Namespace Resolution in Future Internet Architectures
draft-wang-fia-namespace-01

Abstract

This document presents the architecture and implementation of an open
and flexible namespace resolution mechanism to be used with Future
Internet Architectures. This resolution mechanism allows the
resolution of different network entities and can be adapted to the
needs of network, application and service providers alike.

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 http://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 April 7, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the
document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
1. Introduction

A number of future network architectures have been proposed to address existing problems of the current Internet, denoted as future Internet Architectures (FIAs). The naming and addressing of network entities including content, users, devices, services etc. are an common to all FIAs. Thus, no matter toward which FIA the Internet will evolve, there will be a need an open namespace management and
Resolution system to provide flexible definition of network entities, optimal name resolution and management, extra mobility consideration and improvement of security issues.

Such a system will:

- Allow multiple namespaces to co-exist;
- Enable dynamic name resolution among multiple namespaces through policies;
- And facilitate interpolation between networked systems with different namespaces.

This draft presents the architecture and implementation of such an open namespace resolution system.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Definitions and Abbreviations

3.1. Definitions

This document uses the following definitions, that are mostly inspired from [RFC5052], [RFC6363].

Namespace:

Resolution:

To Be Completed

3.2. Abbreviations

This document uses the following abbreviations:

ASP: Application Service Provider
CCN: Content-Centric Network
CS: Content Store
DPI: Deep Packet Inspection
4. Background

In the current Internet, DNS servers take charge of mapping URL (name) to the actual network address of a target resource before initiating the communication. This name resolution policy (i.e., from domain name to IP address) is usually static.

In a Named Data Network (NDN) [ndn], data is requested and located by its name. A NDN uses a recursive mechanisms to resolve and forward from one namespace (named objectives) to another namespace.
The Mobility First architecture [mobility] uses a Globally Unique Identifier (GUID) and a network address (NA) as the namespace to identify network entities. The length of the GUID is fixed to 160 bits and the name resolution is also fixed and consists of mapping the GUID to the actual network address (NA). A switch will forward the packet based on the NA.

The Independent Virtual Id Routing [virtualID] or VIRO decouples naming from routing via Virtual ID (VID) and allows different identifiers such as IPv4/IPv6, DNS names or some other names, to co-exist within the namespace. However, all of these namespaces will be mapped to the VID of the particular network entity. It only allows one level name resolution, e.g., from one name to VID, and flexible name resolution driven by policies among multiple namespaces is not accommodated.

The Interface based Naming System [minami] or IFNS also allows multiple namespaces to co-exist in the architecture. A Multi-Name Service System (MNSS) will have different Name Service Components (NSC). IFNS is just one of NSC in this Multi-Name Service System. Each NSC has its own name resolution strategy and cannot map from one NSC to another one.

What can be seen from the above it that each FIA has defined it’s own mechanisms. We propose that namespace resolution not only should have universality to adapt to general usage scenarios, but also should be flexible enough to meet some new requirements as the Internet evolves. A namespace management system should only be defined by the properties of network entities. Furthermore, a name resolution strategy should also provide name resolution from any source namespace to any destination namespace.

5. Requirements

5.1. Generic Namespace Management System

As was seen above, for currently existing network architectures, the namespace and resolution policy is fixed. As such a fixed name resolution policy cannot facilitate the deployment of new services. For example, a network service provider may do Deep Packet Inspection (DPI) for some flows. To enable this, a name resolution policy could specify which certain flows satisfied the pre-set conditions to be resolved to a middlebox for DPI while other flows will be resolved to next hop router for forwarding.

In the more and more diverse Internet, a namespace management system should provide unified APIs to define namespaces and resolution policies flexibly and be applicable to network, services and...
application providers alike. Thus many types of network architectures can be supported on the same physical network. In addition, security features are necessary to ensure that a provider can only access its own namespace and the namespace can only be accessed by itself. Only when a source namespace allows resolution to a target namespace and, at the same time, the target namespace allows resolution from the source namespace, then the resolution between the source and the target namespaces can occur. It implies that resolving from namespace A to namespace B, then from namespace B to namespace C may not be equivalent to resolving from namespace A to namespace C. The mechanism described in the rest of this document allows this.

5.2. Definable Routing

Controlling the routing and forwarding procedure based on some QoS and security consideration is a requirement of both service providers (to keep the traffic within their management domain) and of application providers (to control the quality of service). Hence, in a namespace management system, flexible name resolution policy should facilitate the implementation of any particular routing scheme.

5.3. Decoupling Name Resolution from the Application Service Provider

In existing solutions, ASPs usually handle their own name resolution. For example, in Skype, name resolution is conducted by globally synchronized super nodes. But to maintain a namespace management system by application results in infrastructure costs when deploying application services. In consequence, and because the resolution could be done on a remote network, the resolution delay may be higher than when done more locally by the network service provider.

With our generic name resolution system, the resolution process can be moved from the application layer to the network layer with the authentication of ASPs. To achieve this as mentioned above, there needs to be appropriate security for the namespace management system to ensure that an ASP can only define and access its own namespace, that there exist a trust agreement between the ASP and the network service provider and that the resolution policy is mutually agreed by both source namespace owner and target namespace owner.

5.4. Compatibility Issues

In order to support long-term evolution, different networks/protocols must be deployed in a unified framework. Thus, a generic namespace management system will provide a reasonable way to realize both backward and forward compatibility. On the application layer, because of the decoupling of resolution service from ASPs, the
relationship among applications can be defined more flexibly with more interoperability.

5.5. Security Requirements

The following two main features are added to the namespace solution to address the security concerns related to address resolution.

1) The dynamic security strategy is self-contained in the namespace definition. The service provider will be able to deploy different security strategies for specific services or applications in configuring its namespace. The security strategy can be flexibly changed by modifying the namespace.

2) The physical address is hidden. The namespace and name resolution rule are both defined by the service provider, and this whole process is hidden to other traffic. Furthermore, the control plane will do the authentication verification to guarantee the namespace cannot be left without proper authority.

6. Components of a Multi-namespace Management System

To provide a namespace management system with the aforementioned features, we define three main components:

1) A Namespace Management Component that keeps namespace records. Different service providers can flexibly define their own namespaces based on different objectives and requirements. For instance, a network service provider can define a particular network by deploying the network entity namespace. In terms of application service providers, they can use the namespaces to describe the specific forwarding and security strategy of their application.

2) A Resolution Engine (forwarding plane) which processes filter and action setting for namespaces and entities respectively. In order to get the optimal name resolution and routing scheme, a service provider should define a series of appropriate forwarding policies among particular namespaces. Another other type of policy is provided by filters to define the access control for particular namespace. It is allowed that an instruction set contains multiple actions.

A Control Plane which contains the access control module to guarantee that the configuration process is secure and reliable. It provides configuration APIs including: namespace management (register, update, delete), policy setting (filter and action) and system control. The control plane is provided by middleware,
which makes it possible that all entities in the Internet (e.g.,
network devices, user, service, data object) can manage their
namespaces when give appropriate authority.

7. System Architecture

7.1. Control Plane

The Control Plane is a middleware between applications/services and
the physical infrastructure. The configuration messages will be
verified by the access control module in the control plane. This
ensures there is valid authority for deploying the configuration for
each namespace. Verified messages will be transmitted to the
particular switches maintaining the target namespaces.

7.2. Switch

7.2.1. Namespace Management Module

The Namespace Management Module provides two main functionalities:
(i) it maintains the records of all namespaces and (ii) it interprets
and executes policies (filters and actions) for namespace and
entities (Resolution Engine).

7.2.2. Namespace Access Control Module

Any entity (e.g., network devices, user, service, data object) can
create, update, or delete namespaces and resolution policies by
configuration messages. These messages contain the namespace
settings (or changes), the policy setting (or changes) and the
process flag (create, update or delete). A configuration message is
firstly sent to the control plane. Then the control plane processes
the message to find the target switches. Finally, the control plane
pushes the configuration message with its signature to switches. In
the switch, the Access Control Module will verify whether the
configuration message is from the controller with authority to manage
this particular namespace.

7.2.3. Session Control Module at the edge switch

The Session Control Module is designed for managing the sessions and
providing QoS in the edge switch. A session describes a temporary
resolution, which avoids searching and interpreting among namespaces
repeatedly. It highly improves the efficiency of data routing.
Generally, every sessionxxx activity will be managed by its own life-
cycle or providerxxx instruction. Sessions can be managed
individually to achieve some QoS goals. The xxxirst packetxxx
inference of OpenFlow and SDN can be used to trigger a new session.
For OpenFlow, the first packet of a stream (after the DNS resolution) may trigger a control action. The first packet exchange therefore takes on a dual role: data exchange and control trigger. Having an explicit initial exchange that informs the network of the data transfer provides a clean mechanism for separating the data delivery from the control plane.

7.2.4. Forwarding Plane

The Forwarding Plane handles packet forwarding based on the resolution engine integrated in the namespace management module. It forwards packets to particular interfaces according to records in the interface mapping table.

8. Implementation

The recommended namespace format is presented in the figure below. It can use the unity paradigm or a self-defined namespace format.

```
+-----------------------------------------------------+
<table>
<thead>
<tr>
<th>Namespace:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy:</td>
</tr>
<tr>
<td>------------+------------------------------------</td>
</tr>
<tr>
<td>Tag:</td>
</tr>
<tr>
<td>------------+------------------------------------</td>
</tr>
<tr>
<td>Entity Name</td>
</tr>
<tr>
<td>----------------+---------+----------+---------+-----</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>----------------+---------+----------+---------+-----</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>----------------+---------+----------+---------+-----</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
+-----------------------------------------------------+
```

Figure 1: Namespace format

The different elements of the format are defined as follows:

Policy: the access control filters of the namespace compose the policy. Generally, regular expressions are supported. But policy could also be programmed by script language to describe complicated filters and actions like moving to another namespace.

Tag: the tag of a namespace indicates its characteristics. For example, if this is a service namespace, the tag could be the servicexxx name.
Entity Name: the name of the specific entity.

Value: the value is the type of address or of any other network identity. For example, the IP address, the MAC address or some new identity that is defined by the provider.

Action: The action field indicates how the entities will be matched and what will be done after the matching. A particular namespace could have multiple actions to execute. For instance, SENDTO VALUE means forward to the network address recorded in the value field. GOTO means the packet will be send to another namespace. Furthermore, some filter can be added to limit the action.

State: The state of this entity. For example state could show that whether a device is online or offline.

9. Example of Multiple Namespaces

9.1. Deploy ICN instances on Multiple-namespace Network

Existing ICNs all have their own namespaces and mechanisms for resolution. Hence, we have deployed an ICN instance on our system.

For instance, in NDN [jacobson], Forwarding Information Base (FIB), Content Store (CS), and Pending Interest Table (PIT) can be implemented as three namespaces in the proposed system where FIB is a namespace of destinations for Interests, CS is a namespace of cached content, and PIT is a namespace of sources for unsatisfied Interests.

The Forwarding Engine described in CCN [snamp] can also be implemented in our system. The logic of original CCN Forwarding Engine can be defined by policies of namespaces. For instance, when an Interest packet comes in, the our forwarding engine will check if there is match in the namespace of the CS. If not, the forwarding engine will check the namespace of PIT (according to the action defined by the policy of namespace of CS). If no matching entity can be found in PIT again, the forwarding engine will check the namespace of FIB (according to the action defined by policy of namespace of PIT). We note that "prefix longest matching" can be defined by the filter of the policy of the entity in the namespace. Finally, according to the policy (action of forwarding and ID of target interface), the switch forwards this Interest packet to the corresponding interface specified by the matched entity.

Besides NDN, other ICNs can be implemented by the proposed system. For example, PSIRP [psirp], is also supported by multiple namespaces. In PSIRP, users (subscribers/publishers) subscribe/publish content to
rendezvous nodes. Rendezvous nodes actually can be implemented on switches with the namespaces of scope and content. In PSIRP, forwarding solely depends on information identifiers, i.e. RIds and SIds, thus MPLS-like label switching protocols are used. These identifiers can be processed by the filters defined by policies of corresponding namespaces and forwarding engine forwards packet by these identifiers with policies of namespaces.

As a generic solution, our system supports implementation different ICNs, which makes it possible to deploy all these different ICN instances on the same infrastructure. An extra benefit is the possibility of fusioning ICNs. For instance, we can deploy two ICN instances, CCN and PSIRP, on the same infrastructure. In the switch, we can set some policies to define the translation between the namespaces of CCN and PSIRP. A CCN client requests content by sending an Interest packet. When it fails to find a matched entry in namespace of CS (defined by CCN), we may let it search in namespace of published content (defined by PSIRP), by the policies (action of xxxotoxxx and packet modifying to adapt PSIRP) of namespaces in CS. Thus, a publisher of PSIRP may provide the content to a CCN client.

9.2. Supporting Manifests

The manifest is a data object containing meta-data about another object. Therefore, it can be given a name in CCN and the CCN transaction could start in a corresponding manner by requesting this object. There are proposals and discussion to add manifests to CCN. These manifests usually point to hashes of a series of data objects in order to speed up the forwarding and security checks. Manifests could also point to other names for the same object (like the name of an object pinned at a specific location). They could be extended to support other useful network meta-data.

In a generalized multiple namespace management system, a user can issue a request for a manifest data object which consists of a set of data objects from different ASPs or different ICN instances. The name resolution for the data objects contained in the manifest can be done individually in different namespaces according to the name resolution policy specified in the meta data of the manifest. This will allow users to compose and fetch data/service from different ASPs and/or ICN instances in an easy way.

10. The procedure of communication over Heterogeneous Wireless Network

In the current heterogeneous wireless network, communication between two entities which employ different network protocols needs to be bridged by IP. That means the protocol translation is performed twice, which leads to extra transmission overhead (due to redundant
TCP/IP packet header) and operations overhead (brought by format conversion between Ethernet frame in the link layer and data packet in the network layer). To solve the problem mentioned above, we designed a new namespace management system to make the communication more flexible and efficient.

```
+-----------+                       +-----------+
|User Device|                       |User Device|
+-----------+                       +-----------+
     \                       /    
      \ Wi-Fi                ZigBee /  
       \ /                    \ //  
       \                      /|  
       \                       |/  
       \                      |  
       \                      |  
       \                      |  
       \                      |  
       \                      |  
       \                      |  
       \                      |  
       \                      |  
       AP \                  / AP  
       \                    /  
       \                   /   
       \                  /    
       \                /     
       \               /      
       \             /       
       \           /        
       \          /         
       \        /          
       \      /            
       \     /              
       \   /                
       \  /                 
       \ /                  
       \                    
       BlueTooth            AP --- +-----------+
                      |
                      |                   
                      |                   
                      +-----------+
```

Figure 2: Demo Scenario

In the sensor network scenario shown in the Figure 1, three different protocols, i.e., Wi-Fi, ZigBee, and Bluetooth, are adopted by different devices. Instead of encapsulating the original packet with different protocols into a TCP or UDP packet and routing packets based on IP, Our proposed namespace management system can be used to support protocol and packet translation by extending the Ethernet frame. The format of the designed new frame is shown in the figure 2.
In this frame, MAC Header field records the source and destination MAC addresses for link layer transmission. SrcName, DstName and Protocol Type in the Data field are used for routing purposes where the name resolution will be used by our proposed multiple namespace management system.

Upon receiving the frame, the router will do name resolution based on the SrcName and DstName to find out where this frame should be forwarded to and will change the DstMACAdd to the Mac address of a physical device’s specific port for next hop. When the target device receives the frame, it will translate the frame to the target format according to the protocol stored in the Protocol Type property. Figure 3 presents an example of the translation between Bluetooth and WiFi.

With the application of the new namespace management system, the whole procedure is expected to be accomplished at the link layer.
without switching to the network layer. Figure 4 indicates the results of the comparison of the throughput rates between the solution using multiple namespace management system and UDP under different situations (Throughput: Packets/sec, Packet Size: Bytes).

<table>
<thead>
<tr>
<th></th>
<th>One Threads</th>
<th>Two Threads</th>
<th>Three Threads</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Packet Size</strong></td>
<td>10 100 1000</td>
<td>10 100 1000</td>
<td>10 100 1000</td>
</tr>
<tr>
<td><strong>Our System</strong></td>
<td>50120 47875 38533 94231 90652 72153</td>
<td>136246 126053 98152</td>
<td></td>
</tr>
<tr>
<td><strong>UDP</strong></td>
<td>36983 36720 33023 63034 60992 60238 53221 52965 53393</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Optimizing Percentage</strong></td>
<td>35% 30% 16% 49% 48% 19% 156% 137% 83%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5: Comparison of the throughput capacity between our system and UDP

This demo shows that multiple namespace management system can enable flexible protocol translation by self-defined name resolution.

11. Acknowledgements

The authors would like to acknowledge Dr. Marie-Jose Montpetit for supporting editing of this draft.

12. IANA Considerations

This document includes no request to IANA at this point.

13. Security Considerations

To be defined when appropriate, see RFC 3552 [RFC3552].

14. References

14.1. Normative References
14.2. Informative References


Internet-Draft

Namespace Resolution in Future Internet Archit

October 2015

Afanasyev, A., "SNAMP: Secure Namespace Mapping to Scale
NDN Forwarding.", Proceedings of IEEE Global Internet
Symposium., 2015.

Lu, G., Jain, S., Chen, S., and et. al, "Virtual id
routing: a scalable routing framework with support for
mobility and routing efficiency.", Proceedings of the 3rd
International ACM Workshop on Mobility in the Evolving
Internet Architecture., 2008.

Authors’ Addresses

Jianping Wang
City University of Hong Kong
Email: "jianwang@cityu.edu.hk

Shusheng Liu
Huawei
Email: liushucheng@huawei.com

Cedric Westphal
Huawei
Email: Cedric.Westphal@huawei.com