Conceptual Architecture for Evaluating Inter-Domain Solutions within OneLab2

TR09-0002

Document Properties:

Title of Contract	Publish-Subscribe Internet Routing Paradigm	OneLab2: An Open Federated Laboratory Supporting Network Research for the Future Internet
Acronym	PSIRP	OneLab2
Contract Number	FP7-INFSO-ICT 216173	224263
Start date of the project	1.1.2008	1.9.2008
Duration	30 months, until 30.6.2010	27 months, until 30.11.2010
Document Title:	Conceptual Architecture for Evaluating Inter-Domain Solutions within OneLab2	
Date of preparation	July, 2009	
Author(s)	Christopher Reason, Dirk Trossen	
Responsible of the deliverable	Christopher Reason	
	Phone: +44 7501 231909	
	Email: christopher.reason@bt.com	
Target Dissemination Level:	Public	
Status of the Document:	Completed	
Version	1.0	
Document location	http://www.psirp.org/	http://www.onelab.eu/
Project web site	http://www.psirp.org/	http://www.onelab.eu/

Production Properties:

Reviewed by:	

Revision History:

Revision	Date	Issued by	Description
0.1	28/5/2009	Chris Reason	
0.3	13/7/2009	Dirk Trossen	Added PSIRP use cases, formatting
0.4	14/7/2009	Chris Reason	Minor grammatical changes, formatting
1.0	15/7/2009	Chris Reason	Production version

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°224263-OneLab2.

Table of Contents

1 Introduction

Projects like PSIRP aim, among other problems, at evaluating inter-domain technologies, such as for routing and rendezvous. Evaluating these experimentally requires an inter-domain setting. While the 'standard' Planetlab environment provides such setting over the 'standard' Internet, it is crucial for the evaluation of Future Internet approaches to evaluate the technologies in settings that *emulate* potential future inter-domain relations, these relations stemming from the business and regulatory relations to be in the future.

This report outlines the general problem space, a conceptual solution for the problem and specific sets of technologies being used in building an actual solution. For this, we place the PSIRP project in the centre as an example for a customer applying this solution. But the proposed solution should be applicable for similar inter-domain evaluations.

OneLab2 [ONE2008], an EC funded FP7 collaborative project is an open federated laboratory supporting network research for the future internet, building upon the original OneLab project's PlanetLab Europe (PLE) testbed. OneLab2 started in September 2008 and will run for 27 months, with the aim to extend PLE's functionality and scale

PSIRP (Publish-Subscribe Internet Routing Paradigm) [PSI2008], another EC funded FP7 project aims to develop, implement, and validate an information-centric internetworking architecture based on the publish-subscribe paradigm, which appears to be one of the most promising approaches to solving many of the biggest challenges of the current Internet. PSIRP started in January 2008 and will run for 30 months.

2 Problem Space

Most researchers have at some point faced questions such as “what is the performance of my new protocol”, “how does my new technology perform in a highly distributed environment” or “how does my pre-commercial code perform under more realistic networking conditions”. When developing inter-domain technologies this is no different. In the most part, these questions are solved either experimentally or using models, and it is the former of the two which this report focuses on; “How do I experimentally verify inter-domain technologies in a realistic setting”.

It is important to note the inclusion of the “in a realistic setting” clause, as testing an inter-domain protocol designed to be run over a potential future internet architecture over the current internet will rarely provide irrefutable evidence for its performance.

It is therefore important to identify factors including:

1. Which technology or technologies will I need to evaluate? (e.g., routing, rendezvous, resource discovery)
2. What architectural or topological properties do I need control over? (e.g., number of nodes, links, ASes, routers, bandwidth, latency, link failure rate)
3. What is the motivation behind the evaluation? Is it to provide security, scalability, or performance guarantees, investigate the effect on business peering/transit relationships, or something else?
4. What software and hardware resources do you require/have which are suitable to evaluate the inter-domain technology over?
5. In which environment should the evaluations take place over – the current internet, or a potential future internet architecture?

This then leads onto the question of which method will be used for the evaluation – using a testbed, simulation or emulation, and will depend upon the available resources identified in point 4 above.

It is due in part to the factors outlined above that one evaluation environment won’t suit all, and will often only be appropriate for one particular technology or method of evaluation, and this is a common problem encountered when looking for experimental evaluation.

As such, this paper will outline a conceptual solution to the problem of experimentally evaluating inter-domain protocols in section 3, explore the existing technologies to solve this problem (the solution space) in section 4, before focusing on how the outlined solutions could be applied to provide experimental evaluation for the inter-domain technologies used in the PSIRP project.

3 Conceptual Solution

As mentioned earlier, the three main methods of experimentally evaluating inter-domain technologies are emulation, simulation and (experimental and usually local) testbed. All three have particular strengths and weaknesses, so an evaluation architecture which combines all three should provide the greatest flexibility while retaining the best features of each.

	Pros	Cons
Emulation	• Can potentially be run over existing infrastructure using a virtual machine or various pieces of software • Enables technology to be run over a software implementation of required hardware	• Potential performance issues due to emulation environment (e.g., Virtual Machine) • Rooted in current networking environment which may impair experiment at hand, e.g., through underlying business relations, usage of particular wireless technologies and alike.
Simulation	• Scalability • No specific hardware requirements • Can simulate any inter-domain constellation	• Hard to tell with complex simulations whether it is in fact an accurate representation of real life • Potentially difficult to implement technology to evaluate in a simulation language • Typically high performance computers required for larger simulations
Testbed	• Actual implementation can be used without need for modification • Provides native environment	• Costly to build and maintain • Scalability issues • Rooted in current networking environment which impairs experiment at hand, e.g., through underlying business relations, usage of particular wireless technologies and alike. • Often localised and therefore hard to re-use

Table 1 Evaluation Methods

To summarise the three approaches, evaluating inter-domain technology is best suited using emulation when you already have access to a particular testbed, but it doesn't currently meet all the resource/software requirements. For example, the use of a virtual machine may enable evaluation of the technology over the existing testbed.

Simulation is best suited when scalability of a particular technology is required and the cost of implementing this as a testbed is impractical for any reason (money, space, time etc)

Testbeds are best suited for evaluating inter-domain technologies in a real world environment using real world hardware (providing the most realistic evaluation environment).

With all three evaluation approaches, the key features remain the same – the ability to change routing protocols, the access to raw network resources, network topology, peering and transit relationships and network resource availability/constraints. The two most important features of a conceptual solution however are the access to the raw network resource, in addition to easily being able to create any network topology, even aggregated (on the AS level).

The first point is particularly important, as the greater the access to the raw network resources (such as Ethernet or fiber), the less restrictions are placed upon which protocols or technologies can be evaluated.1 The same is true for topology creation – the existing topology may not contain the ideal number or type of features, and this is especially poignant when trying to evaluate a protocol designed to run over a potential future internet topology over the current internet topology.

Taking into account all of the above requirements, an ideal conceptual architectural solution contains all three outlined evaluation methods above, in addition to the use of both the Internet (and the IP protocol) as well as a raw Ethernet network provided, for example, by the EC FP7 project FEDERICA [FED2009].

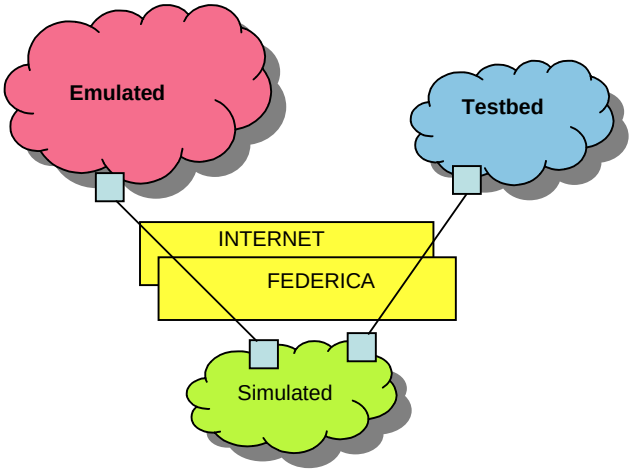

Figure 1: Evaluation environment combining all approaches

[1] While most current protocols run over IP, the future internet may not. It could for example run directly over Ethernet.

4 Current Solution Space

In this section, a number of relevant resources have been identified which could be combined to provide an appropriate environment for solving the inter-domain experimental evaluation problem.

4.1 PlanetLab

PlanetLab [PLA2008], the open federated laboratory supporting network research for the future internet is the largest network testbed in the world with 1000+ nodes at 480+ sites. Each of the nodes is directly connected to the internet, without network address translators (NAT's), firewalls or proxies, and usually offers bi-directional network bandwidth in excess of 400kb/s. PlanetLab only offers layer 3 access however.

In order to use the PlanetLab testbed, you must be allocated a slice[2], which provides an empty virtual machine running Fedora r8 on each sliver[3] in your slice. In order to create a network overlay, the topology information would have to be manually configured.

4.1.1 VNET

As PlanetLab uses the Linux VServer [VSE2008] technology to virtualise the underlying nodes resources, it means that users have no direct access to the hardware or network interfaces. As the network stack in not virtualised, creation and configuration of the network device is disabled to ensure traffic isolation and ensure that one user can't affect how traffic generated or received by another slice is handled. The VNET [HUA2005] module ensures that the standard Linux/BSD socket API's continue to work correctly though. In order to track connections, VNET relies on Linux's NetFilter system, ensuring that sent packets are associated with the slice that sent them and received packets are associated with connections which they initiated or bound.

As all packets pass through the VNET module, and each connection is defined on a per-protocol basis, TCP, UDP, ICMP, GRE and PPTP protocols are supported.

To support reservation for IP protocols beside IP, safe raw sockets may be created, however they, along with raw IP sockets have extra restrictions imposed upon them by VNET. Restrictions on sent packets include:

- The packet must be trackable by NetFilter (i.e. IP multicast packets are disallowed)
- The connection with which the packet is associated must not be owned by another slice
- The packet must be correctly formed
- The TCP or UDP source port must be 1024 or above (or if not, the port must have been bound using the appropriate PlanetLab service)

4.1.2 TUN/TAP Devices

The VNET module emulates a single static TAP interface, tap0, to ship packets between the kernel and user space (for each slice). While this interface isn't configurable, it does enable the user to implement a network stack outside of the kernel.

4.1.3 Bandwidth Caps

In order to ensure fair utilisation of the network resources, each slice is apportioned a percentage of the outbound bandwidth. Each slice receives one share of the bandwidth, and the root slice on the node receives five shares. This means that for a 10Mb connection, if 5

[2] A slice is a set of distributed resources across any or all of the 1000+ PlanetLab nodes.

[3] A sliver is a set of allocated resources on a specific PlanetLab node

slices plus the root slice are all actively competing for outbound network capacity, the root slice will receive 5Mbs, and each of the remaining five slices will receive 1Mbs each.

While it is unlikely that every slice will be competing for their portion of the outbound bandwidth, it becomes important to understand the complex bandwidth limits imposed by PlanetLab on every slice, especially when transmitting a large amount of data per day. This is especially true as each slice has a has a daily [global] tx limit of 5.4 gigabytes. While this won't be a problem for 99% of slices, this transfer limit soon seems small when a large number of nodes are involved in a large experiment.

4.2 PL-VINI

PL-VINI [FEA2006] is the prototype of VINI, a piece of software which enables arbitrary virtual layer 2 networks connected by software tunnels to run over a PlanetLab slice. Initially developed during 2006, PL-VINI is now unsupported and somewhat aged, although it has successfully been run over the current PlanetLab infrastructure.

4.2.1 PL-VINI tools

PL-VINI uses a variety of software to create an appropriate environment to run user configurable routers and tunnels within an overlay, detailed in the table below.

Software	Role
Click Modular software router [KOH2000]	Provides tunnelled virtual network topologies and IP forwarding
XORP (eXtensible Open Routing Platform) [XOR2009]	Implements routing protocols such as BGP, OSPF and RIP
User-Mode Linux [UML2008]	Provides the environment for running routing software
OpenVPN [OPE2009]	Allows external clients to connect to the virtual network (and inject traffic)

In addition to the above tools, PL-VINI includes a set of scripts to manage topology formation; however the mapping between the virtual network topology and the physical underlying network topology is an arbitrary one, in that the user must make that choice.

4.2.2 PL-VINI architecture

As a result of having to run over a prohibitive PlanetLab base architecture, PL-VINI uses a UML virtual machine to give the appearance of configurable network interfaces to the routing and tunnelling software. In practise, the packets are tunnelled from UML to the single tun/tap device provided to each sliver.

Figure 2 illustrates the PL-VINI architecture; the blue boxes represent the topology, which includes the virtual network devices and tunnels, the red boxes represent the routing and forwarding (data traffic does not enter UML) and the green box represents the virtual Ethernet device provided to every slice, which allows traffic to enter and exit the overlay.

As a result of the layered architecture and the routing having to occur within a second virtual machine in user space, forwarding speeds are significantly reduced from that of a native implementation, as illustrated in Figure 4. As Pl-VINI is based upon Click, due to the overhead of other components in the virtualised architecture, actual packet forwarding speeds will be less than 70,000 pps.

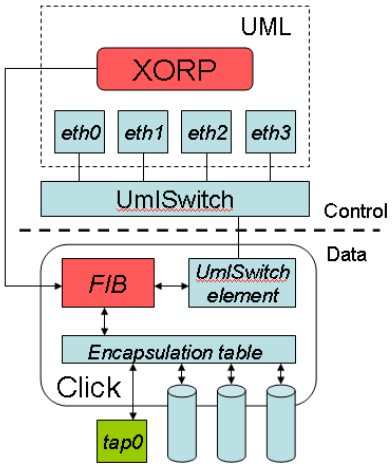

Figure 2: PL-VINI architecture [FEA2006]

4.3 VINI

Following the success of the PL-VINI prototype, a full software implementation called Trellis [BHA2008] was developed to improve forwarding speed (and therefore scalability of user-defined layer 2 & 3 virtual networks), and thus the VINI network [BAV2006] established. Whilst Trellis was based upon the PlanetLab boot image (which meant Linux VServer host virtualisation), it was modified to virtualise the network stack using Network Namespaces (NetNS) [NET2009].

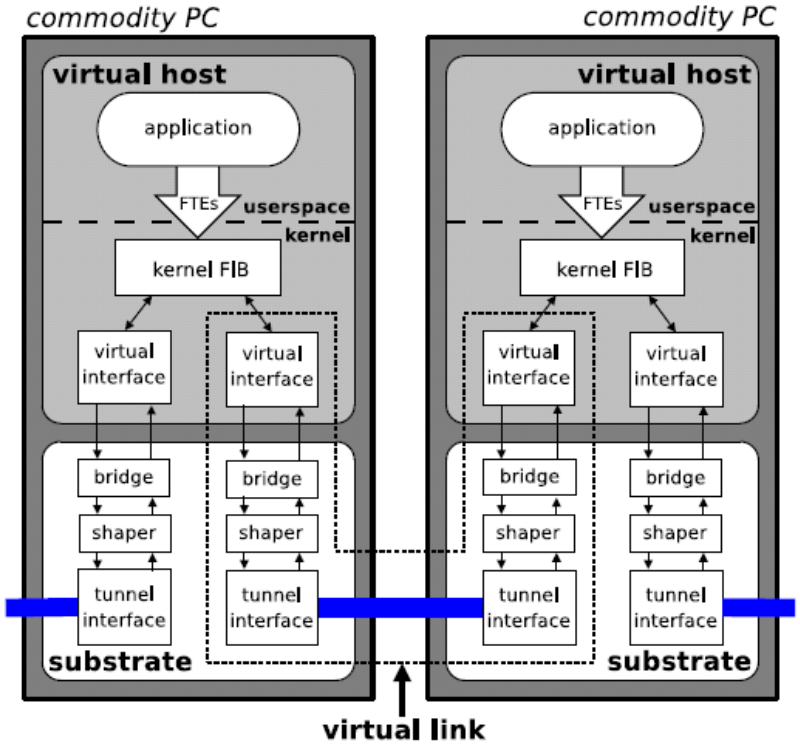

Figure 3: Trellis Architecture [BHA2008]

This meant that much of the PL-VINI architecture could be simplified, which resulted in a marked increase in packet forwarding speeds. Additionally, a Topology Manager (TM) module was created to help simplify topology creation; however it was only designed to create a virtual topology to mirror the physical one, which in most cases wouldn't be appropriate. As a result, topology creation is still largely configured through scripts similar to those used in the VINI prototype, PL-VINI.

The Trellis design decisions, aimed to balance speed, flexibility and isolation, provide features including the appearance of direct layer two links between nodes, custom data plane operations such as supporting non-IP packets and high packet forwarding speeds. A detailed view of the Trellis architecture is shown in Figure 3. The Trellis implementation uses two types of bridging; the standard Linux 'Bridge' module and a 'shortbridge', a custom high performance device created especially for bridging a single virtual interface directly to another tunnel interface. Where possible, the 'shortbridge' device is used, however it can't be used for point-to-multipoint links.

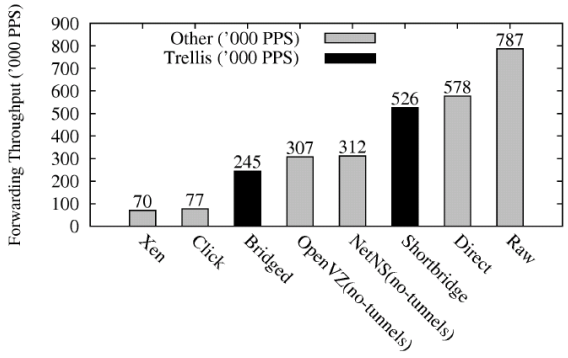

Figure 4: Peak forwarding performance in pps with 64byte packets [BHA2008]

As the Trellis software is no longer able to run on standard PlanetLab nodes, the VINI network was set up to run directly over Internet2, LamdaRail and CESNET – layer 2 academic and research networks. At present, the VINI network consists of 37 nodes at 22 sites.

4.4 FEDERICA

FEDERICA[4] is a technology agnostic network infrastructure that will provide layer 2 (and above) network technologies to be trialled over existing production networks such as GEANT2 and NRENs (National Research and Education Networks). FEDERICA, unlike PlanetLab, imposes far less restrictions upon the users of the testbed, enabling users to run any type of virtual machine on the network nodes. It is expected that FEDERICA will provide a topology creation tool or interface, although details haven't yet been released.

As of July 2009, there isn't a UK point of presence (PoP) on any of the associated underlying FEDERICA networks. Additionally, the required level of access required for advanced users (SuperUsers) isn't expected to be completed until phase three of the FEDERICA project in mid 2010.

[4] "Federated E-infrastructure Dedicated to European Researchers Innovating in Computing network Architectures"

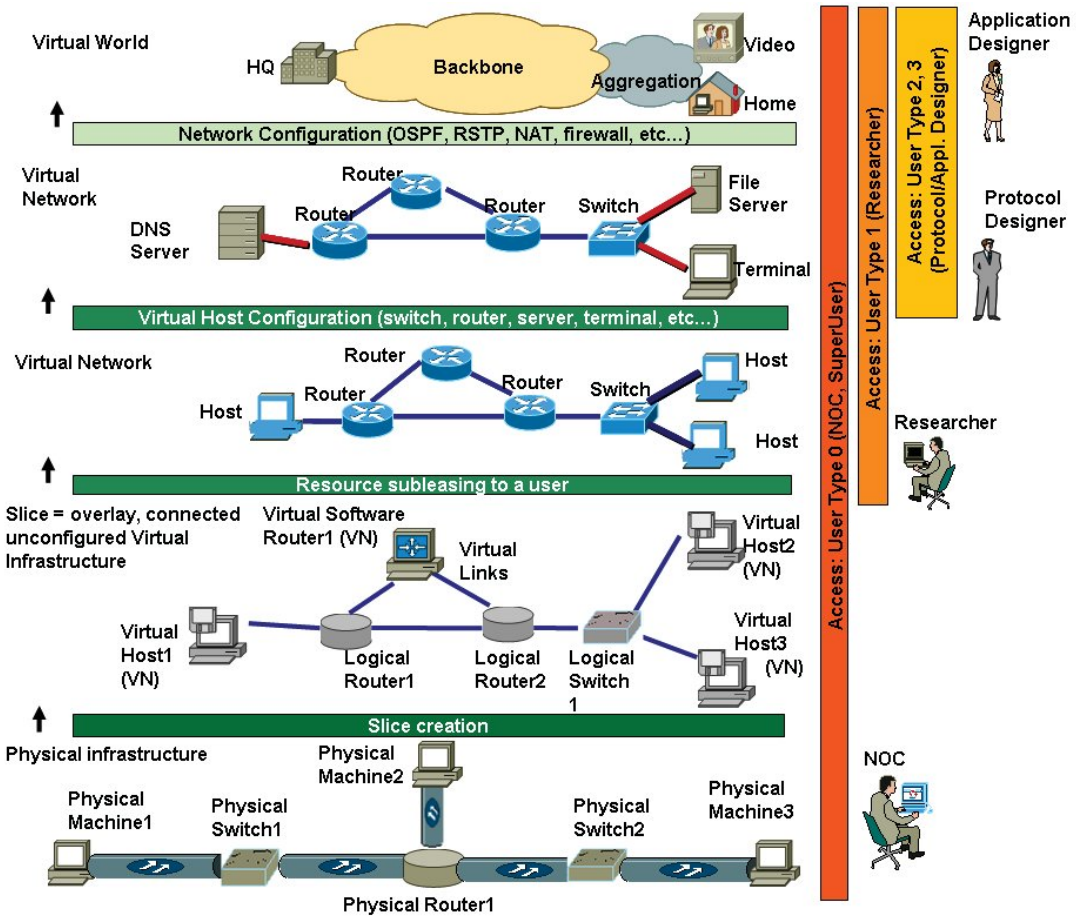

Figure 5: FEDERICA infrastructure [FED2009a]

4.5 ns-3

Ns-3 [NS32009] is a completely new piece of software following on from the widely used open-source Ns-2 simulator for internet systems, targeted at network researchers to enable the study of internet protocols and large-scale systems in a controlled environment. While ns-2 extensions won't be compatible with ns-3, it does bring numerous new features, including:

1. Extensible software core – extensively documented API.
2. Attention to realism – nodes modelled more like a real computer, with support for the sockets API and IP/device driver interface in Linux.
3. Software integration – conforms to standard input/output formats so other tools such as pcap, trace, ns-2 can be reused.
4. Support for virtualisation and testbeds.
5. Flexible tracing and statistics – decoupling of trace sources from trace sinks.
6. Attribute system – enables researchers to easily view and configure all values in a simulation.
7. New models – updated from ns-2.

Of particular note is bullet 4, "Support for virtualisation and testbeds", as ns-3 will provide two methods to integrate with real systems:

1) Virtual machines which run on top of ns-3 devices and channels

2) Ns-3 stacks which run in emulation mode and emit/consume packets over real devices

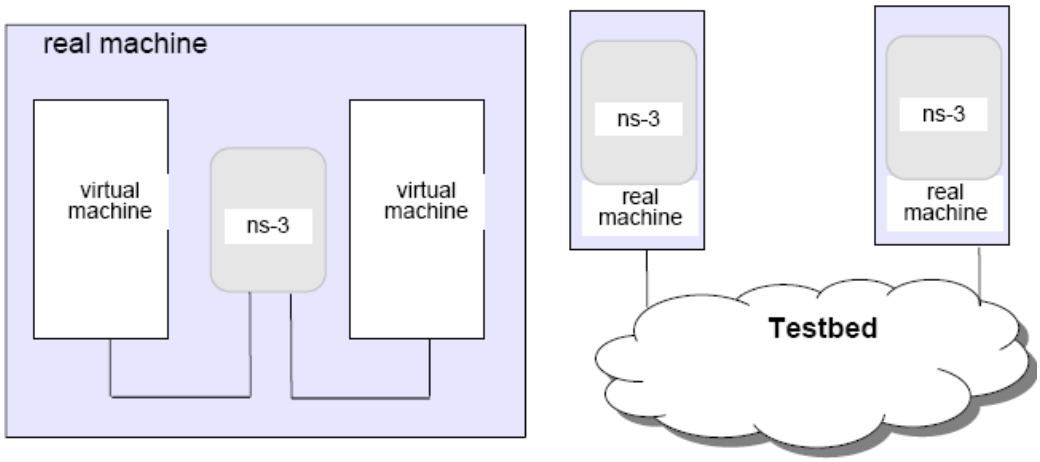

Figure 6: ns-3 interconnecting virtual machines, and testbeds interconnecting ns-3 stacks [NS32009a]

4.6 OpenVPN

OpenVPN is open-source software for creating SSL VPNs, with the ability to use either bridging or routing mode to create a virtual network using a tun or tap device respectively. Routing mode enables the user to create a wide area IP VPN by creating separate subnets with routes in-between. Bridging on the other hand creates a virtual wide-area Ethernet LAN, running on a single subnet. Due to the restrictive PlanetLab network interface permissions, OpenVPN can't run over a PlanetLab node outside of an additional virtual machine in a user's slice, such as UML, as used by PL-VINI.

4.7 VDE (Virtual Distributed Ethernet)

VDE, like OpenVPN is a piece of software which allows users to create virtual Ethernet links between both physical and virtual machines. VDE is listed as having the following features [VDE2005]

- VDE is Ethernet compliant.
- VDE is general, it is a virtual infrastructure that gives connectivity to several kinds of software components:

  Emulators/ virtual machines

  Real operating systems

  Other connectivity tools.
- VDE is distributed.
- VDE does not need administrator privileges to run.

Unlike OpenVPN, VDE gives the user control over the routing functions and uses the concept of 'Virtual Switches', 'Virtual Plugs', Virtual Wires' and 'Virtual Cables' to separate and simplify the various components the software emulates. Unfortunately, as in order to construct the virtual network, VDE requires the ability to create and modify tun/tap devices, meaning it too would have to be run within a virtual machine inside a users slice.

4.8 Ethernet Generic Routing Encapsulation (EGRE)

Ethernet Generic Routing Encapsulation enables Ethernet packets to be encapsulated using the GRE tunnelling protocol, and one particular implementation was developed for use in the Trellis software back in 2006 [BAV2006]. More recently, EGRE patches have been created to

give EGRE support to some of the latest linux kernels, however EGRE details are sparse. Using EGRE to create an overlay would however require the use of additional software to correctly label the encapsulated Ethernet frames with the routing path before sending.

4.9 OneLab2 D7.3 Routing in a Slice (RIAS)

Within the OneLab2 project, one team is looking at the issue of “routing in a slice” [LIS2009]. Using the PlanetLab Europe federation of PlanetLab, the RIAS team will identify and design extensions to the existing PlanetLab platform to enable experimental evaluation of technologies by providing virtual layer 2 links, customisable routing entries and efficient virtual to physical topology mapping tools. This portion of the OneLab2 project is still under development, however the ultimate aim is to integrate these extensions into the PlanetLab Europe build.

5 Customer Project: PSIRP

Within the PSIRP project, it is desirable to be able to create arbitrary network topologies in order to evaluate inter-domain technologies including rendezvous, forwarding, routing and caching. Although this could be performed within a network simulator such as ns2 (or ns3), the more complex the simulator, the more difficult it becomes to identify whether the simulator is an accurate representation of reality or not. Therefore, being able to experimentally evaluate the afore-mentioned inter-domain technologies is essential. While there exist current test beds which would allow the PSIRP project to evaluate the inter-domain technologies over the current internet, it is most useful if such a testbed existed (or could be created) to enable testing over networks (and network topologies) which emulate the potential future internet.

5.1 Use Cases

There are two use cases that are of interest to the PSIRP project, one relates to the evaluation of the global rendezvous solution and another to the evaluation of an intra-domain forwarding mechanism. We outline these use case in more detail in the following.

5.1.1 Rendezvous Evaluation

The current inter-domain rendezvous solution developed within the PSIRP project foresees the operation of regionalized rendezvous networks (RENEs) that are being interconnected via an interconnection overlay (IO) which can be provided by one or more providers. The operation within the RENEs is based on a tree mechanism, similar to DONA [DONA2007], while the IO utilizes a content-addressable network (CAN) approach. The relations of the RENEs as well as the usage of the overlay(s) are determined by commercial relations between the different providers, both infrastructure and overlay ones.

Evaluating the performance of the chosen approach, e.g., on level of stretch and delay, highly depends on the underlying commercial relations that build the basis for the peering and transit relations of the forwarding networks. Evaluating a deployment of the solution in the current Internet would merely yield in performance results that inherently reflect the current commercial relationship of ISPs in the Internet.

In order to truly evaluate the performance of the solution, one needs to be able to ‘emulate’ potential commercial relationships of ISPs – relationships that are unlikely to be found in the current Internet. For instance, an increased localization of traffic, utilizing heavy caching of data, is a desirable environment to evaluate the solution in. For this, one needs to be able to create autonomous system (AS) relations that reflect such changed business relations. The investigated techniques in this report, allowing for creating such AS relationships, are crucial for this evaluation. With that, arbitrary AS peering and transit relations would be set up, with the actual rendezvous solution operating in such AS environment. In other words, the nodes in the Planetlab environments would be the rendezvous-relevant nodes, while the tunnelling would reflect the changed AS relationships.

5.1.2 zFilter Evaluation

The mechanism described in [Jok2009] describes an intra-domain forwarding mechanism that is based on a zFilter flow identifier through a local AS. The mechanism allows for expressing a delivery tree within such local AS through a Bloom filter within a limited space header of a forwarded header. Forwarding decisions in intermediary nodes are based on simple logical AND operations, which can be performed at line speed.

As with all Bloom filter based approaches, false positives of such AND operations can occur, leading to false deliveries along the AS links. Hence, determining a rate for such false positives is a typical performance evaluation objective. Mechanisms to minimize such false positives have been proposed in [Jok2009], such as the introduction of *virtual link identifiers*, which combine certain AS paths into a virtual (single) link, ‘thinning’ out the Bloom filter space and therefore reducing the potential for false positives. Decisions for such virtual link identifier

creation can be based on real-time monitoring of such false positives but can also be based on other (resource optimization) objectives.

Evaluating the effectiveness of a variety of link identifier mechanisms as well as the evaluation of performance throughput in the core and metropolitan network deployment case are objectives for experimental verification, in addition to simulation and emulation. One can also envision the usage of emulated and experimental experiments.

5.2 Requirements

In order to experimentally evaluate PSIRP, there are some testbed and node requirements which need to be fulfilled.

Node requirements:

- 64bit capable processor (Multi core)
- 500GB Hard Disk (Recommended)
- 4GB+ RAM (ideally 800MHz+)
- One (or more) gigabit Ethernet NICs
- Jumbo frame support upto 8192 bytes
- FreeBSD 7.x 64bit edition

At present, the PSIRP prototype has only been developed on the FreeBSD operating system, however there are plans to increase the number of supported operating systems.

Testbed requirements:

- Definition of a custom Ethernet type
- Jumbo frame support up to 8192 bytes (Optional)
- Support for 200 or more Ethernet multicast groups
- Potential exclusive use of all Ethernet resources
- Virtual PC co-located to switch
- Access to VLAN tags (Optional)
- Measurement of delay on Ethernet (logical) layer and VLAN usage level

5.3 Addressing the Platform Problem

5.3.1 PlanetLab

The current PlanetLab testbed doesn't provide any network guarantees other than unfiltered, direct internet access (layer 3), which would require all PSIRP ethernet frames to be tunnelled over IP, and therefore limit the prototypes performance. Additionally, due to the lack of internet bandwidth guarantees, each PlanetLab node in a slice may have wildly different available network bandwidth; ranging from 400kbs to 1Gbs for those directly connected to particular NRENs.

With respect to network topology formation, as PlanetLab doesn't have an inbuilt tool for creating overlays, nor the required network interface permissions, the PSIRP prototype would most likely have to run within a virtual machine such as UML (User Mode Linux), to allow the creation of an overlay between all slivers in a users slice. As identified in section 4 there are several pieces of software available which can create an overlay network given the appropriate operating system and network interface.

Given the restrictions illustrated, it would require a fair amount of effort to modify the PlanetLab platform into an appropriate evaluation environment for the PSIRP prototype.

5.3.2 PlanetLab Europe

PlanetLab Europe would encounter the same issues as stated above for PlanetLab, as at present there are no additional PLE specific topology creation tools, nor stricter network (layer 3) guarantees.

5.3.3 RIAS

Assuming the RIAS project was successfully integrated back into the PlanetLab Europe nodes (potentially mid 2010), it would offer the ability to create virtual layer 2 connections between nodes (albeit over IP links), in addition to the use of topology management tools which would intelligently map a virtual network to the most appropriate area of the PlanetLab physical network in order to provide the best network and node performance.

5.3.4 PL-VINI

Running the prototype over PL-VINI, installed on PlanetLab nodes will be challenging, if not impossible. This is due to the layered virtual machines within which the PSIRP prototype would be running, all of which are now significantly outdated. While it would be possible to update the software (and indeed the operating systems) used by PL-VINI, a high level of understanding of the internals would be required to ensure that the underlying network was accessible through the virtual machine layering (at present it isn't). Assuming these problems could be overcome, PL-VINI would indeed provide virtual point-to-point Ethernet links between nodes, and a simple topology manager which when given a topology map would create the desired virtual links. This would rely upon the user intelligently choosing the mapping between the virtual and physical network. As such, the use of PL-VINI would not be ideal.

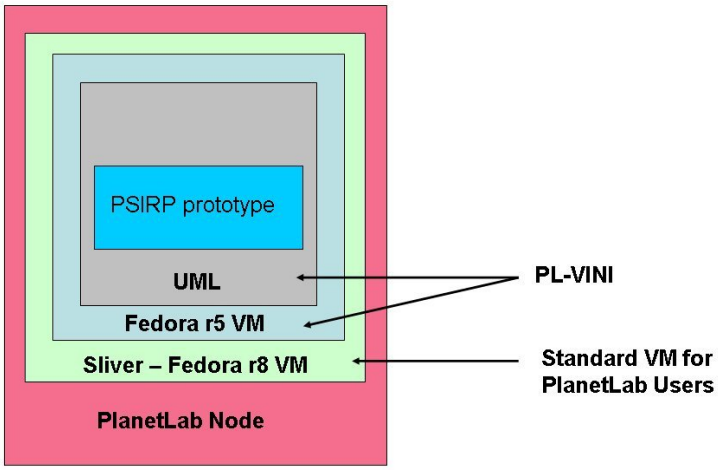

Figure 7: PSIRP prototype virtual machine layering on a PlanetLab node

5.3.5 VINI

Based on the published specifications of the Trellis software and the underlying network infrastructure, the VINI network provides an ideal network environment to run the PSIRP prototype over. As it provides layer 2 access over multi-gigabit speed connections, it would provide the best network performance to run the prototype over. Again, like PL-VINI, it has a simple topology manager which when given a topology map would create the desired virtual links (which relies upon the user intelligently choosing the virtual to physical network mapping).

5.3.6 FEDERICA

Out of all of the identified solutions, FEDERICA is the only environment which would enable the prototype to be run, without modification to either the evaluation environment or the prototype itself. The access to the raw Ethernet networks would enable realistic experimental evaluation over a potential future internet environment (without the overheads of Ethernet to IP tunnelling). Due to FEDERICA also being an EC FP7 project however, it is unlikely to be in a position to provide 'SuperUser' access as required for the prototype (as shown in Figure 4) until mid 2010. In addition, while there will be topology formation tools, as the project is still ongoing, the scope or features of the tools are currently unpublished.

5.3.7 NS-3

As ns-3 is a simulation environment, it would provide both a simulated Ethernet network, in addition to tools to create any network topology. As the PSIRP prototype wouldn't be able to run directly in the ns-3 environment as-is, it would require some degree of re-writing using the ns-3 API to create an appropriate model.

6 Proposed Solution

Having explored the current solution space for experimental inter-domain evaluation, the following solution (illustrated in Figure 8) has been proposed:

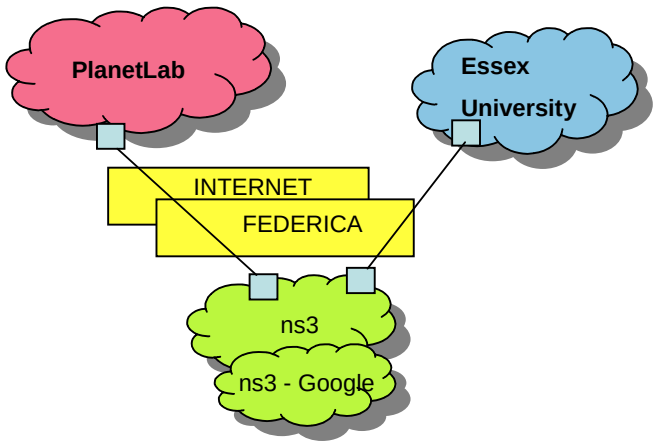

Figure 8: Proposed environment for testing the PSIRP prototype

It combines all three evaluation environments, simulation (ns-3), emulation (over PlanetLab) and a small general purpose testbed (at Essex University) to provide the best of all worlds while enabling a larger evaluation environment to be built than using just one of the environments alone.

The testbed, located at Essex University would enable the creation of large, high bandwidth tier-one equivalent ASes, with a direct connection to the raw Ethernet network provided by FEDERICA through a UK NREN.

The ns-3 simulator(s), connected either directly to FEDERICA or to the internet would allow the creation of a large number of smaller ASes, which could then be used to generate traffic to be injected into the PlanetLab and Essex University testbed.

The PlanetLab environment, connected over the internet, could be used to evaluate the performance of technologies over the current internet and its associated protocols, and would be a prime candidate for creating a number of smaller ASes (depending on the available node bandwidth) with real world traffic properties (such as delays, link failures etc).

6.1.1 Network access

The utilisation of both the Internet and FEDERICA would provide the best of both worlds for evaluating inter-domain technologies, as it enables protocols which don't run over IP (but perhaps over Ethernet) to be evaluated. It also gives the experimenter the flexibility to choose the network environment with the most appropriate underlying physical topology to partially (or fully) match the overlay topology.

6.1.2 Topology formation

The above solution would also provide a large amount of flexibility when creating a network topology, as it is expected that FEDERICA will contain it's own topology manager, ns-3 will by design enable any network topology, and there are an number of options for forming topologies over PlanetLab (some of which are outlined in section 4). It is through this flexibility that it is envisaged the three evaluation environments will be able to be connected seamlessly to create the large evaluation environment depicted in Figure 8.

7 References

[BAV2006] A. Bavier, N Feamster, M. Huang, L. Peterson, J. Rexford, *"In VINI Veritas: Realistic and Controlled Network Experimentation"*, in Proc. ACM SIGCOMM, pp 3-14, 2006

[BHA2008] S. Bhatia, M. Motiwala, W. Muhlbauer, Y. Mundada, V. Valancius, A. Bavier, N. Feamster, L. Peterson, J. Rexford, *"Trellis: A Platform for Building Flexible, Fast Virtual Networks on Commodity Hardware"*, ROADS, 2008.

[DONA2007] T. Koponen, M. Chawla, B. G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A Data-Oriented (and Beyond) Network Architecture," *ACM SIGCOMM Computer Communication Review*, vol. 37, issue 4, Oct. 2007, pp. 181-192

[FEA2006] N. Feamster, *"Customizable, Fast, Virtual Network Testbeds On Commodity Hardware",* Presentation at http://www.cercs.gatech.edu/iucrc08/material/feamster.ppt, 2006.

[FED2009] FEDERICA, *FEDERICA Homepage*, at: http://fp7-federica.eu [Accessed on 22 April 2009]

[FED2009a] FEDERICA fact sheet, *FEDERICA Homepage*, at: http://www.fp7-federica.eu/documents/20081124-Federica_lyon_poster_v6.1.pdf [Accessed on 22 April 2009]

[HUA2005] M. Huang, "VNET: PlanetLab Virtualized Network Access", Tech Rep. PDN-05-029, PlanetLab Consortium, 2005

[JOK2009] P. Jokela, A. Zahemszky, C. Esteve, S. Arianfar, P. Nikander, "*LIPSIN: Line speed Publish/Subscribe Inter-Networking"*, ACM SIGCOMM'09, 2009

[KOH2000] E. Kohler, R. Morris, B. Chen, J. Jannotti, M. Kaashoek, *"The Click modular router"*, ACM Transactions ON Computer Systems, vol 18, pp. 263-297, 2000

[LIS2009] J. Lischka, H. Karl, I. El Kayat, *"Deliverable 7.3, Routing-in-a-slice extension requirements for OneLab2"*, Deliverable D7.3, OneLab2, 2009

[NET2009] Linux Containers – Network Namespace (NetNS), "*Linux Containers Homepage*", at http://lxc.sourceforge.net/network.php [Accessed 22 April 2009]

[NS32009] NS-3, *Network Simulator 3 homepage*, at http://nsnam.org [Accessed 22 April 2009]

[NS32009a] NS-3 features presentation, *Network Simulator 3 homepage*, at http://www.nsnam.org/docs/ns-3-overview.pdf [Accessed 5 May 2009]

[ONE2008] OneLab2, *OneLab2 Homepage*, at: http://onelab.eu [Accessed on 17 April 2009]

[OPE2009] OpenVPN, *OpenVPN HomePage*, at http://openvpn.net [Accessed on 5 May 2009]

[PLA2008] PlanetLab Central API Documentation, *PlanetLab Homepage*, at http://www.planetlab.org/doc [Accessed 17 April 2009]

[PSI2008] PSIRP, *PSIRP Homepage*, at: http://wiki.psirp.org [Accessed on 17 April 2009]

[UML2008] UML, *User Mode Linux Homepage*, at http://user-mode-linux.sourceforge.net [Accessed 23 April 2009]

[XOR2009] XORP, *XORP Homepage*, at http://www.xorp.org [Accessed 14 April 2009]

[VDE2005] R. Davoil, *"VDE: Virtual Distributed Ethernet"*, in Proc. IEEE TRIDENTCOMM, 2005

[VSE2008] Linux VServer Project, *Linux VServer Homepage,* at http://linux-vserver.org [Accessed 22 April 2009]

8 Target Use Cases

8.1 PLE and FEDERICA

Suppose a researcher wanted to evaluate their experiment over PlanetLab and FEDERICA, and the experiment required a specific overlay topology and link/node characteristics. On PLE at present, there is no way to choose such nodes intelligently, or even create an overlay with a specific topology. On FEDERICA, it is expected that there will be an interface to input your topology file/network constraints and the network will be configured as appropriate.

As it is expected that FEDERICA will be federated with PLE, FEDERICA nodes should appear as nodes [with special characteristics] in the current PLE node list. It is uncertain how this would affect the use of the FEDERICA network configuration interface however.

Ideally, the user would be able to use a single configuration tool or interface which when provided with a topology description file would automatically configure the overlay to use both PLE and, where appropriate, FEDERICA nodes and links.

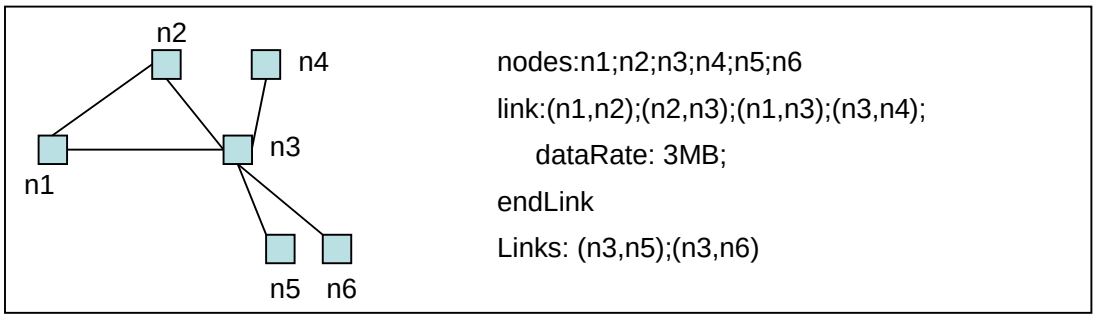

Figure 9: A possible example of a topology description language

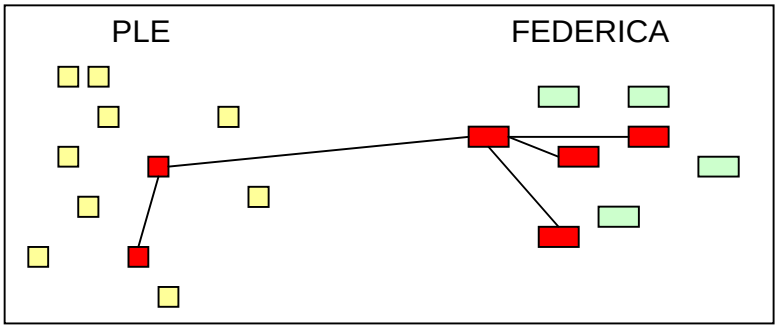

Figure 10: An example physical mapping of nodes and links based on the topology information in Figure 9