The Split Naming/Forwarding Network Architecture

Andreas Jonsson
Luleå University of Technology

Mats Folke
Luleå University of Technology

Bengt Ahlgren
Swedish Institute of Computer Science

August 20, 2003

Abstract

The evolution of the Internet is inhibited by the routing system's lack of ability to provide instant routes at host granularity. We argue that it is necessary to divide the addressing into naming and location and use separate mechanisms for the two functions. We present the split naming/forwarding architecture (SNF), which divides the network layer into naming and forwarding layers. The design rationale of SNF is discussed and we describe an implementation for deployment on top of current network infrastructures.

1 Introduction

In the traditional Internet architecture, the IP-number is used both to identify hosts and by the routing system to find paths for the purpose of delivering packets between any two of them. The work we present in this paper is based on the assumption that there is an inherent conflict between flexibility and scalability in a shortest path routing. This assumption is supported by examining the evolution of the Internet routing system, which reveals that there currently are two imposed constraints. First, IP-numbers are network topology dependent to enable route aggregation, tying them to specific subnetworks. Second, the reaction to topology changes is fairly slow.

To work around this problem it has been proposed to divide the addressing into two different functions: naming and location. Naming and location functions only differ by the performance constraints the mechanisms that implement them have. It must always be possible to contact a host via its *name*. Extra delay is acceptable to make the naming robust and flexible. On the other hand, packets sent between two locations should be forwarded along a path which is shortest in some respect. It is necessary to use two different routing systems to accomplish both at once.

In this paper we present the split naming/forwarding architecture (SNF). The network layer is conceptually split into two layers: *naming* and *forwarding*. Naming and forwarding are implemented by different routing systems matching the expected performance of respective function.

The contribution of this paper is that we identify the naming as the globally unifying function. As will be discussed, this does not mean that there must be a single global name space, although the protocols used for naming must be uniform. We discuss how SNF matches the tenets suggested by Clark *et al* [2], particularly *design for change* and *the centrality of the tussle space*. Although SNF does not explicitly support *controlled transparency*, there is a wide range of options to implement this within the architecture. Finally, we present a strategy for how SNF can be deployed on top of the current network infrastructure with minimum effort.

2 The split naming/forwarding architecture (SNF)

SNF divides the network layer in the protocol stack into *naming* and *forwarding* layers. Conceptually, packets are passed down from the transport layer to the naming layer. Before starting to communicate, an application has in some way obtained a *name* of a service or a host and configured the transport layer to send packets to that name. The naming layer establish a mapping from the name to an addressing unit which can be used by the forwarding layer to deliver the packet. We call this addressing unit a *locator*. Next, we discuss the properties of the forwarding and naming layers respectively.

2.1 Forwarding layer

The forwarding layer does not require globally uniform protocols or global address spaces. The only requirement on the forwarding layer is that it provides locators in some form and that the network is capable of delivering packets to the corresponding locations. This simplistic design imposes only a minimum set of constraints on implementations, which increases the probability that novel technologies can be introduced without violating the architecture.

The network at the forwarding layer may be divided into different *addressing domains* which may use different forwarding protocols and different locator spaces as long as there are *translation gateways* between the addressing domains. Communication between domains may be implemented through source routing, or by setting up state in the translation gateways (similar to NAT).

2.2 Naming layer

The naming layer is responsible for providing name to locator mappings. Two entities that want to communicate must agree on a common protocol at some level. Also, at least one of the entities must be able to name the other one. We believe that the naming layer implements the "least common denominator" and therefore the protocols at the naming layer must be specified in detail and be globally uniform. This does not exclude the possibility for different implementations and different name spaces as we discuss below.

Conceptually, the naming layer forwards packets to the location of a named entity. There are two different ways of accomplishing this: First, the naming system may announce the current locator of the corresponding host, which can be cached at the naming layer. If there is a cached locator for the name in question, the packet is sent to that locator (i.e., the naming layer requests that the forwarding layer sends the packet to the corresponding location). Second, there is a default locator for names not in the cache. Packets forwarded to the default locator reaches an *agent* that is part of a naming system. The naming system is an overlay network which can span different addressing domains and can perform name based routing or act as a distributed database for name-to-locator mappings.

It is desirable to have different naming systems for two reasons. First, naming is a potential tussle space [3], which may be avoided by allowing multiple name spaces. Second, it should be possible to replace the implementation of the naming system. In SNF, a host may be configured to access different naming system, which may be using different naming schemes and may be implemented differently.

A naming system may operate across several addressing domains. If source routing is used, the naming system must find the source routes; if locator overwriting is used, the naming system must establish state in the translation gateways on behalf of the hosts when a name from another addressing domain is used. Thus, when querying for a locator to a host in a different addressing domain, the naming system must be involved. Cached locators must be invalidated when moving between different addressing domains.

3 Implementing SNF

SNF is designed from the general ideas discovered by experimenting with a prototype, which we call Sing. The design of Sing we present here has been revised to match SNF. The major advantage of this design is its reuse of current infrastructure. IP is used at the forwarding layer and DNS at the naming layer.

3.1 Addressing units

Sing use three addressing units (shown in Table 1): the *name* and *locator*, and the *ephemeral correspondent identifier* (ECI). The transport layer may use the ECI combined with port numbers to identify different packet flows to replace static addresses.

Unit	*Implemented by*	*Purpose*
Name	FQDN	Identifies a node.
Locator	IP number	Denotes the position of node.
ECI	64-bit number	Identifies a packet flow.

Table 1: The three addressing units of Sing.

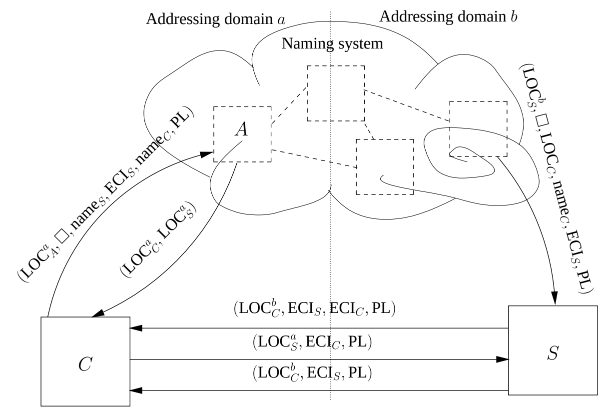

Figure 1: The arrows represent packets sent and the tuples display their contents. name_H is the name of host H, LOC^d_H is the locator for host H as seen from addressing domain d, ECI_H is the ECI chosen by some host for host H, and PL is the payload. The symbol □ indicates a default ECI used to indicate a new corresponding host.

3.2 Locator agent

Each host uses the services provided by a *locator agent* (A in Figure 1), which acts as the outmost part of the naming system (i.e. DNS). The locator agent maintains the current locators used by its hosts and is able to perform naming layer forwarding.

In order to communicate with the locator agent, the host must know the agent's locator which is set up during network initialisation, either statically, or dynamically (i.e., through neighbour discovery).

3.3 The communication procedure

Figure 1 illustrates a client-server scenario. C wants to communicate with S over the forwarding layer, but only knows the name of S. C must request two mappings, $C \rightarrow S$ and $S \rightarrow C$ from the naming system. Furthermore, C and S share their respective ECI during the initial phase. The locator agents acts as portals to the naming system. The naming system transparantly takes care of setting up inter-addressing domain communication if necessary. (I.e., by including source routes in the locators or by preparing translation gateways for locator overwriting.) Any locator changes are anounced through and with assistance of the naming system.

3.4 Compatibility with SNF unaware hosts

Using a DNS server connected to the SNF naming system, an unmodified IPv4 host can communicate with SNF hosts. In this case, the naming system sets up a mapping in the translation gateway and returns an IP-number to the IPv4 host which comes from the translations gateway's set of IP-numbers[1].

It is also possible to introduce an IPv4 compatibility mode where the naming system informs the SNF host about the source and destination IP-numbers seen by the IPv4 host. The SNF host may use these as ECI to communicate with the IPv4 host without the translation gateway having to recalculate transport layer checksums.

4 Related Work

FARA is a recent proposal by Clark *et al* [1], which decouples end-system names from network addresses. The main goal of FARA is the same as for SNF: a clear separation of identity and location. We argue that it is easier to adapt different kinds of networking technologies to the SNF architecture than to FARA for two reasons. First, since less requirements on locator management is put on the forwarding layer, less mismatch is likely to occur. Second, locators in SNF are specific to an addressing domain (they need not in FARA), which often makes it possible to use the already existing addresses for that technology as locators. It can, on the other hand, be argued that it is harder for the application to use multiple naming services in SNF.

IPNL [4] is another proposed Internet architecture inspired by NAT-technology. IPNL is focused on alleviating the address depletion problem associated with IPv4 and being a more cost-efficient solution than IPv6 by reusing as much as possible of the current IPv4 infrastructure. This is achieved in a NAT-fashioned way by partitioning the global network into smaller realms where the address space can be reused. IPNL also uses FQDNs to provide an identity to each host in the network and it also intro-

[1] The translation gateway must have multiple IP-numbers to allow each IPv4 host to simultaneously communicate with multiple hosts on the outside.

duces a new protocol layer between the network and the transport layer.

*i*3 (Internet Indirection Infrastructure) [5] is a suggestion to introduce a layer of indirection, to separate sending from receiving. *i*3 is an overlay capable of routing on names. SNF was designed to allow *i*3 and similar naming systems to be used.

5 Conclusions

A network architecture can be viewed as a set of restrictions which must be obeyed to allow all system components to operate together. Functions that can be implemented in ways that is not prohibited by the architecture should not break the system. A network architecture designed to evolve must allow new unforeseen functions. This can only be accomplished by an architecture that imposes as little restrictions as possible on the networks.

In this paper we have presented the SNF architecture which is designed according to the above philosophy. The SNF design also provides a framework and terminology to further discuss the need for separating naming and forwarding. SNF conceptually separates naming from location by dividing the network layer into a naming layer and a forwarding layer. SNF puts little requirements on the forwarding layer in order to easily utilise existing networking technologies for the forwarding layer. The naming layer, on the other hand being a unifying layer, needs to be specified in more detail as it provides functionality to map names to locators. We have also taken the approach to make the naming layer itself capable of forwarding data packets using the name of an end-point. Forwarding using the locators and the forwarding layer can thus be seen as an optimisation.

References

[1] David Clark, Robert Braden, Aaron Falk, and Venkata Pingali. FARA: Reorganizing the addressing architecture. In *ACM SIGCOMM, Workshop on Future Directions in Network Architecture (FDNA-03)*, Karlsruhe, Germany, August 2003.

[2] David D. Clark, Karen Sollins, John Wroclawski, and Ted Faber. Addressing reality: An architectural response to real-world demands on the evolving internet. In *ACM SIGCOMM, Workshop on Future Directions in Network Architecture (FDNA-03)*, Karlsruhe, Germany, August 2003.

[3] David D. Clark, John Wroclawski, Karen R. Sollins, and Robert Braden. Tussle in Cyberspace: Defining Tomorrow's Internet. In *Proceedings of ACM SIGCOMM*, Pittsburgh, PA, USA, August 2002.

[4] Paul Francis and Ramakrishna Gummadi. IPNL: A NAT-extended Internet architecture. In *Proceedings of ACM SIGCOMM*, San Diego, CA, USA, August 2001.

[5] Ion Stoica, Daniel Adkins, Shelley Zhuang, Scott Shenker, and Sonesh Surana. Internet Indirection Infrastructure. In *Proceedings of ACM SIGCOMM*, Pittsburgh, PA, USA, August 2002.