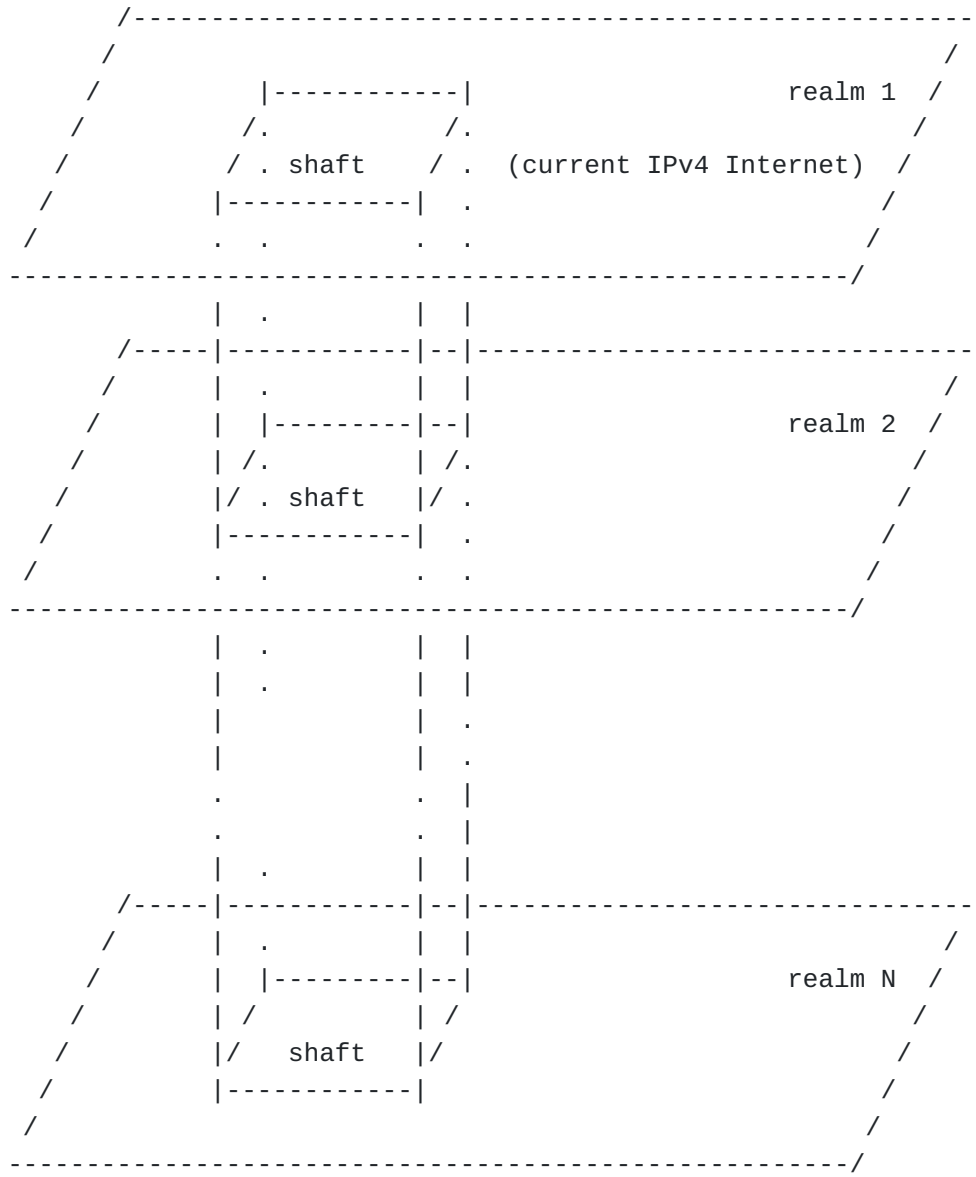

Figure 1: The shaft

By analogy, YADA assigns IPv4 prefixes to a multinternet shaft; those prefixes are common across the realms that are interconnected by the shaft. A single /24 IPv4 prefix assigned allows for > 250 times the capacity of the Internet as we know it at the time of this writing. Multiple prefixes can be assigned to the shaft for unicast and multicast communications, and each realm needs at least one unicast address in the shaft called its realm address. A YADA address is formed by the tuple (realm address, IPv4 address) and is advertised in DNS as a new double-A record.

YADA leverages IP-in-IP encapsulation to tunnel packets across the shaft while normal IPv4 operations happen within a realm. YADA requires a change in the stack in the YADA endpoints that communicate with other realms to support the IP-in-IP YADA encapsulation. YADA also provides a bump in the stack method for legacy applications. More in Section 5.

A second mechanism called Yet Another Translation Technique (YATT) translates the YADA format into flat IPv6 [IPv6]. For unicast addresses, YATT forms an IPv6 prefix by collating an well-known assigned short prefix, the realm address (in the shaft), and the host IPv4 address (locally significant within the realm). The resulting IPv6 prefix is automatically owned by the host that owns the IPv4 address in the realm. YATT then forms an IPv6 address for that host by collating a well-known Interface ID, so there's a one-to-one relationship between the YADA and the IPv6 address derived from it. More in Section 6.

2. Terminology

2.1. Glossary

This document often uses the following acronyms:

YADA: Yet Another Double Address
YATT: Yet Another Translation Technique
NAT: Network address Translation
IID: Interface ID
CG-NAT: Carrier Grade NAT

2.2. New Terms

This document often uses the following nex terms:

IPv4 realm A full IPv4 network like the current Internet. YADA does not affect the traditional IPv4 operations within a realm.
The shaft The shaft refers to a collection of IPv4 unicast and multicast prefixes that are assigned to Inter-realm

communications and cannot be assigned to hosts or multicast groups within a realm.

realm address An IPv4 address that derives from a shaft prefix.

Uni-realm address A realm address that is unicast or anycast. A realm may have more than one Uni-realm address.

Multi-realm address A realm address that is multicast and denotes a collection of realms.

YADA realm Prefix A Prefix assigned to the shaft and from which realm addresses can be derived.

YADA NAT Prefix A Prefix assigned to the YADA bump-in-the-stack NAT operation.

Double-A or YADA address A YADA address is a tuple (realm address, IPv4 address) where the IPv4 address is only significant within the realm denoted by the realm address.

YATT Space An IPv6 range that is assigned for YATT operation.

YATT Prefix An IPv6 prefix that is derived from a YADA address by appending the YATT space prefix, the (truncated) realm address and the IPv4 address.

YATT-IID: A 64-bit assigned constant that is used in YATT to statelessly form an IPv6 address from a YATT prefix.

Multinternet A collection of IPv4 realms interconnected using a common shaft.

3. Extending RFC 1122

YADA extends [INT-ARCHI] to add the capability for an IPv4 host to recognize an special IP-in-IP format as an inter-realm IPv4 packet and process it accordingly. It also adds a new DNS double-A record format that denotes a YADA address.

4. Extending RFC 4291

YATT extends [IPv6-ADDRESSING] to add the capability for an IPv4 host to recognize an special IPv6 format as an YATT address embedding a YADA address and process it accordingly. It also automatically derives the ownership of the YATT prefix associated to a owned YADA address.

5. YADA

YADA assigns IPv4 prefixes to a multinternet shaft; those prefixes must be the same across all the realms that are interconnected by the shaft. Multiple prefixes can be assigned to the shaft for unicast and multicast communications, and each realm needs at least one unicast address in the shaft called its realm address. A YADA address is formed by the tuple (realm address, IPv4 address) and is advertised in DNS as a new double-A record. Because the YADA prefixes are assigned for YADA, a packet that has either source or

destination IPV4 address derived from a shaft prefix is a YADA packet.

YADA leverages IP-in-IP encapsulation to tunnel packets across the shaft for inter-realm communications, while the IPv4 operations within a realm are unaffected. The YADA address is found by using both inner and outer header and combining that information. The pair of IP headers is seen by a YADA stack as a single larger header though a non-YADA forwarder only needs the outer header and plain IPv4 operations to forward.

YADA requires a change in the stack in the YADA endpoints that communicate with other realms to support the YADA encapsulation. YADA also provides a bump in the stack method for legacy applications. YADA also requires a change for the routers that serve the shaft. Those routers play a special role for packets that are delivered from the shaft to the destination realm, and for ICMP errors across realms. All other IPv4 nodes in the realm continue to operate as before.

Routers serving the shaft advertise the shaft prefix(es) in their respective realms, and their realm addresses within the shaft, as host routes for unicast and anycast addresses. A stack that resolve a DNS name with a double-A record indicating a different realm generates an IP-in-IP packet, with the outer header indicating the source and destination realms, and the inner header indicating the source and destination IPv4 addresses within the respective realms, as shown in Figure 2. The packet is forwarded down the shaft as is, using the normal longest match or multicast operation.

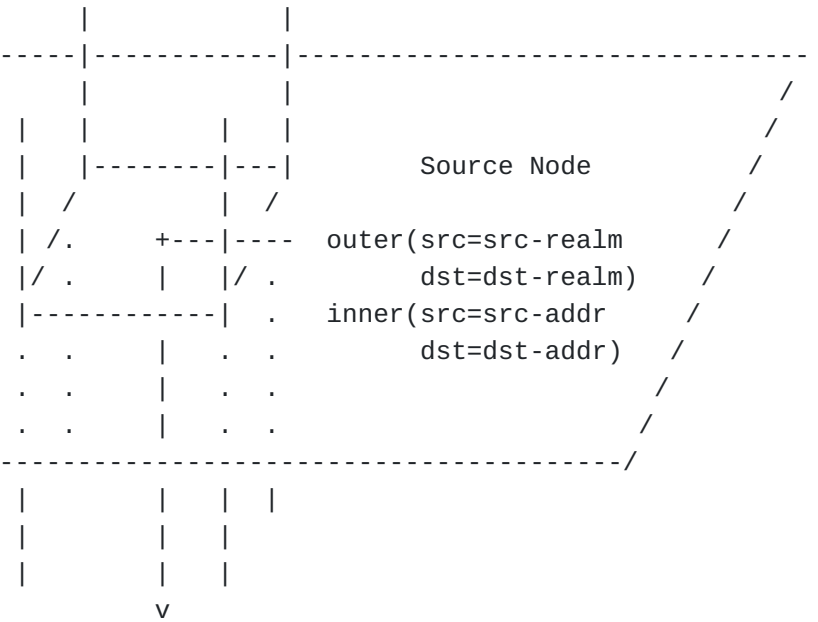

Figure 2: Packets Entering the shaft

The packet destination is an address is the shaft and it is attracted by a router that serves the shaft and advertises its prefixes in the source realm. Based on longest match, the router forwards the packet inside the shaft following the host route to a router that serves the destination realm. That router swaps the destination address in the inner and outer headers and forwards within its realm to the final destination, as shown in Figure 3. In normal conditions, the stack of the destination node recognizes the YADA format and replies accordingly.

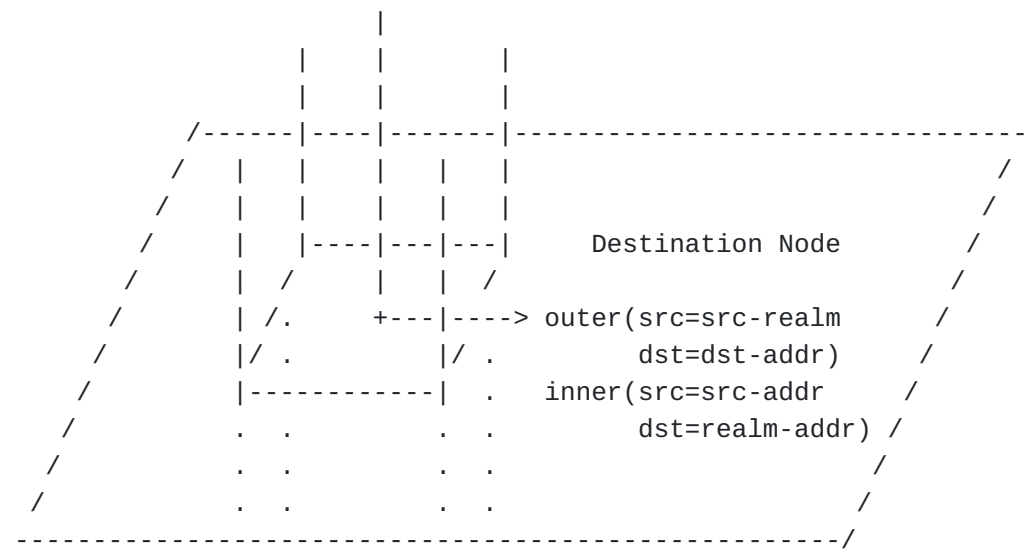

Figure 3: Packets Outgoing the shaft

In case of an error down the path or at the destination, if an ICMP message is generated by a node that is not YADA-aware, the message reaches the router that serves the shaft in the source realm. If the inner header is present in the ICMP payload, then the Router extracts it and forwards to the packet source. If the destination stack does not support YADA and decapsulates, the message reaches the router that serves the destination realm which logs and drops. based on the log, the node may be updated, or the DNS records may be fixed to avoid pointing on a node that does not support YADA.

YADA requires the assignment of a second IPv4 prefix, this time for a internal NATing operation. A bump-in-the-stack intercepts the DNS lookups, and when the response yields a double-A record with a foreign realm, the record is augmented with an IPv4 address taken from a local NAT pool. When the stack sends a packet to that particular address, the bump-in-the-stack translates to the YADA format, using the information in the double-A record for the

destination, and the local realm as source realm. The other way around, if a packet arrives with a YADA format but the stack does not support it, the bump-in-the-stack allocates an address from the pool, and NATs to IPv4 using that address as source.

YADA was initially published as USPTO 7,356,031, filed in February 2002.

6. YATT

A second mechanism called YATT translates the YADA format into flat IPv6.

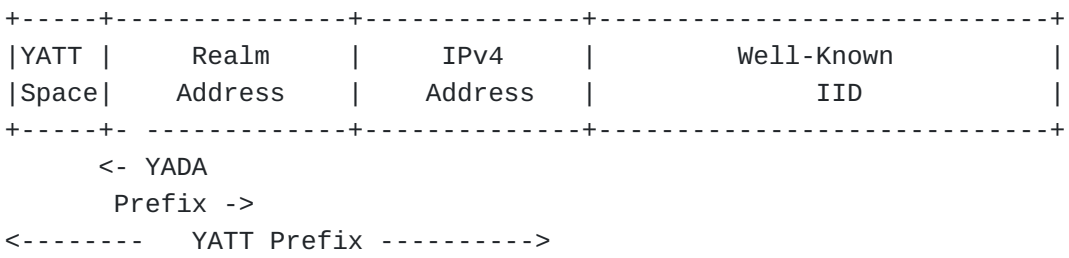

Figure 4: YATT format

For unicast addresses, YATT forms an IPv6 prefix by collating an well-known assigned short prefix called the YATT space, the realm address, and the host IPv4 address (locally significant within the realm). The resulting IPv6 prefix is automatically owned by the host that owns the IPv4 address in the realm.

Depending on assignment, the leftmost piece realm prefix may be truncated if it is well-known, to allow the YATT space and the realm address to fit in a 32-bit DWORD. This way, the YATT prefix can be a full /64 prefix that is entirely owned by the host that owns the associated YADA address.

YATT then forms an IPv6 address for that host by collating a well-known Interface ID, so there's a one-to-one relationship.

The formats can not be strictly provided till the YATT space and YADA prefix are assigned. But say that the YATT Space is F000::/6 and the YADA prefix is 240.0.0.0/6. In that case the values perfectly overlap and the YATT format becomes as follows:

Figure 5: YATT format using 240.0.0.0/6

In that case, the NAT operation is a plain insertion. Depending on the assignment, it might be that the Realm address must be placed in full after YATT space. In that case, the length of the YATT prefix will be more than 64 bits.

If the network supports IPv6 to the shaft, it makes sense for the YADA host or the bump-in-the-stack to generate the packets in the YATT form natively. The shaft router must then attract the shaft YADA realm Prefix in both IPv4 and YATT forms.

If the network is IPv4 only, the packets are still generated using IP in IP, and the YATT NAT operation may happen at the router that delivers the packet in the destination realm, if it is v6-only, or in the destination host, if its stack is v6-only.

YATT was initially published as USPTO 7,764,686, filed in December 2002.

7. Applicability

YADA And YATT enable communication between YADA-enabled IPv4 nodes across realms, and with IPv6 nodes that own a YADA address from which a YATT address can be derived. Communication from a legacy IPv4 application/stack that is not YADA-enabled, or to an IPv6 address that is not a YATT address, is not provided.

Since the YATT translation is stateless, the header translation can happen anywhere in the network, e.g., as a bump in the stack at either end, or within the network, e.g., at the routers that serve the realms on the shaft. The shaft itself is expected to be dual stack to forward packets in their native form, either v4 or v6.

For a legacy IPv4 node to communicate with YADA-enabled IPv4 node in another realm, a NAT operation similar to NAT46 [NAT-DEPLOY], but between IPv4 and YADA addresses, is required. The same would be required to allow an IPv4-only YADA node to communicate with

8. Backwards Compatibility

YADA operation does not affect the intra-realm communication. The only affected stacks are the endpoints that communicate between realms leveraging YADA.

9. Security Considerations

10. IANA Considerations

This document requires the creation of a registry for IPv4 YADA realm prefixes, and the assignment of at least one YADA realm prefix.

This document requires the creation of a registry for IPv4 YADA NAT prefixes, and the assignment of at least one YADA NAT prefix.

11. Acknowledgments

12. References

12.1. Normative References

[IPv4] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/info/rfc791>.

[INT-ARCHI] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

[IPv6-ADDRESSING] 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>.

[IPv6] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

12.2. Informative References

[NAT-DEPLOY] Palet Martinez, J., "Additional Deployment Guidelines for NAT64/464XLAT in Operator and Enterprise Networks", RFC 8683, DOI 10.17487/RFC8683, November 2019, <https://www.rfc-editor.org/info/rfc8683>.

Author's Address

Pascal Thubert (editor)
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
06254 Mougins - Sophia Antipolis
France

Phone: +33 497 23 26 34
Email: pthubert@cisco.com