Routing Namespace Patterns

Christophe Jelger, Ghazi Bouabene, and Christian Tschudin

Technical Report CS-2008-001
University of Basel
September 2nd, 2008

Abstract

In this paper, we adopt the more and more accepted view that future inter-networks will consist of a rich and diverse collection of networks with multiple and potentially incompatible namespaces. In contrast to today's Internet, there would *not* exist a single global namespace federating all the networks. The main design challenge in such a *catenet* is to offer global connectivity over a heterogeneous environment *without* having to design dedicated translation and resolution mechanisms between all possible network-pairs. In this context, "inter-network routing" implies that given a certain destination name or address, one has to find the corresponding network among "the sea of networks" that compose the catenet. In this paper, we describe a strawman design for performing universal inter-network routing: our generic solution is based on *self-classifying names* using Unix regular expressions to both advertise and enable "namespace patterns" lookup across a heterogeneous inter-network.

Keywords:
Inter-network routing, regular expressions, catenet.

Computer Science Department, Computer Networks Group, Bernoullistrasse 16, CH – 4056 Basel Switzerland. Phone: +41 61 267 05 55 Fax: +41 61 267 05 59 URL: http://cn.cs.unibas.ch/

1 Introduction

Recently, the networking community has witnessed a renewed interest in the design of *clean-slate* network architectures (see e.g., [1, 2, 3, 4]) that could better support disruptive evolution and emerging network paradigms such as content-based, ad hoc, sensor, and delay-tolerant networking. A key motivation is to foster innovation by freeing researchers from providing backwards compatibility with the current Internet.

1.1 Global reachability without a global namespace

Somehow stepping back to the roots of inter-networking, many proposals [5, 6, 7, 8, 9, 10] favor a future inter-network architecture that explicitly supports the coexistence of independently managed networks which can use distinct and potentially incompatible naming and addressing schemes and routing protocols. The goal is to embrace heterogeneity at the network layer as a core architectural concept, which is a clear demarcation with the *hourglass* model of the current Internet.

A key design challenge in such a *catenet*[1] is to avoid imposing a federating "one-size-fits-all" network layer with a global address space [5, 6]. This restriction implies that routing must be capable of resolving a remote address or name across multiple and potentially incompatible networks with no guarantee that the communicating entities actually share a given network layer. A core requirement is to be able to add networks with new address and name types in a dynamic and non-disruptive way without having to globally modify existing protocols and applications and without having to re-design the inter-network routing framework.

In this paper, we describe an inter-network routing framework for catenets which does not require a global namespace for federating networks. In our approach, we add to each packet an extra header containing the source and destination addresses (or names) in a human readable form.

While there might be other implementations, what matters is that intermediate gateways can classify an address or a name (and route the corresponding packet) *based on its format*: i.e., the main activity is to identify to which namespace the address or name belongs in order to forward the packet across the catenet. To build their inter-network routing tables, gateways exchange "namespace patterns" encoded as Unix regular expressions: in contrast to traditional routing, this creates "routes for namespaces" rather than routes for specific addresses or names. Because of this specific operation, our proposal is called the Namespace Routing System (NRS).

1.2 Motivation

Before describing our inter-network routing framework, we want to briefly clarify our motivation in order to avoid mis-understandings about the objectives and scope of this work.

First of all, inter-network routing is clearly an old research topic which has lead to the development of the Internet as we know it today. In this work, we are not questioning past solutions and nor are we trying to solve a problem of the current Internet, although our solution could be used to route packets between IPv4 and IPv6 hosts. As part of the "clean-slate" research effort of the network community and with the re-emerging topic of designing heterogeneous inter-networks

[1] or "concatenation of networks", introduced in [11].

[5, 6, 7, 8, 9, 10], our objective is to explore new research directions for the design of future inter-network routing protocols.

On the technical side, we revisit the question of inter-network routing from the start and explore a new technique for solving this problem. In contrast to almost all the previous work in this area, our proposal does not rely on a global and overlaying namespace such as IP or HIP. While our proposal does create a routing overlay on top of heterogeneous networks, we do not superimpose an additional global namespace on top of the existing networks. In particular, we do not assign any global name or address to the individual hosts of the networks that form the catenet.

Following this introduction, the paper is organised as follows. In the next section, we present the design of our NRS proposal and detail various architectural aspects. Section 3 then discusses our proposal and outlines some differences with previous work; we also discuss open issues and future work. The paper is then concluded in Section 4.

2 Basic design

2.1 A concatenation of networks

We assume a networking environment where multiple heterogeneous networks are interconnected and form a catenet. Figure 1 shows an example with four different networks. Here the term 'network' refers to a distributed set of entities that use some common namespace(s) and protocols to communicate[2]. Examples include the IPv4-Internet, the IPv6-Internet, an Ethernet segment, a non-IP MANET, a sensor network, or an MPLS domain. As can be seen on Figure 1, the networks are inter-connected by "dual-stack" gateways which are hosts being able to communicate with at least two different networks.

Note that in this paper, the 'namespace' of a network refers to the set of symbols used to identify the *members* of the network. The term 'member' is by purpose vaguely defined and, depending on the network, could refer to e.g., a host, a set of hosts, a program, a file, a web page, etc. In the rest of this paper, we also use the term 'name' to refer to a particular element of a 'namespace', whether in today's networking terminology it is sometimes called a name (e.g., `www.example.com`) or an address (e.g., `10.1.2.3`).

2.2 Inter-network routing

The basic design of our NRS proposal is illustrated by Figure 2: the gateways participating in inter-network routing form a classic overlay of some arbitrary structure which could for example be manually configured. Each peering connection uses the corresponding underlaying network to "tunnel" routing messages between two running instances of the inter-network routing protocol. For example on this figure, the peering between the gateways A and B is based on some sensor network protocols while the peering between the gateways A and C is based on IPv6.

At the overlay level, the gateways exchange inter-network routing messages in order to advertise reachability information for the networks they are connected to. Note that any existing routing strategy such as link-state or distance-vector could be used: this has no influence on the fundamental design of our proposal. However, for resiliency reasons, multi-paths routing should be supported in order to provide alternative routes for each possible destination network. While we have so far described a classic overlay approach, the difference on which we will elaborate

[2]Note that we do not strictly restrict this definition to "traditional" network layer protocols.

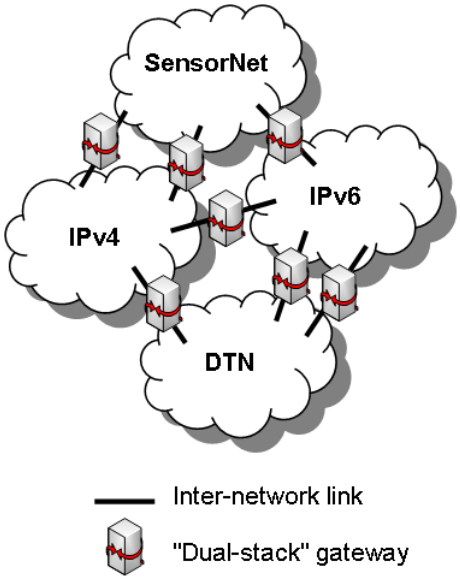

Figure 1: A basic catenet.

in the next section is on the organisation of the corresponding overlay namespace where we will make the overlay structure *disappear*.

If (for now) we assume that we have a way for describing "namespace patterns", each instance of the inter-network routing protocol can compute routing and forwarding information for all the networks attached to the catenet. For example, using some simple notation, the gateway A could derive the inter-network routing information shown by Table 1.

Note that this simple table differenciates *network count* and *hop count*. The network count gives the hop distance with respect to the number of networks while the hop count gives the hop

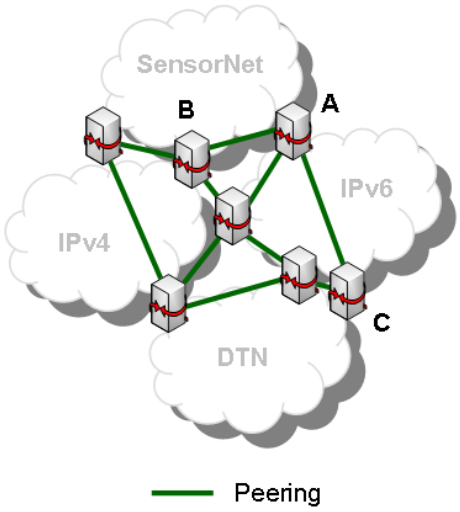

Figure 2: The NRS inter-network routing overlay.

Dest. Net	Next Hop	Net count	Hop count
SensorNet	-	0	0
IPv6	-	0	0
IPv4	B	1	6
DTN	C	1	8

Table 1: Routing table for gateway A.

distance with respect to the total number of hosts. In addition to the total length (in hops) of a path, the idea is to indicate how many networks will be traversed by packets following a certain route.

For example, from the gateway A the DTN network is at a one network-hop distance with say, an 8-hop distance between gateways A and C through the IPv6 network. Also note that for the networks attached to gateway A the two counters are set to 0 by convention.

2.3 Namespace naming and matching

While in itself the distribution of routing information is straightforward, the main challenge of our inter-network routing protocol is actually to be able to encode the "namespace patterns" of networks in a way such that, while looking up a forwarding table, it is possible to check whether a name belongs to some namespace. Our fundamental design assumption is that the latter is something that can easily be achieved with human-readable strings: for example, a network engineer can immediately identify that the string `"10.1.2.3"` is an IPv4 address while the string `"example.com"` is a DNS name. However, it is virtually impossible to classify byte patterns: for example, the 16-byte pattern

```
0x04736d747006756e6962617302636800
```

could either be the IPv6 address `473:6d74:7006:756e:6962:6173:0263:6800` or the DNS-encoded name `smtp.unibas.ch`, but it could also represent four IPv4 addresses or the hash of a filename. However, by just looking at the byte pattern it is impossible to derive what is actually encoded.

Following this observation, we have decided to work with the string representations of names as a way to have self-classifying names. Like others [12], we observe that using strings for representing names is very flexible as strings can easily be manipulated and modified. In particular, this permits to use Unix regular expressions to encode namespaces and check whether a name matches a certain namespace. Regular expressions are indeed a very mature and powerful tool to match a set of strings according to certain lexical rules. Moreover, they are supported by all major programming languages and especially by `C` and `C++` which are widely used for developing network protocols.

To illustrate the use of regular expressions for encoding a namespace pattern, let's consider the example of IPv6 addresses. If we ignore the compressed form `'::'` for encoding consecutive zeros, an IPv6 address consists of eight hexadecimal numbers separated by `':'` characters, with each number having one to four digits. With a regular expression, the namespace pattern of IPv6 addresses is easily expressed as

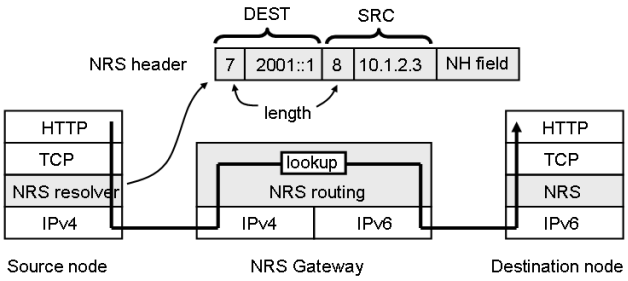

Figure 3: Inter-network forwarding.

```
^(\x{1,4}:){7}\x{1,4}$
```

with a simplified notation where `\x` is a short form for any hexadecimal digit `[[:xdigit:]]`. Note that the starting `'^'` and ending `'$'` characters forbid sub-matches: for example, they prevent this regular expression from matching addresses with more than eight hexadecimal numbers.

A key assumption of our design is that each namespace has a distinct representation format for which a selective regular expression can be written. While this may appear as a limitation, we observe that this requirement is fulfilled for almost all the major namespaces currently being used in the Internet (i.e., IPv4 and IPv6 addresses, DNS names, Ethernet addresses, HIP HITs). For "problematic" namespaces, one way (e.g. with universal resource identifiers [13]) of enforcing this requirement is to add a unique symbol to colliding namespaces in order to avoid any possible confusion. For example, URLs always contain the "protocol" prefix (e.g., `http:`, `ftp:`, `file:`) and SIP addresses typically contain the `sip:` prefix (e.g., `sip:bill@example.com`) to differentiate them from email addresses. Looking at networking architectures, it is worth mentioning that URIs are used in the DTN architecture (see [12]) to encode *endpoint identifiers*.

2.4 Packet forwarding

Inter-network packet forwarding is illustrated by Figure 3. To forward packets across a set of heterogeneous networks, we propose to use a dedicated end-to-end NRS header which contains the string-encoded source and destination names of the two communicating end nodes. In this example, each name in the header is preceeded by a fixed-length field (say, one byte) indicating the length in bytes of the following name. The header also contains a next-header (NH) field[3] indicating to which layer the payload is to be delivered (in that case, TCP). Finally, the header could also contain other fields such as hop count, checksum, payload length, etc, but in this paper we focus on the overall description of our proposal and let engineering details to be defined at a later stage.

To forward packets, each inter-network gateway must extract the destination name from the header and try to match it with all the namespace patterns of its forwarding table. Upon a match, the packet is simply forwarded to the next hop inter-network gateway. Depending on the lookup strategy and with a NRS forwarding table of size N, this would typically require N comparisons if one wants to find all matching entries or, if the first matching entry is selected, $N/2$ comparisons if packets are sent equally to all possible destination networks. Note that this can be optimised

[3] In today's IP or IPv6 headers the next header value is encoded with one byte with TCP=6 and UDP=17.

if forwarding entries are ordered according to the number of times they match names. That is, "popular" entries could be moved towards the start of the forwarding table in order to speed up the lookup process.

As in existing IP-based routers, forwarding could also be optimised with the design of a forwarding cache containing recently seen names. As such, only the first packet of a flow would require a costly lookup in the forwarding table: upon a match, the pair <name, entry> could indeed be temporarily stored in the cache to fast forward all the following packets. However while this optimisation is very easy to implement, the size of the fast forwarding cache could easily grow up exponentially as this mechanism essentially creates the equivalent of a virtual circuit[4] per name. Implementing such a fast forwarding mechanism hence requires further study to better understand the trade off between efficiency and storage cost.

Finally and although this is out of the scope of this paper, one could also use more advanced techniques for indexing regular expressions (i.e., forwarding entries) in an efficient way. For example in [14], Chan et al. propose a tree based structure for indexing a large number of regular expressions. With synthetic data sets, they show that the lookup time is reduced compared to sequential techniques.

2.5 Deployment model

Although this was not explicitly stated yet, our catenet routing system would only involve a small number of gateways compared to the total number of devices in the catenet. These machines would basically constitute the NRS overlay and would take part in the inter-network routing protocol and packet forwarding. Conceptually, we distinguish two roles: NRS lookup servers and NRS gateways.

As for the DNS, "client" hosts would only have to implement an inter-network resolver system which could query NRS servers to retrieve the name or address of the next hop NRS gateway towards some "alien"[5] destination. Note that the address or name of NRS servers inside the network of a given client host could typically be acquired dynamically via e.g., DHCP as is done today for configuring DNS server addresses. Another alternative is that each client host could simply be configured (e.g., also via DHCP) with an inter-network default route and NRS gateway to which it could send all the packets with "alien" destination names.

2.6 Routing policies

In the current Internet, inter-network (or actually inter-domain) routing is inseparable to routing policies. In the BGP world, ISPs collaborate and compete according to complex business relationships. In a catenet architecture, the same situation would probably also emerge. We believe that our proposal could be extended to support routing policies by adding some meta-information to the routing information being exchanged. For example, it would be possible to add AS Numbers or AS paths information in the inter-network routing messages in order to setup routing policies in a similar way to BGP. Actually, since our inter-network protocol is in essence built like "traditional" routing systems, routing policies could be supported with techniques similar to what is done in BGP routing.

[4]Note that the term "virtual circuit" is not fully adequate here as our proposal does not involve any signalling protocol: the caches along a path are indeed exclusively populated and maintained by the flow of data packets.

[5]i.e., located outside the sender's network and namespace.

Finally, deploying such an inter-network routing scheme would also require that namespace management issues are resolved. That is, network operators would have to decide who could advertise certain namespaces or potentially subparts of them like for example a subpart of the IPv4 address space. This relates to the current situation with BGP where ISPs filter routing information according to business relationships. For example on Figure 2, the gateway A could filter out some routing information to prevent that data is routed from the IPv6 Internet to the IPv4 Internet via the sensor network.

2.7 Network address translation

We note that that our proposal can also operate via network address translation (NAT) devices. Indeed, if traditional NAT is performed inside a certain network, one should simply make sure that the "source name" inside the NRS header is translated accordingly in order to allow the destination end-node to reply back and to prevent unwanted "name leaks". As for existing NAT, this would require NAT boxes to setup appropriate connection tracking state to properly NATify packets flowing in the two directions of a communication session. While this would require the development of dedicated extensions, there seems to be no critical issue that would prevent our proposal to operate across NAT boxes.

3 Discussion

To the authors' knowledge, there exist no inter-network routing proposal similar to our NRS protocol. The key concept of our proposal is that the "address family" of an address or a name is actually *implicitly* encoded in its string representation. Such self-classifying names fully avoid the need of having an explicit indication of the address family when forwarding a packet across a heterogeneous inter-network.

3.1 What we gain

First, our solution promotes evolution by allowing the catenet to gracefully and easily incorporate new networks and namespaces without revising standards for protocol numbers. In practice, adding a new network "type" to the catenet simply means injecting a new regular expression in the inter-network routing infrastructure. Second, the fact that our proposal does not introduce a global namespace removes all the cost, stress, and overhead associated with a global numbering authority. More important, this avoids placing an addressing scheme at the center of the inter-network layer and architecture. Third, our solution embraces heterogeneity and makes it explicitly visible at the inter-network level. In particular, it avoids the current approach where "everything must run over IP and IP must run over everything".

3.2 What we loose

With our solution, we accept to trade some lookup and forwarding performance for long term evolution and adaptation. Looking at the Internet, we observe that performance has constantly increased with adoption and deployment; in contrast, the first evolutionary step of the Internet towards IP version 6 has required an incredible amount of work, especially for developing IPv4 and IPv6 inter-networking mechanisms. From these observations, we argue that one should explicitly

"design for change" [15], while (as seen with various parts of the Internet) performance can always be increased at a later stage with adequate or dedicated hardware, optimizations, parallelism, etc.

3.3 Open issues

At this stage of the design we are still facing a number of open issues. First, our proposal requires that each regular expression advertised in the inter-network routing system is selective enough to avoid **false positives**. That is, a regular expression should not match names that do not belong to the corresponding namespace. While this can be achieved with careful engineering practices, we plan to introduce a mechanism to notify the sender when a packet cannot reach its destination because it somehow reached the wrong destination network. In essence, this "namespace unknown" message would be similar to today's ICMP "node unreachable" and "network unreachable" error messages. To then further prevent the use of the faulty forwarding entry, we envisage to add a "negative cache" which, in contrast to the fast forwarding cache, would contain <name, entry> records which are known to be false positives.

A second issue concerns **multiple matches**. When forwarding data and with imperfect regular expressions, it might be possible that a name is matched by more than one regular expression of the forwarding table. Since there is no straightforward "best- or longest-prefix match" granularity with regular expressions, we will have to add some extra mechanism to handle such a situation. A straightforward solution is to simply select one matching entry according to some pre-defined rules. For example, the shortest-path entry or the least-loaded entry could be selected. However, if the selected entry is a false positive, this simple selection system would temporarily lead to a dead end until an error message is received (as just described in the previous paragraph) and another matching entry is selected. An alternative and more costly solution is to temporarily explore all possible paths by duplicating packets until some feedback indicating if a path was successful is obtained. While appealing, this alternative could however generate a substential overhead.

A third issue common to all routing protocols is **security**. That is, one should not be able to compromise the entire inter-network system by injecting malicious regular expressions to e.g., create packet black holes or intercept communications. As with other routing protocols (for example BGP), these security issues can be reduced by setting up secured peerings between inter-network gateways, by filtering out routing information from untrusted sources, and by encrypting and/or signing the NRS header to prevent forgery. It is worth noting that despite its original application, the routing part of our NRS proposal basically operates like any traditional routing protocol and as such does not introduce security problems that are not already known.

3.4 Alternative design?

A seemingly straightforward solution for handling inter-network routing inside a catenet is to explicitly add a header field in order to flag all packets with the appropriate "address or network family" (i.e., typically a well-known value similar to today's AF_xyz family types in sockets) and have an inter-network routing system operating on this field. This could be done in each namespace in a "local" way.

This "enumeration" approach has been chosen with IP (v4, then v6), but also, for example, with the Ethernet ethertype and SNAP field. These cases are demonstrating well the shortcoming of that approach. First, it is virtually impossible to get new ethertype numbers and more and more difficult to get IPv4 estate; Second, each change or addition of an address format requires a revision

of the forwarding machinery, as we well know from IPv6. Another limitation is that namespace numbers have to be put into standards before vendors upgrade their devices for handling new addresses. Moreover, the adoption decision depends on the perceived potential of the new network family in the eyes of vendors, which might prefer to defend their current devices and business model, thus acting as an innovation-adverse force.

Another observation regarding inter-domain routing is that global coordination cannot be avoided: Each packet in a catenet needs to be classified at some point before it can be forwarded at the inter-network layer. In the case of the enumeration approach, either the sender or the first router in the inter-network layer has to add the family type of the destination name. Whatever the mechanisms to perform this classification, they must be globally agreed, distributed, and synchronously updated in a global manner and in an automated way.

However while both solutions require that one can classify names based on their format, our proposal does not require maintaining well-known and globally unique network family types. With our approach where regular expressions are "injected" in the inter-network layer in a coordinated but decentralized manner, we enable that other, more BGB-like negotiation models can be envisaged to manage the catenet. This is in line with our design "philosophy" of keeping our inter-networking layer as *transparent* as possible by e.g., not imposing any global and overlaying namespace to name and address the resources of the federated networks.

3.5 Related work

As already stated in the motivation section of this paper, inter-network routing is a very old research topic and the literature in this area is very large and diverse so we restrict this discussion on the prominent *design patterns* in this area.

Including the Internet and recent proposals using flat routing, most inter-networking schemes are based on a *global* namespace which federates heterogeneous networks. In this model, all hosts that want to be globally reachable must be assigned an address or name in the global namespace. While our proposal also introduces an overlaying "layer", it does not require that individual hosts are assigned an address or name in a new global namespace. This is a major demarcation compared to previous work.

There exist other techniques for performing inter-network routing without introducing a global namespace. For example, IBM's SNA supported inter-network routing by "proxy mapping" some addresses of a network into other networks. The major issue is that this solution requires careful management of namespaces and tight coordination between network operators. Furthermore, it makes resolution and routing quite complex as each inter-network gateway essentially operates like a NATP router. In contrast, our proposal allows each individual network to manage its namespace fully independently and it restrains coordination between network operators to inter-network routing matters (as with BGP).

Although not directly related to inter-network routing, one could see similarities between content-based networking [16], its commercial facets with *content distribution networks (CDNs)*, and our proposed framework, mainly because both proposals route packets based on strings. However, with a closer look and beside this superficial similarity, there are obvious differences: CDNs operate at the application layer and try to optimize content delivery by redirecting (and potentially rewriting) queries to nearby content caches. In contrast, our proposal operates at the (inter-)network layer and aims at routing packets across heterogeneous networks. While there might be similarities on how strings are matched during the packet forwarding process, the scope and

application of our proposal is clearly different than those of CDNs.

3.6 Future work

As an initial proof of concept, a first prototype of our NRS gateway has been implemented using the standard POSIX regular expression `C` library. We have run multiple tests with the regular expressions provided in the Appendix and have used as input a combination of randomly generated IPv4, IPv6, Ethernet, and Email addresses, and DNS names with up to 50,000 names for each namespace. On a Linux PC with a Dual-Core Pentium CPU operating at 2.4 GHz, we have measured an average lookup time of 1μs which corresponds to 1 million lookups per second. This is of course very far from the lookup speed of specialised devices like e.g., TCAM based routers which can perform a few hundred million lookups per seconds. However, we believe that with proper code optimisations and specialised hardware (e.g., including parallel forwarding engines), the lookup performance could be greatly increased if such a routing protocol is to be deployed in the future.

The implementation of the inter-network routing protocol is on-going and a first version should be publicly available by the end of 2008. We also plan to demonstrate the operation of our proposal with existing namespaces and networks, including IP4 and IPv6 gatewaying. Like most routing protocols, open problems such as security and trust, load balancing, and multi-path routing still need to be resolved. We however believe that there are sufficient existing solutions for solving (at least partially) these problems at a later stage.

4 Conclusion

In this paper, we describe an inter-network routing framework for routing packets across a set of heterogeneous networks using incompatible namespaces and protocols. In contrast to the Internet, our solution does not require that each host of the federated networks is assigned a globally unique name at the inter-network layer, and it does not require globally agreed network types. The key insight from our proposal is that it is sufficient that two namespace patterns be different, rather than they be identified by a global number assigned by some numbering authority.

With this approach, heterogeneity at the network layer is made explicitly visible at the inter-network level, hence making it possible to reach destination hosts and resources with the names and addresses they have been assigned in their local network. Our solution offers global reachability without imposing a global namespace, and hence promotes evolution and diversity rather than homogeneity (and ossification) at the network layer.

References

[1] T. Anderson, L. Peterson, S. Shenker, and J. Turner. Report of NSF Workshop on Overcoming Barriers to Disruptive Innovation in Networking, January 2005. Available at `http://www.geni.net/documents.php`.

[2] J. Crowcroft and P. Key. Report from the Clean Slate Network Research post-SIGCOMM 2006 Workshop. *ACM CCR*, 37(1):75–78, January 2007.

[3] FIND – Future Internet Design (FIND) - US National Science Foundation. At `http://www.nsf.gov/pubs/2006/nsf06516/nsf06516.htm`.

[4] Future Internet Research and Experimentation (FIRE) initiative - European Commission. At http://cordis.europa.eu/ist/fet/comms-fire.htm.

[5] D. Clark, R. Braden, A. Falk, and V. Pingali. FARA: Reorganizing the Addressing Architecture. In *Proceedings of FDNA'03*, August 2003. Karlsruhe, Germany.

[6] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield. Plutarch: an argument for network pluralism. In *Proc. of FDNA'03*, August 2003. Karlsruhe, Germany.

[7] S. Schmid, L. Eggert, M. Brunner, and J. Quittek. TurfNet: An Architecture for Dynamically Composable Networks. In *Proc. of WAC'04*, Oct. 2004. Berlin, Germany.

[8] G. Goodell, S. Bradner, and M. Roussopoulos. Building a Coreless Internet without Ripping out the Core. In *Proceedings of HotNets-VI*, Nov. 2005. College Park, USA.

[9] Ambient Networks - EU project (2006-2007) http://www.ambient-networks.org/.

[10] C. Jelger, C. Tschudin, S. Schmid, and G. Leduc. Basic Abstractions for an Autonomic Network Architecture. In *Proc. of AOC'07*, June 2007. Helsinki, Finland.

[11] L. Pouzin. A Proposal for Interconnecting Packet Switching Networks. In *Proceedings of EUROCOMP'74*, May 1974. London, UK.

[12] K. Fall and S. Farrell. DTN: An Architectural Retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836, June 2008.

[13] T. Berners-Lee, T. Fielding, and L. Masinter. RFC-3986 - Uniform Resource Identifier (URI): Generic Syntax, January 2005.

[14] C.-Y. Chan, M. Garofalakis, and R. Rastogi. RE-tree: an efficient index structure for regular expressions. *VLDB Journal*, 12(2):102–119, August 2003.

[15] D. Clark, K. Sollins, J. Wroclawski, and T. Faber. Addressing Reality: An Architectural Response to Real-World Demands on the Evolving Internet. In *Proc. of FDNA'03*, August 2003. Karlsruhe, Germany.

[16] A. Carzaniga, M.J. Rutherford, and A.L. Wolf. A Routing Scheme for Content-Based Networking. In *Proceedings of IEEE INFOCOM 2004*, March 2004. Hong Kong, China.

Appendix 1

In this appendix we provide some examples of (imperfect) regular expressions matching the most common namespaces of the Internet. In order to simplify the notation, we use \x for [[:xdigit:]], \d for [[:digit:]], and \a for [[:alnum:]]. The starting '^' and ending '$' characters are used to forbid sub-matches: for example, they prevent the regular expression of Ethernet addresses from partially matching some IPv6 addresses.

IPv4 addresses:
```
^((\d|[1-9]\d|1\d{2}|2[0-4]\d|25[0-5])\.){3}
(\d|[1-9]\d|1\d{2}|2[0-4]\d|25[0-5])$
```

IPv6 addresses (except compressed form):
```
^(\x{1,4}:){7}\x{1,4}$
```

Ethernet addresses:
```
^(\x{2}:){5}\x{2}$
```

DNS names:

```
^((\a(\-|_)?)+\.)+[a-zA-Z]{2,4}$
```

Email addresses:

```
^(\a(\-|_|\.)?)+@((\a(\-|_)?)+\.)+[a-zA-Z]{2,4}$
```