UNIVERSITEIT VAN AMSTERDAM

System and Network Engineering

Feasibility of ILA as Network Virtualization Overlay in multi-tenant, multi-domain Cloud

Research Project #2

Author:
T.T.N. Marks BSc. (11336137)

Supervisors:
Dr. Paola Grosso & Łukasz Makowski MSc.

July 27, 2017

Abstract

During this research Identifier Locator Addressing (ILA) is studied as a multi-domain Network virtualization Overlay (NVO) solution. Two ILA addressing schemes are proposed that can be used to achieve such overlays. In addition, a suitable control plane for such an environment is analysed. By hierarchically dividing address blocks of identifier space it is possible to allow a thousand domains to work in one ILA overlay with the possibility of 260 thousand subdivisions or projects. Multi-Protocol Border Gateway Protocol (MP-BGP) is the best suitable control plane to control the distribution of ILA mappings information in a multi-domain environment because it offers native filtering capabilities that can be used together with firewalls to provide access controls.

Contents

1 Introduction **3**
1.1 Research Questions 3

2 Related Work **4**
2.1 Network Virtualisation Overlays 4
2.2 Identifier/Locator Split 5

3 Identifier Locator Addressing **7**
3.1 Control plane 9

4 NVO Requirements **10**
4.1 NVE Requirements 10
4.2 NVA control plane requirements 12

5 ILA configuration **12**
5.1 Test setup 13
5.2 General addressing setup 14
5.3 Single SIR scenario 15
5.4 Multiple SIR scenario 16
5.5 Other considerations 16

6 ILA control plane **17**
6.1 Existing control planes 17
6.2 ILA control plane fit 18

7 Discussion **18**

8 Conclusion **19**

9 Future Work **20**

Acronyms **21**

References **23**

Acknowledgements

I would like to thank my supervisors Paola and Łukasz for all of their input and suggestions during this research project. I enjoyed our discussions and it really has made this project more interesting as a result. Additionally, I want to thank those who have supported me during throughout project, especially my roommate Michelle and my parents.

1 Introduction

Containers are increasingly used to drive computing workloads to the cloud by increasing flexibility and reducing costs. More and more infrastructure tenants have come to expect to have their own "private" networks to come with their virtual compute infrastructure. These "private" networks are often realised using NVOs [12]. ILA [8] is an NVO developed at Facebook that can provide Layer 3 connectivity to tenants. Contrary to common overlay networks that use some form of encapsulation ILA uses Internet Protocol (IP) version 6 (IPv6) Network Address Translation (NAT) together with a concept called IP Identifier/Locator split. ILA provides each process, container or Virtual Machine (VM) running on host machines with their own IPv6 address that is mobile [8]. Furthermore, on-the-wire addresses are regular IPv6, so no changes in network hardware are needed to perform routing of overlay traffic.

ILA's design is based on intra-domain networking only. The author would like to investigate ILA as a possible candidate for a multi-tenant, multi-domain environment. In order to do this, the following research goals were defined. The first is to get a thorough understanding of ILA internals, concepts and operation to be able to identify its pros and cons, especially in regards to multi-tenancy. The second goal is to assess the applicability of ILA for use in container networking solutions that consist of multiple domains that are operated by different providers. In particular, looking at the specific use-case of creating a shared container infrastructure with container mobility. As such the following research questions have been defined.

1.1 Research Questions

Is it feasible to use ILA in as a Network Virtualization Overlay for a multi-tenant, multi-domain Cloud?

To answer this main research question the following sub questions have been defined:

1. What are the requirements for Network Virtualization Overlay in a multi-tenant, multi-domain environment?
2. What are possible ILA configurations that satisfy the multi-tenant, multi-domain environment requirements?
3. What would be a suitable control plane for ILA when used as an NVO in a multi-tenant, multi-domain environment?

2 Related Work

2.1 Network Virtualisation Overlays

The increasing scale of datacenter deployments led to the development of various virtualization techniques for networks. These NVOs allow for the creation of virtual network topologies that are subsets of the underlying physical network. These overlays consist of virtual links that connect virtual nodes (VMs, containers or other objects) together while isolating them from any other traffic. A graphical representation can be found in Figure 1. As a result, most cloud infrastructures run some form of NVO to support the creation of virtual tenant networks that are isolated from each other while sharing a physical underlay network.

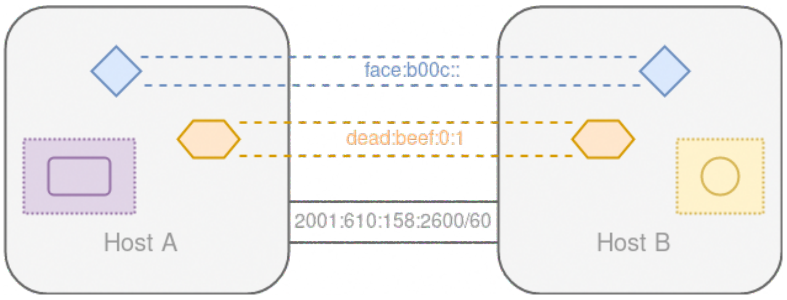

Figure 1: Virtual nodes communicating over overlay subnets that share a physical connection while being isolated from eachother.

There are many solutions for creating NVOs using a multitude of different technologies. The solution chosen by the provider should not matter to the tenants that make use of the overlays. For tenants, it is only important to know what kind of connectivity is provided by the overlay. Just as regular networks overlays consist of (virtual) links. The links can either be Layer 2 (Ethernet) or Layer 3 (IP version 4 (IPv4)/IPv6) depending on where the underlying technology chooses to virtualize the network.

An NVO consists of an underlay physical network and a number of hosts that are connected to this network. Each of these hosts can fulfil one or two functions that create the overlay. These can be Network virtualization Edge (NVE) and/or Network virtualization Authority (NVA) [4]. An NVE is a host or other endpoint that implements the network virtualisation functions such as translation or de-/encapsulation of packets that belong to a particular overlay [10]. These NVEs thus enable one or more overlay data planes for the tenant systems to use. NVEs are connected through an underlay network, which is the physical network infrastructure that carries all traffic. The NVA is a second entity that provides control plane information of reachability to the different NVEs that participate in overlay networks. Both the NVEs and the underlay network are usually located in one or more Data Centers (DCs). The overlays created can span several DCs if needed.

Many older techniques for virtualising networks were actually called Virtual Private Network (VPN) solutions. Perhaps the most well-known and easiest solution to create a Layer 2 overlay would be to use Virtual Local Area Networks (VLANs) from the 802.1Q standard [17]. However, these are quite limited as they only allow for 4094 Local Area Networks (LANs). To be able to allow for more virtual networks that span over Layer 3 networks, Multi-Protocol Label Switching (MPLS) based VPNs were often used. Virtual Private LAN Service (VPLS) [9] networks can create Layer 2 "overlays" by assigning specific labels for each VPN path. Different control planes could be used to distribute label mappings to participating NVEs. There also exists a Layer 3 MPLS based VPN that offers IP connectivity to tenants [14]. Both VPLS and Layer 3 MPLS solutions can use a control plane based on Border Gateway Protocol (BGP) [13] to exchange labels between NVEs.

One protocol that is often used nowadays to create overlays is Virtual Extensible LAN (VXLAN) [11]. It encapsulates Ethernet packets in a VXLAN header which is then transported using User Datagram Protocol (UDP) over the network. As such VXLAN offers Layer 2 connectivity within the overlay. The VXLAN header has a 24-bit Virtual Network ID (VNID) field that allows for 16 million virtual networks. VXLAN can either be used together with MP-BGP [3] to disseminate Layer 2 addressing information within the overlay. This is a flavour of Ethernet VPN (EVPN) [15]. Or the built-in Broadcast, Unknown unicast, and Multicast (BUM) resolution can be used, where addressing information is flooded between NVEs. The last option would be to store this information in some kind of central controller/database.

In order to be able to use any of these protocols and related technology for container networking, additional components are needed. Most often containers are deployed on host machines using some kind of orchestration software that manages placement of containers within a cluster of hosts. These orchestration tools use network plugins that manage NVE and NVA functions for the containers running on that host. Several existing plugins offer different kinds of technologies. For example, flannel [1] uses a distributed Key/Value store as NVA, together with Linux bridges and VXLAN to create an overlay. Another network plugin called Project Calico [2] uses BGP to create Layer 3 overlays and Weave [3] uses the previously discussed VXLAN encapsulation.

2.2 Identifier/Locator Split

ILA is an implementation of a general concept that is known as the Identifier/Locator split. The defining characteristic of these systems is IP addresses can have different semantics compared to the current system. Nowadays, IP addresses are used for two

[1]`https://github.com/coreos/flannel`
[2]`https://github.com/projectcalico/cni-plugin`
[3]`https://github.com/weaveworks/weave`

purposes; to identify endpoints, as well as for routing. In the current system, the address of a machine is bound to its current location. As a result, whenever a device changes its location the address has to be changed as well. This creates a problem where an increasing number of connected devices move not only in one network but also between networks boundaries. Identifier/Locator split systems aim to solve this problem by splitting these functions into separate namespaces. To illustrate this let us use the following example. A student has connected his mobile phone to his home Wireless Local Area Network (WLAN) router. While listening to a music streaming service he/she walks outside the range of the WLAN access point. The mobile phone switches to the mobile operators network in order to maintain network connectivity. The network location of the phone has now changed and results in a new IP address. Because of the new IP address, all of the previously existing connections will be terminated as they were bound to the old IP address. Thus the music stream is interrupted and has to be re-established.

If the identifying function of an IP address were to be separate from the locating function (routing) it would be possible for the identity address to remain the same whenever the location changes. This would enable application connections to keep functioning whenever a location change occurs. In our previous example, the music stream would just continue without interruption. Table 1 provides an overview of how the IP address would be updated semantically. This separation of identity and location of IP address has been brought up numerous times along the history of the internet. The first written occurrence of this idea in a formal document was back in 1977 in the first Internet Experiment Note (IEN) [18]. A more formal discussion of these ideas was written down in Request for Comments (RFC) 1498 in 1993 [20]. The idea resurfaces again during Internet Architecture Board (IAB) workshop on the Internet Network Layer architecture hosted by SURFnet in 1999 [21]. Despite the idea resurfacing multiple times, it has never been implemented on a large scale, even though both IPv4 and IPv6 do allow for incremental changes or updates to the protocol. The IPv4 Options or IPv6 Extension Headers are designed to carry additional information to enable new functionality. The recurring problem that stopped the widespread use of extensions is that both endpoints need to support the extensions. It has become difficult to get enough parties to invest in a certain extension, to such an extent that there is a reasonable chance of the host that is connected to also supports it. This problem is even more pertinent when core protocols themselves are replaced. A good example of this is the terribly slow adoption rate of IPv6 which only just recently started gaining momentum in the wider internet community. This problem is often called network ossification [16].

Table 1: IP address semantics

Protocol Level	**Current identifier**	**Identifier/Locator split**
Application	FQDN / IP Address	Identifier
Transport	IP Address (+port)	Locator (+port)
Network	IP Address	Locator
Interface	IP Address	Locator

Luckily the increased adoption of IPv6 on the internet enables new possibilities to experiment with Identifier/Locator options. Because IPv6 address space is so large it allows for experimentation, which in IPv4 is no longer possible. A result of this experimentation was Identifier-Locator Network Protocol (ILNP) where the Locator/Identifier split was done entirely in the IPv6 address field [2]. This is done by (ab)using the second half of the address space to encode identifiers, and by using the first half of the address space to do routing with Locators. However, as ILNP is architected as a global system it will be hard to use for the same reason as protocols extensions, as the other party also needs to have enabled it.

3 Identifier Locator Addressing

Even though the ideas in ILNP are very similar to ILA they differ in several important aspects. ILA is IPv6 only and cannot run on IPv4 underlay networks. More importantly, it is focused primarily on introducing the Identifier/Locator split within one administrative domain. This makes it easier and more likely that ILA is actually deployed and used. Simply because network domains are often under the control of one entity that can enforce said technology, whereas in ILNP you are dependent on other network operators to implement as well before it can be used. To illustrate the general concepts and how ILA overlays operate an example is given below.

ILA's primary goal is giving each container a location independent IPv6 address. It does this by putting a fixed placeholder in the first 64-bits of the IPv6 address that is normally used for routing. This fixed placeholder also indicates that the address is part of a specific overlay. ILA hosts translate the fixed placeholder to current locations of the nodes and back whenever packets travel over the network. As a result of this back and forth translation, there are two kinds of addresses. Applications address ILA nodes on sockets using addresses that contain the fixed placeholder that is called the Standard Identifier Representation (SIR) prefix [8]. The remainder of the IPv6 address is the identifier of the specific node. The full address with the SIR prefix and an identifier is called a SIR address.

Before the packets can be sent through the underlay network ILA hosts have to substitute the SIR prefix to an actual routable IPv6 address of the underlay network. This is called a Locator, which is the /64 prefix of the host on which a specific ILA node currently resides. Having replaced the SIR prefix with a Locator creates an ILA address that can be routed just like regular IPv6 traffic in the underlay network based on the /64 prefix. An overview of the translation between these address types can be seen in Figure 2.

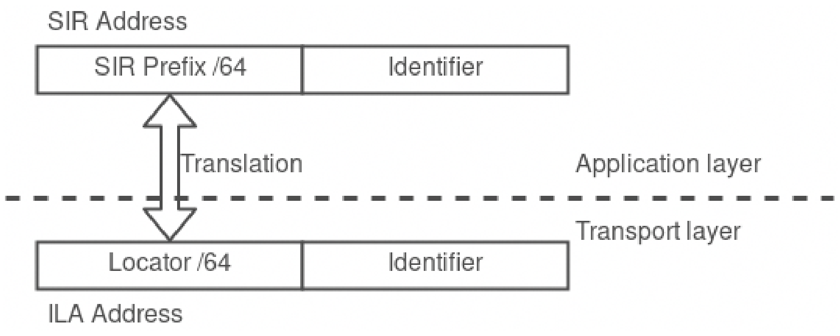

Figure 2: ILA translation between SIR address and ILA address

To illustrate this idea one example is depicted in Figure 3. It shows that both loopback interfaces of host A and host B have an identifier in the SIR prefix subnet of *dead:beef:0:1*. In Step 1, A opens a socket to host B using the SIR address of the respective loopback on B. Whenever a packet is sent from A to B at Step 2 the SIR address of B will be translated from SIR B to the ILA address of B by substituting the SIR prefix with the Locator *2001:610:158:2603*. The packet then travels on the underlay network to host B. When the packet arrives at Step 4 host B recognises that this address is an ILA address and replaces the Locator with the SIR prefix again. Finally, the packet is passed to application socket at Step 5. In order to send a packet back, the same steps are applied in the reverse order. Because of the translations, the application layer (more specifically the network sockets that they use) is now unaware of the location changes of nodes and addresses them solely based on the identifiers. As a result, communication can continue when location changes occur.

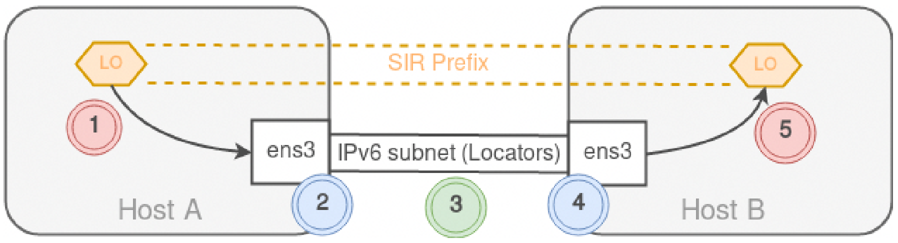

Figure 3: ILA translation example

The identifiers that are used in ILA can be configured in several ways depending on how ILA is deployed. All identifiers start by a 3-bit value that indicates their type. The most important types of identifiers are locally unique identifiers and virtual networking identifiers for IPv4 and IPv6. Following the type is one bit to indicate whether or not check sum-neutral mapping is used (when this feature is not used ILA packets on the wire will have an incorrect checksum). The remaining space is used for the actual identifier. Locally unique identifiers can use 60 of the 64-bits to encode a unique identifier. The virtual networking identifiers encode a VNID of 28-bits within the identifier, making the size of the identifier smaller. Encoding the VNID within the identifier makes it possible to subdivide the IPv6 overlay into smaller subnets. This can be used to give tenants their own virtual subnet that spans the entire overlay and to do Access Control List (ACL) based on the ILA address. A figure with the formats of these identifiers can be found in Figure 4.

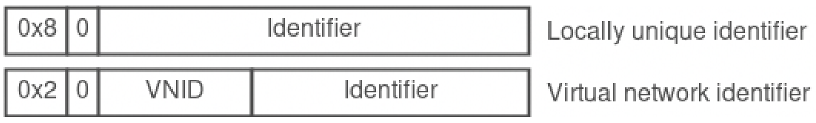

Figure 4: ILA identifier types

The basic ILA functions and primitives that have been discussed up until now are implemented in the Linux kernel since version 4.3 [19]. To be able to do the translations that ILA needs all of the hosts participating in the overlay need to exchange mappings. These mappings serve to find the Locator at which the identifier currently resides in the overlay. For this task, an additional piece of software is needed as this functionality is not handled by the Linux kernel code. The control plane used by ILA host to learn, distribute and configure ILA mappings is not standardised in the ILA draft RFC and it is up to the implementers to find a suitable mechanism.

3.1 Control plane

As mentioned in the previous section ILA needs a control plane in order to distribute mappings of identifiers to a location of the underlay network. Previous Locator/Identifier split systems needed such mechanisms as well. As such, there are already multiple mechanisms that have been designed to enable this functionality. These existing solutions can be categorised into roughly three groups based on their origins. A short overview of each of the groups is given below.

The first group of mapping distribution mechanisms comes from similar Identifier/Locator split protocols. ILNP [2] as well as Locator-Identifier Separation Protocol (LISP) [5]. ILNP, as well as LISP Delegated Database Tree (LISP-DDT) [6], opted to use distributed databases in order to do lookups for mappings.

Other mapping systems have turned to using routing protocols to disseminate mapping information. An older version of LISP, LISP Alternative Topology (LISP+ALT) [7], was such a system. The mapping information was exchanged using the BGP [13] routing protocol. This was possible because LISP IPv4 addresses have the same format as regular

IPv4. In order to leverage BGP to exchange routing layer information for other protocols than IPv4, an extension was made to BGP called MP-BGP [3]. Using this extension Layer 3 MPLS and/or IPv6 network information can sent over the same channel.

The last existing mechanism of distributing mappings stems from Service Oriented Architecture (SOA) using micro-services. These systems often need to communicate or retrieve information from other subsystems. In order to do this many of these systems store and retrieve small pieces of information in a (distributed) key/value store. ILA mappings could easily be disseminated in the same manner if such systems are already in place. Such a system is currently in use at the Facebook deployment of ILA [22].

4 NVO Requirements

First, the requirements of a multi-tenant, multi-domain overlay network are gathered. Next, a comparison to general industry requirements of overlay networks is made to identify differences. The result is a comprehensive set of requirements to test different ILA overlay configurations and control planes against.

In order to describe requirements, the same terminology found in the NVO working group documents will be used. A brief overview of this terminology is given in 2.1. The NVO requirements are discussed first from the perspective of NVEs as the data plane of the overlay and subsequently from the perspective of the NVAs that distribute mapping information as the control plane.

4.1 NVE Requirements

As the title of this paper suggests this research into networking technologies is focused on multi-tenancy in a multi-domain environment. During this research, a use-case was envisioned as an example to evaluate ILA for such purposes. Specifically, the use-case is to run research related data processing in a shared cloud. In this shared cloud different research groups and/or other participating organisations should be able to do computing and simulations with minimal interference from other users. A graphical representation of this is visible in Figure 5. From a high level, there are two specific aspects, namely the multi-tenancy combined with multi-domain. A short discussion on both aspects follows.

The multi-tenancy aspect is comparable to industry standard overlay requirements. Basically, each virtual network needs to be separated from other virtual networks and the underlying physical network. The official NVO requirements also state that address space should be able to be overlapping [12]. When using IPv4 on the overlays this is understandable since there is only limited private address space. However, since this research is focused on ILA and IPv6 only this is not a hard requirement in this case because address space of ILA tenants is IPv6 only and as such is much larger. The second difference between NVO working group requirements and this research is that tenants should be able, by explicit configuration, to access nodes of other overlays. The NVO working group has defined that virtual network packets should be bound to the scope of that specific virtual network and optionally a gateway to outward networks [12]. During this research, an addition is made to this working group definition that it should also be possible for tenants to interconnect overlays with explicit configuration. In this case, it is preferred that the private overlay addresses do not overlap because otherwise routing issues start to appear.

The second perspective is multi-domain. This stems from the idea that in order to save costs each organisation (domain) shares their available resources of computing and network infrastructures into one logical overlay. Within this overlay, it should be possible for containers to not only move intra-domain but if possible also inter-domain. This enables organisations to leverage computing resources of other participating organisations if need be. As a result of this multi-domain property, the overlay should be able to connect NVEs through Wide Area Network (WAN) links. This is because most likely each organisation will host the infrastructure at their own site. The cloud term also indicates that a certain level of scalability is expected. The whole cloud overlay should be able to serve at least several thousands of containers located on hundreds of NVEs.

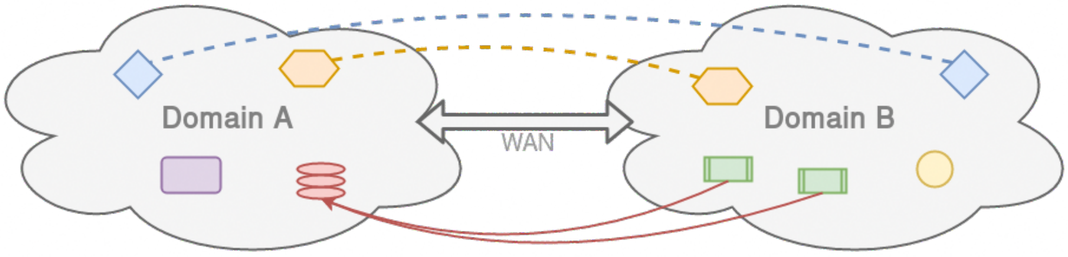

Figure 5: Envisioned multi-tenant, multi-domain ILA overlay

Most overlays only span a single administrative domain, but due to the multi-domain aspect, this does not hold for this use-case. Participating organisations would want to a certain extent remain in control of their subset of an overlay. This creates a multi-stakeholder situation where some, but not all information and resources are shared. It should thus be possible to configure this by the owner of a domain with some mechanism of ACLs. This is an important difference from common overlay solutions that enable tenants to leverage resources at a single large provider and as such a single administrative domain.

Finally, there are some general requirements. The chosen solution should be performant, meaning that there are no significant performance penalties when creating, deleting or moving containers. Secondly, it should not require complete re-architecture of underlying systems, but only on the NVEs and NVAs. And lastly, the overlay should be easy to debug, maintain and use without requiring major changes in operational practices.

4.2 NVA control plane requirements

An NVO needs some kind of control plane, without this critical piece it is not possible to make an overlay function as NVO. This control plane is implemented in NVAs [12]. These nodes distribute and control the NVEs data plane that makes the overlay function as intended. As such it is a critical piece of infrastructure that should have high performance, high resiliency and most-likely also a fail-over mechanism. This can be implemented in several ways. Possibilities are to use traditional directory databases that are replicated for reliability or using some form of gossiping on the network to spread this information. In a single domain setting, building such systems is quite straightforward as the chosen system can be managed by one entity in a local setting. However, in a multi-domain setup, more thought has to be put into how such systems handle inter-domain communication over longer WAN connections. For example what happens when new domains join or leave? Should mappings incrementally synced or sent in bulk? Are all mappings synchronised by default? Or is it preferable to only share a subset to other domains?

The multi-domain aspect thus brings more security requirements. It is important in the multi-domain environment to have some kind of ACL mechanism that gives each domain owner control over what is published to other domains and what not. A good example of this would be that a subset of the overlay is used for internal data processing. Ideally, it is not preferable to announce this to other participating domains. This is different from regular NVO architectures where the infrastructure is under the control of a single entity where such ACL is not needed. Secondly, the information that is passed through the control plane needs to be authentic and cannot be compromised before reaching other domains.

5 ILA configuration

After having established the requirements, the subsequent step is to show in what configurations ILA can fulfil the previously defined requirements. This is tested by creating a test setup of two virtual machines that are ILA hosts. On these VMs different ILA addressing scenarios are experimented with. Each of these addressing schemes should enable ILA overlays that meet the previously defined requirements.

Configuring ILA is quite challenging because most of the “configuration” of NVEs is implicit. There are no configuration files and/or variables to set. Instead, the well-known `ip` command is used to manipulate the kernel routing tables to include ILA translations. As a result, the “configuration” of ILA is largely dependent on which addressing decisions are made up front.

When making the addressing decisions it is important to keep in mind that there are two constraints within ILA addresses that must be met. The first is that only one SIR prefix can be mapped to one locator address. This is to prevent that the mapping can become ambiguous. The second constraint is that the identifier portion of ILA addresses should be unique. When the locator is substituted back to the SIR prefix the remaining (identifier) address should be unique in order to prevent routing issues just like regular duplicate IP addresses.

5.1 Test setup

In order to test different configurations, a minimal ILA setup was created. In this setup, two VMs were running on top of a virtual bridge. The virtualisation stack that was used to create the VMs was `libvirt` [4] on `qemu-kvm` [5]. On both VMs an IPv6/64 globally unique locator address was configured to give the underlay network IPv6 connectivity. This required some manual network configuration on the host as `libvirt` only supports one /64 subnet for all VMs by default. On each of the two VMs Ubuntu 17.04 was installed, as older versions of Ubuntu did not have a recent enough version of the `iproute2` package. For the underlay network to function on both VMs, static routes were configured and the connection was tested using the `ping` program.

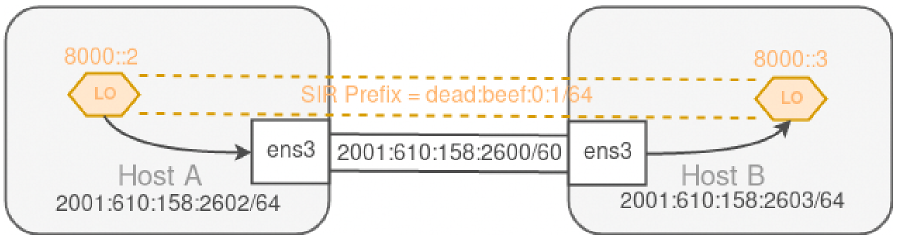

Figure 6: ILA test setup

To configure ILA first an ILA address was assigned to the loopback interface. This step was quite straightforward, there was no special ILA configuration needed. The address that was assigned did need to adhere to some requirements. First, the address must start with a fixed /64 prefix that is used throughout the overlay (the SIR prefix). Secondly, the last 64 bits should contain an identifier other than ::1, as this is the reserved ILA host address.

Once both VMs have an SIR address assigned the ILA NAT functions have to be setup. This requires the ILA kernel module to be loaded. To load this module manually `modprobe ila` was executed. Now the `ip` commands become ILA specific, and the address used previously are checked to adhere to the ILA specifications. At this point, the first issue arose. Some addresses were accepted as ILA addresses and others were not. After a lot of trial and error, the following explanation was found. The identifier

[4] `https://libvirt.org/`
[5] `https://www.linux-kvm.org/`

used must also adhere to a specific format, more specifically, the first 4 bits are used to indicate a type. The type with 0 is a local scope interface identifier [8], causing the translation rules to fail. In order to be able to add an address successfully the first hexadecimal digit of the identifier must be set to account for the type. These can either be a locally unique identifier or virtual network identifiers that have a VNID field specified. Either can be added using the `encap ila` statement as ILA NAT routes. The `encap` statement is used to indicate to the kernel that for these routes the kernel LightWeight Tunnel (LWT) infrastructure must be used to do the NAT step before the packet is passed on. The precise commands can be found in Listing 1.

Listing 1: `iproute2` commands used to setup ILA

```
#Load the kernel module
modprobe ila

#Add ILA address to the lo interface
#SIR prefix: dead:beef:0:1, Identifier: 8000::2
#Identifier type: 8 -> 1, Locally unique identifier
ip addr add dead:beef:0:1:8000::2 dev lo

#Add ILA translation rule for incoming traffic
#Locator: 2001:610:158:2602, Identifier same as above
ip route add table local local 2001:610:158:2602:8000::2/128 \
 encap ila dead:beef:0:1 dev lo

#Add ILA translation route for outgoing traffic
#Locator: 2001:610:158:2603, Identifier 8000::3
ip route add dead:beef:0:1:8000::3 encap ila 2001:610:158:2603 \
 via 2001:610:158:2600::1 dev ens3
```

5.2 General addressing setup

As seen in the example setup in Figure 6 ILA needs a underlay network that is IPv6 capable so that packets, once translated, can be carried over the links. This underlay network provides the locator (physical) addresses for the ILA NVEs. These can either be an IPv6 Globally Unique Address (GUA) or IPv6 Unique Local Address (ULA) addresses [24]. When ULA addresses are used the NVEs in the ILA overlay can reach each other when they are within the same private domain. When GUA addresses are used as locator addresses it is possible to create ILA overlays that span over subnets. In this case, the ILA NVEs leverage the fact that they use regular IPv6 routing to the other NVE and do not require any tunneling between domains. Based on the multi-domain requirement, NVEs in our overlay do not necessarily belong to the same subnet, it is thus required to use GUA addresses as locators in order to create a multi-domain ILA overlay.

The second set of addresses that need to be configured are the SIR addresses that are used within the overlay. Two different types of SIR prefixes can be chosen depending on mobility requirements. When mobility is only required within (and possibly between) ILA domains ULA addresses are required. In order to distinguish multiple tenants and/or domains, this must be encoded in the full SIR address somehow. It was theorised that there are two different methods of how this could be implemented. By either creating one overlay that spans over multiple domains or linking separate overlay domains together.

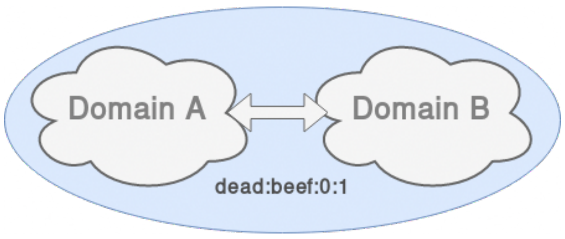

Figure 7: Multiple domains share a single SIR prefix

5.3 Single SIR scenario

The first option consists of one SIR prefix that identifies the overlay as pictured in Figure 7. In this scenario, the virtual network identifier type sacrifices some identifier address space to encode a VNID. This VNID in the spec indicates tenants. But we could change its semantics to be broader. To use the VNID functionality the identifier portion of the address needs to be in a particular address format. The correct type should be set to a virtual network identifier. Now the following 7 digits (28 bits) of the IPv6 address form a VNID field that can be used to encode hierarchical information before the actual identifier. The remaining 8 digits (32 bits) are used as the actual identifier of a specific container. In the scenario where only one SIR prefix was used for multiple domains, the identifier space was used to encode the domain. An example would be to encode the domain in the first 10 bits and using the remaining 18 bits for further subdivision within each domain. This would give 1024 possible domains and a further 260k sub-domains within each domain. Containers are able to have mobility within the entire overlay in this setup even though all subnets are using different address spaces.

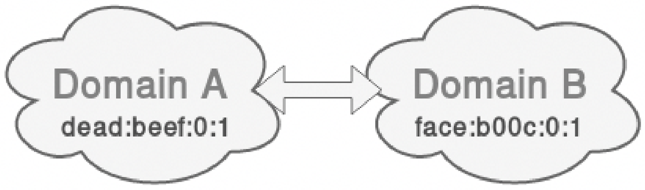

Figure 8: Multiple SIR, each domain has own SIR prefix

5.4 Multiple SIR scenario

The second option would be chaining separate domains together as pictured in Figure 8. Instead of sacrificing some VNID address space it could be possible to distinguish the domain by different SIR prefixes. This would mean that a virtually unlimited number of domains are able to chain together their overlays and would allow for using significantly more subdivision of each domain.

However, there are also some downsides/complications to such a setup. First of all the control plane that is used within NVAs must be able to not only share mapping information but it also has to announce the corresponding SIR prefix. This is because locators can now be tied different prefixes. This also has the implication that containers cannot move between domains without a change of address, as the prefix part has to be updated to reflect the domain where they are in.

5.5 Other considerations

ILA itself does not offer any particular method for isolating tenant traffic. However, because within ILA overlays none of the addresses should be overlapping, the IP addresses can be leveraged to use for ACL. In ILA the isolation of tenants/domains must be handled on the IP level. One of the methods would be to enforce this separation on NVEs using firewall rules based on IP addressing. When using the VNID to separate tenants it is possible to make rules based on IP prefixes, consisting of the full SIR prefix and part of the identifier space, in this case, the VNID, to allow traffic from same tenant/domain. This same concept is applied in Romana [6] for tenant isolation based on IPv4 subnets.

As most firewalls are only able to match on the leading bits of a prefix it is important to make sure hierarchical separation is done properly in leading bits. In the case of the single SIR scenario, it is possible to make an optimisation if inter-domain communication is allowed. In that case, it might be worthwhile to minimise ACL rules by using a global "project" ID in front of the organisation, thereby enabling easy ACL rules by matching on the project ID. By creating a project ACL rule automatically all of the participating domains are allowed.

One other consideration that might be of importance is the effect of ILA translation on packets that are sent over the network. ILA has two operating modes, with and without checksum neutral mapping. Without checksum neutral mapping the ILA translation will mangle transport layer checksums as they are based on SIR addresses. Packets will thus travel over the wire with incorrect checksums. As the translation back to the SIR address happens on the NVE before the packet is passed to the transport/application layers upper layers do not notice this. But "malformed" packets because of incorrect checksums are expected when listening on the wire. Network appliances are expected to

[6] `http://romana.io/`

forward packets regardless of their checksum, only end hosts should drop packets that are malformed. If in any case, these malformed packets on the wire are undesirable it is possible for ILA to keep transport layer checksums intact. However, this requires that the last 16 bits of identifier space are used to correct the one complement value algorithm of the transport layer checksum. This limits the pseudo-random address space for specific containers and requires more computation during translation leading to a performance penalty.

6 ILA control plane

To find a suitable control plane to use with an ILA addressing scheme we start by looking at previously implemented mapping systems for other Identifier/Locator systems. After analysing their design and characteristics a candidate is picked. Finally, we look at how this candidate can be adapted or extended to be suitable as an ILA mapping system in a multi-tenant and multi-domain environment. Ideally, this is demonstrated by a working proof of concept that incorporates all of the aspects of an NVO.

6.1 Existing control planes

As mentioned before the first group of mapping distribution mechanisms comes from older Identifier/Locator split protocols. These solutions have chosen to use various forms of distributed databases. In ILNP [2] the mappings were distributed using the existing Distributed Naming System (DNS) by adding new Resource Record Types (RRTYPEs) in which the mappings were described [1]. LISP [5] has had multiple systems to do mappings, the latest of which is LISP-DDT [6]. This system uses hierarchical delegation just as DNS to be able to look-up mapping information. Architecturally this system is the same as DNS but instead of using the existing DNS infrastructure like ILNP does the LISP-DDT runs on separate nodes and the tree is completely separate from DNS.

Other mapping systems have turned to using routing protocols to disseminate mapping information. The predecessor of LISP-DDT, LISP+ALT [7], was such a system. To do lookups a global overlay network was constructed of General Routing Encapsulation (GRE) tunnels between all border routers. These routers would exchange mapping information using the BGP [13] routing protocol. However, this approach was cumbersome due to the setup of all the GRE tunnels. Other protocols simplified disseminating mappings by using MP-BGP [3] as mentioned in related work such as MPLS based VPNs as well as recent popular layer 2 NVO technologies, such as VXLAN [11] in combination with EVPN [15].

The last existing mechanism of distributing mappings stems from application development and container orchestration systems. It is common these days to design applications using micro-services. These systems often need to communicate or retrieve information from other subsystems. In order to do this many of these systems store and retrieve small pieces of information in a (distributed) key/value store. Examples of software that is often used to provide these look-up services are Redis[7] and Etcd[8]. In ILA mappings could easily be disseminated in the same manner if such systems are already in place.

6.2 ILA control plane fit

Each of the previously described mechanisms has particular characteristics that stem from design decisions and the way they were implemented. We give a short overview of three characteristics for each of the options; scope, organisation and ACL. These can be found in Table 2. These characteristics are interesting because of the multi-domain requirement that we have.

Control Plane	Scope	Organisation	ACL
Distributed K/V store	Intra-Domain	Flat	Limited
ILNP	Global	Hierarchical	Limited
LISP+ALT	Global	Flat	Limited
LISP-DDT	Global	Hierarchical	Limited
MP-BGP extensions	Selective (global)	Hierarchical	Many

Table 2: Existing control plane characteristics

As can be seen in Table 2 most existing control planes are scoped in one of two ways; Either global or entirely intra-domain, and providing limited or many ACL options. Most offer little to no mechanisms out of the box to do any form of ACL when the only option is to either publish mappings or not. This makes it harder to adapt these systems to fit the multi-domain environment.

This quickly leads to the MP-BGP option that does offer filtering and extended policies native to BGP. These can be used as ACL mechanisms for announcing routes together with their mappings when peering with other domains when handling Exterior Border Gateway Protocol (eBGP) connections. This gives each domain control over what information they want to share with peers and together with a fitting addressing scheme and accompanying firewall rule-set gives a complete ACL implementation.

7 Discussion

Although ILA is a viable option to use as an NVO it is still very experimental. During this research, documentation was almost non-existent. Only the draft RFC provided

[7]https://redis.io/
[8]https://coreos.com/etcd

insight into the inner workings of ILA. Using the Linux kernel data plane implementation required a lot of trial and error. This process took a lot more time than it should have in order for ILA to be considered worthwhile to implement.

Also as it stands at the moment of publishing this paper there is no actual working ILA control plane available. The author of this paper has made some progress in implementing an ILA AFI/SAFI extension [23] to BGP in the exaBGP [9] daemon for experiments but this code is nowhere near production ready.

8 Conclusion

Using an NVO in a multi-tenant, multi-domain environment introduces additional complexities to the standard NVO design. Apart from regular NVO features such as tenant isolation, the multi-domain aspect creates some new requirements. Because there is no longer a single entity that has full control over the overlay domain additional ACL methods are needed for each domain to be able to limit permissions from other domains. Additionally, the overlay needs to span WAN links so that containers are able to move not only within one SIR domain but between SIR domains. To be able to materialise these requirements the data plane and control plane needs to be flexible.

During the research, two possible ILA configurations were looked at. The single SIR and multiple SIR scenarios. It turns out that in order for ILA to provide mobility over the whole overlay a single SIR is required. By using the virtual network identifier VNID space and creating a hierarchy it is possible for thousands of domains to live in one overlay in the proposed addressing setup. Other options are also possible if the semantics of how SIR addresses are divided is determined up front. ILA is very flexible and can match specific needs of overlays as long as network engineers can be creative enough with the address space to create hierarchies that fit the use-case, as this is the implicit way in ILA is "configured".

In order for the ILA data plane to work it needs a control plane that has the right characteristics that match the environment. The MP-BGP control plane suits the multi-domain requirement best. As the original BGP was designed for similar goals it offers existing methods to make sure only specified information is received/announced to other domains. But MP-BGP can also serve mapping information between NVAs internally by using Interior Border Gateway Protocol (iBGP) and optionally a route reflector. This simplifies the NVO setup and operation as a single routing protocol can be used for regular routing, distributing mappings internally, regular peering and exchanging mappings with other domains. The last benefit is that BGP has already proven itself to be reliable and able to scale. It is important however to keep BGP's (lack of) security in mind.

[9] `https://github.com/Exa-Networks/exabgp`

It is thus feasible to create an ILA overlay in a multi-tenant, multi-domain environment by designing an addressing scheme that fits the use-case and pairing it with a control plane that enables fine-grained controls.

9 Future Work

In order to be able to use ILA as a network virtualisation layer for containers a network plugin for container orchestration systems needs to be developed that actually automatically sets up ILA within containers and instructs the control plane to distribute mappings. It might be worthwhile to try and extend a project called Calico[10], which is an existing BGP control plane plugin for most container orchestration plugins.

Finally, during this research, no performance measurements were done to see how ILA impacts regular routing performance. This could be looked at more in-depth as more ILA implementations begin to surface that leverage other technologies than the regular Linux kernel. Examples would be Virtual Packet Processing (VPP)/Virtual Packet Processing (DPDK) or extended Berkeley Packet Filter (eBPF)/eXpress Data Path (XDP) implementations.

[10]`https://www.projectcalico.org/`

Acronyms

ACL Access Control List. 9, 11, 12, 16, 18, 19

BGP Border Gateway Protocol. 5, 9, 10, 17–20

BUM Broadcast, Unknown unicast, and Multicast. 5

DC Data Center. 4

DNS Distributed Naming System. 17

DPDK Virtual Packet Processing. 20

eBGP Exterior Border Gateway Protocol. 18

eBPF extended Berkeley Packet Filter. 20

EVPN Ethernet VPN. 5, 17

GRE General Routing Encapsulation. 17

GUA Globally Unique Address. 14

iBGP Interior Border Gateway Protocol. 19

IEN Internet Experiment Note. 6

ILA Identifier Locator Addressing. 1–3, 5, 7–20

ILNP Identifier-Locator Network Protocol. 7, 9, 17

IP Internet Protocol. 3–6, 13, 16

IPv4 IP version 4. 4, 6–10, 16

IPv6 IP version 6. 3, 4, 6–10, 13–15

LAN Local Area Network. 5

LISP Locator-Identifier Separation Protocol. 9, 17

LISP+ALT LISP Alternative Topology. 9, 17

LISP-DDT LISP Delegated Database Tree. 9, 17

LWT LightWeight Tunnel. 14

MP-BGP Multi-Protocol Border Gateway Protocol. 1, 5, 10, 17–19

MPLS Multi-Protocol Label Switching. 5, 10, 17

NAT Network Address Translation. 3, 13, 14

NVA Network virtualization Authority. 4, 5, 10–12, 16, 19

NVE Network virtualization Edge. 4, 5, 10–12, 14, 16

NVO Network virtualization Overlay. 1, 3, 4, 10–12, 17–19

RFC Request for Comments. 6, 9, 18

RRTYPE Resource Record Type. 17

SIR Standard Identifier Representation. 7, 8, 12, 13, 15, 16, 19

SOA Service Oriented Architecture. 10

UDP User Datagram Protocol. 5

ULA Unique Local Address. 14, 15

VLAN Virtual Local Area Network. 5

VM Virtual Machine. 3, 4, 12, 13

VNID Virtual Network ID. 5, 9, 14–16, 19

VPLS Virtual Private LAN Service. 5

VPN Virtual Private Network. 5, 17

VPP Virtual Packet Processing. 20

VXLAN Virtual Extensible LAN. 5, 17

WAN Wide Area Network. 11, 12, 19

WLAN Wireless Local Area Network. 6

XDP eXpress Data Path. 20

References

[1] Atkinson. R.J. et al. *DNS Resource Records for the Identifier-Locator Network Protocol (ILNP)*. RFC6742. Nov. 2012. URL: https://tools.ietf.org/html/rfc6742.

[2] Atkinson. R.J. et al. *Identifier-Locator Network Protocol (ILNP) Architectural Description*. RFC6740. Nov. 2012. URL: https://tools.ietf.org/html/rfc6740.

[3] Bates. T. et al. *Multiprotocol Extensions for BGP-4*. RFC4760. Jan. 2007. URL: https://tools.ietf.org/html/rfc4760.

[4] Black. D. et al. *An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)*. RFC8014. Dec. 2016. URL: https://tools.ietf.org/html/rfc8014.

[5] Farinacci. D. et al. *The Locator/ID Separation Protocol (LISP)*. RFC6830. Jan. 2013. URL: https://tools.ietf.org/html/rfc6830.

[6] Fuller. V. et al. *LISP Delegated Database Tree*. draft-ietf-lisp-ddt-09. Jan. 2017. URL: https://tools.ietf.org/html/draft-ietf-lisp-ddt-09.

[7] Fulrer. V. et al. *Locator/ID Separation Protocol Alternative Logical Topology (LISP+ALT)*. RFC6836. Jan. 2013. URL: https://tools.ietf.org/html/rfc6836.

[8] Herbert. T. et al. *Identifier-locator addressing for IPv6*. draft-herbert-nvo3-ila-04. Mar. 2017. URL: https://tools.ietf.org/html/draft-herbert-nvo3-ila-04.

[9] Kompella. K. et al. *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*. RFC4761. Jan. 2007. URL: https://tools.ietf.org/html/rfc4761.

[10] Lasserre. M. et al. *Framework for Data Center (DC) Network Virtualization*. RFC7365. Oct. 2014. URL: https://tools.ietf.org/html/rfc7365.

[11] Mahalingham. M. et al. *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*. RFC7348. Aug. 2014. URL: https://tools.ietf.org/html/rfc7348.

[12] Narten. T. et al. *Problem Statement: Overlays for Network Virtualization*. RFC7364. Oct. 2014. URL: https://tools.ietf.org/html/rfc7364.

[13] Rekhter. Y. et al. *A Border Gateway Protocol 4 (BGP-4)*. RFC4271. Jan. 2006. URL: https://tools.ietf.org/html/rfc4271.

[14] Rosen. E. et al. *BGP/MPLS IP Virtual Private Networks (VPNs)*. RFC4364. Feb. 2006. URL: https://tools.ietf.org/html/rfc4364.

[15] Sajassi. A. et al. *BGP MPLS-Based Ethernet VPN*. RFC7432. Feb. 2015. URL: https://tools.ietf.org/html/rfc7432.

[16] Turner. J. et al. "Diversifying the internet". In: *Global Telecommunications Conference, 2005. GLOBECOM0́5. IEEE*. Vol. 2. IEEE. 2005, 6–pp.

[17] IEEE. *802.1Q-2014 IEEE Standard for Local and metropolitan area networks*. 802.1Q-2014. 2014. URL: https://standards.ieee.org/findstds/standard/802.1Q-2014.html.

[18] Bennett. C. J. *Issues in the interconnection of datagram networks*. IEN1, retrieved on: 15-06-2017. July 1977. URL: https://www.rfc-editor.org/ien/ien1.pdf.

[19] Corbet. J. *Identifier locator addressing.* retrieved on: 08-06-2017. Sept. 2015. URL: https://lwn.net/Articles/657012/.

[20] Saltzer. J. *On the Naming and Binding of Network Destinations.* RFC1498. Aug. 1993. URL: https://tools.ietf.org/html/rfc1498.

[21] Kaat. M. *Overview of 1999 IAB Network Layer Workshop.* RFC2956. Oct. 1999. URL: https://tools.ietf.org/html/rfc2956.

[22] Lapukhov. P. *Internet-scale Virtual Networking using ILA.* https://www.nanog.org/sites/default/files/20161018_Lapukhov_Internet-Scale_Virtual_Networking_v1.pdf. retrieved on: 05-06-2017. Oct. 2016.

[23] Lapukhov. P. *Use of BGP for dissemination of ILA mapping information.* draft-lapukhov-bgp-ila-afi-02. Oct. 2016. URL: https://tools.ietf.org/html/draft-lapukhov-bgp-ila-afi-02.

[24] Hinden. R. *Unique Local IPv6 Unicast Addresses.* RFC4193. Oct. 2005. URL: https://tools.ietf.org/html/rfc4193.