TKK Dissertations 133
Espoo 2008

SECURE MOBILITY AT MULTIPLE GRANULARITY LEVELS OVER HETEROGENEOUS DATACOM NETWORKS

Doctoral Dissertation

Jukka Ylitalo

Helsinki University of Technology
Faculty of Information and Natural Sciences
Department of Computer Science and Engineering

TKK Dissertations 133
Espoo 2008

SECURE MOBILITY AT MULTIPLE GRANULARITY LEVELS OVER HETEROGENEOUS DATACOM NETWORKS

Doctoral Dissertation

Jukka Ylitalo

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Faculty of Information and Natural Sciences for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 14th of November, 2008, at 12 noon.

Helsinki University of Technology
Faculty of Information and Natural Sciences
Department of Computer Science and Engineering

Teknillinen korkeakoulu
Informaatio- ja luonnontieteiden tiedekunta
Tietotekniikan laitos

Distribution:
Helsinki University of Technology
Faculty of Information and Natural Sciences
Department of Computer Science and Engineering
P.O. Box 5400
FI - 02015 TKK
FINLAND
URL: http://www.cse.tkk.fi/
Tel. +358-9-451 3228
Fax +358-9-451 3293
E-mail: jylitalo@cc.hut.fi

© 2008 Jukka Ylitalo

ISBN 978-951-22-9530-2
ISBN 978-951-22-9531-9 (PDF)
ISSN 1795-2239
ISSN 1795-4584 (PDF)
URL: http://lib.tkk.fi/Diss/2008/isbn9789512295319/

TKK-DISS-2499

Yliopistopaino
Helsinki 2008

HELSINKI UNIVERSITY OF TECHNOLOGY P. O. BOX 1000, FI-02015 TKK http://www.tkk.fi	ABSTRACT OF DOCTORAL DISSERTATION

Author Jukka Ylitalo

Name of the dissertation

Secure Mobility at Multiple Granularity Levels over Heterogeneous Datacom Networks

Date of manuscript 14.9.2007	Date of the dissertation 14.11.2008

☐ Monograph ☒ Article dissertation (summary + original articles)

Department	Department of Computer Science and Engineering
Laboratory	Telecommunications Software and Multimedia Laboratory
Field of research	Computer Science
Opponent(s)	Dr. Bengt Ahlgren
Supervisor	Professor Antti Ylä-Jääski
Instructor	Dr. Pekka Nikander

Abstract

The goal of this thesis is to define a set of changes to the TCP/IP stack that allow connections between legacy applications to be sustained in a contemporary heterogeneous datacom environment embodying multiple granularities of mobility. In particular, the thesis presents a number of solutions for flow mobility, local mobility, network mobility, and address family agility that is mobility between different IP versions. The presented mobility solutions are based on the so-called identifier-locator split approach. Due to the split, the mobile and multi-homed hosts that employ the presented solution are able to simultaneously communicate via multiple access networks, even supporting different IP versions and link layer technologies.

In addition to the mobility solutions, the thesis also defines a set of weak and strong security mechanisms. They are used to protect the mobility protocols from redirection, Denial-of-Service (DoS), and privacy related attacks. The defined security mechanisms are tightly bound to the presented mobility architecture, providing alternative ways to optimize mobility management signalling. The focus is on minimizing end-to-end signalling latency, optimizing the amount of signalling and optimizing packet forwarding paths. In addition, the architecture provides identity and location privacy for hosts.

The presented work defines one specific kind of engineering balance between the security, privacy, and efficient mobility signalling requirements. This thesis indicates that the added security, indirection, backwards compatibility, and inter-operable mobility solutions can overcome several of the current TCP/IP restrictions. The presented mobility architecture also provides a migration path from the existing Internet architecture to a new cryptographic-identifier-based architecture.

Keywords	IPv4, IPv6, Security, Privacy, Multi-Homing, Flow Mobility, Local Mobility, Network Mobility	
ISBN (printed)	978-951-22-9530-2	ISSN (printed) 1795-2239
ISBN (pdf)	978-951-22-9531-9	ISSN (pdf) 1795-4584
ISBN (others)		Number of pages 190 p. + app. 140 p.
Publisher	TKK, Telecommunications Software and Multimedia Laboratory	
Print distribution	TKK, Telecommunications Software and Multimedia Laboratory	

☒ The dissertation can be read at http://lib.tkk.fi/Diss/2008/isbn9789512295319/

TEKNILLINEN KORKEAKOULU PL 1000, 02015 TKK http://www.tkk.fi	VÄITÖSKIRJAN TIIVISTELMÄ

Tekijä Jukka Ylitalo

Väitöskirjan nimi

Secure Mobility at Multiple Granularity Levels over Heterogeneous Datacom Networks

Käsikirjoituksen jättämispäivämäärä 14.9.2007	Väitöstilaisuuden ajankohta 14.11.2008

☐ Monografia ☒ Yhdistelmäväitöskirja (yhteenveto + erillisartikkelit)

Osasto	Department of Computer Science and Engineering
Laboratorio	Telecommunications Software and Multimedia Laboratory
Tutkimusala	Computer Science
Vastaväittäjä(t)	Dr. Bengt Ahlgren
Työn valvoja	Professor Antti Ylä-Jääski
(Työn ohjaaja)	Dr. Pekka Nikander

Tiivistelmä

Tämä väitöskirja määrittää joukon muutoksia TCP/IP-pinoon. Muutokset koskevat liikkuvuudenhallinnan eri asteita. Ne mahdollistavat yhteyksien ylläpidon olemassa olevien sovellusten välillä nykyaikaisissa ja heterogeenisissä tietoliikenneverkoissa. Väitöskirja esittää joukon ratkaisuja datavuon, verkkojen ja osoiteperheiden liikkuvuudenhallintaan. Viimeisin näistä tarkoittaa liikkuvuutta IP-versioiden välillä. Lisäksi työssä esitetään ratkaisu alueelliseen liikkuvuudenhallintaan. Väitöskirjassa kaikkien isäntäkoneiden on tuettava esitettyjä liikkuvuudenhallintaratkaisuja. Esitetyt ratkaisut pohjautuvat jakoon tunniste- ja paikkatiedon välillä. Jako mahdollistaa, että liikkuvat isäntäkoneet kykenevät samanaikaisesti kommunikoimaan useiden liittymäverkkojen kautta tukien sekä eri IP-versioita että linkkikerroksen tekniikoita.

Liikkuvuudenhallintaratkaisuiden osana väitöskirja määrittää joukon heikkoja ja vahvoja turvallisuusmekanismeja. Ne suojaavat yhteyskäytännöt uudelleenohjaukseen, palvelunestoon ja yksityisyydensuojaan liittyviltä hyökkäyksiltä. Määritetyt turvallisuusmekanismit on tiukasti sidottu työssä esitettyyn liikkuvuudenhallinta-arkkitehtuuriin, mikä tarjoaa vaihtoehtoisia tapoja optimoida viestintää isäntäkoneiden välillä. Painopiste on viestinnän viiveen ja määrän minimoinnissa sekä pakettien reitin optimoinnissa. Lisäksi arkkitehtuuri tarjoaa yksityisyydensuojan tunniste- ja paikkatiedolle.

Väitöskirja määrittää tarkasti yhden tasapainoisen ratkaisun turvallisuuden, yksityisyydensuojan ja tehokkaan liikkuvuudenhallintaviestinnän vaatimusten välillä. Työn tulokset osoittavat, että lisätyt turvallisuusmekanismit, esitetyt sidonnat tunnistetietojen välillä, yhteensopivuus vanhojen sovellusten kanssa ja yhteensopivat liikkuvuudenhallintaratkaisut poistavat yhdessä useita nykyisiin TCP/IP-yhteyskäytäntöihin liittyviä ongelmia. Esitetty liikkuvuudenhallinta-arkkitehtuuri tarjoaa myös yhdenlaisen siirtymätien nykyisestä Internet-arkkitehtuurista uuteen kryptograafisiin tunnistetietoihin perustuvaan arkkitehtuuriin.

Asiasanat	IPv4, IPv6, Security, Privacy, Multi-Homing, Flow Mobility, Local Mobility, Network Mobility		
ISBN (painettu)	978-951-22-9530-2	ISSN (painettu)	1795-2239
ISBN (pdf)	978-951-22-9531-9	ISSN (pdf)	1795-4584
ISBN (muut)		Sivumäärä	190 p. + app. 140 p.
Julkaisija	TKK, Telecommunications Software and Multimedia Laboratory		
Painetun väitöskirjan jakelu	TKK, Telecommunications Software and Multimedia Laboratory		

☒ Luettavissa verkossa osoitteessa http://lib.tkk.fi/Diss/2008/isbn9789512295319/

Acknowledgements

I wish to express my gratitude to my mentor and friend Dr. Pekka Nikander for his inspiration and guidance during this work. I want to thank Pekka for his encouragement, support, and insights. I believe that one of the most valuable lessons that I have learned from him is the way of doing research.

I am thankful to my supervisor Professor Antti Ylä-jääski for his advice, feedback and support during this work. I want to thank him and TML laboratory for providing me a research chamber at TKK for finalizing this thesis.

I am deeply grateful to the pre-examiners Dr. Tuomas Aura and Professor Petri Mähönen for reading the manuscript. I owe my gratitude to Dr. Göran Schultz for proofreading the language in the manuscript.

I wish to express my gratitude to managers, Raimo Vuopionperä, Tony Jokikyyny, Jari Arkko, Bengt Sahlin, Stefan von Schantz, Jan Höller, Eva Fogelström, and Olle Viktorsson, at Ericsson for giving me opportunity to write this thesis. Without their support the work would not have been possible.

The funding for this work has been received from Ericsson, TEKES, and EU projects. Namely, from the national TEKES GIGA technology program and EU's sixth framework Ambient Networks project.

Several people have influenced this thesis. I want to thank all of you for your involvement and help with this work.

I want to thank Jan Melén for mentoring me in numerous theoretical and implementation related issues. I also want to thank Patrik Salmela for his unselfish support, feedback, and time spent for elaborating several research ideas together. Thanks to Jan and Patrik for their persistent implementation effort. I also want to thank Petri Jokela for his feedback, excellent project management, time scheduling, and thesis related standardization efforts.

I have had a privilege of working with Teemu Rinta-aho, Kristian Slavov, Jorma Wall, Vesa Torvinen, Göran Schultz, Tero Kauppinen, Martti Kuparinen, Heikki Mahkonen, Sebastien Pierrel, Fabrice Colliot, Vesa Lehtovirta and Björn Melén.

In addition, I want to thank Yuri Ismailov, András Méhes, Gonzalo Camarillo,

Göran Selander, Jani Hautakorpi, Hannu Kari, Mats Sågfors, Mikko Särelä, Christian Vogt, Teemu Koponen, Miika Komu, Hannes Tschofenig, Henrik Petander, Antti Tuominen and Jaakko Laine for fruitful discussions around the topic.

My family, parents, sister's family and friends have been a source of hope and belief at hard times.

I started writing the summary of this thesis soon after my lovely daughter Viola was born. The best time of the days have been the time with Viola and my beloved wife Jutta. Lastly, I thank with all my heart Jutta for all her understanding and support.

Jorvas, September 2007

Jukka Ylitalo

Contents

Appendix A State Establishment at SPINAT Nodes

List of Abbreviations

ACL Access Control List

AID Application Layer Identifier

API Application Programming Interface

ARP Address Resolution Protocol

BEET Bound End-to-End Tunnel mode

BEETIN Bound End-to-End Tunnel mode for Intermediate Nodes

CIDR Classless Inter-Domain Routing

DHT Distributed Hash Table

DNS Domain Name System

DoS Denial-of-Service

ESP Encapsulating Security Payload

FQDN Fully Qualified Domain Name

HI Host Identity

HIP Host Identity Protocol

HIT Host Identity Tag

HMAC Keyed-Hash Message Authentication Code

HTTP Hypertext Transfer Protocol

IETF Internet Engineering Task Force

IPsec Internet Protocol Security

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

IRTF Internet Research Task Force

LPM Longest Prefix Matching

LSI Local Scope Identifier

MIP Mobile IP

MIPv6 Mobile IPv6

MitM Man-in-the-Middle

MTU Maximum Transmission Unit

NAT Network Address Translation

NIC Network Interface Controller

QoS Quality of Service

P2P Peer-to-Peer

PKI Public Key Infrastructure

PF Protocol Family

SA Security Association

SPI Security Parameter Index

SPINAT Security Parameter Index multiplexed Network Address Translation

SPKI Simple Public Key Infrastructure

TCP Transmission Control Protocol

TLI Transport Layer Identifier

TLV Type-Length-Value

UDP User Datagram Protocol

VIID Virtual Interface Identifier

List of Figures

1 Introduction

An essential problem with the existing mobility and multi-homing approaches is that they do not offer integrated security and mobility management mechanisms for different mobility dimensions such as host mobility, flow mobility, network mobility and local mobility (mobility terminology is explained in [118]). Due to the design of the existing TCP/IP architecture, it is difficult to achieve flexible and granular mobility solutions without modifying the host stack [185]. Here, *granularity* means the scope of a hand-off to which a mobility mechanism can be adjusted. In this thesis, the author defines a set of required modifications to the host stack to achieve a flexible and scalable mobility architecture. The presented architecture integrates multiple authentication, privacy protection, key-management and mobility-management mechanisms together in a seamless way (security and privacy terminology is explained in [148]). The integrated security and mobility protocols presented in this thesis reduce significantly the mobility and key management signalling and hand-off latency without causing substantial trade-offs between security and mobility requirements. The following sections define a number of basic terms used throughout the rest of the thesis. While the terminology used in this thesis largely adheres to the way these terms are typically used in the literature, the definitions are included here to achieve the desired level of preciseness.

1.1 Heterogeneous Datacom Networks

In a heterogeneous data communication (datacom) system, hosts are connected to a network via attachment points employing different kinds of link technologies. A datacom system can roughly be divided into software and physical devices. The physical devices are connected to each other via *links* providing physical communication media for software. *Hosts* are devices that work as endpoints for routing paths. The *routing paths* consist of multiple links that are connected to each other via packet forwarding devices [96].

Processes are instances of programs that are executed at hosts. They establish communication channels that are called connections to communicate with each other. This is also known as *interprocess communication* (IPC) [193]. A *connection* consists of a set of protocol machine instances that are located on the packet forwarding

path between processes and have tightly coupled states (see also [89]). A finite state machine keeps each *protocol machine* in a correct state based on the phase of a connection [9][43][89]. Moreover, a *protocol* denotes a set of rules that define a communicating policy between two nodes [15]. The hosts may even spread connections over the multiple link technologies simultaneously. In this thesis we are primarily interested in wireless link technologies. The availability of these link technologies is bound to geographical locations. Typically, high-bandwidth access is offered in urban areas, while rural areas may be covered by radio technologies providing wider-ranging coverage with limited bandwidth.

Hosts may send and receive packets but they cannot forward packets between links in the same way as intermediate nodes do [118]. An *intermediate node* denotes a device that is on a routing path between hosts. Both hosts and intermediate nodes are just called *nodes* in the context of this thesis. Depending on the role of node in the system, the node may either have a fixed or a dynamic attachment to the network. An *attachment point* is a place where a node's network interface is attached to the network [96]. A network interface is also known as a Network Interface Controller (NIC). The parts of the network providing attachment points for hosts are called *access networks*.

The *Internet protocol (IP)* is used for transmitting blocks of data between hosts over globally-interconnected datacom networks, i.e., the Internet. The datacom networks may support different IP versions, namely, Internet Protocol version (IPv4) [81] and Internet Protocol version 6 (IPv6) [163]. An IP address is used for identifying both a node and its topological location in the current Internet [68]. The shortage of the IPv4 addresses coupled with privacy requirements has led several operators, companies, and individual end users to decouple their networks from the Internet with intermediate nodes called Network Address Translation (NAT) [136] devices. NAT devices map IP addresses used in the private networks to global IP addresses used in the Internet. It seems that when the IPv4 address space runs completely out, the role of the IPv6 protocol will be increased in the operator networks. A problem will then be inter-operation between IPv4 and IPv6 protocols.

The IPv4 and IPv6 protocols are part of the so-called TCP/IP stack that consists of the physical, link, network, transport and application layers [185]. Typically, a specific end-to-end routing path consists of several combinations of link and network layer technologies in a heterogeneous datacom network. The main purpose

of a mobility-management mechanism is to keep the end-to-end connections alive independent of the underlying network technology. In this thesis, a *mechanism* denotes specific technical processes used to implement well-specified functions, such as mobility-management or key management (explained later). In a basic case, *mobility* denotes a hand-off [118] procedure where a network interface changes its point of attachment to the network. Typically, the different parts of a mobility-management mechanism are distributed between multiple nodes. To synchronize the distributed parts of the mechanism, mobility-management protocols are used for data communication between nodes.

1.2 Research Methodology

> *"When you step into an intersection of fields, disciplines, or cultures, you can combine existing concepts into a large number of extraordinary new ideas."*, F. Johansson [56].

In this thesis, the focus has been on removing obstacles in the TCP/IP host stack design to achieve a mobility architecture that integrates multiple mobility dimensions together. The dimensions consists of host mobility, flow mobility, local mobility, network mobility and address-family agility as illustrated in Figure 1.1, on page 22. Additionally, the author has been looking for countermeasures to some security vulnerabilities. This has resulted in a set of research problems, and in a set of conflicting goals. The author has been looking for the right engineering balance in the intersection of the different mobility and security requirements. Finally, the different kinds of security and mobility concepts are combined together in a novel and seamless way resulting in a new kind of mobility architecture. An essential part of the research methodology has also been testing and verifying the architecture using a real code[1].

1.3 Mobility Dimensions

This thesis defines secure and efficient mobility-management signalling in different kinds of mobility scenarios in heterogeneous datacom networks. From the protocol-design point of view, mobility can be divided into different categories based on the

[1] *"Code is Law."*, L. Lessig [111].

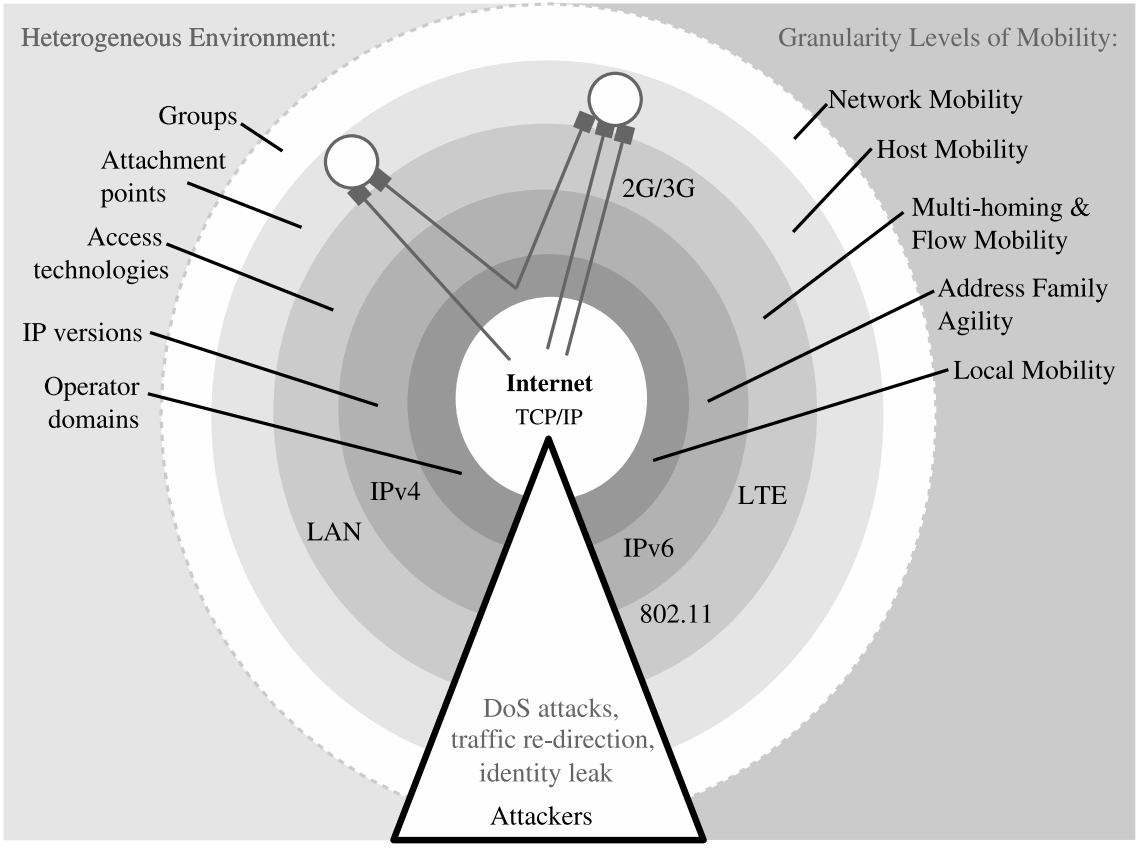

Figure 1.1: Heterogeneous datacom network environment.

scope of the hand-off. Figure 1.1 illustrates the relationship between the different granularity levels of mobility and the network environment.

Typically, a geographical movement of a host or a change in the experienced Quality of Service (QoS) triggers a hand-off between access networks. This is called *host mobility* [118]. *QoS* describes some characteristics of communication that a user of a host desires [89]. In addition, a multi-homed host that is simultaneously attached to multiple access networks may divide its connections between its network interfaces. *Flow mobility* makes it possible to move data communication between network interfaces even on a per connection basis. In addition, the traffic flows can be moved between IP versions. This kind of IP *address-family agility* enables communication between different IP versions.

A group of hosts can benefit from *network mobility* when the group is reachable via the same mobile router, and the hosts in the network are moving to the same

geographical direction [118]. The (mobile) routers connect links together and forward IP packets between hosts using the IP addresses carried in the packets for *routing* decisions [15]. In all cases, *local mobility* mechanisms can be used to minimize hand-off latency and to offer location privacy for hosts that are moving in a restricted region; i.e., typically in the same operator domain [118]. Local mobility, sometimes also called *micro-mobility*, denotes mobility signalling between the mobile node and an intermediate node that is located in the local operator network while *macro-mobility* results in mobility signalling between end hosts [118]. The *hand-off latency* denotes the time spent by detaching from the previous attachment point and receiving the first packets from existing connections after attaching to the new attachment point.

1.4 Identifier, Identity and Name

In the current Internet, most security problems are related to mechanisms that associate identifiers and identities with each other. The relationship between identifiers and identities is illustrated in Figure 1.2, on page 24. According to RFC 4949 an identity is "the collective aspect of a set of attribute values (i.e., a set of characteristics) by which a system user or other system entity is recognizable or known". An *identifier* represents a specific identity in the network and it is used to distinguish an identity from all others [148]. However, this definition is too broad for the purposes of this thesis. It is necessary to consider separately *non-verifiable and verifiable identities* based on the type of the association between the identifier and the identity. The difference is that a non-verifiable identifier, i.e., a *name* is associated to an identity using a non-cryptographic mechanism, while a verifiable identifier, i.e., a cryptographic one can be associated with an identity using a cryptographic mechanism. In other words, a name is associated to a *non-verifiable identity* using a non-cryptographic mechanism, while a cryptographic identifier is provable associated to a *verifiable identity* [131] by cryptographic means. It is good to notice that an identifier that is associated to more than one identity may cause misidentification. For example, if a single Mobile IP's [26] home address would be bound to two different mobile hosts, packets could be forwarded to the care-of addresses of both mobile hosts. Therefore, it is important to have a one-to-one mapping between identifiers and identities.

In a way, names are a subset of identifiers, like hosts are a subset of nodes. For

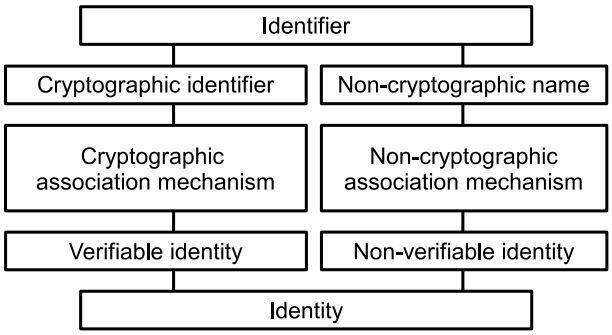

Figure 1.2: The relationship between the identity, identifier and name concepts.

example, a legacy IPv6 address [163] is a name but a Cryptographically Generated Address (CGA) [175] is an identifier that can be associated to a private key using public-key cryptography [14]. A private key of a public-key pair [14] and a root value of a one-way hash chain [110] are examples of verifiable identities. The *root* is an initial value of a hash chain while the last generated value of a hash chain is called the *anchor* value. The identity, identifier and name concepts are discussed in detail in publication [P2].

1.5 Key-management Mechanism

The earlier mentioned cryptographic associations between identifiers and identities can be used with key management mechanisms. *Key management* controls the process of creating and updating cryptographic keying material at nodes and sharing the keying material and associated information between the communicating nodes [148]. A *key exchange* protocol is an essential part of the key-management mechanism. It is used for generating keying material and distributing the material between hosts for establishing Security Associations (SAs). *Security association* denotes a cryptographic coupling of two identities using a mechanism that provides a way for protecting data communication between the identities [148]. *Re-keying* procedure is another part of the key-management mechanism that replaces the used cryptographic keying material with new material.

In this thesis, the author defines a set of new key exchange mechanisms and utilizes parts of the Internet Protocol Security (IPsec) security protocols and databases.

The original IPsec security architecture consists of security protocols, security associations, key management, authentication and encryption algorithms [166]. *Authentication* denotes a cryptographic mechanism used for verifying that an identifier is associated to an identity in a provable way. From the key-management point of view, the author replaces the Internet Key-Exchange (IKE)[44] with other key-management mechanisms. The new key-management mechanisms presented in this thesis utilize the Encapsulating Security Payload (ESP) [165] protocol from IPsec. In addition, the author enhances the IPsec databases for the purpose of this thesis. The IPsec databases are called the *Security Policy Database (SPD)* and the *Security Association Database (SAD)*. The SDP contains static policies that determine the decisions concerning inbound and outbound IP traffic. The SAD contains the dynamic protection state, including the confidentiality and integrity protection keys that are used for encrypting and authenticating the IP traffic. Basically, a policy indicates which security association should be used for a specific IP packet. Each security association in the SAD is identified by a Security Parameter Index (SPI). The SPI is also included in the *ESP header* that is carried in an IP packet after the IP header and before the upper layer protocol headers [165].

1.6 Internet Architecture

This section briefly presents some essential parts of the current Internet architecture. Section 1.6.1 discusses packet forwarding functionality. Section 1.6.2 introduces mobility-related security problems, and sections 1.6.3-1.6.5 cover basic mobility protocols.

1.6.1 Packet Forwarding Functionality

The basic infrastructure required for packet forwarding can be roughly divided into two kinds of intermediate nodes: *switches* that operate at the link-layer, and *routers* which operate at the network layer. Switches forward IP packets between nodes based on the Media Access Control (MAC) addresses carried in the headers of link-layer frames where each link-layer *frame* encapsulates an IP packet. The *MAC address* is a hardware address that identifies a NIC in the local network. It is also called a link-layer identifier.

> *"The function or purpose of Internet Protocol is to move datagrams through an interconnected set of networks."*[81].

In other words, IP packets are forwarded via multiple links and the associated MAC addresses are dynamically changed at the routers [36]. From the topological point of view, the IP address carried in the packet header defines the destination host for the packet, while the MAC address determines the next hop router. From the routing point of view, a router maps the destination IP address of each packet to the IP address, and then to the MAC address of the next-hop router. The mappings are stored in the routing tables. Each *routing table* consists of multiple entries, where a single entry contains a mapping between an IP-address prefix and an IP address or a MAC address. A *prefix* denotes the network-identifier part of the address. The structured address prefixes carry information of the hierarchical network topology. The closer the packet is delivered to the host, the longer part of the prefix is used for routing. Finally, the host identifier part of the address, also called *suffix*, identifies a host in the destination subnetwork. From the security point of view, the main problems are related to the integrity of routing tables. In particular, only authorized nodes should be able to update the routing tables.

To minimize the amount of data that needs to be stored in routers, they should maintain only the minimal set of IP-address prefixes that are required to deliver packets in an efficient way to the destinations. Depending on the granularity of the routing-table entries, routing protocols can be divided into interior and exterior ones [117]. *Interior routing* denotes routing inside an autonomous network while *exterior routing* denotes routing between autonomous networks [15]. In addition, there are some proposals to replace the legacy IP routing with flat-namespace, packet-content-based or geographic routing [29][33][184]. The design goals of all types of routing protocols are mainly related to scalability, security, extensibility, performance, robustness and supporting different kinds of QoS requirements (see e.g. [78][30]).

1.6.2 Security and Privacy Aspects

In the protocol stack, protocols for host mobility and related authentication are implemented above the packet forwarding functionality. However, the authentication, key management, and mobility mechanisms cause overhead in control signalling,

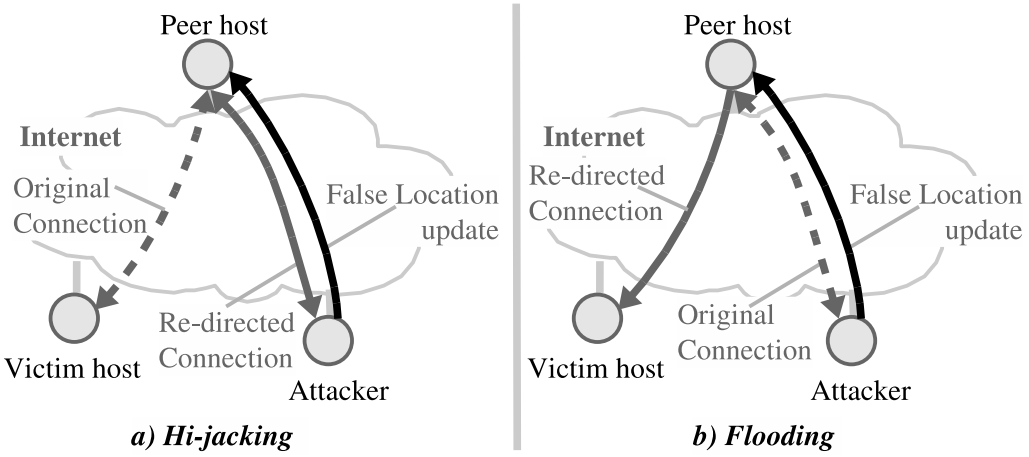

Figure 1.3: Two kinds of re-direction attacks.

and add complexity to the current Internet architecture. Security problems may also arise from the current IP-address-based host identification. Here, *identification* denotes a non-cryptographic mechanism used for recognizing that an identifier is associated to an identity, as opposed to cryptographic authentication. The problems caused by location-bound identification are related to host authentication during hand-offs and to the way currently deployed TCP/IP protocol machines identify connections.

Mobility-management protocols may even contain flaws that result in Denial-of-Service (DoS) vulnerabilities, and in leaks of identity and location information (see [148][91][176]). *Identity privacy* denotes controlled sharing of identity related information, while *location privacy* denotes controlled sharing of topological location information [148]. To mitigate identity leaks, it is possible to use *identity protection* mechanisms that provide identity privacy.

From point of view of this thesis, the essential DoS attacks can be divided into re-direction and Distributed-DoS (DDoS) attacks. In the former, an attacker may re-direct a connection to a new location, *hi-jack* the connection, and pretend to be the victim host as illustrated in Figure 1.3 (a). This kind of identity theft is the result of unauthorized use of an identifier. An attacker may also cause DoS by re-directing big data streams to the victim host's IP address or to its subnetwork. This is called a *flooding attack* and it is illustrated in Figure 1.3 (b). One form of a DoS attack is called a Distributed DoS (DDoS) attack where multiple (zombie) hosts

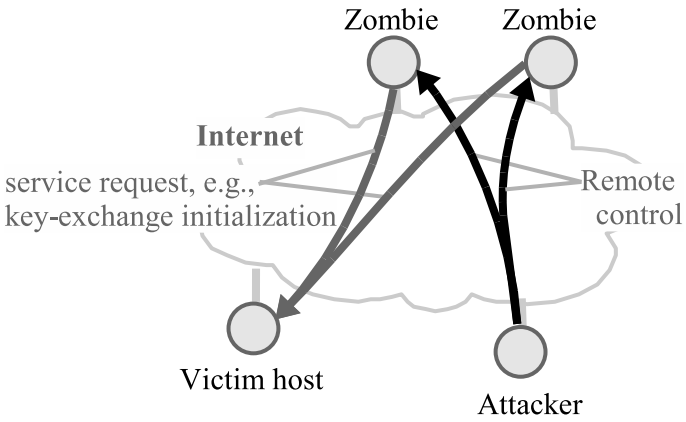

Figure 1.4: Distributed Denial-of-Service (DDoS) attack.

bomb the same IP address with multiple service requests, as illustrated in Figure 1.4. This may result in CPU and memory exhaustion if the victim host cannot process (consume) the incoming service requests as fast as than they are produced. From the system-security point of view, it is important to protect mobility protocols from all these attacks.

1.6.3 Host Mobility Support

In the current Internet, Mobile IP [26] and Mobile IPv6 [45] protocols provide basic mobility support for IPv4 and IPv6 hosts. The basic functionality of the protocols is illustrated in Figure 1.5, on page 29. To bring out the essential the differences between the two protocols, Mobile IP foreign agent [26] has been omitted from the Figure. The many extensions to Mobile IP and Mobile IPv6 are out of scope for this section. However, it is still good to notice that there are proposals that provide an integrated solution for IPv4-IPv6 dual stack mobility [71].

In both Mobile IP and Mobile IPv6, the mobile host registers with the home agent to acquire a so-called *home address* from the home link. The mobile host is reachable via the home address, which has a one-to-one mapping to the mobile host's topological IP address, called the *care-of address*. The mapping between the addresses is called a *binding*. In Mobile IP, the binding is stored only at the home agent and the connections to and from the mobile host are routed via the home agent. In Mobile IPv6, the binding may be stored also at peer hosts. As a result, the peer hosts are

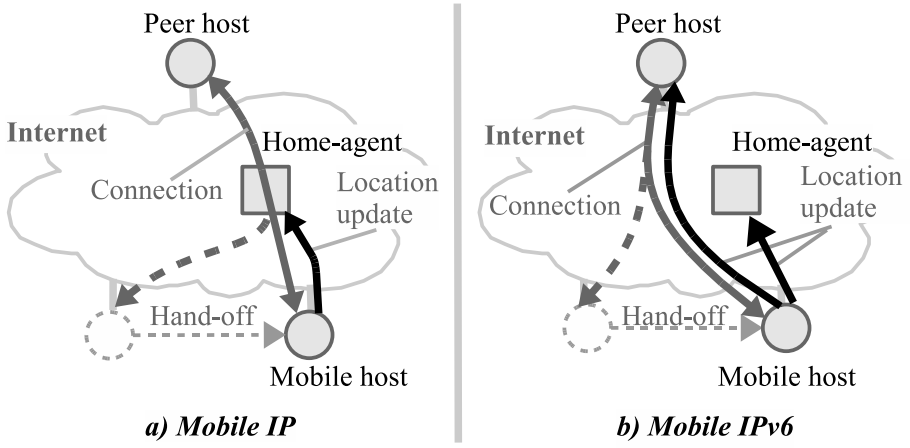

Figure 1.5: host mobility support in IPv4 and IPv6 networks.

aware of the mobile host's actual location and are able to send packets directly to the mobile host. This is also called *route optimization*. In both cases, the mobile host updates the binding at the home agent after each hand-off from one care-of address to another. In the route-optimization case, the mobile host also updates the bindings at the peer hosts. The location update messages are called *binding updates*.

To overcome the hi-jacking attack (Figure 1.3 (a), on page 27), the mobile host establishes an IPsec tunnel with the home agent. Using the IPsec security association, the home agent is able to authenticate the binding updates that are sent by the mobile host. However, the security association does not completely solve the flooding attack (Figure 1.3 (b), on page 27). Basically, a mobile host may re-direct its traffic to a victim host's location because the home agent does not verify that the mobile host is in the claimed location. However, the research community has not reached a consensus about the seriousness of this attack.

In Mobile IPv6, the mobile host and the peer host do not establish an IPsec security association. Instead, Mobile IPv6 uses so called Return Routability (RR) test to mitigate re-direction attacks. Basically, the peer host uses the RR test to verify that the mobile host is reachable at the claimed care-of address. The RR test is based on sending a challenge cookie via two different paths between the hosts. The first packet is routed via home agent from the mobile host to the peer host. In addition, the mobile host sends another packet directly to the location of the peer host. The

peer host replies to both of these messages and sends the messages via reverse routes back to the mobile host. Both of the messages contain different tokens that are used at the mobile host to generate the same key that is known by the peer host. The key is used to protect the final binding update message between the mobile host and the peer host. The RR test protects the hosts from the re-direction attacks presented in Figure 1.3, on page 27. Basically, the peer host does not send traffic to unverified locations. It is good to notice that the packets are forwarded via the IPsec tunnel between the mobile host and its home agent.

1.6.4 Local Mobility Support

Figure 1.6 (a), on page 31, illustrates Hierarchical Mobile IPv6 (HMIPv6) [34][72] approach to local mobility-management and signalling optimization. It defines a network node called the *Mobility Anchor Point (MAP)*. Basically, a MAP works like a home agent in a visited network. A mobile host can register with a number of MAP nodes at the same time. MAP hides the actual location of the mobile host from its home agent and from the peer hosts. The mobile host has two different kinds of care-of addresses. The on-link care-of address (LCoA) is the address assigned to the mobile host on its local link. A regional care-of address (RCoA) is an address that a MAP device has assigned to a mobile node. The RCoA is not changed as long as the mobile node moves only within the same MAP region. The mobile node signals changes of the LCoA to the MAP using local binding updates. HMIPv6 does not use the Return Routability (RR) test for the local binding updates.

The local binding updates, exchanged between the mobile host and a MAP, can optionally be protected with IPsec. The main reason to use IPsec is to protect the mobile host from connection hi-jacking. IPsec requires a pre-shared secret or public-key certificates for authentication. This is a problem because the mobile host needs to have prior knowledge of the MAP. However, since the RCoA addresses are ephemeral, the IPsec security association (SA) does not necessarily need to be strongly authenticated.

A scalable way to create the SAs is opportunistic authentication which is not supported in IKE. Here, *opportunistic authentication* denotes a mechanism that uses the Internet's routing infrastructure for binding an IP address to an authentic identifier. Typically, the first key exchange message is sent to an IP address without

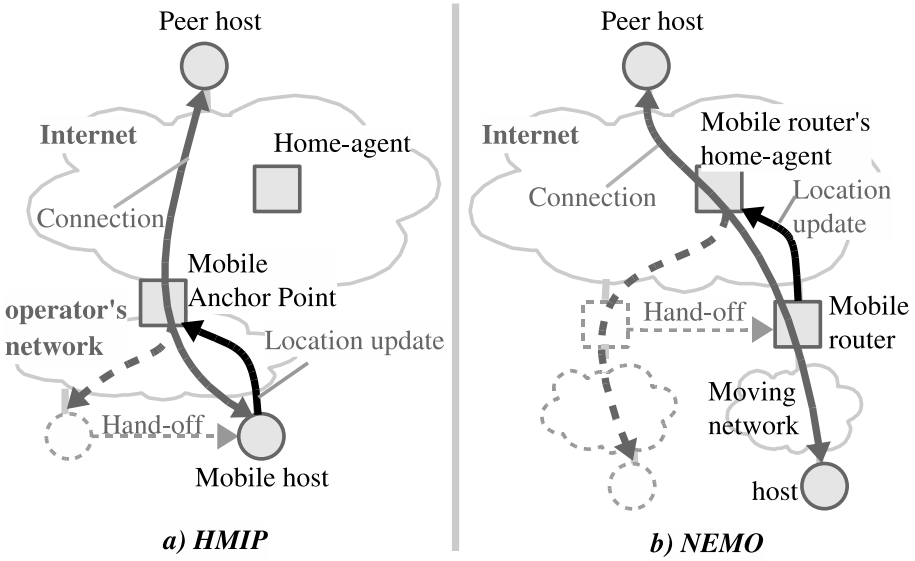

Figure 1.6: Local and network mobility support.

knowing the peer's public-key. The peer's public-key is received in the reply message. The solution is vulnerable to Main-in-the-Middle (MitM) attacks where the attacker is located on the mobile-host-to-MAP signalling path.

1.6.5 Network Mobility Support

Figure 1.6 (b) illustrates the Network Mobility (NEMO) basic support protocol [50]. In NEMO, a mobile router acquires a single IP address at an access network and establishes a bi-directional IPsec tunnel with its home agent. Connections to and from the moving network are routed via this tunnel. The clients attached to the moving network may additionally support host mobility or they may be fixed to the moving network. In a basic case, the mobile router registers a moving network prefix with its home agent and advertises the same prefix for address configuration in the moving network. Once the home agent receives packets carrying the moving network prefix, it tunnels the packets to the mobile router. From the peer host point of view, the hosts on the moving network are reachable via the location of the mobile router's home agent. Once the mobile router changes its point of attachment to the Internet, it sends a binding update to the home agent. As a result, the endpoint of the IPsec tunnel is updated and all the end-to-end connections are routed via the

new location of the mobile router. The mobile router and its home agent do not run any RR test during the hand-off.

1.7 Structure of the Thesis

The rest of this thesis is organized as follows. Chapter 2 discusses identifier bindings in the current TCP/IP protocol stack. This chapter provides the necessary background for the rest of the thesis.

Chapter 3 presents the main research problems and goals. The chapter also discusses the engineering trade-offs between the goals and defines the scope of this thesis.

Chapter 4 provides an the architectural overview of the proposed solutions. The description is divided into the architectural components, functional architecture and implementation architecture. This chapter binds the different contributions of the thesis together.

Chapter 5 describes the mobility and multi-homing functionality that is common to the different mobility solutions considered in this thesis. The chapter presents the functionality from the connection's, end host's, and intermediate node's point of view. The focus is on secure state establishment and update events.

Chapter 6 describes four granularity levels for mobility: flow mobility, address-family agility, local mobility, and network mobility. The different mobility solutions are integrated together at the implementation level.

Chapter 7 analyzes the presented mobility solutions from the viewpoints of bindings, cryptographic association mechanisms, signalling optimizations, and implementation. Finally, chapter 8 concludes the thesis.

A list of the author's contributions and original publications is presented after the conclusions.

2 Background: TCP/IP Stack Identifier Bindings

This chapter provides a background to internals of the TCP/IP stack, describing how the various identifiers are *bound* together. The author will focus on the reasoning behind the choices made in the current TCP/IP stack for binding identifiers together, and on the resulting tensions. The chapter describes the author's best understanding of the situation, and acts as a deep problem description for the rest of the thesis. The interested reader may look, e.g., at [12][13][38][40][42][84][112][147][155] to get familiar with the generic evolution of the Internet architecture.

An introduction to identifier bindings at TCP/IP layers is presented in sections 2.1-2.2. Sections 2.3-2.6 analyze the bindings in a bottom-up order, starting from the link layer and ending with the application layer bindings.

2.1 Definition of Binding

A *binding*, created by a protocol machine, associates some identifiers together in the TCP/IP stack; such identifiers include IP addresses, port numbers, protocol values and various credentials. In this thesis, a *binding* denotes a minimal shared state required for coupling a given set of identifiers together (see [89]). It is created by either a cryptographic or a non-cryptographic association mechanism[2]. A binding may associate two or more identifiers together, typically belonging to different namespaces.

Namespace is a set of identifiers associated to a given collection of identities [89]. The instances of protocol machines are a representation of identities in the current TCP/IP stack [96]. The states of protocol machines are typically identified using a subset of the associated identifiers. In a NAT device, a single IP address and a port value, carried in a packet, are used for identifying a binding that associates a pair of IP addresses and a pair of port values together. In addition, a state transition of a protocol machine may replace an associated identifier with a new one, such as a Mobile IP [26] host that updates a binding between its home address and care-of address.

[2]The different kinds of association mechanisms used in the TCP/IP stack are discussed later in sections 2.3-2.6

Each binding, created by a protocol machine, is stored in a data-communication state [43] at some TCP/IP layer. The multiple data-communication states form a chain of bindings that results in a connection between hosts. In other words, the multiple protocol machines at different nodes create bindings during connection establishment that are required to deliver packets between end hosts. The different protocol machines, distributed between nodes, depend on each other during the lifetime of a connection.

From the protocol-design point of view, one-to-one, one-to-many and many-to-many bindings are used for implementing identifier translation, tunneling and (de)multiplexing mechanisms. A one-to-one binding at Mobile IP's [26] home agent between a mobile host's home address and care-of address is an example of an identifier translation mechanism. Here, *identifier translation* denotes label switching operations for identifiers carried in packet headers. Typically, a header is located at a beginning of each packet, containing routing information and describing the content of the carried data.

Another example are Virtual Private Network (VPN) gateways that use tunneling techniques for encapsulating IP packets, from different sources, in IPsec [166] protected packets carried between the gateways. In other words, the gateways implement many-to-one and one-to-many bindings. In general, *tunneling* denotes a mechanism that encapsulates a packet in another packet.

A binding between IP addresses at NAT [136] devices is an example of a (de)multiplexing mechanisms implementing one-to-many, many-to-one and many-to-many bindings. For example, port-multiplexed NAT (NAPT) [136] devices implement *many-to-many* bindings between port and IP address values. *Port* values in the TCP/IP stack are local abstractions used together with IP addresses and protocol values to uniquely identify connections at hosts [48]. Basically, *multiplexing* denotes many-to-one bindings where several identifiers are mapped to a single identifier [48]. In this thesis, multiplexing denotes a mechanism that is used for identifier translation for outgoing packets. *Demultiplexing* is a reverse operation for multiplexing and it is applied to incoming packets where a single identifier is mapped to multiple identifiers [48].

To clarify the type of different kinds of bindings in this thesis, the author includes the direction and type of the mapping in the notation for essential bindings. The $A \overline{\overset{1 \leftrightarrow 1}{\longrightarrow}} B$ binding denotes a bidirectional one-to-one mapping between A and B identifiers, while the $A \overline{\overset{m \rightarrow n}{\longrightarrow}} B$ binding denotes a unidirectional many-to-many mapping between A and B identifiers. The bindings described in the following chapters have different combinations of the direction and arity.

2.2 Binding Identifiers

All data communication between nodes is based on a set of different kinds of bindings. This section analyzes bindings from viewpoint of their origin, scope, lifetime and location. The goal of this section is to illustrate the limitations of the bindings used in the current TCP/IP stack. These limitations explain many key problems in the existing mobility and multi-homing protocols.

In this thesis, an *intermediate node* denotes any device that is located between end hosts on the routing path. Depending on the context, the intermediate node may be, e.g., a switch, a router, or a NAT device. Based on this flexible definition of an intermediate node, a connection can be seen as a result of multiple identifier bindings at the end hosts and different kinds of intermediate nodes on the routing path; this is discussed in example 1 on page 36 and the generalization of the observation is illustrated in Figure 2.1 on page 36. The scope of the bound identifiers, the lifetime of bindings, and the number of bindings per connection determine the flexibility of a mobility architecture as presented later in chapters 4-7. Briefly, the *scope of an identifier* is defined by the corresponding namespace [89]. The *lifetime* of a binding depends on the lifetime of an instance of a protocol machine. The scope and lifetime of bindings are discussed later in sections 2.2.3 and 2.2.5.

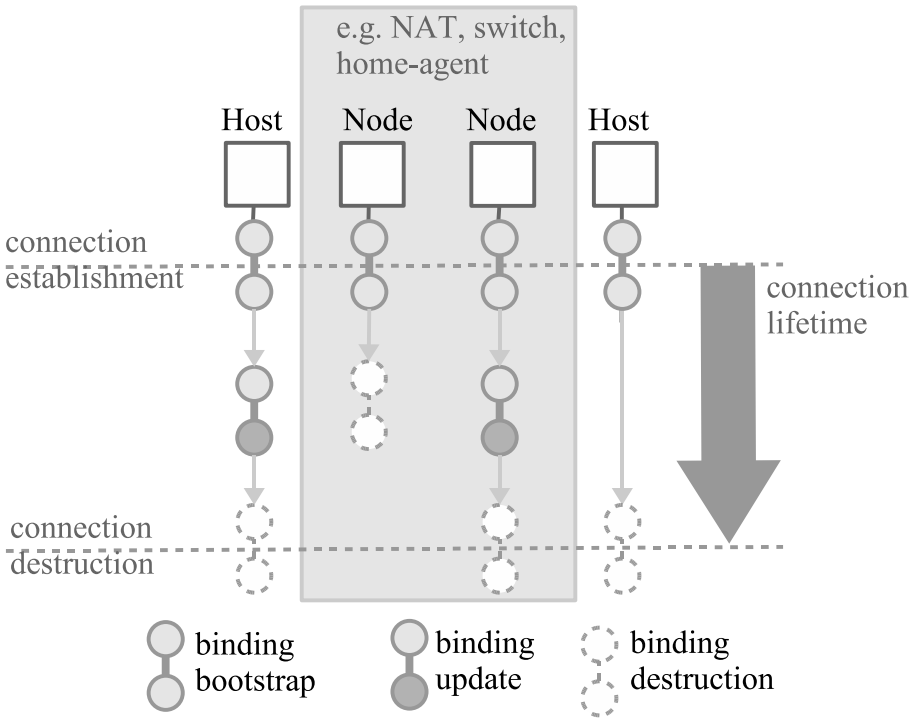

Figure 2.1: Connection consists of multiple bindings.

Example 1. *A Mobile-IP [26] host is located in private network behind a NAT node and a peer host is attached directly to the Internet. When the client host opens a connection, the packets are forwarded via the home agent to the peer node. The on-the-path switches, the NAT node, and the home agent each establish states bound to the different identifiers carried in the packet headers. For example, each switch creates a binding between a MAC address and an attachment point. The NAT node creates a binding between the IP addresses and port values. Finally, the home agent implements a binding between the home address and the care-of address. Due to the mobility of the host and the limited lifetime of bindings, new bindings may be established and old ones destroyed during the lifetime of a connection.*

2.2.1 Saltzer's Principles

Saltzer presented the main principles for naming and binding of objects already in 1978 [95]. Later on, Saltzer et al. presented their end-to-end communication

arguments in 1981 [171]. From the protocol-layering point of view, Saltzer identified four kinds of fundamental network objects and relationships between them (see [96]). Saltzer's network objects were: services, nodes, network attachment points, and paths. *Layering* denotes the way of developing protocols in layers where each layer is responsible of implementing a specific part of the communication procedure [89]. Nowadays, the rough correspondence between the presented objects and the operating system layers is visible. Operating system layers can be divided into application processes, a socket layer, a protocol layer, an interface layer and a physical medium [63].

Saltzer also identified three bindings between the objects: bindings between services and nodes, between nodes and attachment points, and between attachment points and paths. According to Saltzer, the resulting four namespaces, together, define the location for a network end point, i.e., a service. Since 1978, the Internet architecture has evolved, and the required granularity of the network objects has become more fine-grained. From the addressing point of view, this phenomenon is analyzed by Day [89]. However, Saltzer's naming and binding principles are still as valid as they were 30 years ago.

In [171], Saltzer et al. presented ideas about avoiding the placement of extra functionality at lower layers of the communication stack. The principle of providing minimal common functionality at each layer has been realized quite well in the TCP/IP stack. Saltzer's end-to-end ideas have been implemented partially (e.g. [38]) but not fully (see [41]) for the namespace part. The choices made in naming network objects differ from Saltzer's ideas [95] in that the different layers use the same namespaces to identify different network objects. An example is the tight binding between services and nodes. Nowadays, both of them use the same IP namespace for identification. For example, legacy applications and services open sockets that work as connection endpoints. A *legacy application* uses standard and widely deployed networking libraries and socket APIs. The sockets are identified with a set of identifiers including IP addresses, which will be discussed in more detail in section 2.6.1. Moreover, the nodes are identified at the network layer with the same IP addresses that are associated with the sockets. Balakrishnan et al. analyze the phenomenon in detail in [17].

2.2.2 A Connection Results in Multiple Bindings

A connection life-cycle between two nodes in the Internet can be divided roughly into three phases: creating a number of bindings, updating the bindings, and destroying the bindings (see Figure 2.1 on page 36). A *binding update* denotes a protocol state machine transition results in a new identifier association. The number of bindings per connection and the number and frequency of binding updates have an influence on the scalability (see [74]) of an architecture.

Example 2. *The explosion of routing table sizes [46] is an example of protocol machine design that produces a large number of distributed bindings. Basically, each routing table entry defines a binding between an IP address prefix and an IP address or a MAC address (or both), requiring a piece of memory.*

Example 3. *The growth of the Internet, and the limited number of IP addresses, have created a demand for NAT devices. The NAT devices dynamically create bindings between IP addresses and port values during the establishment of an end-to-end connection. A major problem with NAT devices is that the protocol machines at end hosts and at the NAT devices cannot be kept in synchrony, at least not easily. In other words, the bindings at NAT devices remain static for their lifetime, even the protocol machines update bindings at end hosts. This is a problem for the mobility-protocol design, because creating new bindings at the current NAT devices after a hand-off, instead of updating them, results in complex signalling during host mobility.*

In the connection-establishment phase, bindings are created at different layers both at the communicating endpoints and at some intermediate nodes. This is illustrated in Figure 2.2, on page 39, and discussed in example 4. In many cases, the intermediate nodes dynamically update and destroy bindings to provide efficient packet forwarding functionality (see [113]). For example, routing protocols can mitigate network failures and update routing table entries according to the dynamically changing routing paths. Another example is NAT devices and switches that destroy their communication states after certain period of connection inactivity.

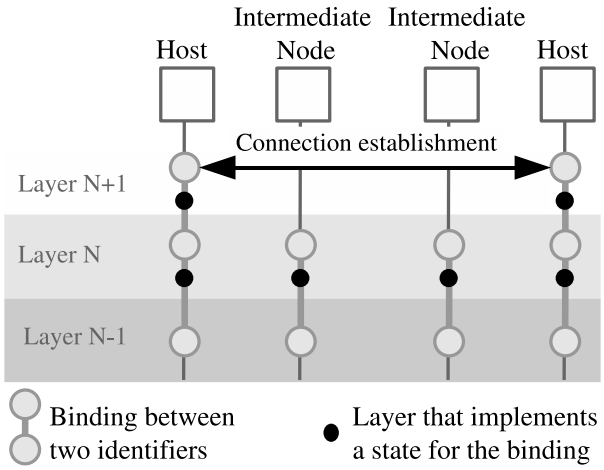

Figure 2.2: Creating communication states for bindings.

Example 4. *This example presents a connection establishment between two hosts on the same local link from the viewpoint of layers 1-3. The layers can be mapped to layers (N-1)-(N+1) in Figure 2.2. The connection, in this case, refers to the states created by Address Resolution Protocol (ARP) [47]. When the two hosts want to communicate with each other, a physical link (layer 1) must be established between them. When the hosts are attached to the link, they initiate ARP at layer 2. During this procedure, the interconnected switches and hosts in the local network establish bindings between MAC addresses (layer 2) and the physical ports (layer 1). Logically, the corresponding states are stored at layer 2 at the hosts and at the intermediate switches. The ARP also creates bindings between IP addresses (layer 3) and MAC addresses (layer 2) at the hosts. Logically, this state is stored at the layer 3 at the hosts.*

When a host makes a hand-off a number of new bindings are created at some intermediate nodes on the new routing path. To provide seamless binding updates and to minimize hand-off related latency, it is possible to create the required bindings below layer$_n$ at the new routing path before updating the bindings at the layer$_n$ at the end hosts (like [70][145][120]). This is also called *make-before-break* and discussed in the next example. In this thesis a *seamless binding update* denotes a hand-off that does not affect the experienced QoS [118].

Example 5. *A Mobile-IPv6 [45] host may have a NIC that supports simultaneous communication with two access points. Before the host breaks an attachment with the current access point, it may initiate an attachment with the new access point (see Petander et al. [70]). During this procedure, the host may acquire a new IP address at the new link. Basically, the host creates a new binding between its MAC address (layer 2) and the new IP address (layer 3) leased from the new link. After that, the host updates the binding between its home address and the new (care-of) IP address.*

2.2.3 Scope of Identifiers

Each protocol layer defines a logical scope for its identifier namespace. Whenever the protocol machines at two different layers use the same identifiers for state identification the protocol layers become coupled with each other. The same identifier may be used for different purposes in different protocol machines.

The states of the protocol machine are defined by the type of a connection and by the role of the node. In Mobile IPv6 [45], a mobile node, a home agent and a correspondent node have different kinds of protocol machines depending on the role of the node, albeit they all have bindings between the same set of home addresses and care-of addresses. In addition, the protocol machine defines, e.g., the identification, routability and QoS properties of the identifiers. For example, a TCP state machine uses a pair of IP addresses for connection identification, while the same set of IP addresses is used for packet forwarding by the Internet routing mechanism. It is also good to notice that the same identifier may be called a static identifier in one protocol machine and a temporary identifier in another protocol machine.

Example 6. *A Mobile-IP [26] host is located behind a NAT node. It establishes a connection through the NAT node with a peer host. The NAT node creates a binding for the IP address of the mobile host. From NAT protocol machine's viewpoint, the IP address is static because it is not changed during the lifetime of the binding. However, from the Mobile IP protocol machine's viewpoint, the same (care-of) IP address is a temporary identifier because it is changed during hand-offs.*

2.2.4 Bindings and Mobility

Mobility protocols update bindings between static and temporary identifiers. An example of this is a binding between Mobile IP's [26] static home address and temporary care-off address. The scope and number of bindings that are updated during each hand-off define the *granularity level* of mobility. A micro-mobility hand-off (section 1.6) may result in a binding update between a MAC and an IP address, while a macro-mobility hand-off may update a binding between a home address and a care-of address [26]. The connection related static bindings set constraints for the granularity of mobility design (discussed later in section 2.4.5).

Adding a dynamic binding between layer_n and layer_{n+1} identifiers provides a way to sustain connections above layer_n during hand-offs. For example, a TCP connection is bound to a static Mobile IP's home address at layer-4 while the care-of address at layer-3 is dynamically bound to the home address. In addition, a right balance between dynamic and static bindings that are part of a connection can minimize the amount of signalling during node movement and provide backward compatibility for applications.

> **Example 7.** *Depending on the mobility approach, the dynamic binding between identifiers takes place at different layers in the TCP/IP stack. From the legacy application point of view, it is important to provide static identifiers at the application layer. From the TCP point of view, it is essential to provide fixed identifiers for the TCP protocol machine. From the reachability point of view, it is important to have a static IP address. The presented viewpoints should be accommodated during a mobility protocol-design process. If a mobility approach is based on re-establishment of TCP connections after each hand-off, the mobile host must communicate with all peer applications. On the other hand, if the mobility approach hides the change of an IP address from the TCP, like Mobile IP[26] does, it is possible to optimize the required hand-off signalling.*

2.2.5 Lifetime of Bindings

A connection consists of multiple bindings and lifetime of the bindings may differ from each other. Whenever a single binding is destroyed, the whole connection will be broken at least temporarily. For example, if a NAT device or a router loses its

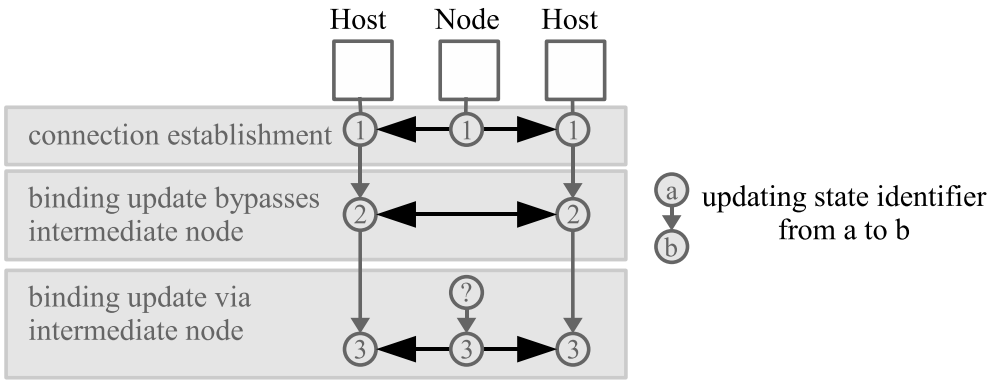

Figure 2.3: Binding identification problem.

state, the broken routing path cannot be used for packet delivery between hosts. Thus, the bindings must be updated or re-created to keep the connections alive between hosts. One design choice is to identify bindings at intermediate nodes with identifiers that do not change during hand-off or rekeying events. In this way it is possible to sustain connections and to avoid synchronization problems between the multiple states of a connection during hand-offs. However, this causes privacy and traceback problems.[3]

When connection identifiers are changed during hand-offs, e.g., for identity protection reasons, this may result in synchronization problems at intermediate nodes, as illustrated in Figure 2.3 on page 42. The main problem is that a binding identifier is replaced with a new one during the hand-off. Figure 2.3 illustrates a situation where an intermediate node cannot update the binding identifier to correspond to the situation at the end hosts. The dynamically changing connection identifiers can be used for providing identity protection for hosts. However, there is a trade-off between the privacy requirements and connection identification requirements at intermediate nodes. To support identity privacy, the connection identifiers should be changed during the lifetime of a connection. On the other hand, to identify connections at intermediate nodes the connection identifiers should not be changed during the lifetime of the connection.

[3]Here, *traceback* denotes identification of the source of a packet.

Example 8. *The following example is based on Figure 2.3. A host without any mobility-management mechanism establishes a UDP connection with a peer host through a NAT node. The NAT node creates a state for the source IP address (identifier 1 in Figure 2.3). Later on, the host is attached directly to the Internet without explicit mobility support. Even the host allocates a new IP address for itself, the connectionless UDP based communication continues (identifier 2 in Figure 2.3). Soon, the host joins again the previous private network and allocates a new IP address for itself (identifier 3 in Figure 2.3). The earlier created state at the NAT device is not expired. The NAT node is not able to identify the earlier created binding because the host uses a new identifier, i.e. IP address. The same problem applies also to later presented SPINAT [P7] approach, but in a different form.*

2.2.6 Churning Saltzer's Bindings

In this section, the author analyzes protocol machines that bind the Saltzer's network objects together in ways that were not initially preferred in [95]. Multiplexing and tunneling techniques are examples of this kind of protocol machine implementations. For example, overlay routing can be implemented with either of these techniques. From the IP viewpoint, overlay routing denotes that identifiers above network layer are used for routing packets between end hosts. Another example is the definition of size of a specific namespace in the different parts of the TCP/IP stack. Typically, size of an identifier namespace is based on the estimated maximum population size. Some initially made estimations have turned out to be insufficient, like with the IPv4 address namespace. This has resulted in multiplexing namespaces with others.

Multiplexing is mainly used to enlarge an existing namespace with another one, and to optimize payload size (see example 9 on page 43). Multiplexing also adds bindings at intermediate nodes that are identified with multiple identifiers, like IP addresses and port values at NAT devices. On the other hand, tunneling techniques minimize the number of bindings at intermediate nodes but increase the header size (see example 10 on page 44). They can also be used to optimize the bindings updates. For example, using triangular routing via a Mobile IP's home agent [26].

Example 9. *In a way a NAT device enlarges the existing IP address space by multiplexing public IP addresses with port values.*

Example 10. *A Virtual Private Network (VPN) connects two private networks to each other via an IPsec tunnel [166]. When one of the security gateways is behind a NAT node, the NAT node establishes a single binding for the tunnel, not per connection between end hosts.*

Protocol headers carried in the packets are not always in the same order as the corresponding protocol layers in the TCP/IP stack [168]. Alternative tunneling techniques provide a way to implement cross layer bindings at the TCP/IP stack. As a result, the packets are not processed in the TCP/IP layering order. An upper layer protocol may carry a lower layer protocol; the obtained benefit varies depending on the combination. For example, an IPsec [166] tunnel consists of IP packets that carry encrypted IP datagrams.

2.2.7 Services

A requirement for naming different network objects and binding them to each other comes from the interprocess communication (IPC) [193]. Typically, services are implemented at the application layer as user-space processes. Operating System (OS) can be divided into *user and kernel spaces* based on control of memory access. Even lower layer services are typically implemented as user space daemons, like Dynamic Host Configuration Protocol (DHCP) [143] servers.[4] Thus, the realization, i.e. implementation, of a communication service may be closer to Saltzer's end-to-end principles than the actual specification of the service.

As long as processes are located at the same host, they may communicate with each other using a local communication channel and local scope identifiers. When processes are running at the same host they still are logically executed at different locations, i.e. different parts of the memory address space. The host defines the scope of process identifiers, and thus the number of the alternative logical locations. The current TCP/IP stack is based on the assumption that the port numbers remain the same during the connection.

However, with additional support a process may change its location that is also called process migration. *Process migration* is a mechanism supporting process movement between hosts during execution of processes [179]. In other words, the process

[4] *Daemon* processes run in the background and are not associated to any terminal [192].

can update its location, and inform the peer process about the new port number bindings. While process migration is out of scope for this thesis, the purpose is to illustrate that each dynamic binding is bound to a specific granularity scope of mobility (discussed in chapter 6).

2.3 Link layer Bindings

The link-layer is the second layer in the TCP/IP 5-layer model that refers to a physical frame format and addressing [48]. The link-layer identifiers name a network interface in a physical link. Each access technology defines a link layer namespace, like IEEE 802.11g[79] or IEEE 802.3[80]. From the connection point of view, the different link layer namespaces provide the same semantics, while each NIC attached to the node may support different kind of link layer namespace.

2.3.1 Attachment Point Identifiers

The link-layer identifiers do not carry information about the services. From the location information viewpoint, the current link layer identifier namespace is flat. *Flat namespace* denotes that an identifier of the namespace is unstructured and cannot carry hierarchical routing information. Structured identifiers, like IP addresses, contain information of a hierarchical network topology. However, part of the link layer namespace is typically divided into hierarchies by the manufacturers to avoid identifier collisions. Here, *identifier collision* denotes the probability of the same identifier being associated to multiple identities in a specific namespace scope. When the collision probability is low enough it can be ignored in real life use-cases [133].

The size of the link layer identifiers defines their collision probability. In the case of collision, a colliding host is assumed to change its link local identifier. The collisions may result in process mis-identification, which is also used for identity theft. The existing link layer protocols like ARP [47], and neighbor discovery in IPv6 [180][86] are examples of protocols that bootstrap link layer communications and handle the collisions. A *bootstrapping* mechanism defines which event takes place first in a system. This is also called the chicken or the egg dilemma.

2.3.2 Identifier Binding Problems

The security problems with the link layer identifiers are related to the ownership [123] of identifiers and to privacy issues. Non-cryptographic link layer identifiers cannot be used for strong authentication. *Strong authentication* denotes a public key based cryptographic security mechanism that is used to verify an identifier-identity association [148]. ARP spoofing and neighbor discovery related redirection attacks are examples of the link-layer vulnerability (see [88]). In a *spoofing attack*, an attacker pretends to be another authorized host in the network [148]. In a link-local re-direction attack an attacker tries to give a false link-local address to the router and other hosts in the link to redirect data to this address. The identifier ownership problems [123] are similar at the link and network layers. The scope is just different, and therefore also the severity of the potential DoS attacks.

The privacy issues are related to static link layer identifiers and traceback. As long as the node uses the same link layer identifier at different links, it is possible to identify the host. Dynamically changing identifiers provide anonymity but make also the authentication and access control more difficult to implement. *Anonymity* denotes that an identity is unknown. Here, *access control* denotes the protection of local network resources against unauthorized access [148].

2.3.3 Node Namespace Required

The scalability issues related to the flat link layer namespace are visible when lots of links are connected at the link layer via intermediate nodes to each other [52][160][154][83]. Depending on the access technology the intermediate nodes are called switches, relays or repeaters at the link layer [104]. When a process initiates a connection with another process at a peer host, the local host triggers a peer host discovery. Due to the flat namespace structure, the discovery procedure is typically based on local broadcasting. Local broadcast means the delivery of data to every node in a link. The messages are forwarded between links via the intermediate nodes. As a result of the discovery procedure, the intermediate nodes, e.g. switches, create bindings between the link layer identifiers and network interfaces.

The size of the population defines the scalability limits for simple flooding based spanning tree approaches. In a spanning tree approach a source node (a root) sends packets to other nodes (leaves) that belong to the group. Therefore, alternative node

discovery proposals are based on overlay of tunnels (like [52][154][140]), Distributed Hash Table (DHT) techniques (like [28][191]) and sophisticated Peer-to-Peer (P2P) flooding techniques (like [18]).[5] *Overlay of tunnels* denotes here that identifiers above link layer are used for forwarding packets between NICs. Moreover, *DHT* is based on key-value mappings that are distributed between several intercommunicating nodes. Each of the nodes stores a subset of entries of a complete hash table. In *P2P networking* clients communicate directly with each other instead of communicating via a server (a client-server model).

2.4 Network Layer Bindings

Due to the limited scope and large spectrum of link layer identifiers an additional global node identifier namespace is required to avoid identifier collisions and to provide transparent [13] end-to-end communication [35]. Following the layering design, the global IP address namespace is defined at its own layer. That is, at the network layer. Network layer is the third layer in the TCP/IP 5-layer model. It provides connectionless packet delivery service for upper layers.

2.4.1 Routing Paths between Attachment Points

Logically, unicast communication at the network layer is based on one-to-one bindings between IP addresses, while multicast traffic is based on one-to-many bindings. The multicast traffic uses group identifiers, also called multicast addresses, where multiple NICs are associated with the same multicast address. From the end host point of view, an IP address pair defines a routing path between the hosts. From the identifier binding point of view, the routing path may consist of several sub-routing paths that are defined by link-layer identifier pairs. To overcome failures on the routing paths, routers have a set of alternative bindings between IP addresses and link-layer identifiers. The scope of the routing path identifiers is just different, as illustrated in Figure 2.4 on page 48.

It is good to notice that the network layer does not keep a state for each end-to-end path. Thus, the network layer provides a stateless delivery mechanism between locations of hosts. Stateless denotes that the network layer does not need to dynamically

[5]Here, *flooding* denotes forwarding of packets to every node in the scope of the local link layer namespace [118].

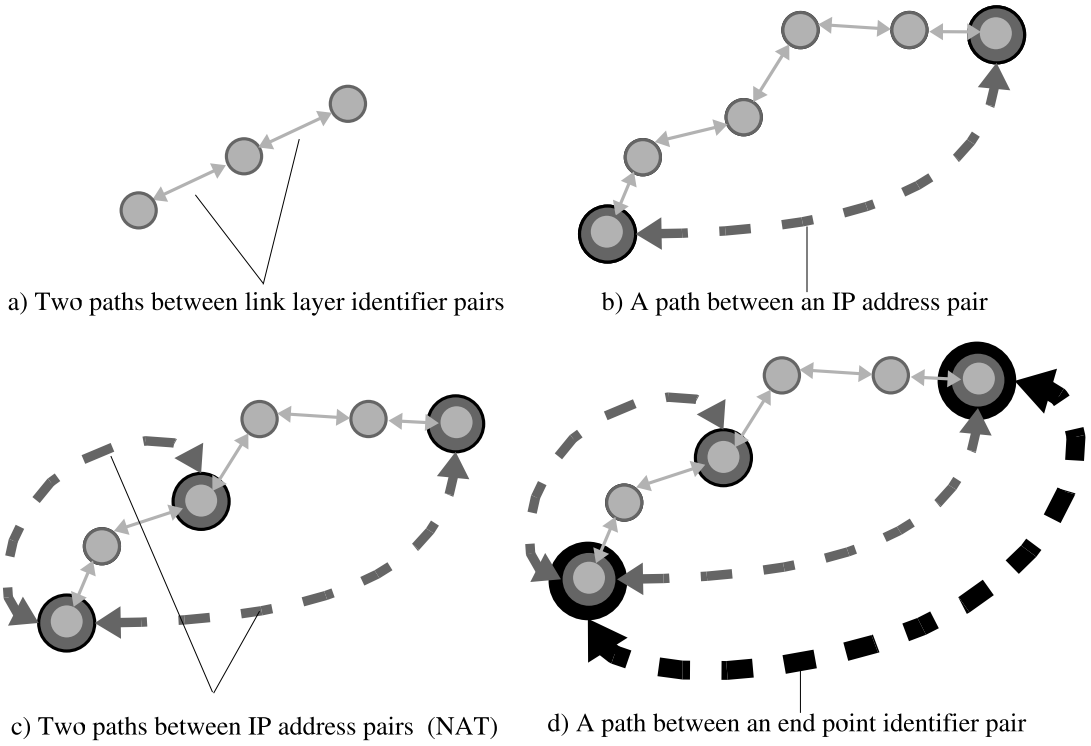

Figure 2.4: The scope of a routing path.

create new protocol machine instances. However, when the size of routing tables depends on the number of destination locations, the network layer is not totally independent of the end-to-end paths.

2.4.2 Dynamically Changing Identifiers

The lifetime of the link layer identifiers and IP addresses naming the routing path may differ from each other. Due to network failures the routing path between IP addresses may be changed from the link-layer identifiers point of view. The packets are forwarded via a topologically different routing path. As a result also the chain of link layer identifiers that identify the path is changed. In addition, an IP address of a node may be changed due to topological movement or a prefix renumbering event. *Renumbering* denotes the procedure of changing the IP addresses or a network prefix of each network interface attached to the network [129]. The IP based mobility results in dynamic bindings above the link layer.

Either the end hosts or intermediate nodes may dynamically update the bindings related to the end-to-end routing path. Depending on the approach, the changes on the path are visible or transparent to one or both of the end hosts. This is discussed in the following section 2.4.3. A common thing with the mobility architectures is that almost (excluding e.g. [124]) all of them depend on intermediate node support.

To manage the changes on the routing path, the mobility approaches need protocol machines at intermediate nodes to provide rendezvous functionality. A rendezvous node works as an initial (or constant) contact point for a moving node. The intermediate nodes also help to mitigate double jump related problems. Double jump means that both end hosts are moving simultaneously and the binding update messages are sent to the old locations of the nodes. The rendezvous nodes hide the changes on the routing path temporarily or completely. A non-optimal routing path that goes via a rendezvous node offers location privacy for an end host because the rendezvous node hides the actual location of the host.

The rendezvous nodes may also interact with each other and form a logical overlay routing infrastructure above the network layer (see [122][67]). An IP-address-independence overlay infrastructures use namespaces above the network layer for routing packet between end hosts. This kind of routing path naming allows to bind different kinds of network layer identifiers together.

2.4.3 Mobility Approaches

The network layer mobility protocols can be analyzed from the routing path update point of view as illustrated in Figure 2.5, on page 50.

Mobility Model 1. Some of the mobility protocols update the end-to-intermediate routing path, like Mobile IP [26] (see also [32]). This is illustrated in the first row in Figure 2.5 and discussed in example 11 on page 50. In this kind of approach, the routing path remains the same from the peer node's viewpoint. The mobile node updates the forwarding path at intermediate nodes, which results in non-optimal packet forwarding paths. Typically, only the mobile node is authorized to communicate with the intermediate nodes.

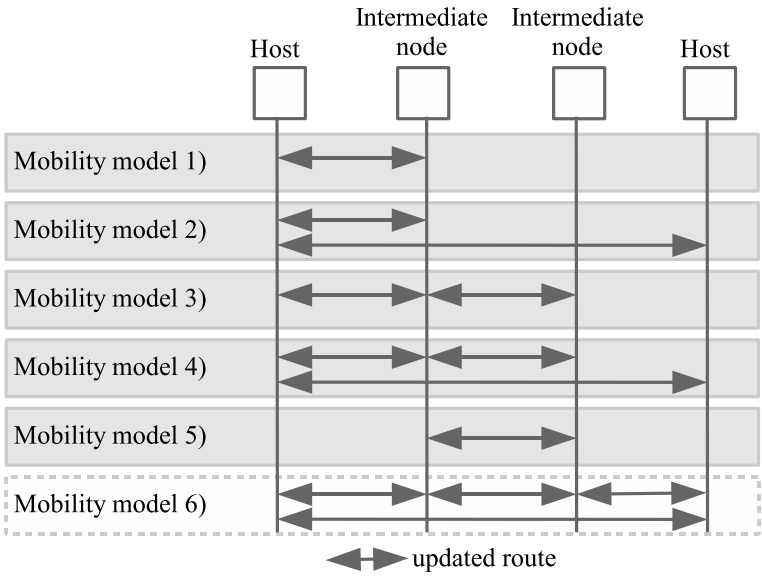

Figure 2.5: Routing path update scenarios.

Example 11. *In Mobile IP [26], packets are routed via a home agent, through a tunnel, to the mobile host. This kind of routing is also called triangular or dog-leg routing. A mobile host has a security association with the home agent. After each hand-off, the mobile host updates its location to the home agent only. Due to the triangular routing the mobility is invisible to the peer host.*

Mobility Model 2. Another mobility approach is to update the end-to-end and end-to-intermediate routes, like in Mobile IPv6 [45] (see also [173][152][58][31]). This is illustrated in the second row in Figure 2.5 and discussed in example 12 on page 51. The approach extends the mobility model 1. The main difference is that the packet forwarding path is updated both at the intermediate and peer nodes. The end-to-end signalling provides routing path optimization. The target of the routing path optimization is to minimize the end-to-end packet forwarding latency. A direct routing path means topologically shortest path between two nodes.

Example 12. *In Mobile IPv6 [45], the mobile host does not have a security association with its peer host but with it home agent like in Mobile IP [26]. However, the mobile host updates its location both to its home agent and to the peer hosts. The security is based on sending different pieces of binding update related information via two different paths. One path goes through an IPsec tunnel via a home agent, and the other path goes directly between end hosts. A MitM attacker must be located on the end-to-end signalling paths to break the security solution as discussed earlier in section 1.6.3.*

Mobility Model 3. It is also possible to update the end-to-intermediate and intermediate-to-intermediate routing paths like Stoica et al. present in Internet Indirection Infrastructure (i^3) [174] (see also [198][61] [186][109][107][172]). This is illustrated in the third row in Figure 2.5 and discussed in example 13, on page 51. In this approach, the routing path remains the same from the peer node's viewpoint. Both the mobile nodes and the intermediate nodes may be authorized to update the routing path. The approach may improve the hand-off latency and provide route optimization.

Example 13. *i^3 [174] defines an overlay routing mechanism for multicast, anycast and mobile communication. The architecture decouples the sender and receiver of each other using rendezvous servers, called i^3 nodes. Packets are routed from the sender to the receiver via these i^3 nodes. After each hand-off the mobile host updates its location at an i^3 node. Due to the triangular routing latency, the mobile host may change the i^3 node and update a binding between the old and current overlay identifiers for reachability reasons. This results also in a routing path update in the i^3 infrastructure.*

Mobility Model 4. Another approach is to update the end-to-end, end-to-intermediate and intermediate-to-intermediate routing paths like Nikander et al. present in Hi^3 [126] (See also [57][139][6]). This is illustrated in the fourth row in Figure 2.5 and discussed in example 14, on page 52. The approach combines the mobility models 2 and 3. The additional end-to-end signalling may be needed for security and/or for additional route optimization reasons.

Example 14. *Hi³ [126] is an instantiation of the i³ [174] architecture that defines a detailed security solution at the protocol level. Hi³ combines ideas from Secure-i³ [5] and HIP [P3]. Hi³ is based on the observation that a DHT extended HIP rendezvous server [90] and the basic Secure-i³ infrastructure are fairly close to each other. From the routing path viewpoint, the main difference between Hi³ and i³ (see example 13) is that the mobile host may update its location also directly to the peer host.*

Mobility Model 5. It is also possible to update only the intermediate-to-intermediate routing path like Proxy Mobile IPv6 [164] (see also [159][156]). This is illustrated in the fifth row in Figure 2.5 and discussed in example 15 on page 52. In this kind of proxy based approach, the mobility is transparent for both end hosts. The intermediate nodes are authorized to update the routing path. Like the traditional routing is transparent to the end hosts, in a similar way the intermediate proxy nodes can make the mobility transparent for end hosts. A proxy node takes care of the signalling on behalf of the end host.

Example 15. *In Proxy Mobile IPv6 [164], a legacy host attaches to an intermediate node called Mobile Access Gateway (MAG). Once a MAG node makes a hand-off also the attached legacy host changes its topological location. The MAG node updates the legacy nodes location to a home agent called Local Mobility Anchor (LMA). The mobility is hidden from both end hosts.*

Mobility Model 6. Finally, this thesis presents an approach for updating the intermediate-to-host routing paths as illustrated in the last row in Figure 2.5. The approach is based on delegation of signalling rights between end hosts and intermediate mobile nodes. The related network mobility solution is presented in publication [P9], and discussed later in section 6.4.

2.4.4 Analysis of Mobility Approaches

The evolution of the presented mobility models 1-6 can be analyzed from the TCP/IP backward compatibility viewpoint. The oldest protocols updating the end-to-intermediate routing path were designed to be backward compatible from the peer host viewpoint. However, the approaches require changes at the mobile host and intermediate nodes. The latest protocols updating intermediate-to-intermediate

routing path provide TCP/IP backward compatibility at both end hosts. Thus, more and more functionality is moved to the intermediate nodes. The rest of the approaches increase the functionality both at the end hosts and intermediate nodes. This breaks the backward compatibility with the existing TCP/IP stack bindings at nodes.

The evolution of the dynamic routing path management can also be analyzed from architectural requirements' viewpoints, e.g., from security, robustness, bandwidth consumption, latency and route optimization viewpoints. However, related analysis is out of scope for this chapter.

2.4.5 Additional States at Intermediate Nodes

Several namespace problems at the network layer are related to the shortage of 32-bit long IPv4 addresses and size of routing tables [46]. The Classless Inter-Domain Routing (CIDR), private address spaces and IPv6 were introduced to solve the problems [183][103][163]. While CIDR reduced the bindings at the routers, the private address spaces added new states at intermediate nodes called NAT devices. The utilization of solutions (e.g. [76]) that solve the intercommunication between IPv6 and IPv4 has been quite slow. Therefore, the larger 128-bit IPv6 address space has not been fully utilized in the current Internet. Instead, usage of IPv4 address space has been optimized at the cost of increasing tensions inside the TCP/IP stack.

The private address spaces add new kinds of bindings to the routing path. Some intermediate nodes, like NAT devices and firewalls, define additional IP address related bindings. As a result, IP address-pairs cannot always be used for defining end-to-end routing paths in a unique way. Instead, the routing path may be identified with a set of IP address-pairs. Thus, other namespaces are needed to efficiently identify a routing path. This has resulted in alternative overlay routing concepts that use additional namespaces for path identification between end hosts.

As discussed earlier in section 2.2.2, a connection consists of different kinds of bindings at intermediate nodes located on the routing path. In the basic form, the intermediate nodes utilize the IP address space only for network layer routing. However, the earlier mentioned shortage of global IPv4 addresses and privacy requirements have led to multiplex IP addresses with other namespaces. Typically, the transport layer identifiers, like port values, are used for this purpose. The network and trans-

port layers are not only bound together at end hosts but also at the intermediate nodes.

From the mobility viewpoint, it is not enough to manage IP related bindings at end hosts but also at the intermediate nodes. However, the end-to-end binding update signalling cannot be used to update the static bindings at the deployed NAT devices as discussed in example 3, on page 38. To overcome the problem, some mobility approaches use tunneling techniques and explicit protocols to initiate static bindings per connection at intermediate nodes which is a clear architectural inefficiency. An example of this are legacy NAT traversal protocols that use UDP tunnelling like STUN [161], TURN [94] and TEREDO [23]. NUTSS [66] and NATBLASTER [20] are proposals for establishing TCP connections through NAT devices. Many of these proposals reuse techniques developed in the P2P file-sharing and gaming community.

The mentioned NAT traversal approaches do not require modifications to the existing and largely deployed NAT devices. However, without incorporating additional logic into the intermediate nodes and support of third party entities, mobile protocol solutions become tricky when both end hosts are behind NAT devices. From the protocol-design point of view the UDP tunneling violates the Saltzer's layering principles, wastes bandwidth, and makes the protocol implementation gradually more complex. To avoid the presented NAT related problems, the end host and new intermediate nodes should use the same static identifiers for connection identification. The used connection identifiers at intermediate nodes and end hosts should be changed during host mobility only for privacy purposes.

2.5 Transport Layer Bindings

Transport layer is the fourth layer in the TCP/IP 5-layer model. The network layer does not keep bindings between nodes and services but the bindings take place at the transport layer. For example, the TCP pseudo header checksum is computed over the IP addresses and port numbers carried in the packet.[6] This kind of design choice binds the network and transport layers to each other. The pseudo header checksum cannot be changed during the TCP connection lifetime. If the identifiers used at transport layer are changed, the packets cannot be delivered between the

[6] A *pseudo-header checksum* is a field carried in a TCP header that is used for error detection. The TCP/IP checksum covers the TCP header and a pseudo header that contains the IP addresses, the protocol value and the payload data length [93].

communicating processes without mobility support.

The transport layer protocols can roughly be divided into connectionless (e.g. UDP) and connected (e.g. TCP) protocols (see [185]). Connected transport layer protocols create static bindings between IP addresses and port values, unlike connectionless transport layer protocols. The intermediate nodes, like firewalls, benefit from the connected and stateful transport layer protocols because it is hard for attackers that are not on the routing path to guess the IP addresses, port values and sequence numbers carried in the packet and required to bypass a firewall. While connected transport layer protocols are able to provide some security protection against identifier spoofing, the bindings at different nodes need to be in synchrony. The synchronization is required to handle time out and state loss events at the end hosts and intermediate nodes.

2.5.1 Decoupling Network Layer from Transport Layer

The initial transport layer protocol machines, like TCP, were not designed to dynamically update their states as a result of host mobility. Therefore, static identifier bindings at the transport layer has resulted in defining logical mobility related namespaces that hide location updates from the transport and upper layers. The bindings between the IP addresses and the mobility related identifiers are logically located between the network and transport layers. An example of such a mobility namespace at the transport layer is the Mobile IP's home address namespace [26]. Another example is the identifier-locator split approach in HIP [P3] where an IP address works as a locator at the network layer while the host is identified with a separate public-key based identifier at upper layers. This differs from the current identification practice at the Internet where the IP address also works an identifier for the host above the network layer. However, decoupling the network layer from the transport layer can be done in alternative ways. For backward compatibility reasons, the identifiers of the mobility namespace must have similar semantics as the legacy IP addresses from the transport layer's viewpoint.

The locations of the mobility related namespaces in the TCP/IP stack has araised argumentation in the research community. It seems that the Saltzer's principle of providing the minimal common functionality at each layer is a good argument for defining a logical macro mobility layer below the transport layer. In practice, the

protocol machines of the logical layer are typically implemented (hooked) at the network layer.

Depending on the mobility approach, the mobility related identifier namespace may be flat or structured. The name resolution and connection bootstrapping procedures depend on the semantics of the static identifiers used above the network layer (see example 16, on page 16). Name resolution means binding an application layer name to a peer host's identifier. In addition, the network layer security protocols (like IPsec [166]) define security related namespaces that logically are also located above the networking layer but below the transport layer. However, most of the mobility protocols do not utilize the security namespaces for mobility management (unlike HIP [P3]).

> **Example 16.** *In Mobile IP [26], a DNS query returns a home address of a mobile host for an application that wants to contact the mobile host. The home address is used at the transport and network layers to initiate a connection. However, in the HIP [P3] case, a DNS query returns a non-routable host identity tag (HIT) to the application and to the transport layer and an IP address to the network layer. HIT is a hash of a public key that is computed using a one-way hash function [14]. The host must bind the HIT and IP address together below the transport layer. If either the HIT or IP addresses is not available for the application it is not able to initiate a connection with the mobile host.*

While the network layer provides a routing path between IP addresses, the logical mobility layer provides overlay routing path between mobility layer identifiers above the network layer. In a way, the overlay routing functionality (e.g. in i^3 [174]) implements the IP address translation without causing tensions in the TCP/IP stack unlike legacy NAT devices.

2.5.2 Mobility Approaches at Transport Layer

The mobility-management can also be implemented per transport layer protocol. Some new transport layer protocols support dynamic identifier binding updates (like [116]). The trade-off resulting from the transport layer related mobility approaches is that each of the protocols need to implement its own mobility-management

functionality, unlike the mobility approaches that support dynamic bindings below the transport layer.

When dynamic IP address bindings are supported both at the transport layer and below the transport layer, the protocol-design becomes complex. For example, combining Mobile IP [26] and mobile Stream Control Transmission Protocol (SCTP) [116]. The alternative locations for dynamic IP address mappings has led to incompatibility between different mobility protocols. One reason for this is that the protocols are designed to solve different subsets of a larger mobility, multi-homing and security problem field.

2.6 Application Layer Bindings

The applications communicate with the transport layer via the socket Application Programming Interface (API) library that is called libc. From the socket API point of view, mobility identifiers provided for applications must be as long as legacy IP addresses in bits for backward compatibility reasons. Therefore, the current mobility proposals use 32-bit and 128-bit long identifiers for IPv4 and IPv6 socket APIs at the application layer. The design of backward compatible mobility protocols depends on the legacy socket APIs. Some protocols break the backward compatibility and define new socket APIs, like DONA [109] and SCTP [149]. A typical problem with the deployment is that most of the current applications support only legacy socket APIs. *Deployment* refers to an issue of getting a new protocol or identifier integrated into a code-base at those nodes that are already located in the Internet.

2.6.1 Socket Interface Bindings

The socket binding ties the application, transport and network layers together. A *socket* was initially defined to allow multiple processes at a host to use multiple instances of transport layer protocol machines simultaneously [93]. A socket associates an IP address pair, a port value pair and a protocol value together. A pair of sockets uniquely identifies a transport layer connection.

While the socket functionality (see [190]) does not have its own layer from the TCP/IP layering viewpoint, it has one from the implementation point of view (see e.g. [63]). Logically, the socket layer is located above the transport layer. A socket

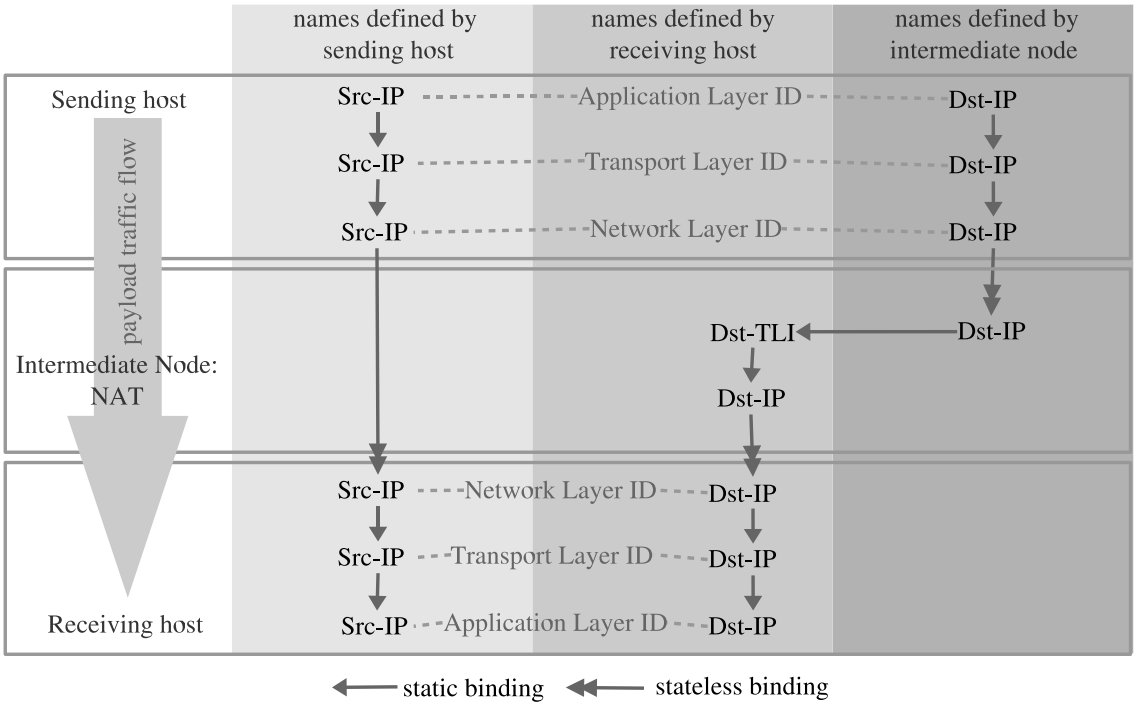

Figure 2.6: IP address bindings for end-to-end data-plane traffic in the current TCP/IP stack.

is bound to a Transport Layer Identifier (TLI) pair that defines a routing path, e.g., for a TCP connection. Each TLI consists of an IP address, a protocol number, and a port value triplet. This causes tensions in the TCP/IP stack because IP addresses are bound both to Saltzer's network attachment points and services as illustrated in Figure 2.6. This differs from the Saltzer's viewpoint of having own bindings between services and nodes, and nodes and attachment points. Some mobility approaches solve the problem by defining an additional binding above the transport layer (like [82][197][170]).

It is also good to notice that each binding requires a state. Therefore, the socket can be seen as a state between the application and the transport layer that provides the binding between the identifiers. The sockets are bound to the same protocol and address-family, e.g. IPv4 and IPv6, during their lifetime. This kind of static binding sets limits on interprocess communication between sockets that are bound to different address and protocol families.

Basically, the socket binding would probably be different if the current port number

namespace were larger (above 16-bits) or the IPv6 address space (128-bits) were directly used to identify services. The current socket structure makes the evolution towards next generation communication schemes difficult. With a large enough global port, i.e. service, number namespace it would be possible to combine the semantics of the node and service namespaces together (see DONA [109]).

2.6.2 Service-to-Node Binding

The interface above the application layer is towards a human being. It differs from the other interfaces in the TCP/IP stack. The human readable Fully Qualified Domain Name (FQDN) [2] offers a way for persons to interact with the TCP/IP stack in a convenient way. FQDNs given to applications are mapped to IP addresses. The bindings are stored at intermediate nodes, i.e. at Domain Name System (DNS)[2] servers. The local *resolver library* at the application layer takes care of the communication between the processes and the DNS servers. During the *name resolution* procedure an application makes an FQDN query to the DNS server. The *DNS* server maps the FQDN to the IP addresses stored in its database and sends them back to the application.

The basic reachability of a host is implemented with the FQDN-to-IP address binding. A user needs to know only a name of the host, not the location of the host. However, the DNS does not support fast binding updates between the identifiers. Therefore, the FQDN is often mapped to identifiers of mobility namespace at a DNS, like to Mobile IP's [26] home addresses. Typically, the weak authentication of the peer host is based on the FQDN$\overset{n\rightarrow m}{\longrightarrow}$IP-address bindings that is received from the DNS. To strongly authenticate the service, security APIs can be used to define security policies for sockets based on the transport layer identifiers. Here, *security policy* denotes a set of rules that determine the integrity and confidentiality protection mechanism applied for the packets sent and received via a socket [148].

2.6.3 Service-to-Service Binding

Some protocols implement bindings between sockets at the application layer, like P2P applications (see [108]). In this kind of approach, the intermediate nodes forward packets between sockets. Although, the processes at the intermediate nodes act as connection end points, payload data is forwarded between processes located

at end hosts. The approach requires an additional namespace to be used above the socket layer. Different P2P protocols define this kind of new namespaces for socket bindings. As discussed earlier, overlay routing can be based on alternative namespaces above the network layer. The logical mobility layer below the transport layer, and the application layer namespaces are alternatives to implement that.

2.7 Summary

The initial set of TCP/IP identifier bindings and the socket concept have been good choices for several reasons. The Internet architecture has not suffered from additional namespace management problems. The initial set of namespaces has been easy to deploy. The basic TCP/IP identifier bindings have fulfilled the needs of most nodes in the Internet.

If the mobility aspect and mobility namespaces had initially been part of the architecture the mentioned issues would probably have scattered the Internet. The trade-off has been that the initial set of identifier bindings is not enough for the current mobility and multi-homing requirements. The Internet community is facing the challenges in the architectural design.

The current TCP/IP namespaces are mixed with each other resulting in identifier binding update problems as illustrated in Figure 2.6, on page 58. Tunneling based approaches are used to minimize additional states at end hosts and to move part of the state management to intermediate nodes. Some other solutions compensate the small namespace sizes by multiplexing existing TCP/IP identifiers at intermediate nodes. All of these cause tensions between network-stack layers.

Currently the transport layer identifiers used with the sockets bind the network, transport and application layers to each other. The initial static bindings between the layers do not support granularity in mobility approaches. It seems that dynamic bindings between identifiers of any adjacent layers increase the flexibility of the TCP/IP stack. In addition, the location of the dynamic bindings, at end hosts and intermediate nodes, defines the granularity level of the mobility approach as discussed in the following solution chapters 5 and 6.

3 Research Problem

One of the main problems with the existing mobility and multi-homing approaches is that they do not offer integrated authentication and mobility-management mechanisms for different granularity levels of mobility. The main goal in this thesis is to define an efficient and flexible mobility and multi-homing architecture that integrates authentication, key-management, privacy protection, and mobility-management signalling between nodes in a seamless way. The research field is related to finding alternative ways to secure dynamic bindings in the TCP/IP stack.

3.1 Research Goals

The author has been looking for a solution that integrates different granularity levels of mobility. The studied granularity levels have been flow mobility [P1], address-family agility [P8], local mobility [P4] and network mobility [P9]. The problem field connects to securing the mobility state establishment and update signalling between end hosts and intermediate nodes [P3][P2][P6][P4][P5][P9]. An additional requirement has been backward compatibility with existing, i.e. legacy, applications [P3][P8].

One of the main research goals concerns supporting mobility and multi-homing in parallel [P3][P2][P1]. Basically, the mobility functionality implies that an end point is serially reachable via different locations. The multi-homing functionality makes the end point reachable via multiple locations in parallel. Supporting simultaneous communication through different access technologies [P3][P1] is part of the multi-homing problem field. The focus on the mobility and multi-homing research has been above the link layer.

The multiple access networks may support different IP versions, IPv4 and/or IPv6. This has resulted in a study of hand-offs and inter-communication between different IP versions from the perspective of applications and network layers [P8][P7]. From the application viewpoint, the main goals are socket-API backward compatibility and transparent address-family hand-offs [P8]. For intermediate nodes, the most interesting problem is identifier-based locator translation [P7].

All the research goals mentioned above are bound to secure and efficient mobil-

ity state management at end hosts and intermediate nodes. The author has also studied the problem field of identity privacy related to the mobility state management [P6][P4][P5].

3.2 Side Requirements

Determining a balance between security and mobility-management signalling overheads has resulted in a set of side requirements. Some essential goals have been to minimize signalling and to provide optimal packet forwarding paths between end hosts. To reduce mobility-management signalling and latency, lightweight state establishment and update procedures at end hosts and intermediate have been studied [P5][P4]. An end-host related goal has been to achieve an alternative to a public-key-based mobility state establishment and update [P5]. Another goal has been to provide transparent security-association establishment and update between mobile hosts and intermediate nodes [P4].

One of the research goals has been avoiding a signalling explosion in densely-populated moving-networks [P9]. This has also led to a study of optimal packet forwarding paths originated from and destined to moving networks [P9]. Furthermore, optimization of forwarding paths depends on IP-address allocation inside moving networks [P7][P9]. The focus in studies has been on authorization of a mobile router and how it can signal on behalf of its clients [P9].

3.3 Initial Set-up and Environment

Working with the existing Internet architecture sets limits on what we can do, compared to defining a new architecture from the beginning. The guiding principle has been making minimal architectural changes and supporting legacy applications and transport protocols. The research goals together with these constraints led the author to study identifier-locator split solutions and cryptographic host-identifier namespaces. The results are different solutions for dynamic identifier bindings.

Throughout, the aim has been the right engineering balance between the existing Internet architecture and the presented new changes. Chapters 4-6 explain the most essential architectural changes. Some design choices have generated a set of benefi-

cial side effects, while others have caused trade-offs and drawbacks. An example of a side effect is the strong authentication requirement and identity privacy problem created by cryptographic identifiers. Some of the requirements in the publications are contradict each other. This thesis presents a solution that provides one possible balance between the conflicting requirements.

3.4 Non-Goals

This section explains which problems and solutions are not covered by the current research. The author has focused on the identifier bindings between the networking and transport layers, and between the transport and application layers. Dynamic bindings at other layers, like link-layer mobility and process migration are beyond the scope of this thesis.

Adaption of the transport layer protocols (like [16][27]) and applications to a dynamically changing network environment has also been out of the scope. Instead, the goal has been on the application backwards compatibility, in other words, to keep the old APIs and semantics for applications.

Legacy intermediate nodes, like access points, legacy NAT devices and firewalls, are beyond the scope of this thesis. The related traversal issues are discussed, e.g., at the Internet Engineering Task Force (IETF) and Internet Research Task Force (IRTF) HIP working groups. In this thesis, the author assumes that each end host implements the new namespace bindings. The alternative proxy approaches that support legacy end hosts are out of scope for this thesis.

Regardless of these exclusions, the presented solutions and results in this thesis are also meaningful in many contexts that are not discussed in this thesis.

4 Architectural Overview

In this section, the author presents an architectural overview that combines the author's publications together in a seamless way. The author briefly discusses the identifier-locator split approach in section 4.1. In the following sections 4.2-4.4, the architecture is discussed from the *component*, *functionality*, and *implementation* viewpoints. The components work as building blocks for the architecture. The author has divided the components roughly into *end hosts*, *intermediate nodes*, and *protocols*. The end hosts and intermediate nodes interact with each other using a set of mobility state management protocols. Each component implements a set of functionalities. Furthermore, the logical functionalities can be divided into implementation modules. Figure 4.1 (page 65) illustrates a rough example topology of the new architecture.

The presented functionality can be divided into three groups. The groups define functionality for 1) *default Internet routing*, 2) *common mobility state management* and 3) *granular mobility state management*. The first and second group of functionality work as a foundation for the third group. The default Internet routing functionality refers to the existing Internet infrastructure and to its IP routing functionality. The common mobility state management denotes the common functionality that is used in all the different granular mobility approaches (chapter 6). The common functionality for the basic state-establishment is presented in sections 5.1-5.3 and for the state update in section 5.4. The state update includes both the location update and rekeying functionality.

4.1 Identifier-locator Split

This thesis defines a secure architecture for multiple granularity-levels of mobility. The architecture is based on an *identifier-locator split* approach where static and globally-unique host-identifiers are dynamically bound to locators [P3]. The *identifier* and *name* concepts were earlier discussed in section 1.4, on page 23. A *locator* is a non-cryptographic and structured name that is defined at the network layer and used for routing packets between network interfaces. From the semantics point of view, the locators are used for identifying a locator$\overset{n\rightarrow 1}{\longrightarrow}$MAC-address bindings while they are not associated with other identifiers above the network layer. However,

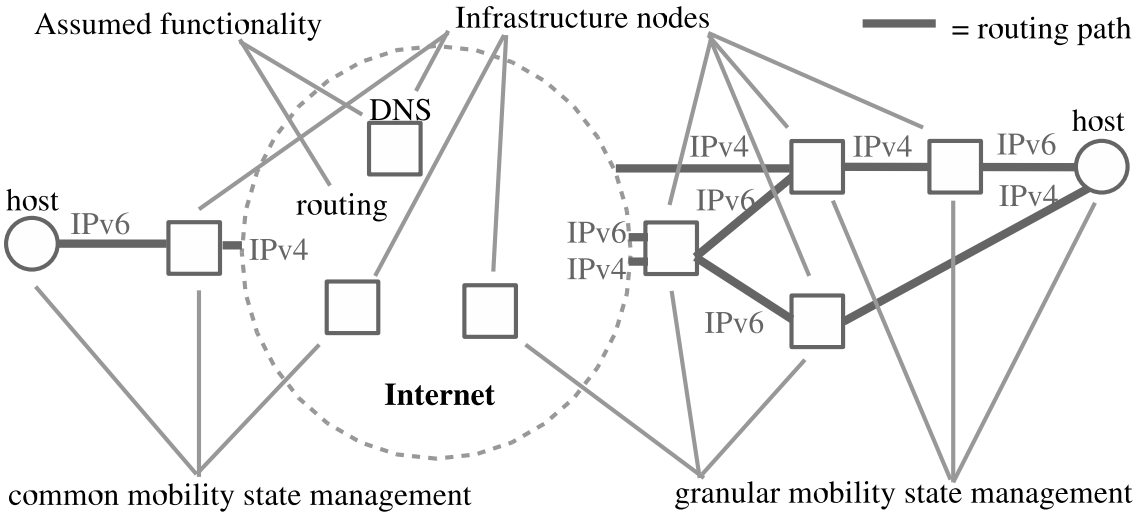

Figure 4.1: A rough example network structure of the architecture.

in the current Internet, IP addresses are used both for locators at network layer and host identifiers above the network layer. This is called semantic overloading of IP addresses. Non-cryptographic names like IP addresses cannot strongly be associated to hosts making it difficult to verify that a specific host is authorized to update a binding. Therefore, this thesis presents alternative ways for host authentication using different kinds of host identifiers. Depending on the presented mobility solution, public keys [P3][P2][P4][P6][P7][P8][P9], Lamport's one-way hash-chains [P4][P6][P5] and structured home addresses [P1] are used for host identification. The solutions are extensions to the basic Mobile IPv6 (MIPv6) [45] and Host Identity Protocol (HIP) [146] protocols.

4.1.1 OPODIS'02 Paper: Research Problem

The semantic overloading and non-cryptographic nature of IP addresses makes it impossible to use them for strong authentication. The problem becomes even worse with mobile and multi-homed host. This kind of hosts have several network interfaces bound to dynamically changing IP addresses. When a host changes its point of attachment to the network or it reroutes traffic from one interface to another, the identifiers associated with a TCP connection are changed. The main problem is that the peer host cannot verify that the new connection identifiers belong to the

same mobile-host. Basically, the peer must have evidence that an identifier belongs to a specific host. An essential problem is that a host using traditional IP-addresses cannot prove that it is authorized to use a specific IP-address at a specific location. A resulting goal is to define the actual identity that is the target of the identification process.

4.1.2 OPODIS'02 Paper: Basic Idea

In publication [P2], the author analyzes the separation between verifiable host-identifiers and non-verifiable location names. The main result of the paper is the relationship between non-cryptographic and cryptographic association-mechanisms. The author argues that a host-identity must consist of a secret that is known only by the authentic host. From the cryptographic association-mechanism point of view, the host must be able to prove to its peers that it knows the secret without revealing it. Therefore, public-key cryptography and Lamport one-way Hash chains are suitable for decoupling the private identity and public identifier from each other [14][110]. It is good to notice that a normal IP address is associated using a non-cryptographic mechanism to a host and it cannot be used for strong authentication. The author also defines a socket association structure for cryptographic identifiers, location names and hand-off related policy identifiers. The dynamic binding between the cryptographic host-identifiers and non-cryptographic location names makes it possible to secure location update messages between hosts. The main limitation of the paper is that it does not contain formal evaluation of the defined bindings.

4.1.3 OPODIS'02 Paper: Relation to Previous Work

The identifier-locator split is discussed earlier, e.g, by Chiappa, Bellowin, Lear and Moskowitz [101][162][53][146]. The author of this thesis analyzes the same problem field from the security point of view in publication [P2]. The analysis has a close relation to the previous multi-homing and mobility work that use non-cryptographic primary identifiers for host identification but depend on TLS or IPsec for host authentication purposes like SCTP, Mobile IPv6, LIN6 and TCP Migrate [149][45][115][173]. As a result, the decoupled security protocols provide secondary cryptographic identifiers for the hosts. The main problem in these kinds of approaches is the separation

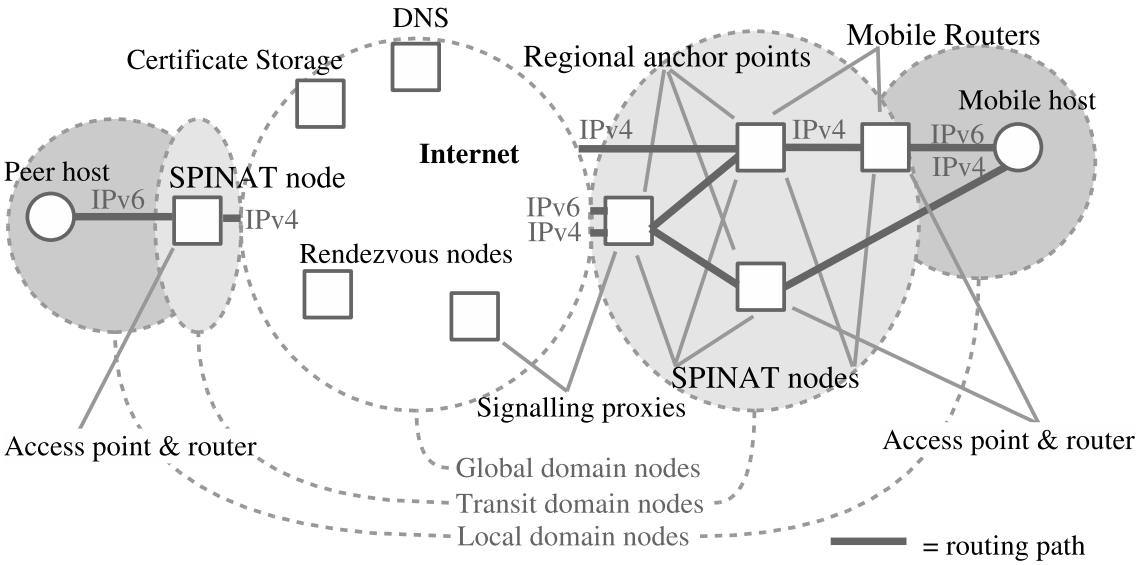

Figure 4.2: Some components in the architecture.

between mobility and security protocols from each other. To increase the flexibility in the mentioned mobility and multi-homing protocols it may be possible to refactor some of them to use the cryptographic identifiers as primary host-identifiers like Cryptographically Generated Addresses (CGAs) in Mobile IPv6 [85].

4.2 Components

The host stack required for interprocess communication is constructed based on the earlier mentioned identifier-locator split approach. The hosts can be named according to their roles, namely, *mobile hosts* and *peer hosts*. In the following context, a mobile host initiates a communication with a peer host. In other words, the mobile host can be called an *initiator* and the peer host a *responder* from the protocol-exchange viewpoint [121]. The intermediate nodes support communication between mobile hosts and peer hosts. It is good to notice that the peer host may also dynamically change its topological attachment to the network and play the role of the mobile host. The nodes discussed in this section are illustrated in Figure 4.2, on page 67.

The hosts require intermediate nodes to communicate with each other (section 4.2.4).

From the host viewpoint, the intermediate nodes can be divided (like in [97]) into local-domain nodes, transit-domain nodes and global-domain nodes (sections 4.2.1-4.2.3). The scope of a locator associated to a network interface defines the scope of the node. The *local-domain* nodes are attached to the same link with the host. The host can negotiate with them using link-local locators. The *transit-domain* nodes provide a private, not globally routable, locator space for their clients. The network interfaces of the *global-domain* nodes are directly attached to the Internet. In other words, the nodes are reachable via globally-routable locators.

4.2.1 Local Domain Nodes

The local domain nodes may support different kinds of wireless and wired access technologies. The most essential local domain nodes are the access points and access routers. An *access point* is an intermediate node that operates at the link layer in the local domain and offers an attachment for nodes to the link [118]. An access point may be decoupled from an access router or it may be integrated with the access router. An *access router* is an intermediate node that operates at the network layer providing IP connectivity to the nodes in the local domain [118]. An access router may be connected to multiple access points. A node runs an attachment exchange with an access point to attach to the local domain (see [87][167][100][11]). Typically, the *attachment exchange* consists at least of authentication and optionally of authorization.

During (or after) the attachment procedure the node communicates with an access router to allocate locators for its NIC. The node may simultaneously be attached to multiple access points via different network-interfaces. A multi-homed node has several locators associated to its network interfaces. Each of the network interfaces may have multiple locator$\overset{n\rightarrow 1}{\longrightarrow}$MAC-address bindings.[7]

4.2.2 Transit Domain Nodes

The same node may work as a local-domain node and a transit-domain node for the hosts. The transit-domain nodes have multiple roles that are 1) local mobility-anchor-points [P4], 2) mobile routers [P9] and 3) static signalling-proxies [P9]. In

[7]The binding between network interfaces and locators is based on Address Resolution Protocol (ARP) and Neighbor Discovery (ND) protocols [47][180][86].

this thesis, all transit-domain nodes implement the Security Parameter Index multiplexed Network Address Translation (SPINAT) functionality [P7][P6]. The *SPINAT* functionality provides overlay routing for host identifiers and identifier-based locator-translation for IPsec ESP protected payload traffic. The SPINAT nodes are located on the end-to-end routing-path, like legacy NAT devices.

The *local mobility-anchor-points* hide the node mobility from the peer hosts and optimize the end-to-end state update signalling. The mobile routers dynamically change their topological attachment to the network, like mobile hosts. Nodes attached to *mobile routers* move together with the routers. The mobile routers can be attached to other mobile routers, thus creating nested moving networks. It is good to notice that a *mobile node* can be either a mobile host or a mobile router. In this thesis, the mobile routers implement the signalling-proxy and access-router functionality. A node may authorize a mobile router or a *static signalling-proxy* to run a mobility state-update-exchange on behalf of it [P9][125].

4.2.3 Global Domain Nodes

The end hosts, local-domain nodes, and transit-domain nodes need support from the global-domain nodes. The global-domain nodes can roughly be divided into the directory services and rendezvous nodes [P3]. The *directory services* provide a name-lookup service and certificate storage. The nodes use the *name-lookup service*, like DNS, to bind a FQDN to a locator set and to an Host Identity (HI). *HI* consists of a public-key-based identifier-identity pair. *Host Identity Tag (HIT)* is a hash value that is computed over a public key of the HI using a one-way hash-function [14][121][P3]. It is typically also stored in the directory service together with the HI [132]. The *certificate storage* provides a way for the nodes to store and retrieve certificates [P9].

The *rendezvous nodes* work as static reachability-points for nodes. The functionality is presented in publication [P3] where the rendezvous nodes are called forwarding agents. Typically, the nodes store their static locators bound to a rendezvous node at the directory service. The rendezvous nodes keep a binding [8] between the HI and the current locator of the node.

[8] See section 5.1.3 for a rendezvous-state initialization for the HI $\underline{n \leftrightarrow 1}$ locator binding.

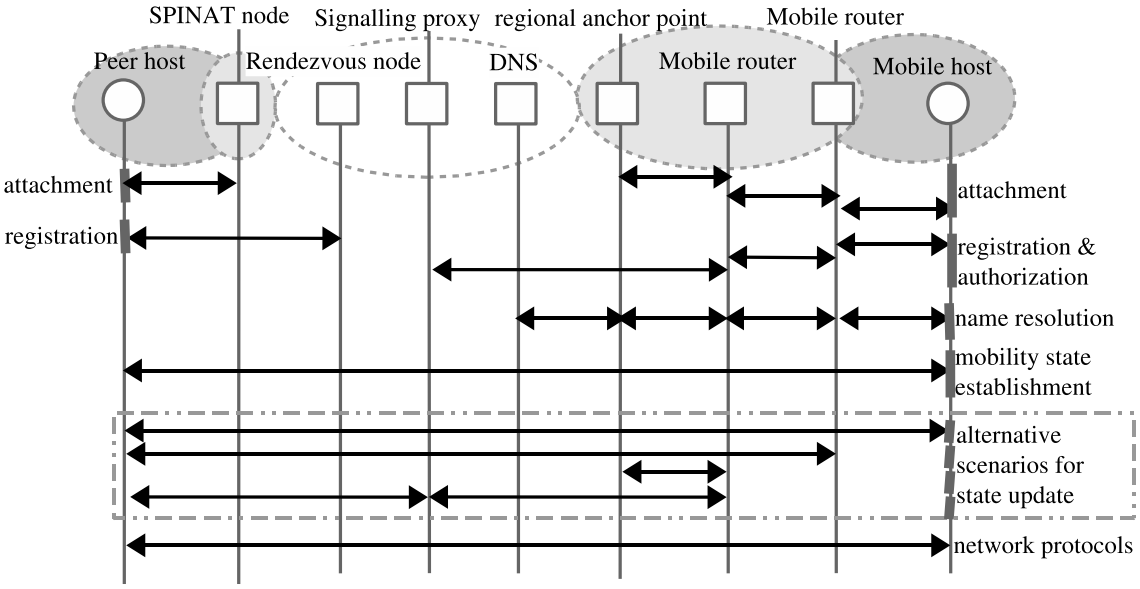

Figure 4.3: Example interactions between components.

4.2.4 Communication Protocols

The communication protocols between nodes can be divided into 1) bootstrapping protocols, 2) control-plane protocols and 3) data-plane protocols. The *bootstrapping exchanges* take places before a node can initiate a control-plane exchange. Furthermore, the *control-plane exchanges* establish and update states for *data-plane protocols*.

The control-plane protocols can be divided into 1) *registration and authorization exchange*, 2) *mobility state-establishment-exchange* and 3) *mobility state-update-exchange*. The data-plane protocols can be divided into 1) *integrity and confidentiality protection protocols* and 2) *network protocols*. In the new architecture, the author has focused on IPsec ESP, IPv4 and IPv6 protocols [166][81][163]. However, it is possible to replace the applied protocols with other protocol families. Figure 4.3 illustrates example interactions between the different components.

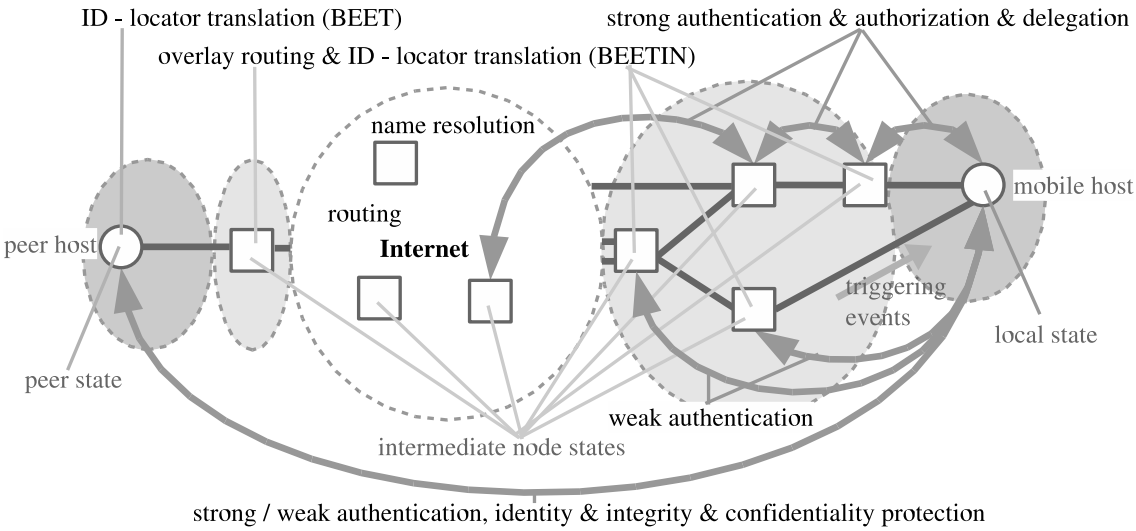

Figure 4.4: Visualizing some parts of the functional architecture.

4.3 Functional Architecture

The previous sections presented the main components in the architecture. The following sections discuss the functional architecture. As described earlier, the architecture is based on the default Internet routing, and on the common and granular mobility state management. The author assumes that the *routing* functionality forwards the IPv4 and IPv6 packets between nodes in the Internet. Figure 4.4 illustrates some parts of the functional architecture.

The *bootstrapping* procedure contains the attachment exchange, locator assignment and *name resolution* (section 2.5.1) required to initialize state establishment and update exchanges (section 4.3.2) with peer hosts. The mobility state establishment results in identifier translation at the nodes (section 4.3.1). The authentication, authorization, and delegation are part of the mobility state management (section 4.3.3). *Authorization* denotes that one identity approves rights to access its resource to another identity [148]. Here, *delegation* refers to the ability of an identity to reassociate its granted authorization with another identity [148].

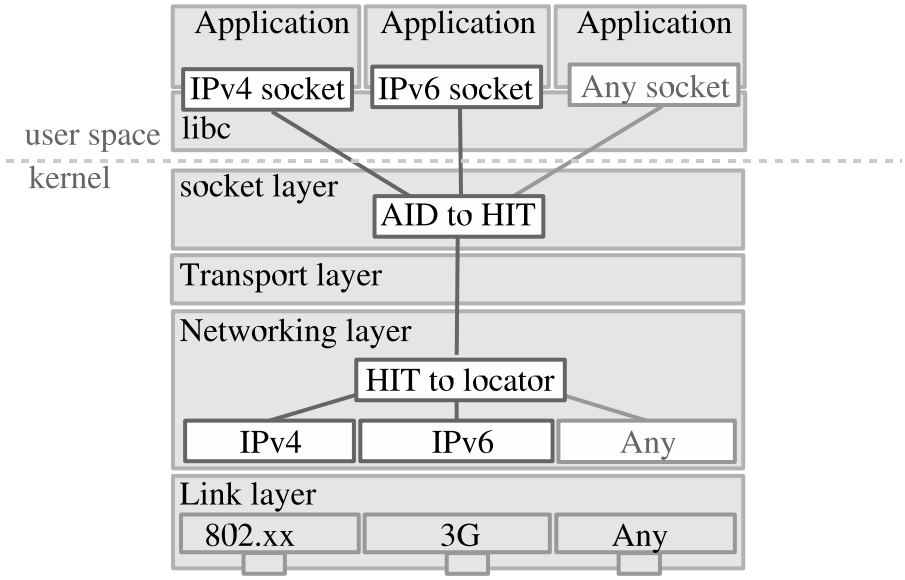

Figure 4.5: Double Namespace Binding at Hosts.

4.3.1 Identifier Translation

An application invokes name resolution to bind a FQDN to an HI and to a related locator-set. The name resolution returns an Application Layer Identifier (AID) to the application, and a HI and a locator set to the local mobility protocol machine. *AID* can be a cryptographic identifier or a non-cryptographic name. The scope of the namespace and the size of the AID defines its collision probability. Depending on the size of AIDs they may be used locally at a host scope or globally in the Internet. From the socket API viewpoint, AID may be structured or flat as long as its size is suitable with the socket API system calls. AIDs replace the IP addresses in the socket bindings at the socket layer. In the scope of this thesis, an AID$\underline{^{n\leftrightarrow 1}}$HIT binding is created at the socket layer.

The translations between AIDs, HIs, and locators are implemented at two different locations in the kernel. The $AID\underline{^{n\leftrightarrow 1}}HI$ translation takes place at the socket layer. The situation is illustrated in Figure 4.5. The translation provides a way to decouple the socket APIs from the transport layer. The second $HI\underline{^{n\leftrightarrow m}}locator$ binding is implemented at the IPsec Bound End-to-End Tunnel mode (BEET) [134][P3] and at the new BEET intermediate node [P7][P9] modes (sections 7.2.2-7.2.4).[9] For

[9]At this point of the thesis it is sufficient to focus on a coarse AID$\underline{^{n\leftrightarrow 1}}$HI and HI$\underline{^{n\leftrightarrow m}}$locator

the purpose of the following sections, the author will use the 'Bound End-to-End Tunnel mode for Intermediate Nodes (BEETIN)' term to refer to the new IPsec intermediate node mode. The new mode differs semantically from the end-to-end BEET mode [134] (see section 5.3.5).

The BEET mode implements the HI$\overline{\overline{n \leftrightarrow m}}$locator translation at end hosts, while the BEETIN mode implements the translation at intermediate nodes. The BEET and BEETIN modes implement HI$\overline{\overline{n \leftrightarrow m}}$locator translation between locator families. The related IPsec [166] policies and Security Associations (SAs) are created during state establishment.

4.3.2 State Establishment and Update

The state establishment can be divided into 1) *local state-establishment* (section 5.1), 2) *peer state-establishment* (section 5.2) and 3) *intermediate state-establishment* (section 5.3). The local and peer state-establishment create BEET SAs at the hosts. The intermediate state-establishment creates new BEETIN SAs at intermediate nodes. The nodes also generate shared keying material during the state establishment. The keying material is used by BEET SAs and optionally by BEETIN SAs to provide *integrity and confidentiality protection*[14] for payload packets [P7].

The received local and network events trigger a state update exchange [P1]. An *event* denotes an occurrence at a host or in a network that is relevant to a mobility or security mechanism [148]. The exchange updates or creates new IPsec policies and SAs. In the following context, the author will use *"SA update"* to describe a situation where the keying material remains the same but the locators bound to the SA are replaced with new ones.

A hand-off event results in a locator update, while the rekeying event updates the packet integrity and confidentiality protection keying material. The state update procedure (section 5.4) can be divided into 1) *local state-update*, 2) *peer state-update* and 3) *intermediate state-update* depending on where the corresponding protocol-machine is located. The architecture also provides functionality for surviving from a *state loss*. A protocol machine may loss its state at hosts or participating intermediate nodes.

---

binding granularity. In practice, the logical HI$\overline{\overline{n \leftrightarrow m}}$locator binding is divided into HI$\overline{\overline{1 \leftrightarrow n}}$VIID, VIID$\overline{\overline{n \leftrightarrow 1}}$SPI and SPI$\overline{\overline{n \leftrightarrow 1}}$locator bindings that are discussed later in section 7.

4.3.3 Authentication, Authorization, and Delegation

The nodes authenticate each other during state-establishment and state-update exchanges. The authentication functionality presented in this thesis can be divided into *weak authentication* and *strong authentication* (section 5.2.2). A weak-authentication solution for hosts is presented in publication [P5] and for intermediate nodes in publication [P4]. A strong authentication solution supporting identity protection is presented in publication [P6].

Weak authentication uses computationally more lightweight algorithms compared to strong authentication. Weak authentication is open for certain Man-in-the-Middle (MitM) attacks but protects the nodes from some CPU related DoS attacks. A *MitM* attacker that is located on the routing path between hosts intercepts or selectively modifies packets to masquerade as an end host or some intermediate-node that is involved with the connection [148]. The strong authentication is based on public-key cryptography [14]. The presented weak and strong authentication techniques are self-identifying and do not require Public Key Infrastructure (PKI). *Self-identifying* denotes that each host generates an identity for itself. The authentication is based on proving the ownership of the specific identity with Lamport one-way hash chain [110] and public-key [14] techniques. *PKI* denotes a system of Certificate Authorities (CAs) that form a set of certificate based authentication and authorization mechanisms [148]. *CA* is an identity that issues certificates. A *certificate* is a formal binding between data and identifiers that are associated together using CA's signature. A *signature* is created by a cryptographic mechanism that associates a public-key pair and some data together [14].

The authentication functionality is extended to support *identity protection*. Identity protection mechanism allows a host to determine which other nodes are able to verify its identity. The strong authentication and related public-key cryptography are used for *delegating binding update rights* [P9] between nodes (section 6.4).

4.4 Implementation Architecture

The previous sections discussed the components and the functional architecture. In this section, the author will briefly describe the architecture from the implementation point of view. The functionalities can be divided into implementation-modules.

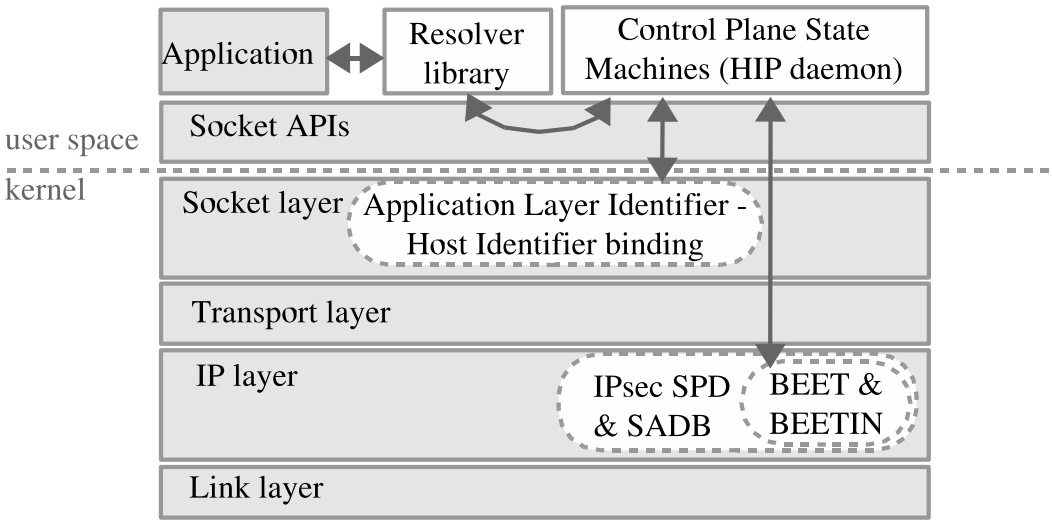

Figure 4.6: Some essential implementation modules.

Figure 4.6 illustrates the most essential implementation modules at the nodes. The implementation is roughly divided into *user-space* and *kernel* parts. Hosts and intermediate nodes use the same code-base. Thus, nodes can dynamically take new roles in the architecture. The author and his colleagues have adopted parts of the extreme programming [102] methods during the implementation process.

4.4.1 AID↔HI Translation

To initialize a connection, an application typically makes a DNS query using the *resolver-library* function-calls. The DNS library returns an HIT and a locator set for each peer-host. The library returns the Local Scope Identifier (LSI) and/or HIT to the application. A globally unique HIT is used with the IPv6 socket API, while a local LSI replaces the IP address in the IPv4 socket API. The resolver library also sends the HI and the received locator set to the HIP daemon. The daemon initializes the AID$\overset{n\leftrightarrow 1}{\underline{\quad}}$HI binding at the socket layer (section 7.2.1). In addition, the daemon initializes an outgoing IPsec policy for the peer's HI.

4.4.2 HI↔Locator Translation

The HI$\overset{n\leftrightarrow m}{\longleftrightarrow}$locator translation is implemented at the BEET and BEETIN modules in the kernel. The corresponding two, new IPsec SA types identify the required translation for each packet. In the outgoing BEET SA case, an outgoing packet carrying HITs matches with an IPsec policy. The policy is bound to a corresponding SA that implements the translation from HITs to locators. In the incoming case, a received ESP packet is decrypted and the locators translated to HITs. The resulting plain-text packet is sent back to 'ip6_input' processing without the ESP header. Hereafter the message is processed as a basic incoming IPv6-packet.

In an incoming BEETIN SA case, the locators in the incoming ESP packet are translated to HITs in the same way as in the BEET mode. However, the ESP packet is not decrypted but it is sent to 'ip6_output' processing. The same outgoing ESP packet matches with an IPsec policy. The policy is mapped with an outgoing BEETIN SA that does not encrypt the packet in the basic case but only translates the HITs to locators. The IPsec policies and SAs are also used for traffic flow management at multi-homed nodes (section 6.1).

5 Common Mobility and Multi-Homing Functionality

This chapter covers the common mobility and multi-homing functionality for four granularity levels of mobility discussed later in chapter 6. The following discussion is based on the components and functionality presented in the previous chapter 4. The different stages of the mobility-management are presented in chronological order. Initially, a node joins a network (section 5.1) before initiating communication (section 5.2) with peer nodes. The intermediate nodes on the packet forwarding path establish states (section 5.3) during the communication initialization. Later on, the node dynamically updates the related states (section 5.4) before destroying or losing them.

5.1 Joining the Network

The bootstrapping exchanges take place before a node can initiate communication with its peer hosts. The bootstrapping can be divided into host-identity generation, network attachment, service discovery and registration at local and transit domains and rendezvous state initialization at a global domain. These different phases of the bootstrapping procedure are discussed in the following sections 5.1.1-5.1.3.

The exact role of a node does not need to be fixed during the local-state initialization. Instead, each node may dynamically change and take new roles in the architecture. A node may simultaneously act, e.g., as a mobile router, a rendezvous node and a local mobility anchor point for other nodes (sections 4.2.2-4.2.3). However, the applied security models, like pre-defined Access Control List (ACL) and public-key sharing [14], set limitations to the dynamics in the architecture. *ACL* is a mechanism that authorize nodes to access a system resource based on enumerated identifiers [148].

5.1.1 Host Identity Generation

Before a node joins a network it generates a set of identities for itself. The semantics of identities are discussed in publications [P2] and [P3]. In the following context, the

author will assume that an identity is a private key, while the identifier is the corresponding public-key or any cryptographic derivate of it, e.g., a hash chain [14][110]. Values in one-way hash chains can work as identifiers for the identity. Publication [P6] uses public keys, while publication [P5] is an example of using one-way hash chains to identify a host.

The identities can be divided into *long term* and *temporary* identities based on their lifetime. Furthermore, the long term identities can be classified into *public* and *private* identities. An *identity protection* mechanism presented in publication [P6] defines a solution for supporting private identities in the Internet. A node may store its long term identifier in a directory service but hide the same identifier from the intermediate nodes during the state-establishment and state-update exchanges. Thus, the classification of identities is bound to the location and time. The public identities can be used, e.g., with ACLs, while temporary identities provide anonymity for the nodes. The *private* and *anonymous* identities differ mainly in that lifetime of the anonymous identities is typically shorter than lifetime of private identities. Anonymous identifiers are not stored in a directory service, and they are typically used per purpose [P6].

An essential benefit of the public-key-based Host Identities used in this thesis is that they can be applied to authorization and delegation mechanisms (see [131]). In addition, the HIs are bound to application layer identifiers and locators using policies. Here, a *policy* denotes a set of rules used by a mobility mechanism to determine a binding between identifiers. The policy-based bindings makes it possible to support different granularity levels of mobility. The applied policies may be locally defined, or they may be initially defined by other nodes, e.g., by operators [P1].

5.1.2 Network Attachment, Local Service Discovery and Registration

Once a node has initialized a local state, it is able to attach to an access network and to assign locators from the access routers. The attachment, service-discovery and service registration exchanges are logically separate protocols. From the protocol-design point of view it is possible to integrate them into a single protocol exchange at the local domain (like in [87]).

The service discovery is related to the moving-network approach presented later in section 6.4. The mobile routers work as access routers for mobile nodes. When the

mobile routers work as access points for their clients, they may use beacons to advertise their services. *Beacon* is a small data frame that is broadcast by a wireless access point to inform mobile nodes about its existence. In this case, the service registration exchange can directly be integrated with the attachment exchange. However, when the mobile router is located in the transit domain, several hops away from the access point, the attachment exchange cannot be used for the service registration. Instead, the mobile host may use a network layer service discovery protocol to locate the mobile router and the provided services.

A mobile node runs a registration exchange with mobile routers and optionally with some rendezvous nodes when it joins the network. The node registers its current location on the rendezvous nodes. During the registration exchange the node may also establish transparently a rendezvous state at on-the-path SPINAT nodes. A *transparent state establish* denotes that the mobile node does not run an explicit exchange with the SPINAT node but the SPINAT node creates a state by intercepting the end-to-end traffic (discussed later in section 5.3). After running the registration exchanges the node is able to initialize communication and receive connection attempts from other hosts.

5.1.3 Rendezvous State Initialization

A node may have prior knowledge of its global-domain rendezvous nodes or it may dynamically discover the rendezvous service located in the Internet. In the former case, the identifiers of the mobile nodes are pre-configured at the rendezvous node. Moreover, the latter case is out of scope for this thesis and is analyzed, e.g., by Yen et al. in [195]. The dynamic service discovery and registration to rendezvous nodes located in the transit domain are discussed later in section 5.3.7.

The author assumes here that the node has prior knowledge of the locations and the identities of the global domain rendezvous nodes. The information is stored in the local state during initialization. The node also stores the static locators bound to the rendezvous nodes on a directory service, e.g., in the basic case on the DNS.

Correspondingly, the static rendezvous nodes may store the host identifiers for authentication purposes. The basic authentication model in the architecture is based on self-identification that does require PKI. The security associations between the mobile hosts and the rendezvous nodes are established with the authenticated state

establishment exchanges (section 5.2.2). It is still good to notice that PKI and ACLs can be applied with the identifiers [P9].

5.2 Initiating End-to-End Communication

The previous section discussed events that take place before a node can initiate communication with its peer hosts. This section discusses name resolution and the mobility state establishment exchanges in sections 5.2.1-5.2.2. Typically, the name resolution is used to bootstrap the state establishment exchange between hosts. The author describes alternative state establishment exchanges in publications [P6][P3][P5]. The corresponding strong and weak authentication exchanges are discussed in sections 5.2.3-5.2.11.

5.2.1 Name Resolution and Global Service Discovery

In this thesis, *global service discovery* is based on name resolution and on the DNS infrastructure. However, it is possible to replace DNS with DHT (e.g. [158][174][191]) and P2P based (like [151]) directory services. *Name resolution* maps FQDNs to HIs and locators. The HIs identify peer hosts running a requested service. The locators may be bound to NICs of rendezvous nodes or peer hosts. FQDN is not mapped to a location of a host running a required service but to an HI that implements the required service. This thesis is based on the many-to-one FQDN$\overset{n\leftrightarrow 1}{\longleftrightarrow}$HI binding [132].

It is also possible to have one-to-many FQDN$\overset{1\leftrightarrow n}{\longleftrightarrow}$HI bindings. As a result, the semantics of FQDNs is changed. A FQDN may be mapped to multiple HIs that are running the same service. In this case, name resolution provides backwards compatible representations of the HIs to applications running in a semantically old environment (see section 4.4.1). The applications assume that all the received representation of the HIs are mapped to the same peer host. Although, the different HIs may identify end points that are not reachable at the same topological-location.

Name resolution may also return a pre-signed state establishment message of the peer host to optimize the state establishment latency (e.g. [126]). In that kind of approach, the directory service logically runs a part [10] of the state establishment

[10]The part of the exchange where the responder remains stateless.

exchange on behalf of the peer host. However, using the DNS for this purpose may be controversial and is out of scope for this thesis.[11]

5.2.2 Alternative State Establishment Exchanges

The name resolution works as an initial step to reach the peer host. The resulting mobility state establishment exchange can be triggered either right after the first datagram has been sent to the socket or after the transport-layer state has been established. The latter approach is called a delayed state establishment (like the SHIM approach [49]) and it is out of scope for this thesis.

Publications [P3][P6] present alternative state establishment exchanges that are triggered by the first outgoing payload message. The benefit of this approach is that the hosts are able to authenticate each other before the payload traffic starts flowing. In addition, the control-plane exchanges can be used to establish states at intermediate nodes. The trade-off with this kind of an approach is that the state establishment causes delay for the transport layer connection establishment.

The state establishment exchanges can be divided into strong [P3][P6] and weak [P5] authentication exchanges based on the used cryptographic techniques. The state establishment exchanges can be negotiated directly between the hosts, or via rendezvous nodes, called forwarding agents in [P3]. Optional bidirectional routing via rendezvous nodes can be used to provide location privacy for hosts [P6]. The resulting location privacy is strongly bound to identity protection.

The state establishment exchange based on strong authentication refers here to the HIP base exchange presented in [P3] and to its privacy enhanced version BLIND presented in [P6].[12] HIP is explained in the following sections 5.2.3-5.2.5 and BLIND in sections 5.2.6-5.2.8. Both of the protocols are two-round-trip authenticated key exchanges providing some DoS protection for the responder [189]. The key exchanges are used to authenticate the hosts to each other and to create IPsec BEET SAs at end

[11]The response messages stored at a DNS are updated frequently. The dynamic DNS [138] does not solve the message update problem because DNS proxies typically cache the earlier made query for some period of time.

[12]It is good to notice the one-to-one relation between the BLIND messages in Figure 1 in [P6] and the HIP base-exchange messages in Figure 13 in [P3]. Basically, the 'Trigger' message is mapped to I1, the 'Challenge' message is mapped to R1, the 'Response' message is mapped to I2, and the 'Conf' message is mapped to R2.

hosts and BEETIN SAs at SPINAT nodes (section 5.3.2).[13] The public-key-based authentication is also used for authorizing intermediate nodes and for delegating state update rights between the nodes (section 6.4).

It is good to notice that in all the three state establishment exchanges presented in [P3][P6][P5], the responder remains stateless during the first protocol round-trip. All the protocols implement a similar message structure, making it easy for the intermediate nodes to establish states. Each message in the exchanges carries the host identifiers of both end hosts. However, the weak authentication protocol in [P5] differs from the two others in that it uses Lamport one-way hash chains instead of public-key technology to authenticate hosts to each other.[14] To keep the exchange lightweight, it does not implement the Diffie-Hellman key exchange either. Thus, the weak authentication exchange cannot be used to share symmetric IPsec keying material between hosts. The weak authentication mechanism provides an alternative for the public-key-based authentication. Basically, the hash-chain-based state establishment exchange, presented in [P5] and discussed later in sections 5.2.9-5.2.11, cannot be used for key-sharing and authorization in the architecture.

5.2.3 NDSS'03 Paper: Research Problem

In the current Internet architecture, an IP address represents both a host's identity and the host's topological location. The processes at the application layer are bound to sockets and the sockets are identified with IP addresses and port values. This kind of structure binds the processes to a specific topological location, thereby making host mobility, and multi-homing difficult. This has led to several security problems, including the so-called address-ownership problem [123]. The address-ownership problem denotes the non-cryptographic association mechanism between IP addresses and devices attached to the network that makes it difficult to authenticate hosts during hand-offs. The main problem is to define an IP mobility and multi-homing protocol that is both efficient and is not vulnerable for re-direction and related DoS attacks (section 1.6.2). Here, efficiency denotes the minimal amount of control signalling during hand-offs and topologically shortest routing paths.

[13]The BEET and BEETIN SA modes were earlier explained in section 4.3.1 and SPINAT in section 4.2.2.

[14]It is good to notice the one-to-one relation between the HIP base exchange messages in Figure 3 in publication [P3] and the weak authentication messages in Figure 1 in Publication [P5]. Basically, the FLoTI message is mapped to I1, the FLoT message is mapped to R1, the SLoTI message is mapped to I2, and the SLoT message is mapped to R2.

5.2.4 NDSS'03 Paper: Basic Idea

In publication [P3], the authors analyze how mobility, multi-homing and security are integrated in the HIP protocol. Logically, HIP introduces a new protocol layer between the transport and network layers. The sockets are no longer named with IP addresses but with separate derivatives of public-key-based host identifiers. The endpoint which holds a particular private key is typically a device, but can basically be a smaller entity, like an application, or a larger entity, like a computer cluster. The logical layer is used for managing the HI-to-locator bindings. The dynamic binding between HIs and locators make it easy to implement host mobility and multi-homing. Due to the public-key-based host identifiers, it is possible to secure the HIP signalling with cryptographic signatures. This kind of cryptographic association mechanism between HIs and locators solves the earlier mentioned address ownership problem because the hosts are not identified with non-cryptographic IP addresses but with public-key cryptography.

From the signalling point of view, HIP consists of the base and update exchanges. Basically, the base-exchange is an authenticated Diffie-Hellman key exchange protocol that uses the public-key based host-identifiers for authenticating hosts to each other. Additionally, the exchange establishes a set of security associations between the hosts. The established IPsec ESP security associations are used for protecting the integrity and confidentiality of the payload packets. In addition, some keying material is used for protecting the mobility and key management messages between the hosts. After the security association are established, a mobile host may run the update exchange to inform the peer hosts about the location of the mobile host during hand-offs. The same packet format is also used for rekeying purposes. In the mobility case, the three-way update-exchange is used for verifying the mobile host's location. Basically, the peer verifies that the mobile host is reachable at the claimed location. The reachability needs to be checked to prevent re-direction and related DoS attacks. To overcome the reachability problems, the authors present a forwarding agent concept in the paper. The forwarding agents are denoted here as rendezvous nodes.

From the payload traffic point of view, HIP uses a new kind of IPsec mode. In standard IPsec, the *transport mode* is used for establishing a secure communication channel directly between any two communication endpoints. Moreover, in the *tunnel mode*, the endpoints of a tunnel are typically not the same as the communication

endpoints. In other words, the tunnel-mode related security associations are bound to different IP addresses than the sockets. For the purposes of HIP, Nikander et al. have proposed a new IPsec mode, denoted as Bound End-to-End Tunnel (BEET) mode[134]. The new mode is a combination of the tunnel and transport mode, using the transport mode packet format but providing limited tunnel mode semantics.

In practice, the BEET mode represents the logical HIP layer in the TCP/IP stack and processes the required HI-to-locator mappings in the host stack. The solution does not change the header structures in IP packets but just the details of the packet handling at the hosts. The HI namespace imposes changes only to the logical packet structure. That is, each packet must logically include the pair of host identifiers. To optimize the payload header size, the IPsec Security Parameter Index (SPI) is used to replace the host identifiers in the ESP protected packets.

From this thesis point of view, the main result of publication [P3] is the identified set of bindings required for implementing mobility and multi-homing in a secure and scalable way. The analysis of bindings is presented through this thesis, especially in chapter 7. The main limitation of the approach is that it requires changes to both communicating end hosts. In addition, some applications that transmit the application-layer identifiers as referrals in the payload data does not work in all cases.

5.2.5 NDSS'03 Paper: Relation to Previous Work

In the following, the author briefly analyzes different mobility and multi-homing protocols that are essential from publication [P3] point of view. Stream Control Transport Protocol (SCTP) is a multi-homing approach at the transport layer where each process is associated with a set of IP addresses [149]. In addition to host multi-homing, some mobility-extension have been presented to support dynamic IP address updates in SCTP [116]. Since basic SCTP [149] does not support static host identifiers like home addresses are used in Mobile IPv6, solving the security issues in a scalable way may be even harder than with Mobile IPv6 [45]. TCP migrate [173] is another transport layer solution allowing the TCP endpoints to migrate from one IP address to another. The main limitation with the transport-layer oriented solutions is that each transport layer protocol must define its own mobility and multi-homing mechanism. Thus, network-layer approaches are more

flexible from the backwards compatibility point of view, because most of them can be used together with the current transport layer protocols. For example, Mobile IPv6 [45] can be used together with the existing TCP and UDP protocols.

In Mobile IPv6 [45], each mobile host is identified with a static home address. The mobile host's home agent assigns the home address to the mobile host. Whenever the mobile host changes its topological attachment to the Internet, the binding between its actual location (care-of address) and the home address is updated at the home agent and at the peer hosts. When the HIP related mobility and multi-homing ideas were published in [P3], some proposals presented the usage of multiple home addresses and multiple care-of-addresses in the Mobile IPv6 context [P1]. One of the experimentation was Homeless Mobile IPv6 that replaces static home agents in Mobile IPv6 with a dynamic set of IP addresses. However, the work did not properly address the involved security problems; instead, the security considerations lead to the definition of the address ownership problem [123] and thereby paved the road for the Mobile IPv6 security solution [45].

One essential difference between HIP and Mobile IPv6 approaches is the authentication model used with location updates. In Mobile IPv6, the security is largely based on the so-called Return-Routability (RR) test where challenge packets are routed via two different paths between end hosts after each hand-off. In addition, to increase the security in binding updates, there have been proposals to use Cryptographically Generated Addresses for home addresses in Mobile IPv6 [175]. Basically, the public-key based CGAs correspond the host identifiers in HIP.

LIN6 [115] is another IPv6 based approach for host mobility and multi-homing. At the time when the author was working with publication [P3] the largest unsolved problems in LIN6 were related to the scalability aspects on the security side. The address update messages were protected with IPsec, thereby requiring some kind of global infrastructure in order to establish the required security associations. The author believes that it is possible to use principles of CGA to generate host identifiers also in LIN6 and therefore apply the security principles presented in [P3] also to LIN6.

Basically, the main difference between the HIP approach and the most essential related work is the seamless integration of the strong authentication mechanism into host-mobility and multi-homing in the HIP solution.

5.2.6 Security Workshop'04 Paper: Research Problem

To provide location privacy for hosts it is possible to route packets via a rendezvous node that hides the actual location of the endpoints. When locators are decoupled from HIs, location translation at the rendezvous nodes alone does not provide identity protection for hosts that are identified using the fixed HIs. As a result, the focus is moved to protection of public-key-based HIs and HITs during end-to-end signalling.

The identity protection problem is essentially related to two-round-trip Diffie-Hellman key exchange protocols that use public-keys for mutual authentication [14]. More precisely, the basic problem is related to the order of messages that carry the Diffie-Hellman parameters and public-keys used for verifying the signed Diffie-Hellman parameters. To avoid message spoofing, the end hosts must authenticate the Diffie-Hellman parameters with signatures. However, the signatures cannot be authenticated without knowing the public key.

Another problem is to keep the responder stateless during the first round-trip of the key exchange without opening the protocol for MitM attacks. The main reason for this is to force the initiator to do computationally more work than the responder before the responder generates shared keying material with the initiator. To achieve this kind of situation, the initiator cannot send the Diffie-Hellman [14] parameters in the first message. Instead, the responder sends its Diffie-Hellman parameters to the initiator in the response message and waits until the initiator has generated a shared Diffie-Hellman secret. However, this kind of approach causes a trade-off between message authentication and identity privacy.

To achieve identity privacy, the responder's public-key cannot be transmitted in a clear form together with the Diffie-Hellman parameters to the initiator. However, without the responder's public-key the initiator cannot verify the signature received in the second message of the key exchange. From the identity protection point of view, it is possible to send the responder's public-key in an encrypted form in the final (fourth) message. However, from the authentication point of view, it is a problem because the initiator cannot authenticate the second message and the Diffie-Hellman parameters in the message may be generated by a MitM attacker. This kind of MitM attacker is able to reveal the initiator's public-key sent in the third message. In other words, an active attacker can easily find out the initiator's

public key because the second message cannot be fully authenticated.

5.2.7 Security Workshop'04 Paper: Basic Idea

To mitigate identity leaks during connection establishment, the author presents a two-round-trip Diffie-Hellman key exchange protocol in publication [P6] that protects the identities of the hosts from passive and active Man-in-the-Middle (MitM) attacks and polling attacks under certain assumptions. The limitations of the protocol are discussed later in this section. In a *passive wiretapping attack*, a MitM attacker intercepts packets without the knowledge of the end hosts and without altering the packets [148]. An active MitM attacker may even alter messages to analyze the behavior of the target hosts. In this kind of *polling attack*, the attacker sends spoofed messages to the target host to reveal its identity.

The solution is based on a two-round-trip authenticated Diffie-Hellman key exchange protocol that authenticates the end hosts to each other and creates a security association between the hosts. The author has focused on HI protection during the end-to-end key exchange. The privacy of payload data is based on the ESP confidentiality and integrity protection [165].

The basic idea of the protocol is to use a nonce to compute additional hashes ('blinded' HITs) of the HITs at the initiating end host. The blinded HITs act as pseudonyms. Here, a *pseudonym* denotes an identifier that cannot be linked together with the original identity by a MitM. The original HITs are replaced with a puzzle in the first key exchange message. The puzzle consist of the 'blinded' HITs and the nonce. The responder uses a brute force method to solve the puzzle. The result of the puzzle is the responder's original HIT. From the responder's point of view, the time that is consumed for finding a solution to the puzzle directly correlates with the number of HIs of the responder.

It is good to notice that the original HITs and HIs are not carried in plain text in the key exchange messages. Thus, the responder's HIT can be used as part of the key-material generation. This property is applied with a delayed authentication mechanism to protect both the HIs. Here, *delayed authentication* denotes a mechanism that buffers a message until the required authentication information is received in later messages. The delayed authentication provides a way for encrypting the HIs between the end hosts without revealing the original public key material

to the MitM.

The novel part of the presented protocol is that it does not require the end hosts to initially know the public keys of each other. It is sufficient that only the initiator knows a hash of the public key of its peer. The actual public keys are transmitted in an encrypted form as a part of the protocol. For the author's best understanding, the presented protocol is the first end-to-end DoS resistant *two-round-trip* Diffie-Hellman key exchange protocol that offers identity protection for both communicating peers and protects hosts from passive and polling attacks based on the following limitations.

The strength of identity protection is directly bound to the strength of the puzzle. The strength depends on the difficulty of finding the responder's original HIT or HI. The first assumption is that an attacker is not able to find the responder's identifier with help of the location information. The relationship between identity and location privacy is analyzed in [P6][119]. Secondly, an attacker is not able to make an educated guess of the responder's public key; e.g., based on traffic analysis [106]. The difficulty of making an educated guess is related to the location of the initiator and to the moment of the BLIND exchange run. An attacker may use social engineering to figure out that the initiator visits a certain host at a specific time. Thus, the on-the-path attacker can listen to the BLIND exchange and use prior knowledge of a public server's identity to verify the educated guess made.

To increase the strength of privacy protection it is possible to distribute the identifiers of the peer hosts to initiators using an out-of-band mechanism. This can be achieved in the following steps. Firstly, the peer host does not publish its public keys, or derivatives of them, in a public directory service. Secondly, the peer host shares the public keys via a protected channel with the initiators. However, this kind of initial public-key sharing can be implemented only in closed networks. The problems in the Internet are related to publishing the identifier and retrieving the identifier from a public directory service. A MitM attacker may be located between the initiator and the directory service. To protect the identifiers from this kind of attackers, the initiator must establish a protected communication channel with the directory service, e.g., using the same BLIND protocol. However, bootstrapping the public-key sharing between the host and the directory service is out of scope for this thesis. In addition, making the brute force attacks harder by 'blinding' the HITs already at the DNS is for further study.

It is good to notice that non-malicious intermediate nodes are not able to learn the public keys of the end hosts either because the BLIND exchange hides the public keys (HIs) from them. There is a trade-off between identity protection and signature verification at intermediate nodes. To overcome the problem in the local mobility context, a weak authentication mechanism is discussed later in section 6.3.

5.2.8 Security Workshop'04 Paper: Relation to Previous Work

In the current Internet, the NAT devices and dynamically allocated IP addresses provide some identity protection for the end hosts [136][143]. However, in the IPv6 case, the situation is worse because the host part of the IPv6 address is not typically translated on the routing path making it difficult to provide identity protection. To overcome the identity and location privacy problems it is possible to use privacy proxies [157]. Such usage necessitates that the proxy is trusted to keep the end host's identity secret. However, this kind of privacy proxies fall beyond the scope of publication [P6]. As long it is possible to use random IP and link layer addresses the problems related to IP address tracking more or less disappear [25]. Therefore, the author uses the forwarding agent to hide the actual location of the end hosts. As earlier mentioned, it is not enough to focus on the location privacy when hosts are identified with public-keys during end-to-end key exchanges.

Basically, it is possible to achieve identity protection for both end-hosts using public-key encryption [4][69]. However, such approaches are vulnerable to CPU related DoS attacks where attackers bomb the responder with pre-generated messages. Therefore, public-key-encryption-based approaches are out of scope for publication [P6]. From the related work point of view, the Internet Key-Exchange protocol (IKE) [44] and the early version of JFK [7] are more close to the author's work. However, IKE does not provide DoS or identity protection against active and passive attacks for both end hosts [44]. To overcome the limitations of IKE, Aiello et al. presented JFK. However, they argue that "it is essentially impossible under current technological assumptions to have a two-round-trip protocol that provides DoS protection for the responder, passive identity protection for both parties, and active identity protection for the initiator." [7]. Contrary to their argument, the author of this thesis managed to define a key exchange protocol (BLIND) that fulfills those requirements under the presented set of limitations.

5.2.9 CMS'04 Paper: Research Problem

Security in Mobile IP and Mobile IPv6 based solutions is based on security associations between the mobile host and its home agent. While a single mobile host may establish an IPsec security association with its home agent it is difficult to achieve a scalable key-sharing solution with local mobility approaches (section 1.6.4). Many mobility and multi-homing protocol proposals assume the presence of some security infrastructure, e.g., PKI or pre-shared secret. However, the usage of PKI and IPsec based authentication mechanisms in the Internet scale results in key-sharing and scalability problems. In some cases, public-key computation requires too much CPU processing power from small mobile-devices. The main research problem is to overcome the scalability and processing-power limitations and to find a security solutions that can be used to verify a location of a mobile and multi-homed host in a secure way.

5.2.10 CMS'04 Paper: Basic Idea

In publication [P5], the authors introduce a mobility and multi-homing state-establishment exchange that utilizes Lamport one-way hash chains and does not require pre-existing security information between hosts. The procedure is known to be vulnerable to an active Man-in-the-Middle attack in the first message exchange. However, the procedure is efficient, and does not have inherent scalability problems.

The presented approach is based on delayed authentication like TESLA [3] where each message is buffered until the related HMAC-key is received in the subsequent message. The difference between TESLA and [P5] is that the presented approach is not intended for broadcast messages and does not depend on global synchronized clocks. Instead, the protocol in [P5] replaces the clocks with the interlocked release of hash-chain values from the end hosts. Similar to the clock ticks in TESLA, this can be used to enforce a causal order on the events, which is required for delayed authentication. In other words, both end hosts use hash chains for message authentication where the values of the hash chain are used for HMAC-keys.

The two-round-trip state establishment exchange can be divided into two phases. During the first round-trip the mobile host informs the peer host about its two locations. In addition to locators, the message contains the host identifiers and a nonce that are protected with HMAC. Once the responder receives the first message,

it generates a temporary hash chain based on the values received in the first message together with a locally generated secret. The same secret is used with multiple initiators for some period of time. Thus, the responder does not need to create a state per initiator during the first round-trip. Using a value of the temporary hash chain, the responder computes HMAC over the initiator's HMAC and the host identifiers. In other words, the responder's HMAC protects the initiator's HMAC in the second message.

The second round-trip of the exchange is routed via the second attachment point of the mobile host. Basically, the third message carries the information of the first and second messages and additionally the anchor value of the mobile host's hash chain. The responder uses the received host identifiers, locators and nonce for reconstructing its hash chain and verifying the two HMACs. After validating the message, the responder's replies with its anchor value to the initiator. The main purpose of the protocol is to exchange the anchor values between the hosts.

The protocol is based on opportunistic authentication denoting that the hosts do not have prior knowledge of each other. The first round-trip of the exchange is vulnerable to an MitM attack where an attacker can replace the responder's message with its own. To mitigate this kind of MitM attacks, it is possible to include a public-key of a host as part of the hash chain generation. The same public-key or hash of it can be used as host identifiers. As a result, each host may sign the first message and associate an anchor value to its identity in a strong-cryptographic way.

Publication [P5] presents also a three-way exchange that uses the initially bootstrapped hash values for updating locator information. The exchange is based on the same delayed authentication mechanism as the two-round-trip state establishment protocol. The first message of the locator binding update exchange updates or destroys a locator at the responder's context. The second and third messages are used for making a reachability test for the new locators.

Publication [P5] illustrates two different use cases where the presented protocol can be applied to. In the first use-case, the protocol is used for managing a multi-homing situation directly between end hosts. The second use-case illustrates an approach for establishing a security context between the mobile host and a local-mobility anchor host via a Mobile IPv6's home agent. The main result of the approach in [P5] is the presented weak authentication mechanism that uses hash chains for mutual authentication. The novel part of the solution is the delayed authentication

that provides causal order on state establishment and update events in the mobility and multi-homing context.

The presented approach [P5] contains certain limitations that are analyzed by Heer in [177]. The most essential limitation of the author's protocol are related using the same hash chain for sending locator-update messages and for acknowledging messages. This kind of approach opens the solution for a MitM attack. When both end hosts move at the same time the causal order of the events may be broken by a MitM. Basically, a MitM can delay the messages in both directions and generate a spoofed location-update messages that are protected with authentic hash values.

In addition, the Lamport one-way hash-chains used in [P5] do not offer an efficient authorization mechanism like public-keys used in [P9]. This is a limitation from the network mobility point of view because a mobile host cannot use the presented exchange for authorizing a mobile router for sending location update messages to the peer hosts on behalf of the mobile host (section 6.4).

5.2.11 CMS'04 Paper: Relation to Previous Work

Publication [P5] is based on the author's earlier Weak Identifier Multi-homing Protocol (WIMP) work [98]. The recovery procedure from a state loss event defined in [98] can also be applied to the presented protocol in [P5]. The original idea of delayed authentication was presented in Interlock Protocol by Rivest and Shamir but the protocol was broken by Bellowin and Merritt in [19]. The close relation with TESLA [3] was analyzed briefly in the previous section. In addition, Anderson et al. present a stream authentication protocol called Guy Fawkes in [8] that uses short hash chains for authenticating and integrity protecting subsequent messages. However, the protocol is not analyzed in the mobility and multi-homing context. Heer has done independent work called Lightweight Authentication for the Host Identity Protocol (LHIP) in [177]. From the author's work point of view, Heer analyzes and elaborates the ideas of [P5] and [98]. The LHIP work contains clever observation about using dedicated hash chain for location update exchanges and for acknowledgement messages [177]. In addition, Heer proposes usage of multiple pairs of hash chains to support parallel exchanges. Another excellent observation is an idea of sending HMAC values of the protected data in the first message before sending the actual data in a subsequent message. This is also called pre-signing procedure in

the Heer's work. As a result, it is possible to send multiple location updates parallel between peers.

5.3 State Establishment at Intermediate Nodes

The previous sections discussed communication initiation between end hosts. In this section, the author describes the SPINAT solution presented in publication [P7]. The basic idea of the solution and its relation to the previous work is presented briefly in sections 5.3.1-5.3.3. The author analyzes and elaborates the approach in sections 5.3.4-5.3.7 to show how it works together with the rest of the solutions presented in this thesis. In addition, the author discusses the SPINAT state establishment from the address-family-agility point of view in Appendix A.

5.3.1 SecureComm'05 Paper: Research Problem

The different overlay routing approaches decouple the endpoint identifiers from locators using new namespaces above the network layer. Overlay routing is based on the dynamic binding between the endpoint identifiers and locators at intermediate nodes called overlay routers. It is possible to find an analogy between the overlay routers and Network Address Translation (NAT) devices. All of them translate locators using additional namespaces above the network layer. From the host identification point of view, the identity (IP address) of the endpoint appears to be changed at NAT traversal. In other words, the current NAT practice does not only translate locators, but it also changes the apparent identity of the communicating parties. To overcome the authentication related problems in the overlay routing, the hosts should be associated to identifiers that can be used for authenticating the end hosts to each other. For example, IPsec provides a cryptographic namespace for that purpose.

From the overlay routing point of view, the IPsec security associations (SAs) uniquely identify the endpoints. Therefore, the SPI values carried in the ESP packets can be used for endpoint identifiers. From the IPsec ESP viewpoint, the main problem is the dynamic binding between the SPI values and locators at the intermediate nodes. One essential problem is that the SPI values in the ESP headers are integrity protected with keying material that is only known to end-hosts. However, due to

the 32-bit long SPI-namespace the overlay routers must translate the SPI values on the routing path to avoid collisions. Another problem is that the receiver defines the SPI value that the sender uses towards the peer host. This is different from the legacy NAT approach where the source port values, used for multiplexing IP addresses, are defined by the senders. Basically, the author focuses on a solution that is independent of the transport layer protocols like UDP header format.

5.3.2 SecureComm'05 Paper: Basic Idea

In publication [P7], the author integrates IPsec into overlay routing using new kinds of intermediate nodes, called SPINAT nodes. The author proposes that the IPsec control-plane signalling is used to create and update HI$^{\underline{n\leftrightarrow m}}$locator and SPI$^{\underline{n\leftrightarrow 1}}$locator bindings at SPINAT nodes. Basically, the control-plane signalling can be divided into key exchange and rekeying exchanges. The HI$^{\underline{n\leftrightarrow m}}$locator bindings are also called triggers and the SPI$^{\underline{n\leftrightarrow 1}}$locator bindings are part of the communication context in [P7]. Each receiver registers HI$^{\underline{1\leftrightarrow 1}}$locator binding at one or more SPINAT nodes to be reachable from the Internet.

The registration can be divided into transparent and explicit HI$^{\underline{n\leftrightarrow m}}$locator binding establishment at SPINAT nodes. In the former case, the mobile host runs an key exchange with the peer host while the on-the-path SPINAT nodes establish bindings based on the intercepted messages. In the explicit registration case, a mobile host knows the locator of the SPINAT node and runs a key exchange with it. In both cases, the SPINAT learns also the required SPI$^{\underline{n\leftrightarrow 1}}$locator binding during the key exchange procedure.

In the presented approach, the SA update exchange is integrated with the locator update exchange. Basically, the SPI value is changed when a mobile node creates a new SA. On the other hand, the locator is changed when a mobile host changes its topological location. It is good to notice that the two events, rekeying and readdressing, are not required to take place simultaneously. The only static information related to a communication context at each SPINAT node is the end-to-end host-identifier pair. The SPINAT nodes must authenticate each SPI-to-locator binding update to avoid re-direction and DoS related attacks. The dynamic SPI$^{\underline{n\leftrightarrow 1}}$locator bindings at SPINAT nodes makes it possible to translate address realms and even address families. For this purpose, the author defines a new IPsec mode called SEET

mode in [P7](here called BEETIN mode). By decoupling host identifiers from locators at end-hosts it is possible to dynamically update SPI$\overset{n\leftrightarrow 1}{\longleftrightarrow}$locator bindings at SPINAT nodes without breaking the transport-layer connections at the end hosts. The locator translation at SPINAT nodes is based on the destination locator and on the SPI value carried in the ESP packets.

In a basic scenario, a mobile node that is located in the local domain initiates a key exchange with a peer host that is located in the global domain. In a more sophisticated scenario both of the hosts are located in local domains behind SPINAT nodes. Basically, the hosts establish transparently states at the on-the-path SPINAT nodes during end-to-end key or rekeying exchanges. The HI-to-locator binding is used for routing IPsec control-plane messages between end hosts. The SPI-to-locator binding is required for routing ESP payload packets between end hosts. Basically, the control-plane messages initiate a set of bindings at SPINAT nodes that are required for overlay routing ESP protected payload packets between end hosts.

The main result of the solution in [P7] is the novel way of using SPI values for multiplexing locators at SPINAT nodes and for identifying end hosts using the same SPI values. An essential limitation of the solution is that it requires some changes to the existing key exchange protocols and optionally also to IPsec ESP integrity protection computation. However, the author believes that the required changes in IPsec can gradually be applied in new overlay network infrastructures. On the other hand, the solution does not work with the already deployed legacy NAT devices without additional UDP tunneling support [136]. In addition, access control at SPINAT nodes is out of scope for this thesis.

5.3.3 SecureComm'05 Paper: Relation to Previous Work

It is good to notice that the presented transparent registration is used, e.g., by local mobility anchor points in publication [P4] while the explicit registration is required by mobile routers and signalling proxies in publication [P9]. In the following, the author briefly presents some essential related work.

From the locator translation point of view, there is a clear analogy between the overlay routers and legacy NAT devices [136] . Both of them translate IP addresses, but using different namespaces for the connection identifiers. From the endpoint identifier point of view, overlay routers, e.g. in IPNL [62] and DataRouter [182],

use new namespaces for locator translation. One frequently cited overlay routing architecture is the Internet Indirection Infrastructure (i3)[174]. i3 defines an overlay routing mechanism for multicast, anycast, and mobile communication. Packets are always routed from the sender to the receiver via rendezvous servers, called i3 nodes. Overlay routing is based on a new endpoint identifier namespace. The i3 architecture uses the existing Internet routing infrastructure for delivering packets between the i3 nodes. i3 suffers from the basic security vulnerabilities that are related to location updates and confidentiality protection of the traffic. However, Stoica et.al. propose that the host identifiers can be generated from public keys, and public key cryptography can be used to secure the i3 architecture [174]. That is very similar to what the Host Identity Protocol (HIP) offers [P3]. Adkins et.al. solve several security vulnerabilities related to i3 in their security enhanced approach, called Secure-i3 [5]. Secure-i3 is a framework that does not go into protocol level details. Nikander et.al. have continued the work by defining Host Identity Indirection Infrastructure (Hi3) [126]. The author of this thesis has continued that work in [P7] and presents a solution that can used for integrating confidentiality and integrity protection of payload packets into different kinds of overlay routing infrastructures like i3, Hi3, DOA, Layered Naming Architecture, FARA, PeerNet, and UIP [174][126][188][17][39][55][60].

5.3.4 Transparent State Establishment and Identity Theft

The presented transparent state establishment mechanism in [P7] is vulnerable to an identity theft. The problem occurs in a situation where an attacker tries to use the victim host's host identifier to establish inconsistent states at SPINAT nodes. However, the SPINAT node is designed to discard an inconsistent soft state when the peer host does not reply to a spoofed message. Here, a *soft state* denotes that a performed state transition can be discarded if a specific event (e.g. a message) is not received. In such a case, the instance of the protocol machine is not destroyed but the state machine returns to the previous state. The creation of soft-states at the SPINAT nodes may still result in a DoS situation from the mobile host's point of view. The attacker may try to bomb a SPINAT node with spoofed packets to establish inconsistent states before the authentic mobile host manages to establish states at SPINAT nodes. Publication [P4] presents different mechanisms to protect from this kind of identity theft attacks. The author will briefly analyze the essential problem field in the following.

Public key signatures [14] do not directly provide a solution to the problem. An attacker may implement a replay attack, and establish a soft state at the SPINAT node. In a *replay attack*, an attacker intercepts, records and retransmits the message [148]. An increasing sequence number does not help either if the mobile host loses its mobility state between the packet recording moment and the replay attack. When time-stamps are used together with signatures they provide a way to protect from replay attacks. In that case, the SPINAT nodes accept the UPDATE messages containing the latest time stamp. Publication [P4] presents an alternative protocol that does not use public-key cryptography for message authentication.

One solution is to use randomly changing binding identifiers in the binding update messages. Publication [P6] describes a way to implement dynamically changing HITs in the control-plane exchanges. In that approach, the HITs are changed during each control-plane exchange even if the identities at the end hosts remain the same. As a result, it is difficult for an attacker to implement identity theft because the mobile host changes the visible identifiers during each update exchange. The transparent SPINAT state establishment and identity protection problems are related to local mobility discussed later in section 6.3

5.3.5 BEET Intermediate Node (BEETIN) Functionality

After the hosts and the on-the-path SPINAT nodes have established communication states, the ESP[165] protected payload traffic starts flowing between the hosts. This section discusses the ESP protected payload traffic translation from the SPINAT's point of view. The rough functionality of the BEETIN mode is presented in [P7] (there called SEET mode).

The end hosts create the initial BEET SAs during the state establishment exchange. The HIP mobility and multi-homing mechanism presented in [135] defines a way for creating SAs between multi-homed hosts. An incoming and an outgoing SA form a so-called SA pair in HIP. The end points may have multiple SA pairs between them. The SAs in the SA pair are bound to the same locators. In other words, they define the end points for a routing path. The SPI value in each SA is selected by the ESP packet receiver.

The SPINAT nodes handle SAs also in pairs. To keep the discussion clear, the author has named the SA pairs as a private SA pair and a public SA pair. This

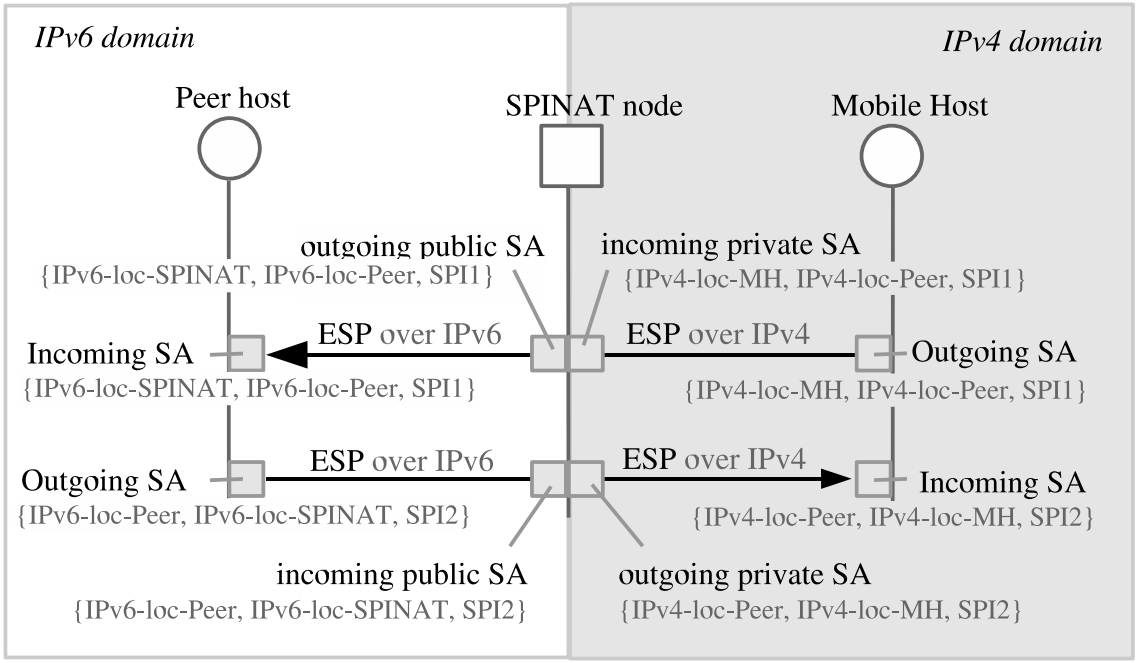

Figure 5.1: Private and public side SAs at SPINAT nodes.

is illustrated in Figure 5.1. The SAs in the pair share the same locators. The destination and source locators in the incoming private SA are in reverse order in the outgoing private SA. The same analogy applies also to the public SA pair.

In the present approach, each outgoing SA has also a corresponding IPsec policy. The author and his colleagues Melén and Salminen have extended the policy mechanism to support SPI based IPsec policy identification. Each BEETIN related IPsec policy is identified with a HIT pair and an SPI value to have an unique mapping between IPsec policies and SAs. This is discussed later in section 5.5.5.

The SPINAT node creates an incoming public SA using the mobile node's incoming SPI value. That is, the peer node's outgoing SPI value. The source and destination locators are the same that were verified with the last outgoing translated control message. The same SPI value is also used with the outgoing private SA.

The incoming private and outgoing public SAs use the peer host's incoming SPI value. That is, the mobile host's outgoing SPI value. The destination and source locators in the private side SAs are bound to the same locators that were carried in the first control message sent by the mobile node. The SPINAT node creates the

SAs after forwarding the last message of the state establishment or update exchange.

The incoming ESP packets are destined to incoming SA processing and forwarded to the outgoing SA processing as presented in section 4.4.2. The locator family translation for ESP protected packets is discussed in Appendix A.4.

5.3.6 SPI Collisions During Locator Translation

Publication [P7] presents alternatives to implement SPI based locator multiplexing and tackles the SPI collisions at the SPINAT node. In addition to presented solutions, the SPINAT node can add a parameter to the control-plane messages carrying SPI information. The parameter contains a translated SPI value that the peer host must use towards the SPINAT node. The SPINAT node may sign the parameter with its private key of the HI pair. However, the peer host does not typically know the HI of the SPINAT node. The result of using this kind of self-signed SPI translation parameter is similar to using User Datagram Protocol (UDP) tunneling [P7]. Any on-the-path attacker can translate a UDP port value and therefore cause an ESP packet misidentification at NAT devices. To overcome the problem, the host inside a private domain may authorize the SPINAT nodes like mobile routers to implement SPI translation (section 5.3.7).

5.3.7 State Establishment at Signalling Proxies

In this thesis, mobile routers and static SPINAT nodes that are located in the fixed network implement signalling-proxy functionality. In the presented moving network approach [P9], the mobile node locates the signalling proxy and registers at signalling proxy to use the provided signalling proxy services. To initiate an authorization exchange with a signalling proxy, a mobile node needs to locate the proxy. During the exchange the signalling proxy is authorized to signal on behalf of the mobile node (section 6.4). In the moving network case, the mobile nodes delegate the location update signalling rights to the mobile routers. The mobile routers can delegate the rights further to signalling proxies in the fixed network.

The mobile nodes may use alternative service discovery mechanisms. In this thesis, the main purpose of the *service discovery* is to locate an intermediate node providing a specific service. Basically, the solutions can be divided into solicited, advertised,

unicast and multicast methods. The common thing with the beacon and router-advertisement based approaches is that they work at the link layer. The mobile host may also send solicited requests to unicast and multicast locators to get a response from a service provider.

The service discovery and registration exchange can be implemented either at the link or network layer as earlier discussed in section 5.1.2. The benefit of the network layer approach is that the exchanges are independent of the link-layer protocols. Therefore, they can be applied with any link technology. The trade-off is the increased amount of signalling. Each protocol-round-trip increases the hand-off time. One alternative is to use the network layer exchange as a fall-back mode if the access technology does not support link-layer service discovery and registration.

In this thesis, the author assumes that all the transit domain nodes implement SPINAT functionality and are located on the end-to-end packet forwarding path. Therefore, the definition, evaluation and analysis of different service discovery protocols is out of scope for this thesis.

5.4 Re-keying and Locator Updates

The previous sections presented the state establishment at hosts and intermediate nodes. This section discusses dynamic, mobility related state update events from the locator update and rekeying points of view [P1][P3][P4]. The goal of this section is to bring up the dependency between the HI, SPI and locator bindings.

5.4.1 Binding between HIs, SPIs and Locators

Traditionally, static IPsec policies, stored at the Security Policy Database (SDP), are not dynamically updated. In such an approach, a single policy rule may cause the instantiation of multiple Security Associations (SAs). However, for the purpose of this thesis the IPsec policies in the SDP are dynamically updated. In addition, policies have a one-to-one relation with SAs. Compared to the traditional policies the new policies contain also SPI values that is discussed in the following.

In the new approach, each IPsec SA is bound to an HIT pair, to a locator pair, and to an SPI value. Furthermore, an IPsec policy is bound to an SPI value, to an HIT

pair and optionally to other transport layer identifiers, like port and protocol values. The SPI bound IPsec policies are discussed later in section 5.5.5. Each IPsec policy has one-to-one binding with an SA that can also be expressed with a Policy$\underline{\overline{n \rightarrow 1}}$SA binding. The policy and the corresponding SA share the same HIT pair and SPI value. The SAs are also bound to groups. A group consists of an incoming and an outgoing SA between an HI pair, and locator sets that are bound to the SAs.

When a multi-homed host is reachable via multiple paths, each path is identified with an SA between end hosts. In this thesis, each SA is bound to a network interface (per HI pair). As a result, each SA identifies one interface, making it easy to bind QoS information to the SA. In addition, the hosts can be made aware of the peer's network interface situation.

A host may dynamically create new, update, and destroy existing SAs. The main issues are related to keeping the local, intermediate and peer states in synchrony in the various multi-homing and locator family scenarios.

5.4.2 Dynamic Policy-to-SA Binding

The mapping between the IPsec policy and the SA defines the binding between the HIs and locators. A policy update event updates the mapping by creating a new policy or by replacing the SPI value in the current policy. A node may dynamically change the SPI value in the policy to renew the policy$\underline{\overline{n \rightarrow 1}}$SA binding. All the connections protected with the same SA can be considered to belong to the same flow. Thus, the IPsec policy defines the flow granularity. This is discussed in more detail in the flow mobility context in section 6.1.

It is good to notice that an SA may have a one-to-many binding with IPsec policies. This may happen in a flow based mobility scenario when several policies are mapped to the same locator pair [P1] that is in the HIP case to the same SA. The HITs may remain the same in the policies and SAs between two hosts but the SPIs in the policies and the locators in the SAs are dynamically changed.

5.4.3 Re-keying and Locator Update from SA Viewpoint

In the HIP approach, the mobility state update exchange consists of three messages [P3][121][135]. The first UPDATE message is sent by the mobile node to the

peer host to inform about the new locator update or rekeying event. The peer host verifies the location update event by sending a challenge message back to one of the mobile node's unverified locations [121][135]. The challenge message verifies that the first message was really initiated by the mobile node and that the mobile node is in the location that was stated in the first message. The challenge message may also contain the peer host's new SPI value in the rekeying event. The mobile node finalizes the exchange by echoing challenge information back to the peer host.

The rekeying procedure replaces the ESP keying material with new material at each end host. As a result, a new SPI value per SA is also created. From a semantic point of view, the rekeying procedure creates a new SA at each node. When the HIT pair and the locator pair remain the same in the current and new SA, the new SA can be said to replace the current SA. In this case, there is a series of SAs associated with the path between two nodes.

A locator update event replaces the locators in an existing SA. The locator update can be separated from the rekeying procedure. A mobile host may change its topological attachment to the network without rekeying. The used locator is replaced with a new one but the keying material and the SPI value remain the same. The SA is updated because the SPI uniquely identifies the SA.

5.4.4 Triggering Re-keying and Locator Update Exchanges

A change in the communication environment [P2] may trigger a rekeying or location update exchange [P1]. *Triggering events* are pieces of information that originate from different *event sources*, like from network, from users or from applications. In practise, this information is collected from the different parts of the TCP/IP stack. The approach taken in this thesis is based on the location update and rekeying related events originating from the link and network layers. Figure 9 in [P1] describes an implementation architecture that handles events received from a variety of sources.

In this thesis, different event sources are registered on a policy driver. The *policy driver* collects information from the different parts of the TCP/IP stack, relevant to the connections associated with the flow control policies, and compares the events against information stored in the policies. The rules in the policies are here called *actions* according to KeyNote Trust-Management System[114]. Basically, a *policy*

is a certificate that consists of a list of actions signed by one HI. An *action* is an operation that is defined by some HI and is controlled by the local mobility-management mechanism. Actions specify a set of network interfaces to be used for connections based on the requirements of the HI. The connections are identified with transport layer identifies in the associated actions. The mobility-management mechanism evaluates actions against received events and decides which interface is to be used with a specific connection. The selection between network interfaces can be based on conditional statements. For example, "use Wireless LAN if it is cheaper than GPRS".

It is also possible to divide a policy into source and destination policies based on the source and destination TLIs expressed in the actions. In the HIP case, a TLI 3-tuple consists of an HIT, a port number and a protocol value. A good design choice is to let an HI that is reachable via a specific locator also to define an action for its locators. However, the actions can be defined locally, remotely by an operator or by a peer host [P1]. The policy based flow mobility is later discussed in section 6.1. The policies can be stored in a user-space *policy database* in certificate format. The applied policies are also stored in a *policy cache* in the kernel in a format that allows fast database access. Whenever a received event matches a policy rule in the policy cache, the policy driver forwards the event to a mobility-protocol state machine.

Changes in the experienced QoS level at the used link, SA time-outs, or changes in the default routers are examples of events that may result in a mobility state update exchange. However, unauthenticated event sources may be a problem. Spoofed events may result in subsequent state update signalling. This may further generate extra control-plane signalling, and cause signalling bursts, e.g., in the moving network case (section 6.4).

5.5 Design Issues in Re-keying and Locator Updates

In the following sections, the author has collected a group of design issues that are related to the rekeying and locator update events. The rekeying problems are bound to SA identification, namely, to frequent rekeying (section 5.5.1), to frequent network leaving and joining events (section 5.5.3) and to locator multiplexing at SPINAT nodes (section 5.5.5). The locator update problems are related to locator selection at multi-homed nodes (section 5.5.7) and at mobile routers (section 5.5.8).

5.5.1 Problem: SA Identification during Frequent Rekeying

This section illustrates a problem related to the current three-way rekeying handshake. A *handshake* denotes an agreement between two hosts that is based on a set of exchanged messages. In the basic scenario, a mobile node initiates a rekeying exchange with a peer node. The peer node replies with a challenge message, and starts waiting for the final acknowledgement message from the mobile node. From the protocol machine's point of view, the *final acknowledgement message* denotes the event that completes a soft state to a hard state.

However, it is possible that the mobile node sends out a valid acknowledgement message to the peer host but the message never reaches the peer host. There can be different reasons why the final acknowledgement message does not reach the peer node. A link-layer connection may be lost right after the mobile node has sent out the acknowledgement message, or an intermediate node may drop the message. This may result in SA synchronization problems, when a fast moving mobile node creates new SAs during each hand-off. Typically this may happen when the mobile node makes subsequent hand-offs before it receives a resubmitted challenge message from the peer node.

The mobile node's mobility state machine is in a state where the final acknowledgement message was delivered correctly to the peer node but the peer node never received it. After the final re-submission, the peer node removes the created soft state for the update exchange and ignores the exchange. Thus, the peer node does not create new SAs because it does not get the final acknowledge message from the mobile node. As a result, the mobile node uses the new SAs, and the peer node still uses the old SAs. This results in a problem when the mobile node sends an UPDATE message from the new location to the peer node.

The UPDATE message contains the SPI of the current SA and an SPI for the new SA. However, the peer node is not able to identify the current SPI because it is still using the old SAs. The peer node drops the received UPDATE message, which results in a DoS situation. In addition, the mobile node uses the current SA towards the peer node. Therefore, the peer host is not able to identify the payload packets either.

5.5.2 Solution: SA Identification during Frequent Rekeying

To overcome the presented problem in section 5.5.1, it is possible to update the SA pair after receiving the first valid ESP packet at the mobile node. However, the solution works with end-to-end mobility but has certain trade-offs with the intermediate nodes as discussed later in section 5.5.3.

In this thesis, the author presents a new Virtual Interface Identifier (VIID) namespace. Basically, a VIID pair is used to identify a set of connections between hosts. Just like with physical network interfaces, multiple locators can be associated with a VIID. VIID is an abstraction that combines of set of locators together that are associated with a common set of characteristics. For example, a VIID may collect together all locators that are associated with different physical Wireless LAN network interfaces. The VIID concept is discussed later in section 6.1 from the flow mobility point of view. The additional VIID namespace provides also a solution for the earlier presented frequent rekeying problem.

A *VIID-pair* identifies a path between two NICs like an SPI-pair but VIIDs have longer lifetime than SPI values. It is good to notice that SPI values are changed frequently during each rekeying exchange. Therefore, VIIDs can be used to identify a set of connections that are dynamically associated with different SPIs. In other words, a VIID-pair binds together a set of subsequent SAs used to protect the same set of connections. Basically, the VIID$\underline{^{n\rightarrow 1}}$SPI binding is required for SPI agility reasons like HIT$\underline{^{n\rightarrow m}}$locator binding is required for locator agility reasons. The VIID$\underline{^{n\leftrightarrow 1}}$SPI binding is illustrated in Figure 6.1, on page 115. In this kind of approach each rekeying and locator update message contains a binding with a static VIID and a corresponding SPI value. When the peer node suffers from the SA synchronization problems it can replace an old SA with a new SA if they are associated with the same VIID. A previously unknown VIID results in new association between the received VIID and the SPI value.

There is an obvious trade-off between the connection identification and privacy requirements with the VIID concept. It is possible to trace a mobile node based on the VIID value. Albeit, an attacker is able to trace an identity using VIID values, it is not able to reveal the original identity of the host when the BLIND approach presented in sections 5.2.6-5.2.8 is used. It is still good to notice that a mobile host may use the same VIID value just for a short period of time to verify that the peer

host remains in synchrony during frequent rekeying and use a new VIID in the later exchanges. In other words, the host may generate multiple VIIDs that point to the identical set of locators.

5.5.3 Problem: SA Identification after Rejoining SPINAT

Another problem may appear when a mobile node causes churn by leaving and rejoining the same local network. *Churn* denotes numerous independent arrivals and departures by nodes in a network [64]. This may cause SA synchronization problems at the SPINAT nodes.

In this kind of a scenario, the SPINAT node has established a state and the related BEETIN SA pairs for the mobile host. After leaving the local domain, the mobile node runs mobility state update exchange at the new location before rejoining the initial local network. A problem arises when the mobile node also rekeys and replaces the old SA pairs with new ones during each mobility state update. The SPI values in the SA pairs are updated after each rekeying procedure.

Later on, when the mobile node rejoins the initial local network, it runs a state update exchange through the same SPINAT node. The exchange contains information about the mobile host's current (to be replaced) and new SPI values. It is possible that the initially created SAs are still alive at the SPINAT node. The SPI values are old and out of synchrony. The latest UPDATE message does not contain a known SPI value from the SPINAT viewpoint. Thus, the SPINAT node is not able to replace the existing SA pair with a new one. The problem is similar to the problem presented earlier at the peer node side (section 5.5.1) but takes place here at an intermediate node (see also Figure 2.3, on page 42).

5.5.4 Solution: SA Identification after Rejoining SPINAT

To solve the presented problem in section 5.5.3, the SAs should be bound to static VIIDs as described earlier in section 5.5.2. Based on the binding between the VIIDs and SAs, the SPINAT nodes can stay in synchrony with the moving hosts. An unknown VIID carried in the end-to-end UPDATE message results in a new BEETIN SA at the SPINAT node. In addition, the hosts should send keep-alive messages at pre-defined intervals for each VIID. In this way, the SPINAT nodes know when to

release the states.

The SPINAT nodes are able to recover from internal state loss and from the state loss events at end hosts. In the former case, the stateless SPINAT node sends an Internet Control Message Protocol (ICMP) [92][1] destination unreachable message back to the ESP packet sender. As a result, the ESP packet sender initiates a new UPDATE exchange with its peer node. However, if the peer node does not reply to the UPDATE message, the ESP packet sender should start a new base exchange. In the latter case, the end host that recovers from a crash situation initiates a new end-to-end state establishment exchange with its peer hosts. During the recovery procedure, the state machines of the SPINAT node replace the earlier created states with the newly created soft states at the end of the successful end-to-end exchanges.

5.5.5 Problem: SA Identification and Locator Multiplexing

SA identification results in a problem at the SPINAT nodes when multi-homed hosts establish multiple SA pairs between each other (section 5.4.2). Each SA pair is used to identify a traffic flow. The initial SA pair between the hosts is created during the mobility context establishment exchange (section 5.2.2). Later on, the hosts may create additional BEET SA pairs using the state update exchange. The on-the-path SPINAT nodes transparently create new BEETIN SA pairs according to the locator and SPI information carried in the end-to-end update exchange messages.

From the SPINAT's point of view, the host identifier pair gets bound to multiple SA pairs. The multiple traffic flows between hosts may result in SA misidentification if the SPI values only are used to identify IPsec ESP packets at SPINAT nodes. This kind of a situation may happen when a multi-homed mobile host is attached to the same local domain via multiple locators. The situation is illustrated in Figure 5.2, on page 108.

The problem is related to the multiplexed locators at SPINAT nodes. The mobile host's multi-homing situation results in several outgoing SAs at the SPINAT node, each of the outgoing SAs having a corresponding IPsec policy. If the policies are only identified with an HIT pair, the outgoing ESP packets may match with several policies.

The single-homed peer in Figure 5.2 suffers from a similar kind of problem as the

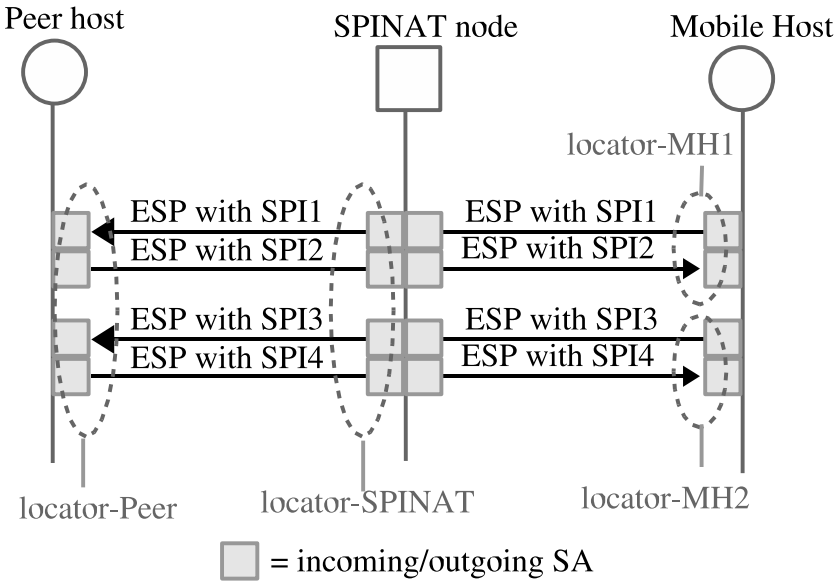

Figure 5.2: An example of having multiple SA pairs between hosts.

SPINAT node, when the multi-homed mobile host creates multiple SAs through the SPINAT node. The result is that the different SA pairs contain the same source and destination locators but different SPI values. It is not enough to use HIT and locator information to bind IPsec policies to SAs.

5.5.6 Solution: SA Identification and Locator Multiplexing

To identify the SAs in the multi-plexed locator case, the approach taken in this thesis uses the SPI value together with the HITs to identify IPsec policies for outgoing packets. The outgoing IPsec policies must contain the same SPI value that is stored in the corresponding outgoing SA at end hosts and SPINAT nodes.

5.5.7 Problem: Direct Route Overrides Locator Selection

The locator selection at nodes can be implemented in various ways. However, studying different locator selection policies and mechanisms is out of scope for this thesis. The locator selection implementation used in this thesis at hosts and intermediate nodes is based on a Longest Prefix Matching (LPM) algorithm. In this thesis, an *LPM* compares the prefixes of source and destination locators with each other. The

compared source and destination locators must belong to the same locator family. The output of the algorithm is a locator pair containing the most common part of the prefix in bits. An efficient way to implement such an algorithm is presented e.g. in [51]. This kind of locator selection approach results in selfish routing decisions at intermediate nodes that are analyzed by Qiu et al. in [142]. In addition, IPv6 [163] locators easily result in better prefix matches than the shorter IPv4 [81] locators. Thus, the algorithm prefers Internet Protocol version 6 (IPv6) locators to Internet Protocol version 4 (IPv4) locators. Each locator is also marked with status information. [15]

However, in some cases the kernel routing table overrides the source locator and route selected by the mobility protocol machine. This may happen for a multi-homed mobile node having two or more network interfaces. One of the network interfaces is attached to the same link with the peer node, while another interface is connected via a SPINAT to the peer node.

When the kernel consults the routing table to select router for the outgoing packet, it finds out that there is a direct link local route to the peer node. Instead of sending the packet via the SPINAT, i.e. the default router, the kernel replaces the source locator with the locator bound to the interface that is attached to the same link as the peer node. The packet is sent out via a different interface than what was defined by the initial source locator selection.

It is still good to notice that this kind of source locator overrule situation happens only when a multi-homed node is on the same link as the peer node. This causes problems for locator update exchange and transparent SPINAT state establishment.

5.5.8 Problem: Private Locator Selection at Mobile Routers

A SPINAT node provides a private locator space for nodes inside the local network. In the IPv4 case, the solution uses locators defined in [194]. For the purpose of the private IPv6 locator space, the present approach uses a pre-defined prefix that is not globally routable. The SPINAT node assigns a static and private locator for its private side interfaces. The public side interfaces are bound to dynamically

[15] A locator may be in UNVERIFIED, TENTATIVE, VALID, DETACHED states. The algorithm prefers locators in VALID state but may also select a locator pair in state UNVERIFIED/TENTATIVE if none of the locators are in the VALID state.

allocated locators. This causes location selection problems at mobile routers that implement the SPINAT functionality. As a result of a hand-off, a mobile router does not have a globally routable locator for a short period of time. However, the loss of locator triggers an event at the mobility state machine (section 5.4.4). The mobility state machine tries to move all the connections bound to the public interface to another interface that has a valid attachment to the network. In other words, the multi-homed mobile router uses its other interfaces to survive from the temporary link loss. As a result, the used LPM algorithm tries to find out a valid locator pair bound to the private interface. Without additional routing logic, the algorithm may provide a private source locator to be used with the peer's destination locator.

5.5.9 Solution: Private Locator Selection at Mobile Routers

To overcome the private locator selection problem presented in section 5.5.8, the LPM algorithm must contain the following extension defined by the author's colleague Melén. *Statically* allocated private locators should never be used with locator selection at mobile nodes. The rule prevents the mobile router from using private side locators with packets that are sent out via the public side interfaces. However, *dynamically* allocated private locators are used in a normal way with globally routable destination locators. Otherwise, the mobile nodes inside private locator spaces could not send out packets. In other words, the locators must be tagged with a flag telling whether they are dynamically or statically allocated.

5.6 Summary

Figure 5.3, on page 111, illustrates the different stages of the common mobility-management functionality presented earlier in this chapter. The initial bootstrapping procedures are required to initiate the end-to-end communication. The presented local service discovery, service registration, and authorization exchanges are used with the local and network mobility solutions later in chapter 6.

During the mobility state establishment the nodes authenticate each other either with weak or strong authentication mechanisms. Identity protection depends on the applied authentication mechanism and whether the routing path goes via rendezvous nodes. It is not enough to protect the identities during end-to-end communication

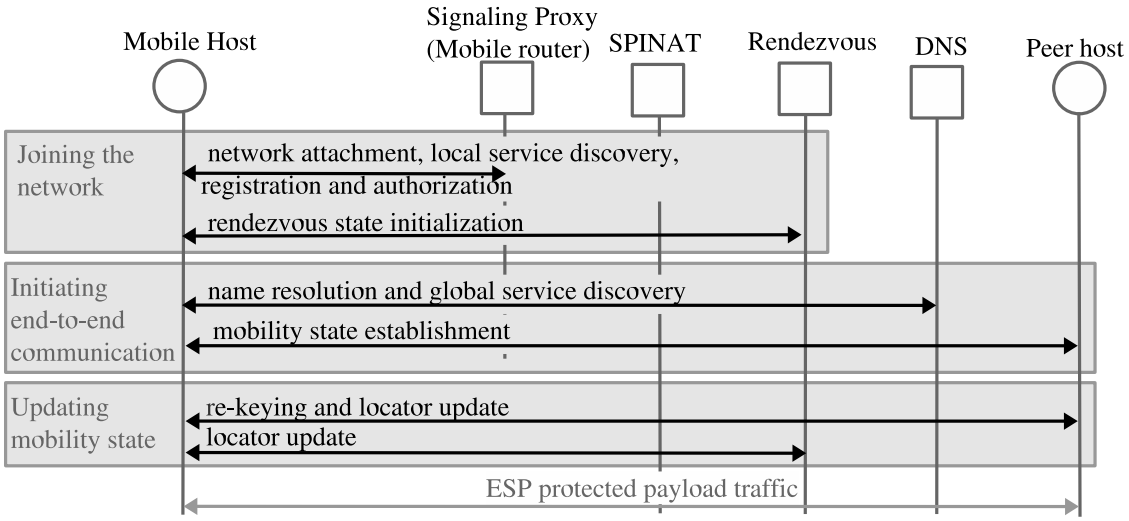

Figure 5.3: The different stages of the common mobility-management functionality.

but also during name resolution and rendezvous state initialization. Later on, the IPsec policies and BEET SAs at hosts are updated during the rekeying and locator update exchanges. In addition, the SPINAT nodes transparently update the IPsec policies and BEETIN SAs during the end-to-end exchanges.

The implementation experiences indicate that an additional VIID namespace is required to identify traffic flows during frequent hand-offs to avoid synchronization problems at end hosts and intermediate nodes. The results also imply that IPsec policies must be identified using both the host identifiers and SPI values when multi-homed end hosts are communicating via intermediate nodes that are using multiplexed locators. Another important observation is to distinguish static and dynamically allocated locators from each other at mobile routers to avoid locator selection problems during moving network hand-offs.

6 Four Granularity Levels of Mobility

This chapter presents four granularity levels of mobility that are flow mobility (section 6.1), address family agility (section 6.2), local mobility (section 6.3), and network mobility (section 6.4). The different granularity levels are based on the earlier presented common mobility state management in chapter 5. The analysis of the solutions is presented later in chapter 7.

A multi-homed host that is able to simultaneously use multiple access technologies can be better reached in the Internet and can better sustain its connections than single-homed hosts [P1][P3]. This requires that the mobility protocol machine is able to dynamically divide connections between different access networks. This kind of flow based mobility [P1] is strongly bound to the compatibility with different IP versions. The existing applications must work regardless of the different IP versions supported in the access networks. The cross communication between different IP versions [P8] is here called address-family agility.

When a group of multi-homed hosts are moving in the same geographical direction they may be attached to the same subnetwork. The benefit of this kind of network mobility [P9] is related to an efficient bandwidth consumption and to lower end-to-end latency during subnetwork-wide hand-offs.

When the hosts are moving inside a subnetwork they can benefit from local mobility [P4]. The local mobility hides the regional movement from the peer nodes. It optimizes the signalling between end hosts which increases location privacy [P6] and minimizes the hand-off related latency.

6.1 Flow Mobility

Here, a *flow consists of one or more connections* that are routed via the same path. In this section, the author describes the flow-mobility solution presented in publication [P1]. The basic idea of the solution and its relation to the previous work is briefly presented in Sections 6.1.1-6.1.3. The main results of [P1] are the identified bindings between static and dynamic identifiers in the host stack. The author discusses the results in more detail from the bindings viewpoint in sections 6.1.4-6.1.10. The discussion elaborates and generalizes the solution to show how it works together

with the rest of the solutions presented in this thesis.

6.1.1 HICSS-36 Paper: Research Problem

A variety of mobility-management protocols support hand-offs between network interfaces. Some of the protocols move all traffic from one network interface to another at once, while some protocols allow simultaneous communication over different network interfaces. However, the solutions presented before [P1] do not propose any means for users or applications to be able to dynamically influence the interface selection during the operation of a mobile node. The earlier solutions combine the hand-off policy and mechanism together where the interface selection has been mostly based on static rules.

6.1.2 HICSS-36 Paper: Basic Idea

Mobile nodes are often equipped with several network interfaces supporting different kinds of wireless and fixed link technologies. The main idea in publication [P1] is to offer the best possible network interface for each connection based on requirements of the different kinds of applications. Changes in the availability or characteristics of an access network behind a network interface may result in a situation where already established connections should be moved from one network interface to another. Publication [P1] presents a network interface selection mechanism for multi-homed and mobile hosts. The mechanism supports hand-offs even on a per connection basis. The local routing is controlled by user-defined rules. The rules define which interface is used for a certain set of connections. The routing decisions are based on the adaptation of the rules into availability and characteristics of the network interfaces and access networks at any given time.

The main result of the solution [P1] is the specific set of identifier bindings that result in the presented policy based flow-mobility mechanism supporting legacy IPv6 peer nodes. For the author's best understanding, publication [P1] presents a novel IP layer solution for flow based hand-offs in a simultaneous multi-access environment. However, the solution has also some limitations. When the authors in [P1] defined the Mobile IPv6 based flow mobility approach, Mobile IPv6 did not support multiple care-of address bindings per home address at the peer node. To overcome the problem, the authors bound multiple home addresses to the same mobile host. From

the peer node's point of view, the home addresses were considered to belong to different mobile hosts. Sockets were bound to home addresses according to the source policy. In other words, the number of home addresses defined the granularity of flows. As a result, an essential limitation of the solution is that the flows are identified with home addresses. In other words, the mobile host must allocate an home address per flow.

6.1.3 HICSS-36 Paper: Relation to Previous Work

In the following, the author briefly presents some essential related work. Since the publication [P1] has been published, a set of flow-mobility proposals have been presented [10][59][169][178]. However, at the time the results in [P1] were published other promising solutions, like LIN6[115], Multi-homed TCP[22] and Stream Control Transmission Protocol (SCTP)[149], did not provide such a flexible policy based flow-mobility mechanism. For example, a mobility and multi-homing protocol LIN6[115] did not provide any means to explicitly select interfaces for different connection in a multi-homed host. Multi-homed TCP[22] did not support fast moving end-hosts or allowed user-defined local routing policies. Basically, SCTP[149] was the only protocol that described a socket API that could be used for implementing interface selection policy at the application layer. However, purely application layer based policy mechanism would result in inefficient implementation. An essential problem with the transport-layer based multi-homing solutions is that the existing TCP and UDP cannot use them. At that time, the Multi6 Working-Group (WG) at IETF was mainly focusing on router based site-multi-homing solutions that increased the amount of required signalling for mobile hosts. Later on, the principles presented in [P1] have been applied to the HIP architecture in [169]. In addition, Monami6 WG at IETF has focused on the same problem field related to Mobile IPv6 [178].

6.1.4 Flow Mobility related Bindings

Figure 6.1, on 115, illustrates a set of bindings from the flow mobility viewpoint. A *flow* is identified with a *VIID* pair where each VIID is bound to a locator set and to one SPI value. The SPI identifies an SA that is bound to one of the locators in the locator set and used with outgoing packets.

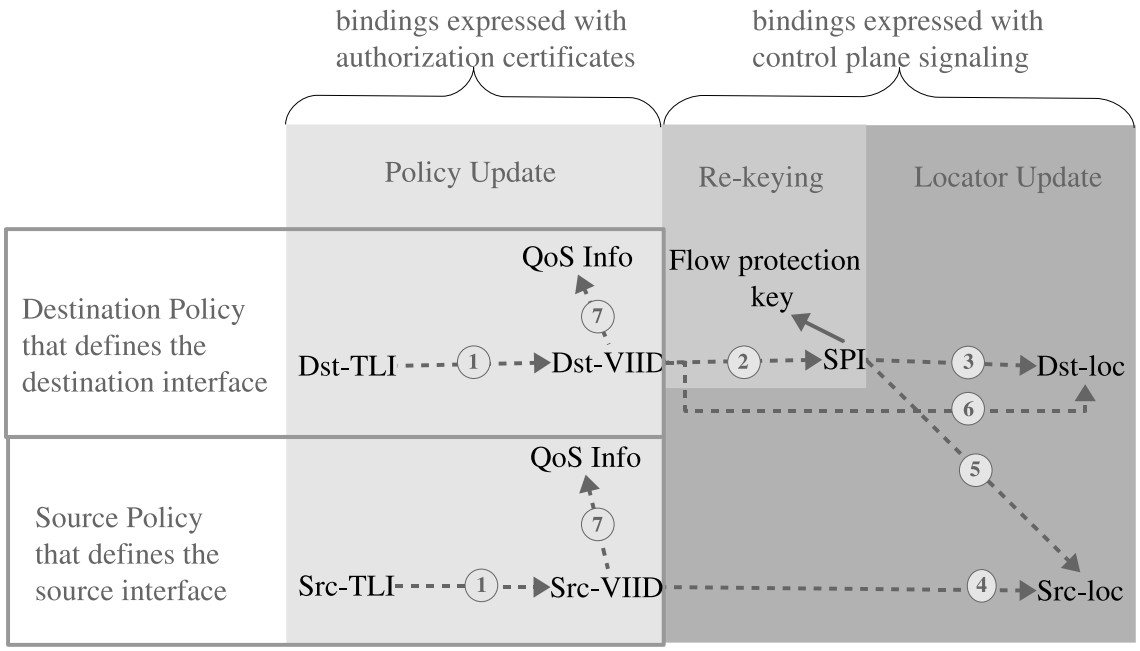

Figure 6.1: The essential identifier bindings in flow mobility.

It is good to notice that in the author's approach SPIs are associated with physical network interfaces. Therefore, the VIID$\underline{^{n\leftrightarrow 1}}$SPI binding depends on the SPI$\underline{^{n\leftrightarrow 1}}$locator binding. Based on this observation, it is possible to define vertical and horizontal hand-offs in the following way. *Vertical hand-off* denotes that the SPI is updated in the VIID$\underline{^{n\leftrightarrow 1}}$SPI binding. *Horizontal hand-off* denotes that the locator is updated in the SPI$\underline{^{n\leftrightarrow 1}}$locator binding. In addition, flow mobility associates a set of connections with a set of VIIDs using the dynamic TLI$\underline{^{m\rightarrow n}}$VIID binding. When the user wants to change a hand-off policy for connections it updates a specific TLI$\underline{^{n\rightarrow 1}}$VIID binding. The number of TLI-pairs associated with a single VIID-pair, consisting of a source and destination VIID, defines the *granularity of a flow*.

The basic idea of flow mobility is to define a hand-off rule set for a node to move a set of connections between alternative paths if the current access networks do not fulfill the needs. A change in the experienced QoS level, e.g., the loss of a connection, is a typical reason for a flow based hand-off. The main principle in the policy expressions is to use identifiers that have longer lifetime than the mobility state machine that uses them. The VIIDs make it possible to dynamically update paths between end hosts without having to update the policies after each hand-off

or rekeying procedure.

In most cases, the policies have longer lifetimes than the locators and SPI values. The locators are changed during each hand-off, and SPIs are bound to SA lifetimes. The short SA lifetimes result in regular SPI updates. Thus, the locators and SPI values cannot be used to identify network interfaces in the policies.

6.1.5 Connection Identifiers at Upper Layers

The flow mobility solution presented in publication [P1] is based on the following 5-tuple that identifies a connection at the socket layer: {source identifier, source port, destination identifier, destination port, protocol}. A *source* denotes the service that sends a packet and the *destination* refers to the service that receives a packet. It is good to notice that the 5-tuple can be divided into source and destination TLIs. A TLI 3-tuple consists of {identifier, port, protocol}. The connection identifiers that are used in the legacy socket APIs are also stored in flow policies (discussed later in section 6.1.7). The coarser the granularity level, the fewer the parameters of the 5-tuple that are defined in a policy. An example of a policy entry is illustrated in Figure 4 in publication [P1].

It is good to notice, that LPM algorithm does not work with flat identifier namespaces. Basically, it would be possible to increase the flow granularity using parameters carried in the upper layer protocol headers, or even in payload data. However, to provide a generic enough flow control for UDP and TCP traffic the solution presented in [P1] is restricted to use the connection identifiers of the 5-tuple. From the policy mechanism point of view, additional protocol namespaces do not change the presented solution.

6.1.6 Expressing Applications in Policies

The global source and destination identifiers of the previously presented 5-tuple remain the same between connections. However, the source ports are dynamically allocated during socket creation. A source port of the 5-tuple can be used to identify a local application. Therefore, the policy engine should keep a table of the bindings between applications and the source ports bound to the applications (see [144]). In that way, users can define the policies before the local application is started, instead

of dynamically updating the source port information to the policy action afterwards. A policy consists of actions where each action defines the preference order of VIIDs that is illustrated in binding (1) in Figure 6.1, on page 115 [P2][24].

For service reachability reasons, the destination port typically remains the same after each application restart. Therefore, the destination port information does not require dynamic mapping at the policy driver (section 5.4.4). However, for human readability reasons both the source and destination port values can be presented as service names like in some firewall rules.

6.1.7 Source and Destination Policies

A header of a payload packet contains the earlier mentioned 5-tuple of connection identifiers when traversing through the host stack. If a match between the connection identifiers carried in the packet and the identifiers stored in a policy is found an action defined in the policy takes place (see also [24]).

As discussed earlier in section 5.4.4, a policy can be divided into source and destination policies based on the source and destination TLIs. *Source-policy actions* are used for finding suitable source locators. Correspondingly, the *destination-policy actions* work for destination locator selection. In practice, the source and destination policy actions can be merged together and expressed in a single policy action. However, it is good to logically differentiate the two kinds of policies. The benefit of having source and destination policies is visible in a distributed flow mobility-management context (section 6.1.10).

6.1.8 VIID-to-Locator and VIID-to-SPI Bindings

In the MIPv6 approach, a source VIID is directly bound to a source-locator set and a destination VIID to a destination locator set. In the HIP case, the VIIDs are bound to a set of locators via SPI values that are further bound to physical network interfaces (like in [24]). The locator set bound to a VIID is typically a subset of all the locators bound to a host. The binding between the VIID and the locator set is dynamic. Basically, a VIID provides a way to bound locators together that have common QoS properties as illustrated in binding (7) in Figure 6.1, on page 115. A host may dynamically create and destroy existing VIIDs. Policies that are bound

to different HIs can use colliding VIIDs.

Each outgoing payload packet is subject to source and destination policy processing. An action in a policy rule defines a priority order for multiple VIIDs. Once matching policy rules are found, the mechanism selects valid source and destination VIIDs, i.e. a VIID pair. The LPM algorithm is used to select the best source and destination locator pair of the associated locator sets that is illustrated in bindings (3)-(6) in Figure 6.1. The resulting locator pair identifies a routing path for a traffic flow.

Each destination VIID is bound to exactly one destination SA per time that is binding (2) in Figure 6.1. Thus, when the binding (6) between the destination VIID and its locator set is updated, the host may need to update also the related binding (2) with SA. It is good to notice that in some cases different destination VIIDs may use the same SA. There is no need to define multiple SAs between the same locators. Thus, in this thesis, the number of destination VIIDs define the maximum number of SAs.

It is possible that the selected source and destination VIIDs may be bound to locators that belong to different locator families. In that case, the sender can prioritize either the source or the destination VIID. In other words, the sender must select the next highest VIID in the priority list to find a valid locator pair for outgoing packets.

> **Example 17.** *For example, a sender may have a source policy saying that all Hypertext Transfer Protocol (HTTP) traffic should be sent via the VIID offering the highest available bandwidth. Currently, that VIID is associated to an IPv4 locator. The destination policy defines that the HTTP traffic should be sent to the cheapest destination VIID that is currently bound to an IPv6 locator. Now, the sender must select the next highest VIID in either of the policy rules to find out a common locator family. However, defining alternative VIID selection algorithms is out of scope for this thesis.*

6.1.9 VIIDs in Payload Packets

The home address information carried in each payload packet provides a way to identify a flow at the receiver side. However, using endpoint identifiers for flow identifiers can be called semantic overloading. A flow identifier does not need to

be a locator. Instead, using the local namespace for flow identifiers makes the mechanism more flexible. The hosts can more easily create new and delete old flow identifiers without having to allocate the identifiers from a third party; i.e., a home agent in the MIPv6 case. The author's colleagues Kauppinen et al. have presented an approach in [178] to support multiple care-of-addresses at the peer host (see also [73]). The approach makes it possible to use a single home address with multiple traffic flows.

In the HIP case, the SPI namespace can be used for flow identification purposes (like in [169]) in the same way as the home addresses are used in Mobile-IPv6-based flow mobility solution [P1]. The SPI value defines a path between end hosts, and it is carried in each ESP-protected payload packet. Thus, the SPI value works as a perfect flow identifier. However, as described earlier, each SA pair is bound to at least one VIID pair. The VIIDs work as stubs for flow end points. They are tagged with the QoS information of the flows. Furthermore, the SAs bound to these stubs can be seen as realizations of the flow end points.

6.1.10 Expressing Policy Rules with Authorization Certificates

The source and destination policy actions can be expressed with authorization certificates (e.g. [21][114][54]). An *authorization certificate* is a signed document defining a set of rights that one identity approves to another identity. This is illustrated in Figure 6.1, on page 115. Publication [P1] describes how the interface selection rules can be presented with the KeyNote Trust System [114] certificates. Still, the rules can be expressed with any authorization certificate system, such as Simple Public Key Infrastructure (SPKI) [54] certificates. Publication [P9] presents a way to apply the authorization certificates in a moving network architecture. The same delegation principles can also be applied with the policy based flow control mechanism.

It is good to logically distinguish the policy rule certificates from the IPsec policy expressions. The IPsec policies and SAs are just one way to implement the policy-based flow mobility-management. Publication [P1] presents the flow mobility-management in the MIPv6 context. The presented principles have also been applied to HIP by the author's colleague Pierrel [169].

The issuer and the subject in the policy rule certificate can be represented with

public-key-based HIs. The issuer authorizes the subject to act according to the policy rules. The issuer of a certificate acts as a policy decision point, while the subject of the certificate is considered to be the policy enforcement point. There are two basic design choices when defining the policy decision point for the destination policies. Either the packet sender or the receiver can define the destination policy. From the security point of view, the host that is bound to a specific location should be able to define the policy rule concerning that locator.

In practice, the receiver may want to restrict incoming traffic and use different kinds of firewall rules for different interfaces. In most cases, the sender cannot affect the receiver's firewall rules, and thus the receiver would not be able to adhere to the destination policy defined by the sender. The author has ended up with a solution where the host that is reachable in a location also defines the corresponding source and destination policy actions related to that locator.

The main purpose of carrying the policy rule information in authorization certificates is to support distributed flow control management. As a result, the policy decision, and the policy enforcement points can be separated from each other. The policy decision point can be located at an end user's, an administrator's, an operator's or a peer host's domain [P1]. Furthermore, the policy enforcement point can be moved from an end host to multi-homed intermediate nodes which is part of future work (see also [37][150]).

6.2 Address-Family Agility

This section describes an address-family agility mechanism that is based on the solutions presented in publications [P8][P7]. Address family agility provides a mechanism for communicating between different versions of socket APIs over different IP versions. The overloaded *'address'* term has different semantics at different TCP/IP layers. In the following context, the author will make a difference between application-layer identifier agility and locator-family agility as earlier illustrated in Figure 4.5, on page 72. The AID concept was presented earlier in section 4.3.1.

The rough idea of the address-family agility solution and its relation with the previous work is presented briefly in Sections 6.2.1-6.2.4. The author discusses the results in more detail from the bindings point of view in sections 6.2.5-6.2.10. The discussion elaborates and generalizes the solution to show how it works together with the

rest of the solutions presented in this thesis.

6.2.1 EW'04 Paper: Research Problem

In the current Internet, an IP address is used both for identifying a host and its topological location. In other words, IP addresses of the same family are used at the application, transport and network layers to identify connections and route packets. As a result, IPv4 applications cannot establish connections with IPv6 applications.

6.2.2 EW'04 Paper: Basic Idea

In publication [P8] , the author decouples the application layer from the transport layer, and the transport layer from the network layer using two kinds of namespaces. The author uses 32-bit long, local scope identifiers with the IPv4 socket API and 128-bit long identifiers with the IPv6 socket API at the application layer. The application-layer identifiers are bound to global, 128-bit long transport-layer identifiers. Further, the transport layer identifiers are dynamically bound to IPv4 or IPv6 locators at the network layer. As a result, the HIP based solution provides end-host mobility and multi-homing between IPv4 and IPv6 based applications. For the author's best understanding, publication [P8] presents the first solution for inter-communication between legacy IPv4 and IPv6 socket APIs for end-to-end connections over IPv4 or IPv6 networks.

The main result of the solution [P8] is the specific set of identifier bindings that result in the presented address-family agility mechanism supporting legacy IPv6 and IPv4 applications. The identified bindings are discussed later in sections 6.2.5-6.2.10. However, an essential limitation of the solution [P8] is that the hosts must be reachable via the same IP version. In other words, the hosts cannot communicate with each other if other is attached to an IPv4 network and the other to an IPv6 network. Another limitation is that application cannot use the application layer identifiers as referrals between each other.

6.2.3 SecureComm'05 Paper: Basic Idea

In publication [P7], the author presents a new variant of IPsec ESP transport mode, denoted as Stripped End-to-End Tunnel (SEET) mode. SEET mode is here called as BEETIN mode at the intermediate nodes. It is based on the initial Bound End-to-End Tunnel (BEET) mode proposed by Nikander et al. [134]. The BEET and SEET modes are a combination of IPsec tunnel and transport modes, using the transport mode packet format but providing limited tunnel mode semantics. In particular, the modes takes care of the translation between locators and the host identifiers. In addition, the SEET mode supports SPI and locator translation at the SPINAT nodes. Because of the connections are identified with SPIs at end hosts, it is easy to replace IPv4 locators with IPv6 locators and vice versa at intermediate nodes.

The approach has also some limitations. The presented SEET mode is not backward compatible with the legacy ESP transport mode used at end hosts [165]. In addition, publication [P7] does not discuss SPINAT functionality from the address-family agility point of view, albeit the locator translation functionality is implicitly included in the solution. However, from the address-family agility point of view, the main result of the publication [P7] is the novel IP-version independent, SPI-based locator translation. The *scalable* solution translates SPI and locator values carried in the ESP protected packet header.

6.2.4 EW'04 and SecureComm'05 Papers: Relation to Previous Work

The legacy socket creation system call requires that an application defines the used address-family. When the peer host is reachable both at IPv4 and IPv6 locators, the application may open a socket per address-family. As a result, the application may establish a transport layer connection per socket. In the basic case, each socket call defines implicitly the used IP mobility protocol family (like [45][26]). The address-family dependence at the application layer may result both in a mobility control channel and a data channel per address-family. That kind of approach does not offer address-family agility for the applications.

One approach is to implement the address-family agility mechanism at the application layer. An application implements a socket selection related functionality. The trade-off is that the approach is not transparent to the application and thus is

difficult to deploy with legacy applications.

The address agility functionality can also be implemented between application and transport layers in the user-space. One approach (like [197]) is to compile an application with an enhanced socket library that offers a virtual socket layer for the legacy socket API. The virtual socket layer provides a legacy socket API to the application. However, the wrapper function at the virtual socket layer maps the socket API calls to legacy socket system calls. In other words, the virtual socket layer multiplexes legacy sockets with virtual sockets. One problem is that the virtual socket layer must open a socket per address-family. In addition, each network-layer hand-off closes the legacy sockets and the legacy socket must be recreated at the logical socket layer. It is possible to depend on the network-layer mobility solutions to keep the connections alive. In that case, the host may implement a network layer based mobility management protocol. Another possibility is to implement a control-plane protocol for mobility purposes at the virtual socket layer, which breaks the transport layer connections.

Network layer tunneling techniques can also be used to implement address-family agility functionality. In the tunneling approach, IPv4 traffic is tunneled over IPv6 or vice versa (like [71]). Each host is simultaneously reachable via IPv4 and IPv6 locators that are bound to intermediate rendezvous nodes. The approach makes it possible to bind an application to an address family different from that used at the network layer. The trade-off is that in several cases the triangular routing does not provide an optimal packet forwarding path for the payload traffic. In addition, the peer host must be reachable all the time from the same locator family that was used in the initial socket binding (see section 2.6.1).

The network-layer tunneling techniques can be used to decouple the used socket API version from the IP version used at the network layer. However, the tunneling techniques do not alone solve inter-communicating between different kinds of socket APIs. On the other hand, the virtual socket layer based approaches require application level gateways, like in P2P overlay networks, to support communication between different addressing domains. A fundamental difference between the virtual socket layer based approaches and the approach in this thesis is the transport-layer connection management during hand-offs. When the solutions in publications [P8][P7] are combined together they are able to overcome the presented restrictions of the other address-family agility solutions.

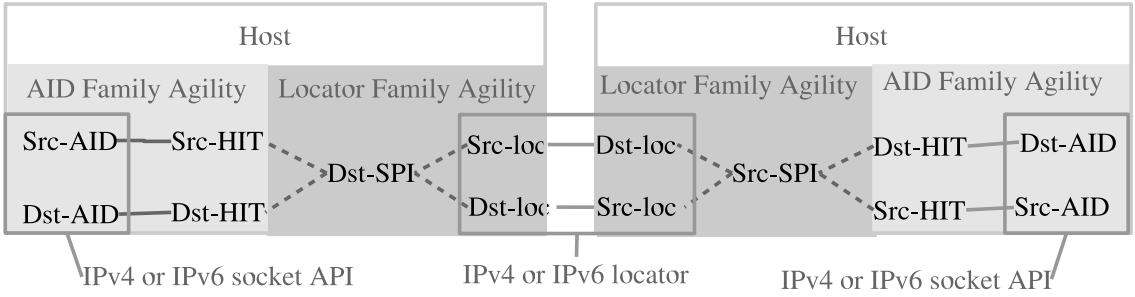

Figure 6.2: Hosts may use different kinds of socket APIs but they are connected to the same locator family domain in this example.

6.2.5 Address Family Agility Solution

In the AID agility case, the identifiers used at the socket APIs (see [190]) can carry the semantics of different IP address families. Two communicating applications running at different hosts can support different kinds of socket APIs. To achieve this kind of flexibility, the solution decouples the socket APIs from the transport layer at the kernel.

In the locator family agility case, the payload packets may contain locators belonging to a different family than the corresponding AIDs. Moreover, the end hosts may be attached to access networks supporting different locator families. The host-identifier-based overlay routing at intermediate nodes (see [P7]) is used to communicate over different locator family domains as discussed in section 5.3.5. The different forms of address-family agility can also take place simultaneously. Depending on the host's reachability situation, the SPI$\underline{\overset{n\rightarrow 1}{\longrightarrow}}$locator bindings (see section 7.2.4) are created at the host and optionally also at the intermediate nodes.

In the basic case, the end hosts are directly attached to access networks supporting the same locator family. However, the applications running at the hosts may still support different socket APIs as illustrated in Figure 6.2, on page 124. The applications implicitly authorize their local environments (see [P2]) to implement the required AID$\underline{\overset{n\leftrightarrow 1}{\longrightarrow}}$HI bindings (see section 7.2.1).

In a more advanced case, the end hosts are attached to access networks supporting different locator families as illustrated in Figure 7.3, on page 150. To support communication between the end hosts, the peer host must be reachable from at

least one locator leased from a rendezvous node that supports the same locator family as the mobile host. The end hosts authorize the intermediate nodes to create the required SPI$\xrightarrow{n\rightarrow 1}$locator bindings (see section 7.2.4).

The dynamic double binding between AIDs and HIs, HIs and locators makes the solution flexible. It allows, e.g., that an IPv6 enabled application, located in an IPv4 network, communicates with an IPv4 application, located in an IPv6 network. From the application's point of view, the identifier remains the same during the lifetime of a created socket.

6.2.6 Routable Application Layer Identifier Considerations

In this section, the routable and non-routable AIDs are discussed from the application's viewpoint. In the routable AID approach, the resolver library returns a routable identifier set to the application as a response to the DNS query. This case results in opportunistic authentication [P3][121] because the sender does not know the receivers HI.

The benefit of the routable AID approach is that the lifetime of the socket and the lifetime of corresponding mobility states do not need to be in synchrony. The mobility protocol may lose its mobility and security state. The first message sent out from the open socket after state loss results in a new mobility state establishment exchange. It is good to notice that in the current HIP approach [121], the peer state must be kept alive until the last application closes its socket bound to the corresponding HIT. There are also drawbacks in giving routable AIDs to the application.

The applications should not depend on the IPv6 based routable identifiers because the current networks are based mostly on IPv4 technology. Using purely IPv6 based routable identifiers does not provide any benefit compared to a flat namespace in the IPv4 networks. Moreover, to support legacy applications and cross IP version communication, the solution has to be backward compatible with the IPv4 socket API. However, IPv6 based identifiers cannot be used with the IPv4 socket API. Due to the lack of global IPv4 locators, they cannot be used for fine-grained identification. In addition, in pure IPv6 networks, the IPv4 based routable identifiers do not provide any benefit. Therefore, the application layer identifiers can neither depend on the existing IPv4 nor IPv6 address spaces to support address-family agility.

6.2.7 Application Layer Identifier Independence

As presented in the previous section, routable application layer identifiers do not provide address-family agility that is flexible enough. Based on that observation, it is possible to remove the routability requirement for the application layer identifiers and use flat namespaces instead of depending on structured identifiers.

HIP provides flat application-layer identifiers for IPv4 and IPv6 socket APIs [121], namely, LSI for the IPv4 socket API, and HIT for the IPv6 socket API. A trade-off is that the solution cannot directly support legacy applications that use identifiers for referrals. It seems that DHTs offer a solution for the HIT$\overset{m \leftrightarrow n}{\longleftrightarrow}$locator lookup problem. However, several administrative and security related problems must be solved before a global DHT infrastructure can be deployed in parallel with the existing DNS infrastructure.

Using variable size identifiers at the application layer has an impact on the transport layer pseudo-header computation [93]. The used transport-layer identifier must be the same at both communicating end points. To support different socket APIs and common pseudo header computation, the solution must implement a binding between AIDs and transport layer identifiers (section 7.2.1).

According to the LSI naming convention, the LSI is unique only inside a host. The host is responsible for taking care of the LSI collisions, and LSIs have only local meaning inside a host. From the address-family agility viewpoint, the HITs are also considered as local scope identifiers at the application layer. In this way, the binding between the socket API and transport layer (section 7.2.1) provides a flexible way to implement address agility between IPv4 and IPv6 applications as illustrated in Figure 4.5, on page 72 [P8].

The problem with this approach is that the applications believe that their peers belong to the same address-family. For example, if an IPv6 legacy application adds its HIT to the payload, an IPv4 legacy peer application does not know how to handle the received identifier. Applications using application layer identifiers as referrals (see e.g. [77]) do not work with this kind of an approach. Still, most of the existing applications work fine with the approach taken in this thesis.

6.2.8 Pseudo Header Computation Considerations

To support current transport layer protocols, the pseudo-header computation must be computed over 32-bit or 128-bit identifiers. In the HIP approach, the pseudo-header is computed using the HIT both in IPv4 and IPv6 socket API cases. In the IPv6 case, the AIDs and HITs are logically bound together, even they are the same at the application and transport layers. The only requirement is that both end points know the same transport layer identifier that is used for the computation. The reason to use HITs, instead of LSIs, is that the HITs are global identifiers that are known by both end points.

Basically, it could be possible to use 32-bit long unique global identifiers for the same purpose but the namespace could not then be flat for collision reasons. As a result, the transport layer identifiers would be similar to MIP (or MIPv6) home addresses. Thus, the home address would work as the transport-layer identifier. In that case, all the nodes would need both a home address and an HI.

The benefit would be that the peer host would be reachable in the case when the mobility-management state is lost before the socket is closed. In other words, the peer's locator information would be stored in multiple states having different life-times in the stack. It is good to notice that in this kind of an approach the application would still be either bound to an LSI or an HIT. However, the benefit of the clean identifier-locator split would be partly lost. This is one reason to use the flat HIT namespace for pseudo header computation. The approach does not require allocating static locators for each host.

In this approach, the flat application-layer identifiers are dynamically bound to HITs that are used for the transport-layer pseudo-header computation. The flat AID namespace makes it possible to implement address-family agility between other kinds of socket APIs than IPv4 and IPv6 sockets. This is illustrated in Figure 4.5 (page 72). As earlier mentioned, the solution has certain restrictions related to referral problems but does not directly require that hosts allocate static locators from the Internet.

6.2.9 Connecting Locator Family Domains Together

The inter-communication between different socket API versions requires also network-layer support when the hosts are connected to different locator spaces. In other words, the network does not offer an end-to-end connectivity over the same IP version.

In this approach, the author assumes that private networks are connected to each other via Internet as illustrated in Figure 4.2, on page 67. SPINAT nodes connect private networks to the Internet. Moreover, private networks supporting different IP versions can be connected to each other via SPINAT nodes.[16] The SPINAT node configures IPv6 and/or IPv4 locators for its public side NICs using IPv6 auto-configuration and DHCP leases [180][86][143]. It sends router advertisements and runs a DHCP server for the hosts attached to same link with its private side NICs. Thus, the SPINAT may offer IPv4 and/or IPv6 locator spaces for its clients. Several SPINAT nodes can be nested together forming a routing tree hierarchy. In this way, the architecture is more fault tolerant because it supports alternative upstream routing paths. When the SPINAT node has multiple public side NICs it uses LPM for source locator selection (*"LPM"* was explained earlier in section 5.5.7, on page 108).

Multi-homed end hosts may simultaneously be attached to several private networks and/or directly to the Internet. In the approach taken in this thesis, the Internet provides global IPv4 and IPv6 connectivity between the private networks. It is obvious that hosts belonging to private networks supporting different IP versions cannot directly communicate with each other without support of an intermediate node. Thus, the locator family agility is based on two kinds of intermediate nodes. Namely, rendezvous nodes located in the Internet and SPINAT nodes located between the locator domains. To be reachable via different address families, the hosts make locator leases from rendezvous nodes located in the Internet per IP version.

6.2.10 Locator Family Translation between Private Domains

To generalize the locator translation between locator family domains, this section briefly describes how the approach works with nested locator domains. Basically, each SPINAT node provides a private locator space for other nested SPINAT nodes

[16] BEETIN mode (section 5.3.5) can also be used to support cross communication with other network layer protocols than IPv4 and IPv6.

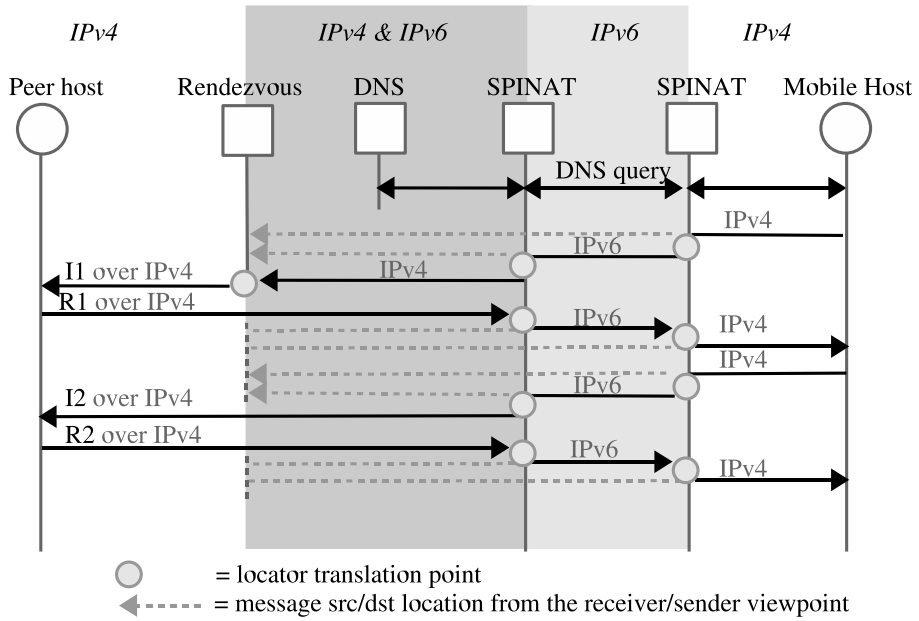

Figure 6.3: An example of initializing locator translation between nested private domains using HIP base exchange.

behind it. It is possible that some private domains support only a single locator family. When a mobile host at the leaf private domain wants to communicate with a peer host, it retrieves the peer's locator set from the directory service. The received locator set should contain at least one locator per locator family. One solution is that the initiator adds the received peer's locator set to the first control-plane message. Once a SPINAT receives the first message, it can use the received locator set for destination locator translation. For example, a mobile host may send a message from a private IPv4 domain via a private IPv6 domain to the Internet, as illustrated in Figure 6.3, on page 129. The SPINAT nodes use LPM algorithm for locator selection between the destination locator set in the control-plane message and its multiplexed locator set. The corresponding state establishment and update exchanges are presented in detail in Appendixes A.1-A.4.

6.3 Local Mobility

In this section, the author describes the local-mobility solution presented in publication [P4]. Local mobility hides regional movement from peer hosts and optimizes the

amount of required hand-off related signalling [31]. The basic idea of the solution and its relation to the previous work is briefly presented in sections 6.3.1-6.3.3. The main results of solution are the identified cryptographic association mechanisms used for authenticating locator update messages at the intermediate nodes. The presented mechanisms make it possible to optimize the amount of security related signalling during hand-offs. The author discusses the results in more detail from the cryptographic association mechanism point of viewpoint in sections 6.3.4-6.3.5. Section 6.3.6 elaborates the solution and presents a minor extension to the local mobility protocol.

6.3.1 ISC'04 paper: Research Problem

A local-mobility anchor point must be able to verify that a mobile host sending a locator update message is in the location claimed in the message. Unverified locator updates open the local mobility protocol for re-direction and related DoS attacks. The security association establishment between the mobile host and the local-mobility anchor point is typically decoupled from the locating update exchange. The problem is that the separate key exchange causes signalling overhead during local-mobility related hand-offs. Another problem with the existing approaches is that the mobile host and the local-mobility anchor point must have a pre-shared secret or the mobile node must have prior knowledge of public-key of the local-mobility anchor point.

6.3.2 ISC'04 Paper: Basic Idea

Publication [P4] defines a secure local-mobility solution that scales well between administrative domains. The solution integrates key-sharing between the mobile node and the local-mobility anchor point into the three-way location update exchange. The security association between the mobile node and the local-mobility anchor point is established using Lamport one-way hash chains and a technique known as secret splitting [110][14]. *Secret splitting* denotes a mechanism that divides a key into multiple pieces. The original key cannot be reassembled without possessing all the pieces. The one-way hash chain and secret splitting techniques provide weaker security than public key cryptography. However, the solution protects the anchor points from traffic re-direction and related DoS attacks. The mobility anchor points

can be located in a tree hierarchy in the administrative domain. The hierarchical model makes it easy for the mobile nodes to locate the mobility anchor points. The HIP based solution optimizes the amount of signalling during hand-offs because it does not require a pre-shared secret or public-key based mechanism between the mobile host and the local-mobility anchor point.

The local-mobility solution is integrated with the SPINAT functionality. The transparent service discovery and registration does not require explicit signalling between the mobile host and the SPINAT node. Therefore, local mobility-management does not cause additional signalling latency to the protocol (see [75]). Once the mobile host is attached to a local network it runs a macro-mobility exchange with each peer host through SPINAT nodes. The SPINAT nodes on the routing path establish states per each end-to-end IPsec SA. Later on, when the mobile host changes its location inside a SPINAT node's region, the SPINAT node hides the movement from the peer hosts. It is still good to notice that each rekeying procedure results in an end-to-end mobility state update exchange [P4].

The local-mobility protocol uses the same message structure both for local and macro mobility signalling. The messages in publication [P4] follow the early HIP mobility exchange terminology. Later on, the messages in the HIP three-way handshake have been renamed and are called 'UPDATE' messages. Currently, the parameters carried in the UPDATE messages define the semantics of the different messages in the handshake [121][135]. Therefore, the message structures presented in [P4] can be seen as extensions to the current HIP-mobility proposal [135].

The main result of the publication [P4] is the scalable return routability test that can be used with a local-mobility protocol. The author believes that solution may turn out be as important thing as the return routability protocol has been for macro-mobility protocols. The presented local-mobility solution is a novel one-and-half round-trip protocol that establishes simultaneously a security association between a mobile node and an anchor point, and updates the locator at the anchor point and at a peer node in a secure way.

However, the main limitation of the protocol is its vulnerability for certain MitM attacks where the attackers are located at both sides of the local-mobility anchor point. In addition, the combination of weak and strong authentication mechanisms results in a trade-off between identity protection and signalling optimizations. Albeit, temporary hash chains provide anonymous identifiers for the hosts, the binding

between subsequent hash chain values makes it difficult to protect from identifier tracking.

6.3.3 ISC'04 Paper: Relation to Previous Work

Several local-mobility schemes, including Cellular IP[31], HAWAII[153], TIMIP[65] and HMIP[34], are integrated with Mobile IP [26] and Mobile IPv6 [45] macro-mobility protocols. The main scalability and hand-off latency related issues with the mentioned protocols are related to preconfigured security associations or heavy authentication related signalling between mobile hosts and the local-mobility anchor points.

The HMIPv6 [72] approach was earlier presented in section 1.6.4. It was under development at IETF at the same time when publication [P4] was published. In HMIPv6, the establishment of an IPsec SA between a mobile host and a Mobility Anchor Point (MAP) requires at least four messages with IKE. The local binding update exchange between the mobile host and the MAP requires two messages. The MAP must also make Double Address Detection (DAD) for the RCoA in its link before replying back to mobile host. In addition, the mobile host must run RR test with its peer host before sending the binding update that requires another five messages. As a consequence, the mobile host must sent eleven (11) messages when it changes securely the MAP region. To reduce the amount of signalling the author of this thesis presents an approach in [P4] that requires three messages together to achieve a secure locator-binding simultaneously at anchor points and at the peer hosts when the mobile node moves between regions of local mobility anchor points.

Many of the problems with the HMIP protocol are related to inefficient security and configuration mechanisms. Whenever a mobile node changes a MAP region and uses IPsec, it must create a new security association with the new MAP. In addition, the operators are assumed to make a careful analysis of their network topology and configure the router advertisement policies according to the locations of MAPs. Moreover, the mobile node has to have a sophisticated algorithm for making an optimal selection between different MAPs.

6.3.4 Authentication between Mobile Hosts and Intermediate Nodes

The presented local mobility solution uses delayed authentication with the location update exchange. The messages are authenticated using hash chain value protected HMACs. The idea is similar to TESLA[3] but applied in a new context. The end hosts reveal the hash chain values in subsequent messages to the SPINAT node. Together, the different cryptographic methods provide a mechanism for a SPINAT node to authenticate the end-to-end update exchange messages and to create shared keying material with the mobile host. Basically, the mobile host indirectly uses the existing end-to-end SA to share keying and hash chain material with a SPINAT node. The weak authentication mechanism is useful in situations where the mobile host does not have prior knowledge of a SPINAT node's identity. Basically, the mobile host could use opportunistic HIP authentication for an explicit registration exchange. However, the approach in [P4] provides better security against MitM attackers than an opportunistic HIP base exchange with the SPINAT node.

Figure 6.4, on page 134, presents a scenario that consists of an end-to-end macro-mobility exchange and a local-mobility exchange. In the figure, the mobile host moves inside the SPINAT node hierarchy. It changes its attachment from the previous SPINAT node's region to a new region. The mobile host initiates an update exchange towards the peer host. Instead of forwarding the messages to the peer host, the SPINAT node uses overlay routing [P7] to forward the update messages to the previous SPINAT node. The previous SPINAT node takes the peer role in the update signalling and replies on behalf of the peer host to the mobile host. The cryptographic association between the mobile host and the SPINAT node is based on the split keying material shared earlier. Each subsequent update exchange updates the shared keying material. From the mobile host point of view, the local and macro mobility update exchanges are similar. However, in the local mobility case, the mobile host does not run the update exchange with the peer host but with the previous SPINAT node.

Whenever the mobile host bootstraps a new hash chain or rekeys, the SPINAT nodes must forward the update messages to the peer host. The same applies when a SPINAT node is out of synchrony with the hash chain material with the mobile host. In this way, the SPINAT nodes in the hierarchy can update their states according to the latest hash chain anchor value and keep in synchrony with the mobile host. The SPINAT node updates its private side SAs during the local update exchange

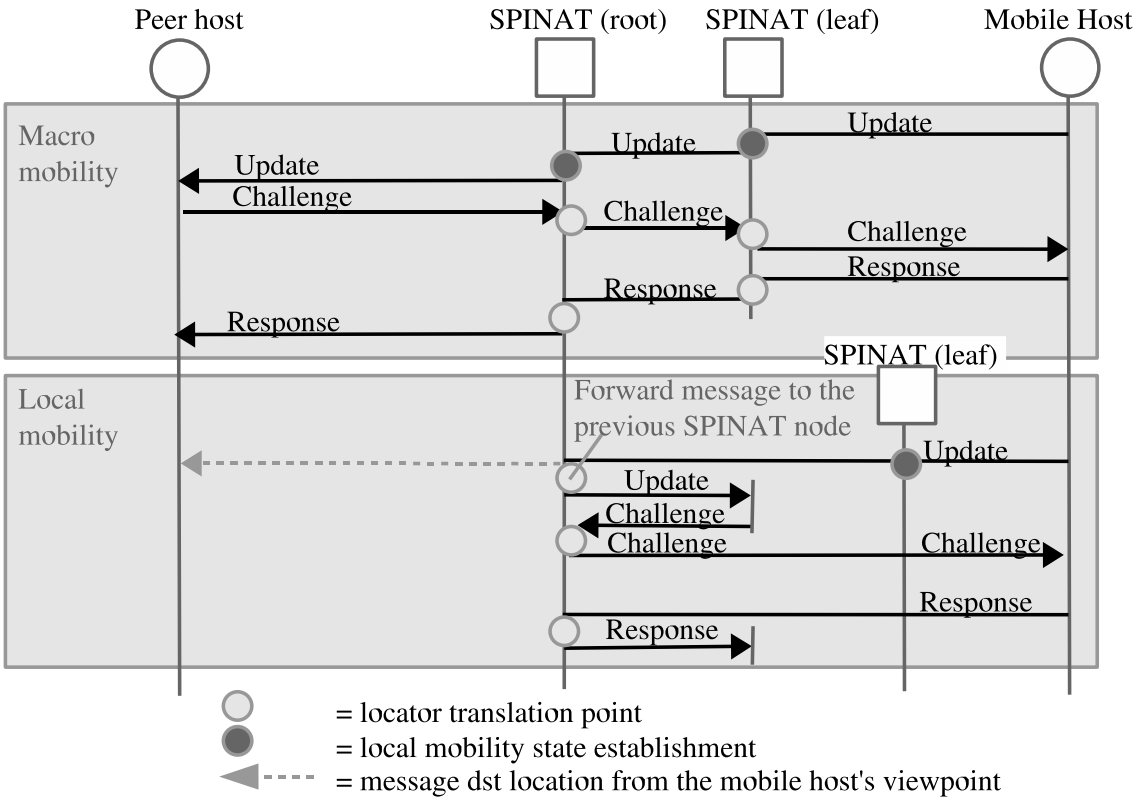

Figure 6.4: An example scenario of local mobility between locator domains.

(see Figure 5.1, on page 98). The SPI values remain the same but the locators in BEETIN SAs are updated.

It is difficult to protect from identity tracking when subsequent update exchanges are bound together using the hash chain values. The SPINAT nodes use the hash chain values to identify the mobile host even when the HITs are changed dynamically. To achieve a balance between local-mobility support and identity tracking, the mobile host may bootstrap new hash chains frequently in order to support local mobility in a restricted area. On the other hand, the mobile host benefits from long hash chains. The shorter hash chains are used for authentication the more frequent macro mobility exchanges are required for synchronizing the on-the-path SPINAT nodes. Finally, the benefit of using dynamically changing HITs [P6] in UPDATE messages makes it possible to protect against temporary identity thefts at SPINAT nodes.

6.3.5 Re-direction Attacks

In the presented local mobility approach, the SPINAT node does not have prior knowledge of the mobile host. Basically, the SPINAT node does not care who the mobile host is as long as the host remains the same during lifetime of the state. The cryptographic association between the mobile host and the SPINAT node is derived from the cryptographic association between the mobile host and the peer host. The SPINAT node assumes that if the peer host is willing to communicate with the mobile host, it has also established an SA with the mobile host. The mobile host uses the end-to-end SA to deliver an encrypted hash chain and splitted secret keying material to the peer host. Only the authentic peer host is able to decrypt the data and send it back in plain text to the mobile host. The SPINAT node is able to bind the update exchange messages together using the Lamport one-way hash chain properties. However, an on-the-path attacker may try to re-direct traffic by intercepting update messages and replacing the locator information with the victim's location. Such an attacker must be located on both sides of the SPINAT node to be able to change parameters carried in the message as discussed in [P4].

6.3.6 Signalling Optimization during Local Hand-offs

The following optimization presented in this section is not included in publication [P4]. In the basic case, the mobile host initiates an update exchange for each of its peer hosts. To minimize the amount of local update signalling during regional hand-off, the mobile host can learn the closest SPINAT node's identifier during the network attachment exchange. When the mobile host finds out that it has moved inside the same SPINAT node's region, it may run a single update exchange with that SPINAT node. This is based on the assumption that the SPINAT node has earlier established states for each end-to-end SA pair. Thus, it is enough that the SPINAT updates the locators of all the private side SAs bound to the mobile host. The mobile host may replace the SPI information, carried normally in the message, with a piece of information that notifies the SPINAT to update all the related SAs. In the case that the mobile host does not get a reply from the SPINAT node, it initiates normal update exchanges with each of its peer nodes.

It is good to notice that the update exchanges are used for creating states and related SAs for end hosts at SPINAT nodes. When the mobile host moves into a

new SPINAT node's region, the new SPINAT node must create a state for each peer host. Because the keying material between the mobile host and the SPINAT node is shared using the end-to-end security association, optimizing the signalling during regional hand-offs is difficult. The authentication mechanism provides a way for the SPINAT node to verify that the peer host really exists. Thus, the mobile host cannot create inconsistent states at SPINAT nodes alone.

6.4 Network Mobility

In this section, the author describes the network-mobility solution presented in publication [P9]. The basic idea of the solution is presented briefly in sections 6.4.1-6.4.2 while the comparison with related work is discussed in more detail in publication [P9]. The main results in publication [P9] are the identified scalability related problems and the resulted signalling proxy based improvements. The author discusses the results in more detail from the authorization viewpoint in sections 6.4.3-6.4.7. The discussion also elaborates the solution to show how it works together with the rest of the solutions presented in this thesis.

6.4.1 WWIC'08 Paper: Research Problem

One problem with the basic Mobile IPv6 NEMO [50] solution is related to the end-to-end signalling latency. In other words, all packets are triangular routed via the mobile router's home agent. To overcome the problem, a couple of optimizations have been presented for the problem [187][105][128][181][99][141]. However, all the solutions depend either on the home agent of the mobile route or home agent of the mobile host. Basically, they use triangular routing for locator update signalling. The main problem is to find a solution where the location updates are sent to the peer hosts via the topologically shortest path without opening the existing solution for redirection attacks. In addition, the solutions do not present how to integrate integrity and confidentiality protection of payload traffic into the system in an efficient way.

6.4.2 WWIC'08 Paper: Basic Idea

In publication [P9], the authors present and evaluate a network mobility scheme based on HIP. The work is based on the author's earlier work in [196]. It is good to

notice that a related approach by Nováczki et al. [127] that is published after the submission of the author's work in [P9] represents a piece of independent work.

The cryptographic host identifiers are combined with an authorization mechanism and used for delegating the mobility-management signalling rights between nodes. While the delegation of the signalling rights scheme itself is a known concept, the way of establishing security associations in [P9] differs from the Mobile IPv6 NEMO [50] based solutions. In the approach presented in [P9], the mobile routers are authorized to send location update messages directly to peer hosts on behalf of the mobile hosts without opening the solution for re-direction attacks. The authorization model makes it possible to support route optimization and minimize over-the-air signalling and renumbering events during hand-offs of mobile routers. Basically, the moving network solution presented in publication [P9] provides a mechanism for delegating signalling rights between public-key-based HIs. The delegation primitive and locator translation together hide the network mobility from hosts attached to the moving network. This minimizes the amount of over-the-air signalling inside the moving network and between the moving network and the Internet.

The delegation may take place between mobile routers but also between mobile routers and signalling proxies located in the fixed network. The rough idea is to aggregate the signalling from the moving network to the fixed network that offers higher bandwidth. In addition, the delegation of signalling rights makes it possible to optimize the packet forwarding paths between the end hosts. The payload packets can be sent directly to the peer hosts instead of tunneling them via rendezvous points. From the payload size point of view, the presented solution is more efficient than the tunneling based solutions (like [50]) because additional tunneling headers are not required.

The required signalling can roughly be divided into network attachment, service discovery, registration, authorization, and location update exchanges (see section 5.1.2). Once a node joins a moving network, it runs an attachment exchange with an access point that is typically managed by the mobile router. The approach integrates the service discovery protocol to the existing end-to-end mobility state establishment and update exchanges (see [130]) to minimize the amount of control signalling.

The novel part of publication [P9] is that it presents and analyzes the characteristics of the new authorization scheme in the HIP network-mobility context using a real implementation. The results in [P9] indicate that it is important to distribute

the location update signalling between multiple signalling proxies to minimize the dependencies between the parallel update exchanges at a single mobile router and to increase parallelism during mobile-router hand-offs. However, the solution contains also some limitations. The same results indicate that the current implementation does not scale well in terms of hundreds of parallel communication sessions.

6.4.3 Signalling Proxy Functionality

The mobile router works as a signalling proxy for the mobile node. During the service registration exchange the mobile node requests the signalling proxy service, and authorizes the mobile router to signal on behalf of the mobile node. The authorization is expressed using public-key-based HIs in authorization certificates. The certificates can also contain flow policy rules as discussed earlier in section 6.1.10. The end-to-end mobility signalling also establishes BEETIN SAs for the end hosts at the mobile router. The main difference between static SPINAT nodes and the mobile routers is that the mobile routers dynamically update the locators of the public side BEETIN SAs during hand-offs (section 5.3.5). The private side BEETIN SAs are updated when the mobile router works as a mobility anchor point and supports local mobility for the mobile nodes as described in section 6.3.

6.4.4 Nested Moving Networks

When a mobile router is attached to another mobile router, they form a *nested moving network*. In this kind of situation, the registered mobile nodes in the moving network authorize the closest, i.e. leaf, mobile router to register themselves to other upstream mobile routers. In other words, each mobile router registers its clients to the next mobile router in the hierarchy to receive traffic from the Internet (see [196]).

The upstream mobile routers are also authorized to signal on behalf of the mobile hosts. This results in signalling aggregation for the mobile routers attached to the Internet, and further for the signalling proxies located in the fixed network. The nested moving networks may also result in a routing loop situation if the mobile routers cannot distinguish downstream and upstream access points from each other. To overcome the problem it is possible to include routing tree related information to the link-layer messages (like [137]).

6.4.5 Delegating Signalling to the Fixed Network

The mobile router may delegate the signalling rights further to a signalling proxy located in the global domain. A *signalling proxy* is a fixed node that is authorized to send location updates to the peer hosts, and optionally to reply to the peer hosts. This requires that the mobile host has initially delegated the rights to the mobile router. Later on, when the mobile router makes a hand-off, it runs the update exchange with the static signalling proxy, not with the peer hosts.

Basically, the signalling proxy may either forward the update exchange to the peer hosts, or run a reachability test itself with the mobile router. In the former case, the peer hosts authorizes the signalling proxy to verify the location of the mobile router. In other words, the signalling proxy runs the reachability test of behalf of the peer hosts. The peer hosts can start sending packets to the new location right after receiving the update message from the signalling proxy. The end hosts benefit from the approach when the static signalling proxy is close to the mobile router, and the latency between the mobile router and the peer host is long. The situation is illustrated in Figure 6.5, on page 140.

Typically, the peer host initiates the reachability test with the mobile router after receiving the update message from the signalling proxy. The situation is illustrated in Figure 6.6, on page 141. It is good to notice that the reachability test is made for the multiplexed locator of the mobile router. The mobile router replies on behalf of the mobile host to the peer host. The payload traffic continues flowing directly between the peer host and the mobile host via the mobile router after the mobile router updates the BEETIN SAs bound to its multiplexed locator.

6.4.6 Integrating Local Mobility into Network Mobility

To minimize the signalling that takes place over the wireless interface, the signalling proxy can be located on the packet forwarding path between the mobile router and the peer hosts. The situation is illustrated in Figure 6.7, on page 142. The signalling proxy works as a local anchor point for the moving network as presented in section 6.3. When the mobile router makes a hand-off, it runs a single update exchange with the proxy. The signalling proxy verifies the mobile router's location.

The signalling proxy runs update exchanges with the peer hosts if the mobile router

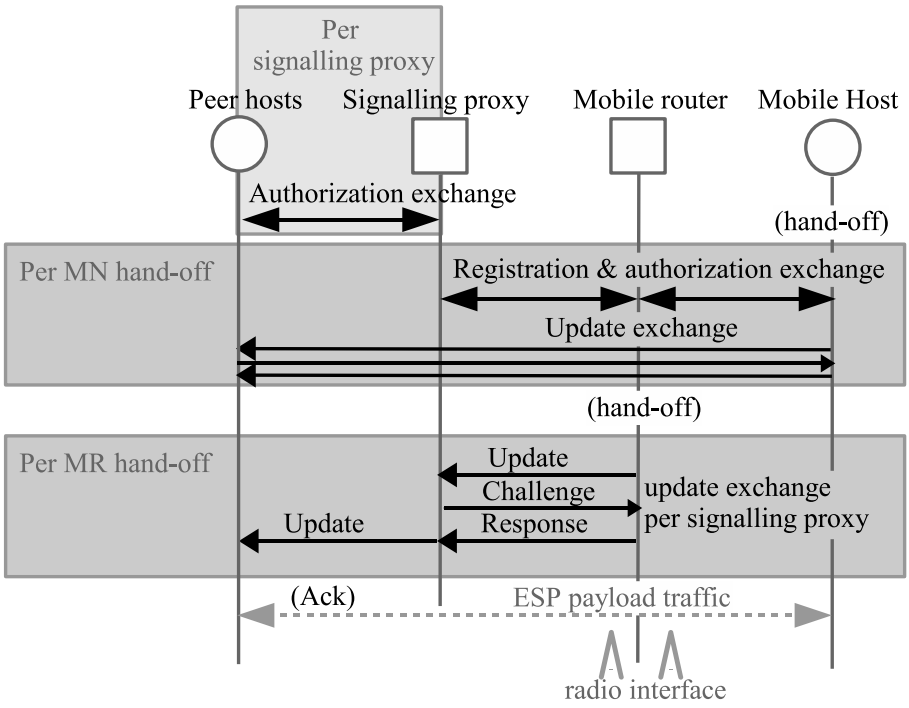

Figure 6.5: Peer hosts authorize the signalling proxy to implement a reachability test.

attaches for the first time to the proxy's domain. The proxy hides the burst of update exchanges from the nodes inside its region. When the local and network mobility-management are integrated together, it is possible to implement a mobile router hand-off with a single update exchange. Therefore, the population in the moving network does not affect the amount of signalling during hand-offs.

Logically, the SPINAT node works as a signalling proxy in the local and network mobility cases. In the former case, the SPINAT node takes the peer role, and runs the mobility update exchange on behalf of the peer node as discussed earlier in section 6.3. In the latter case, the SPINAT node runs the mobility update exchange on behalf of the mobile host. Basically, the mobile node authorizes the SPINAT node to act as a signalling proxy in both cases.

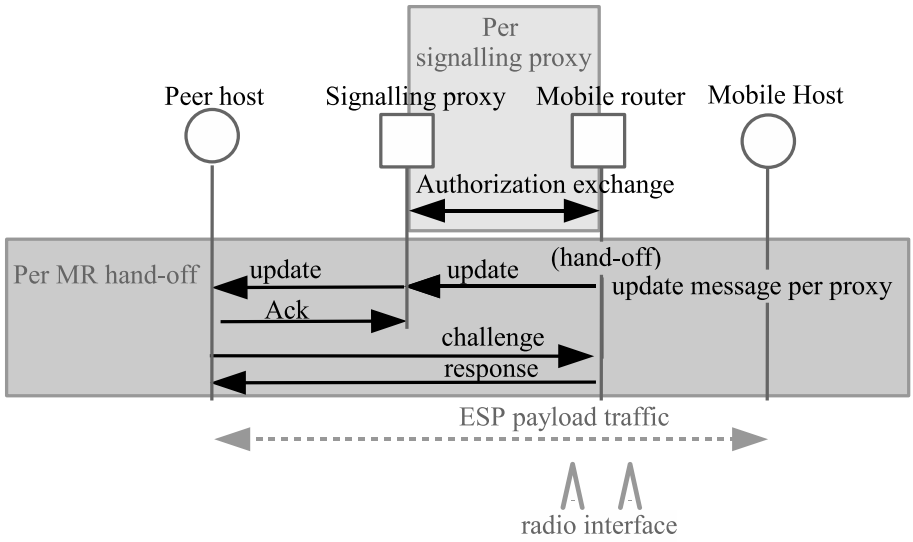

Figure 6.6: Peer hosts run the reachability test.

6.4.7 Leaving a Moving Network

The lifetime of authorization between mobile nodes and signalling proxies is expressed in the authorization certificates. In the basic case, the lifetime is the mobile node's estimated stay at the moving network. If the mobile node stays longer than the lifetime of the approved certificate, it generates a new authorization certificate for the mobile router. The same applies also to the authorization between nested mobile routers and signalling proxies. However, when the mobile host leaves the moving network before the lifetime of the certificate ends, it revokes the valid certificate. *Revocation* denotes a call back mechanism that is typically based on revocation certificates. A revocation certificate identifies another certificate that is inactivated.

The mobile host may include a hash of the revoked certificate to the update messages. The approach requires that the mobile host stores a hash of all active certificates. In other words, it stores certificates who's lifetime has not run out. The revocation of certificates prevents the earlier mobile routers from sending location updates on behalf of the mobile node when the mobile node has left the moving network. The peer host stores the revocation list of the certificates. Basically, the mobile host acts as a Certificate Authority (CA) for itself.

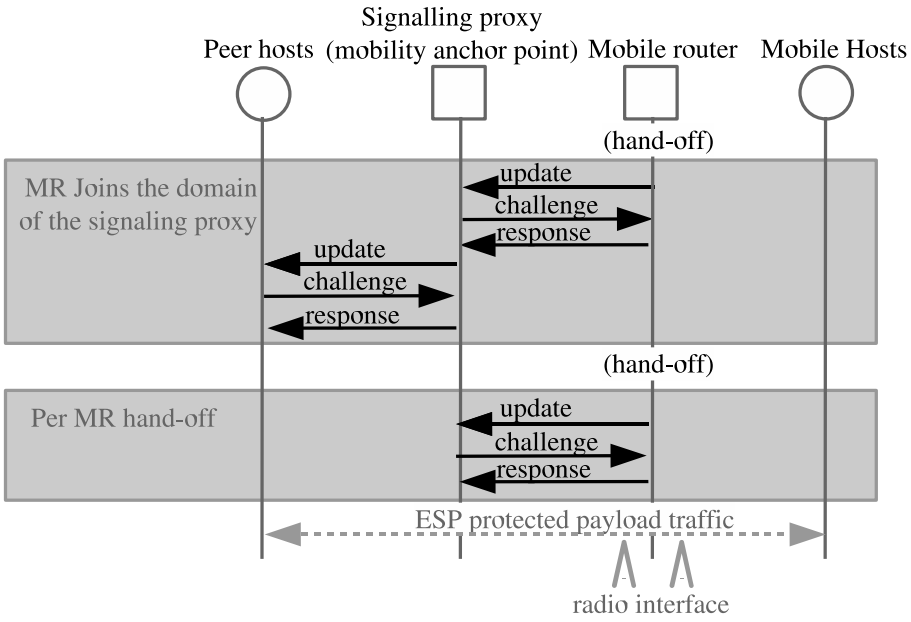

Figure 6.7: The on-the-path signalling proxy acts as a regional anchor point.

The mobile node also unregisters its states from the earlier mobile router. Depending on the situation, the unregistration may take place before or after the mobile node makes a hand-off. In the latter case, the unregistration exchange is run via the previous mobile router's rendezvous node. This requires that the mobile router sends its locator set to the mobile host during the initial registration exchange.

6.5 Summary of Four Granularity Levels of Mobility

This chapter presented four different granularity levels of mobility that are flow mobility, address-family agility, local mobility and network mobility.

Section 6.1 described a flow policy rule based mechanism for managing simultaneous traffic flows. It seems that the presented IP layer based approach was the first end host solution that provided simultaneous multi-access and per flow mobility functionality. The TLIs carried in each payload packet work as triggers for the policy rules. The policy rules define mappings between TLIs and VIIDs. The VIIDs are dynamically bound to locators sharing the same QoS properties. Moreover, the policies are divided into source and destination policies. The source policies are

used for finding a suitable source locator, while the destination policies are applied to destination locator selection. The flow policy rules can be expressed with an authorization certificate. The authorization certificates provide a way for moving the policy decision and enforcement points between nodes.

Section 6.2 presented a solution for AID agility and locator family agility. Together the presented approaches provide a way to support intercommunication between different kinds of socket APIs over locator domains supporting different network protocols. It seems that the presented approach is the first transparent solution for providing inter-communication between different kinds of legacy socket APIs for end-to-end connections over different IP version domains.

The address-family agility approach taken in this thesis is based on two bindings. The first binding takes place between application-layer identifiers and transport-layer identifiers. The second binding between transport-layer identifiers and locators is implemented using the BEET and BEETIN modes. The BEET mode provides dynamic binding between different locator families at the end hosts, while the BEETIN mode supports locator family translation at SPINAT nodes. The SPINAT nodes transparently establish and update the related states based on end-to-end control-plane signalling.

Section 6.3 presented a local mobility-management solution. It is possible to use the local mobility approach together with the other presented mobility approaches. The local mobility solution does not require explicit service discovery or registration exchanges. The on-the-path SPINAT nodes transparently establish states for the end hosts during the initial end-to-end communication.

The local mobility approach uses weak authentication techniques instead of public key cryptography to protect the location update signalling. The weak authentication mechanisms allow optimizing the amount of signalling. The trade-off is that the protocol became vulnerable for certain on-the-path attacks.

Section 6.4 presented a moving network solution that extends the earlier described SPINAT functionality. The mobile routers and static SPINAT nodes work as signalling proxies for the mobile nodes. The solution makes it possible to aggregate the mobility signalling to intermediate nodes located in the fixed network. In addition, the network mobility is hidden from the nodes inside the moving network.

The signalling delegation and the combined locator translation optimize the amount

of over-the-air signalling between the end hosts. The approach provides optimized routing between end hosts without additional tunneling headers. The solution is possible to integrate into local mobility. The nested moving networks can even belong to different locator domains.

7 Analysis

In this thesis, a limited set of authentication and authorization techniques has been applied to different reachability and security problems. This has lead to a flexible and compact mobility solution, even at the implementation level, which will be discussed later in section 7.5. The essential parts of the integrated architecture are illustrated from the bindings point of view in sections 7.1-7.2 and from the authentication and authorization point of view in section 7.3. The results indicate that the added indirection, backwards compatibility and the integrated mobility levels provide signalling efficiency and flexibility in heterogeneous datacom networks (see section 1.1).

Basically, a heterogeneous network environment consists of parallel namespaces at different TCP/IP layers. The author has focused on mobility and multi-homing related bindings in IPv4 and IPv6 networks. The bindings are analyzed from the control-plane and data-plane points of view. The bindings are implemented above the link layer, making the solution independent of the link-layer namespace. The analysis also contains topics for future work.

7.1 Bindings for Control Plane Signalling

The bindings that are created for end-to-end control-plane signalling that traverses rendezvous nodes are described in section 7.1.1 and those for SPINAT nodes in section 7.1.2. The three columns in the following Figures 7.1-7.3 define which of the nodes have generated the identifiers used in the bindings. A binding is illustrated with a line between the identifiers in the figures. The row headers in the figures name the nodes that create, store and depend on the bindings. Binding updates are based on authorization between nodes. Depending on the mobility solution, the authorization is based on different sets of cryptographic and non-cryptographic association mechanisms, as discussed later in section 7.3.

7.1.1 Via Rendezvous Node

Figure 7.1, on page 146, presents the bindings that are required for sending a control-plane message from a peer host to a mobile host via a rendezvous node. The bindings

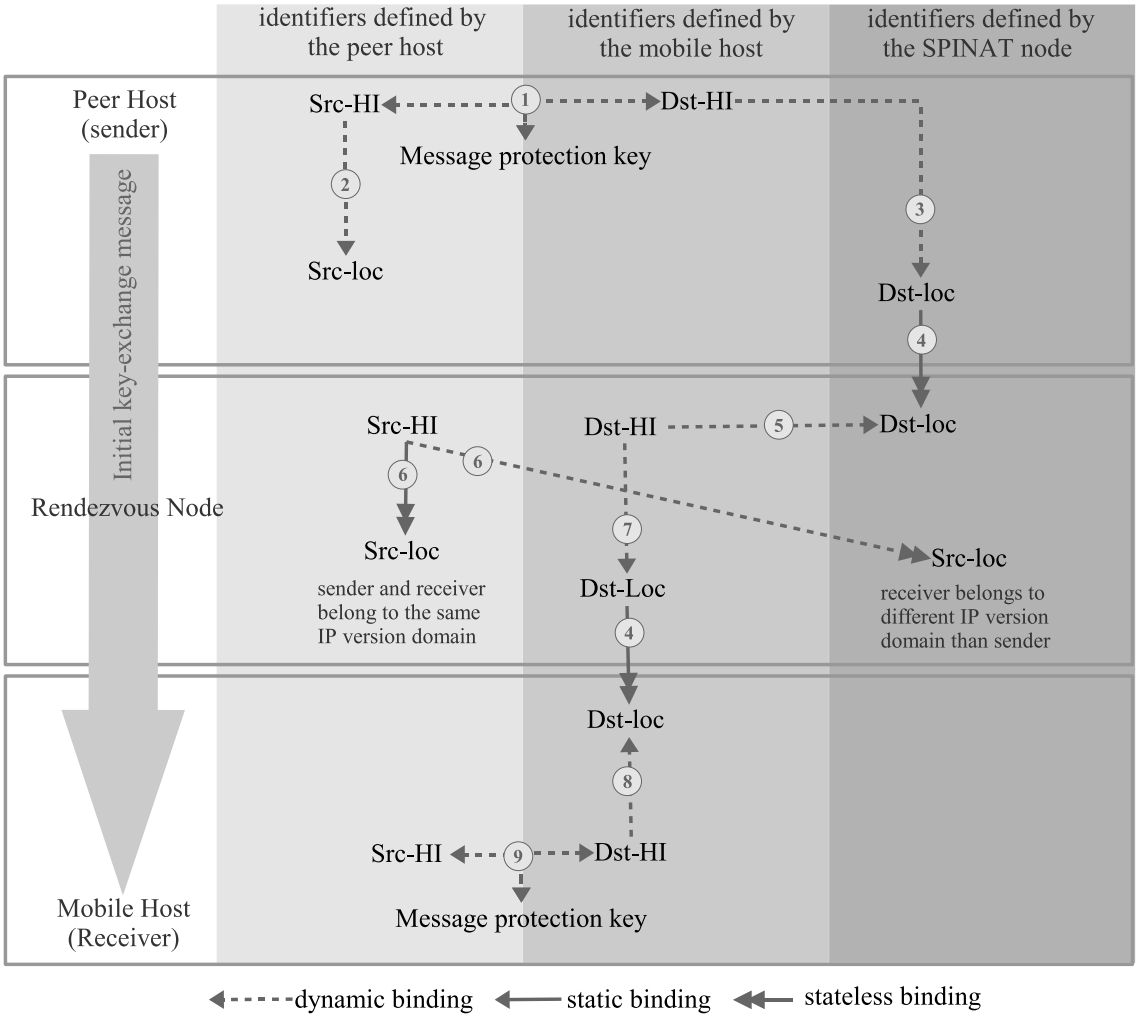

Figure 7.1: Bindings for end-to-end control-plane traffic via Rendezvous node.

between identifiers are numbered from (1) to (9). The stateless locator$\overset{1\rightarrow1}{}$locator binding (4) associates a locator carried in a packet header to a specific topological location on the Internet. The stateless binding depends on the Internet's routing infrastructure also in the following Figure 7.2 (binding 6, on page 148) and Figure 7.3 (binding 8, on page 150).

The peer host generates locally the source HI while the source locator is defined by the topological location of the host. The destination HI and the destination locator are received from DNS. The mobile host has stored its HI and the rendezvous node's destination locator at the DNS. Moreover, the rendezvous node has made a binding between the destination HI and the locator of the mobile host during the registration

exchange. The source HI may be dynamically bound to the rendezvous node's source location if the mobile host and the peer host belong to different IP version domains.

The source and destination HI pair are bound to a message protection key at the hosts. The HI pair$\underline{1 \leftrightarrow n}$message-protection key bindings (1)(9) are created at the end hosts during the authenticated Diffie-Hellman key exchange. The hosts may dynamically update the integrity and confidentiality protection keying material.[17]

The source HI$\underline{1 \rightarrow n}$locator binding (2) and destination HI$\underline{1 \rightarrow n}$locator binding (3) depend on each other. The current implementation uses LPM for source and destination locator selection. The receiving host may be reachable directly at the selected destination location, or the destination location may point to a rendezvous node like in Figure 7.1, on page 146. In the mobile-router case, the destination HI$\underline{1 \rightarrow n}$locator binding (5) is dynamic and updated during each hand-off, like the destination HI$\underline{1 \rightarrow n}$locator binding (8) at the mobile host. The destination HI$\underline{1 \rightarrow n}$locator binding (3) depends on the bindings (5)(8). In other words, the mobile hosts and the routers are authorized to update locators in the binding (3) at the peer host. In the rendezvous-node case, the destination HI$\underline{1 \rightarrow n}$locator binding (7) depends on mobile host's binding (8). In other words, the binding (7) is updated after each mobile host's hand-off.

The rendezvous node does not keep a state for the source HI$\underline{1 \rightarrow n}$locator binding (6). The binding (6) depends on the locator domains. When the receiving host and the sending host belong to different locator domains, the rendezvous node binds the source HI to one of the local source locators. Another reason to use local source locators at the rendezvous node is to mitigate ingress-filtering problems. *Ingress filtering* denotes a mechanism that verifies that an incoming packet at a router or an intermediate node has a source address that matches the direction from which it arrived.

7.1.2 Via SPINAT Node

Figure 7.2, on page 148, illustrates the bindings that are required for sending a control-plane message from a mobile host to a peer host via a SPINAT node.

[17]Sharing symmetric message-protection keying material with intermediate nodes is for future study. It would be an alternative way to implement signalling delegation compared to public key based delegation.

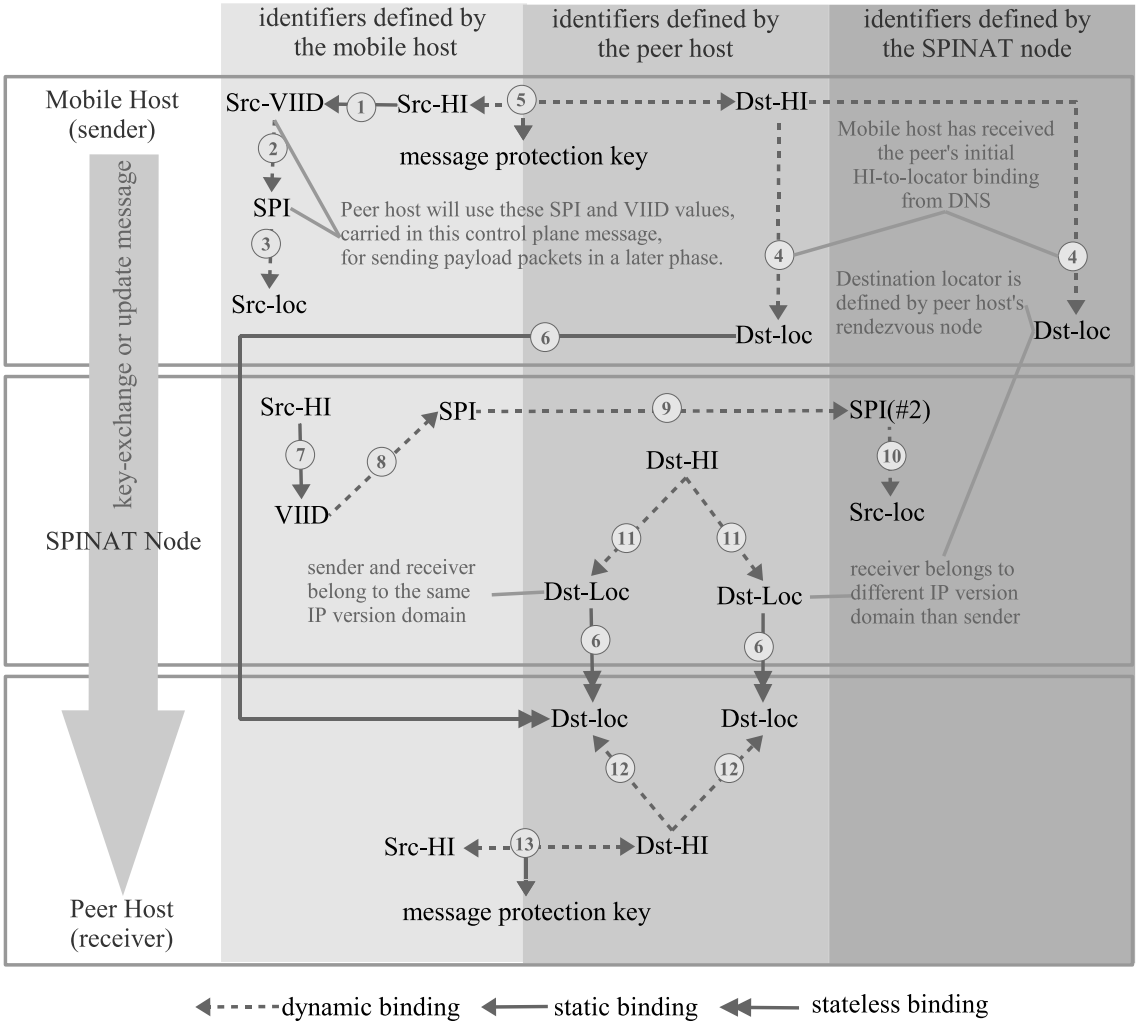

Figure 7.2: Bindings for end-to-end control-plane traffic via SPINAT node.

The bindings between identifiers are numbered from (1) to (13). The dynamic HI pair $\overset{1\leftrightarrow n}{\longleftrightarrow}$ message-protection key bindings (5)(13) at the end hosts are created during a key exchange.

The mobile host generates a VIID and a SPI value that the peer host can later use towards the mobile host with payload packets. This is illustrated in the source HI $\overset{1\leftrightarrow n}{\longleftrightarrow}$ VIID bindings (1)(7). Basically, the VIIDs are used for two purposes. Firstly, the VIID $\overset{n\rightarrow 1}{\longrightarrow}$ SPI bindings (2)(8) provide a way to keep subsequent SPI values in synchrony at hosts and intermediate nodes (section 5.5.1). The bindings (2)(8) are updated during each rekeying exchange. Secondly, the VIID can be bound to QoS information and it is used with flow policy rules as discussed earlier in section 6.1.

The source and destination locators must belong to the same scope and locator family. In other words the SPI$\underline{\overline{n \rightarrow 1}}$source-locator binding (3) and destination-HI$\underline{\overline{n \rightarrow 1}}$destination-locator binding (4) depend on each other. When the receiver is not directly reachable via the same locator family as the sender, the binding (4) points to a locator of the receiver's rendezvous node as discussed earlier in section 6.2.

The SPI$\underline{\overline{1 \rightarrow 1}}$SPI(#2) binding (9) at the SPINAT node is needed for SPI translation. The binding is required if an SPI value collides with another one at the SPINAT node. The SPINAT node can add a parameter to the end-to-end control-plane message that contains information about the translation between the SPI and SPI(#2) values. To mitigate MitM attacks, the mobile host may authorize the SPINAT node to sign the parameter carrying the SPI value in the control message. (The binding (6) in Figure 7.3, on page 150, at the peer host depends on this authorization.)

In addition, the SPINAT node translates a source locator for each message. The SPI$\underline{\overline{n \rightarrow 1}}$source-locator binding (10) depends on the destination-HI$\underline{\overline{1 \rightarrow n}}$destination-locator binding (11). Furthermore, the binding (11) at the SPINAT node depends on selected locator family in bindings (3)(4) at the mobile host. When the peer host is also mobile, the destination-HI$\underline{\overline{1 \rightarrow 1}}$destination-locator binding (12) is dynamic. The bindings (4)(11) depend on the binding (12). The locators in bindings (4)(11)(12) are updated after each peer host's hand-off.

7.2 Bindings for Data Plane Traffic

Figure 7.3, on page 150, illustrates the bindings that are required for data-plane communication between applications. It is good to notice that both end-hosts participate in the generation of the flow protection keys. The payload packets flow from a sending host to a receiving host via an intermediate node that implements the SPINAT functionality. The bindings between identifiers are numbered from (1) to (20).

7.2.1 AID-to-HI Binding

In Figure 7.3, on page 150, the size of the AIDs depend on the socket API. LSI is used in the IPv4 case and HIT in the IPv6 case represents the AID in the AID$\underline{\overline{n \rightarrow 1}}$HI

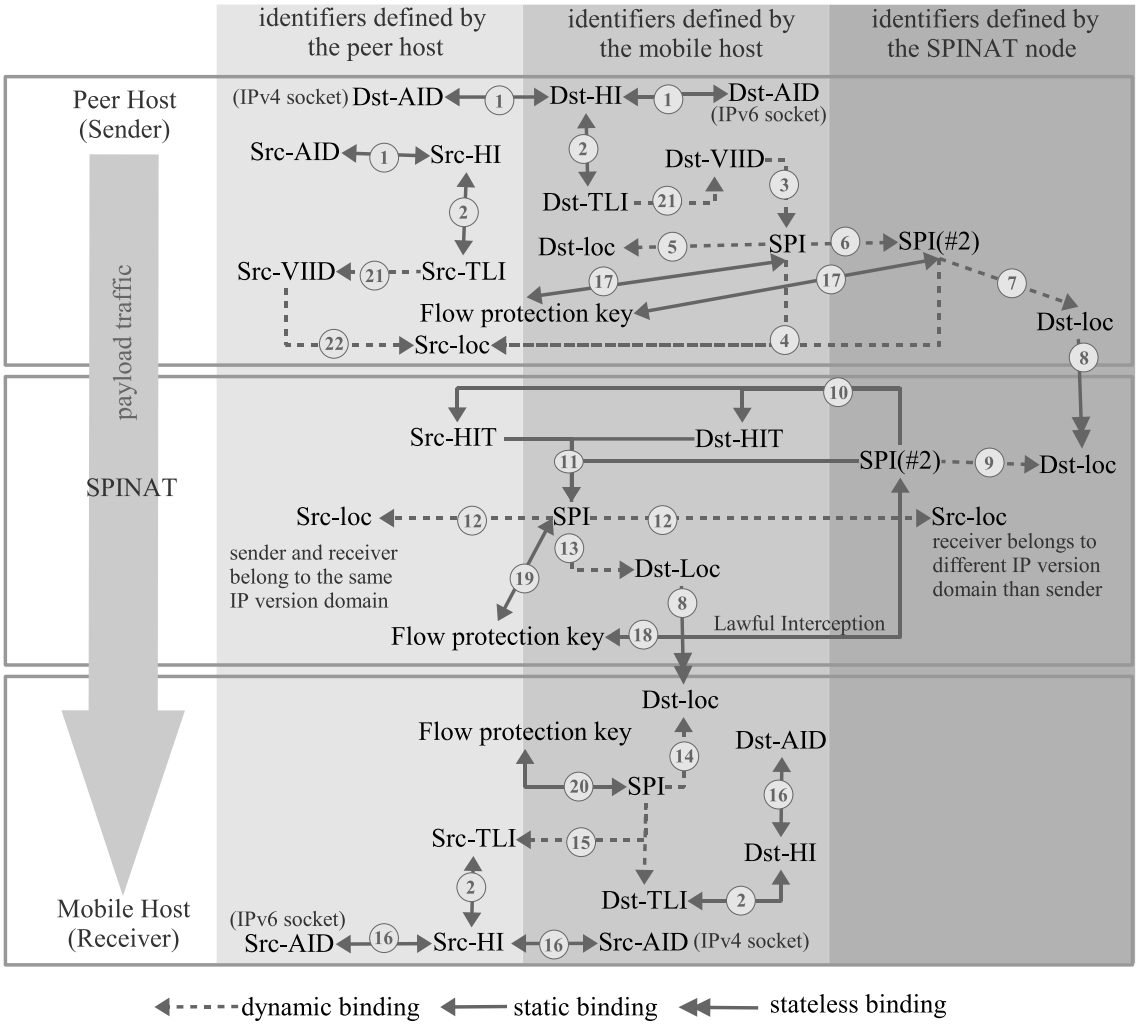

Figure 7.3: Bindings for end-to-end data-plane traffic via SPINAT node.

bindings (1)(16).

The same global scope AIDs can be used at both communicating end hosts, while each host generates its local scope AIDs. The main reason for this is to avoid identifier collisions that may appear with flat namespaces, e.g., with 32-bit long LSIs. The local scope AIDs make it also possible to support different socket APIs at both end hosts. Therefore, the type of AIDs in the bindings (1)(16) can differ from each other.

In this thesis, the one-to-one AID$\overset{1\leftrightarrow 1}{\longleftrightarrow}$HI bindings (1)(16) are static. However, dynamic one-to-many AID$\overset{1\leftrightarrow n}{\longleftrightarrow}$HI bindings make it possible to support process mi-

gration (like in [179]) and change the HIs during a communication session. The one-to-many AID$\overline{1\leftrightarrow n}$HI bindings are for further study.

7.2.2 HIT-to-TLI, TLI-to-VIID and VIID-to-SPI Bindings

In Figure 7.3, on page 150, the source VIID is generated by the peer host while the destination VIID is generated by the mobile host. The destination VIID has been communicated earlier to the peer host during the state establishment. The TLI values are received from the socket at the peer side, and from the packet at the mobile host side.

In Figure 7.3, on page 150, the static HI$\overline{1\leftrightarrow n}$TLI bindings (2) illustrates the translation between application and transport layer identifiers. In the current approach, all AIDs are translated to HITs that are processed like IPv6 addresses at the transport layer. The TLI denotes the {HIT,port value, protocol value}-triplet. The TLI$\overline{n\rightarrow 1}$VIID binding (21) and VIID$\overline{n\rightarrow 1}$SPI binding (3) were analyzed earlier in section 6.1.4 and illustrated also in Figure 6.1, on page 115. The dynamic TLI$\overline{n\leftrightarrow 1}$VIID binding (21) is used for flow mobility as discussed earlier in section 6.1. The SPI$\overline{n\leftrightarrow 1}$locator binding (4) depends on the VIID$\overline{n\leftrightarrow m}$locator binding (22) that is defined in a flow policy.

From the flow mobility point of view, the SPI value carried in the payload packet tags a flow between hosts. The small header size provide efficient bandwidth usage. The trade-off is that a small SPI size may result in collisions. Therefore, SPIs are defined by the receiving nodes. The SPI based multiplexing requires that the SPINAT must be able to translate the SPI.

Figure 7.3 can also be used to illustrate BEET and BEETIN mechanisms. The destination TLI$\overline{n\rightarrow 1}$VIID binding (21) and {SPI,source-HIT,destination-HIT}$\overline{1\leftrightarrow 1}$SPI binding (11) implement the BEET and BEETIN modes for outgoing packets. Respectively, the SPI$\overline{1\rightarrow n}$TLI-pair binding (15) and SPI$\overline{n\leftrightarrow 1}$HIT-pair binding (10) implement the BEET and BEETIN modes for incoming payload packets. Policy based {SPI,source-HIT,destination-HIT}$\overline{1\leftrightarrow n}$VIID binding at intermediate nodes is for future work.

7.2.3 SPI-to-Key Binding

Each SPI is bound to flow protection keying material. In Figure 7.3, on page 150, the static SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$Key bindings (17)(20) at end hosts and bindings (18)(19) at intermediate nodes are used to integrity and confidentiality protect the payload packets. In the transparent SPINAT case, the keying material is not transferred to the SPINAT node and the SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$Key bindings (18)(19) do not exist. In other words, the SPIs are translated without applying any cryptographic operations (see section 5.3.5). However, the receiver may explicitly transfer the key material to the SPINAT node.

The transparent SPINAT node cannot replace the integrity protected SPI(#2) with SPI without additional SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$SPI(#2) binding (6) at the sending host. Otherwise, the packet verification at the receiving host fails. This results in two translations SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$SPI(#2) and SPI(#2)$\overset{1\leftrightarrow1}{\longleftrightarrow}$SPI on the packet forwarding path. It is good to notice that both of the SPIs are bound to same the keying material at the sending host, i.e. binding (17).

The mobile host may also run an explicit exchange with the SPINAT node to transfer the end-to-end flow integrity and/or confidentiality keys to the SPINAT node. If the integrity key is sent to the SPINAT node, the node can verify that each payload packet is received from an authentic sender. If the confidentiality key is transfered to the SPINAT node, it can also implement lawful interception with the SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$Key binding (18). *Lawful interception* denotes interception of packets at some intermediate nodes in accordance with local law.

When multiple hosts establish flows via the same SPINAT node, the key material of the different flows may be transfered to the SPINAT node. In this case, the SPINAT node may create a one-to-many {SPI,source-HIT,destination-HIT}$\overset{1\rightarrow n}{\longrightarrow}$SPI binding (11). In other words, the SPINAT node translates SPIs that are associated with different flow protection keys. This kind of one-to-many binding (11) is useful from the multicast point of view. A single incoming flow is decrypted once using the SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$Key binding (18) and encrypted multiple times per SPI$\overset{1\leftrightarrow1}{\longleftrightarrow}$Key binding (19) before forwarding the copies of the packet to the destinations. However, the approach is for further study.

7.2.4 SPI-to-Locator Binding

In Figure 7.3, on page 150, the dynamic SPI$\underline{^{n\rightarrow 1}}$Locator bindings (4)(5)(7)(14) at end hosts and (9)(12)(13) at intermediate node define the granularity of mobility. When the receiving host is directly reachable from the location of the sending host, the SPI$\underline{^{n\rightarrow 1}}$Locator binding (5) at the peer host depends on the locator update in the SPI$\underline{^{n\rightarrow 1}}$Locator binding (14) at the mobile host. This is also called host mobility (section 5.4). When the peer host is located behind a SPINAT node, the SPI$\underline{^{n\rightarrow 1}}$Locator binding (13) at the SPINAT node depends on the binding (14) update at the mobile host. This is called local mobility (section 6.3).

When the receiving host is reachable via the mobile router, the SPI$\underline{^{n\rightarrow 1}}$Locator binding (7) at the peer host depends on the SPI$\underline{^{n\rightarrow 1}}$Locator binding (9) at the mobile router. This is called network mobility (section 6.4). It is good to notice that the SPI$\underline{^{n\rightarrow 1}}$source-locator binding (12) depends on the SPI$\underline{^{n\rightarrow 1}}$destination-locator binding (9) at the SPINAT node. If the SPIs are bound to different locator families in the bindings (9)(13), the source locator is defined by the intermediate node in binding (12). This is called address-family agility at the network layer (section 6.2).

7.3 Securing Bindings

This section analyzes the earlier discussed granularity levels of mobility (chapter 6) from cryptographic association mechanisms point of view. The presented solutions provide alternative ways to secure identifier-binding updates at the peer nodes. By applying these mechanisms a receiver can verify that the sender is authorized to send the binding update message. The solutions apply two forms of authorization, namely, a cryptographic association between HIs and a non-cryptographic association between HIs and the routing infrastructure. The former authorization mechanism is integrated with (and optionally derived from) the end-to-end HI authentication mechanism. The explicit HI authorization is based on alternative combinations of asymmetric, symmetric and one-way hash chain cryptography. The different cryptographic techniques have reflections on identity privacy. *Asymmetric cryptography* denotes public-key cryptography while *symmetric cryptography* denotes secret key cryptography.

The non-cryptographic association between HIs and locators depends on the as-

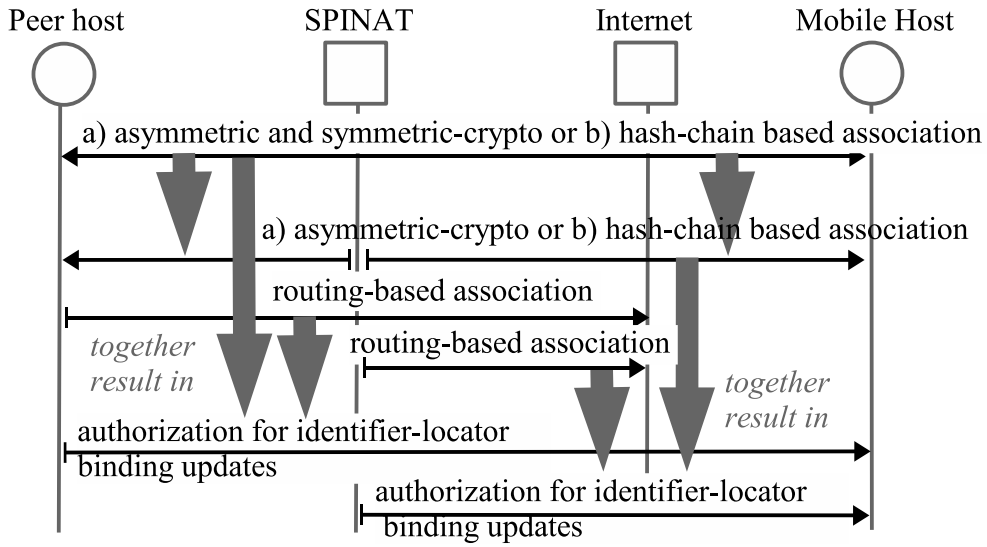

Figure 7.4: Security model for the host mobility and address-family agility solution.

sumption that the routing infrastructure delivers packets to the correct locations according to information carried in the packet headers. Basically, the peer nodes implicitly authorize the routing infrastructure to forward reachability test messages correctly to the destination location that is illustrated in binding (8) in Figure 7.3, on page 150. Vulnerabilities in the routing infrastructure may result in DoS situations and identity leaks for the communicating end hosts. The security problems related to the deployed Internet routing infrastructure are out of scope for this thesis.

7.3.1 Host Mobility and Address Family Agility

Figure 7.4, on page 154, illustrates the security associations that are required for updating bindings in the host mobility and address agility solutions (section 5.4). The model secures the dynamic bindings (3)-(7),(12)-(15) in Figure 7.3, on page 150.

The initial cryptographic association between the HIs of end hosts is established using either asymmetric cryptography [P3][P6] or one-way hash chains [P5]. The former case results also in symmetric-cryptography-based association. The author has focused on identity protection with asymmetric cryptography. The identity privacy solution presented in sections 5.2.6-5.2.8 protects also the identity of end hosts during subsequent binding update signalling. It is good to notice that identity

privacy is also directly bound to the symmetric-cryptography-based association. The symmetric cryptography may result in an identity leak if an attacker is able to bind subsequent SPIs together as a result of traffic analysis. Identity privacy analysis for encrypted payload packets is out of scope for this thesis.

In a basic host mobility scenario (Figure 7.4), the symmetric keying material is not transferred to the SPINAT node. The asymmetric-cryptography or hash-chain-based associations between the SPINAT node and end hosts are derived from the asymmetric-cryptography-based association between the end hosts. From the privacy protection point of view this a problem because the SPINAT node that is also a MitM must be able to verify the identity of the mobile host. The local mobility solution presented earlier in section 7.4 overcomes the key sharing and authentication problems between mobile hosts and SPINAT nodes.

Both the peer host and SPINAT node depends on the routing infrastructure to forward the packets faithfully between the end hosts. Based on the non-cryptographic routing-based associations and the end-to-end cryptographic associations, the peer hosts and the SPINAT nodes authorize the mobile host to update the locator related bindings. When the SPINAT node is not able to verify the identity of the mobile host, a malicious mobile host may establish a cryptographic association with the SPINAT node using spoofed identifiers (section 5.3.4). This requires that also the peer node is malicious and participates the attack. From the identifier spoofing point of view, the SPINAT nodes assume that the peer nodes do not establish cryptographic associations with the mobile nodes sending spoofed messages. In other words, the SPINAT nodes assume that at least one of the end hosts is not malicious.

7.3.2 Local Mobility

Figure 7.5, on page 156, illustrates the security associations that are required for updating bindings in the local mobility solution (section 6.1). The model secures the dynamic bindings (13)(14) in Figure 7.3, on page 150.

The initial asymmetric and symmetric-cryptography-based associations between HIs of end hosts results in hash chain and symmetric-cryptography-based association between the mobile host and the first SPINAT node. Furthermore, each subsequent security association between different SPINAT nodes and the mobile host are derived from the previous security association. The model is based on a chain of security

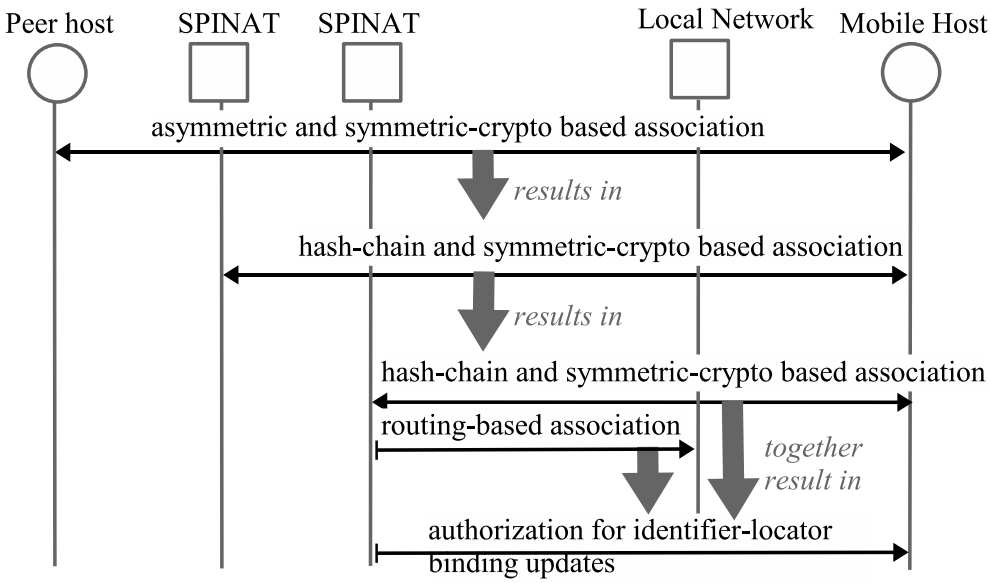

Figure 7.5: Security model for the local mobility solution.

associations between the mobile hosts and SPINAT nodes. The SPINAT nodes reply to the binding update messages on behalf of the peer nodes.

The hash chains provide an alternative authentication mechanism for asymmetric cryptography at SPINAT nodes. The end hosts are able to keep secret their asymmetric-cryptography-based identities. The trade-off is that the hash-chain-based authentication becomes a potential source of identity leaks. The hash chain values make it possible to bind subsequent binding updates together. As discussed earlier in section 6.3.4, the trade-off between privacy protection and the local-mobility-based signalling optimization is bound to the length of the hash chains.

It is good to notice that the scope of the binding update signalling in global and local mobility scenarios is different. Although it is possible to trace a mobile host to some extent in the local network, the solution protects the mobile host's identity from attackers located in the global domain. In addition, the local mobility solution hides the accurate location of the mobile host inside the local domain from the peer nodes.

The SPINAT nodes depend on the local routing infrastructure to faithfully forward packets between them and the mobile hosts. The SPINAT node located in the local network authorizes the mobile node to update the locator related bindings.

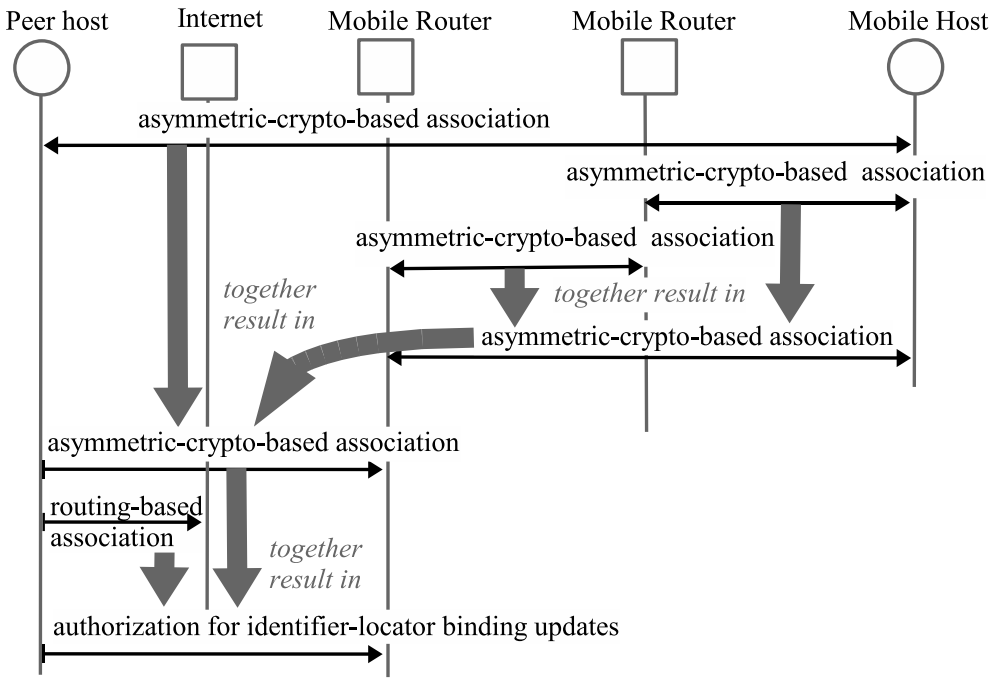

Figure 7.6: Security model for the network mobility solution.

The authorization depends on the routing-based association and the hash chain and symmetric-cryptography based associations.

7.3.3 Network Mobility

Figure 7.6, on page 157, illustrates the associations that are required for updating bindings in the network mobility solution (section 6.4). The model secures the dynamic bindings (7)(9) in Figure 7.3, on page 150.

The initial association between the mobile host and peer host is created using asymmetric cryptography. The mobile host also establishes an asymmetric-cryptography-based association with the closest mobile router. The nested mobile routers create asymmetric-cryptography-based associations with each other. A certificate chain provides a way to express the authorizations between the mobile host and the nested mobile routers. This results in an asymmetric-cryptography-based association between the peer host and the mobile router attached to the global domain.

The security associations in the network mobility and the earlier presented local mo-

bility (section 7.4) are kind of mirror images of each other. The main difference is that the associations in the network mobility are based on asymmetric cryptography, while the associations in the local mobility are based on hash chains and symmetric cryptography. In the local mobility scenario the associations are established between mobile host and intermediate nodes. In the network mobility scenario, the associations are established between the peer host and intermediate node. In the local mobility case, the intermediate node replies on behalf of the peer host. In the network mobility case, the intermediate nodes reply on behalf of the mobile node.

The mobile routers and the peer node do not run mobility state establishment exchanges between them. This makes it difficult to protect the mobile router's identity from attackers. The mobile routers' identity protection and certificate encryption is for further study and is out of scope for this thesis.

7.3.4 Flow Mobility

Figure 6.1, on 115, illustrates three bindings TLI$\overline{\overline{m \rightarrow n}}$VIID, VIID$\overline{\overline{n \leftrightarrow 1}}$SPI and SPI$\overline{\overline{n \leftrightarrow 1}}$locator bindings that must be secured in the flow mobility solution. The same bindings are also illustrated in Figure 7.3, on page 150. The security associations required to update the locator in the SPI$\overline{\overline{n \leftrightarrow 1}}$locator binding were presented earlier in section 7.3.1. Moreover, the association between TLI$\overline{\overline{m \rightarrow n}}$VIID is expressed with an authorization certificate that is signed by the HI that generates the VIID in the binding (section 6.1.10).

Finally, the cryptographic associations that are required to update the SPI in the VIID$\overline{\overline{n \leftrightarrow 1}}$SPI binding is illustrated in Figure 7.7, on page 159. Figure 7.7 is based on the assumption that the communication paths between the end hosts have been established before the flow mobility takes place. In addition, the peer node has made reachability tests for the locations, based on the cryptographic associations presented in Figure 7.4, on page 154.

The mobile host establishes an asymmetric-cryptography based association with the peer node to update the VIID$\overline{\overline{n \rightarrow 1}}$SPI binding. However, the mobile host may also establish an asymmetric-cryptography based association with the SPINAT node and transfer encryption or integrity protection related end-to-end symmetric keying material to the authenticated SPINAT node. This would requires additional {SPI,source-HIT,destination-HIT}$\overline{\overline{1 \leftrightarrow 1}}$VIID binding and VIID$\overline{\overline{n \rightarrow 1}}$SPI binding at a

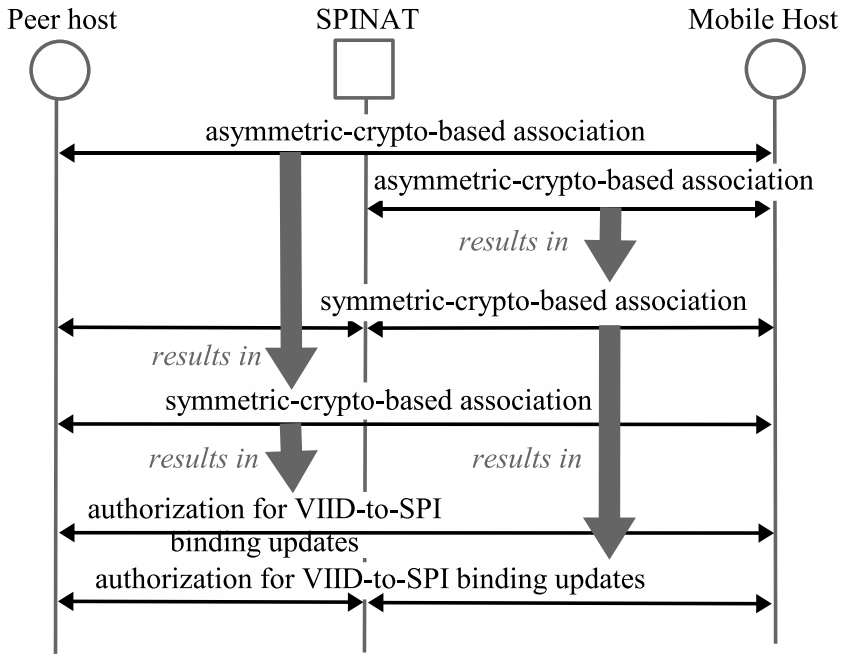

Figure 7.7: Security model for the flow mobility solution.

multi-homed SPINAT node to split flows similar to multi-homed end hosts (see Figure 7.3, on page 150). However, the policy based flow mobility for encrypted payload packets at SPINAT nodes is for further study. The basic approach, presented in Figure 7.2, on page 148, uses the VIIDs that are defined by the end hosts.

7.4 Integrated Authentication, Authorization, Key Sharing, and Location Update Signalling

Figure 7.8, on page 160, illustrates the amount of signalling that is required to establish security associations between nodes and to update the location of a mobile host in different mobility scenarios. The figure also describes the relationships between the exchanges that is also illustrated in [P9].

The required amount of end-to-end state establishment signalling (1 in Figure 7.8) depends on the number of peer hosts. The authenticated two round trip Diffie-Hellman (HIP) exchange is faster than, e.g., a related IKE exchange [P4]. The mobility state establishment must take place before the update exchange is run

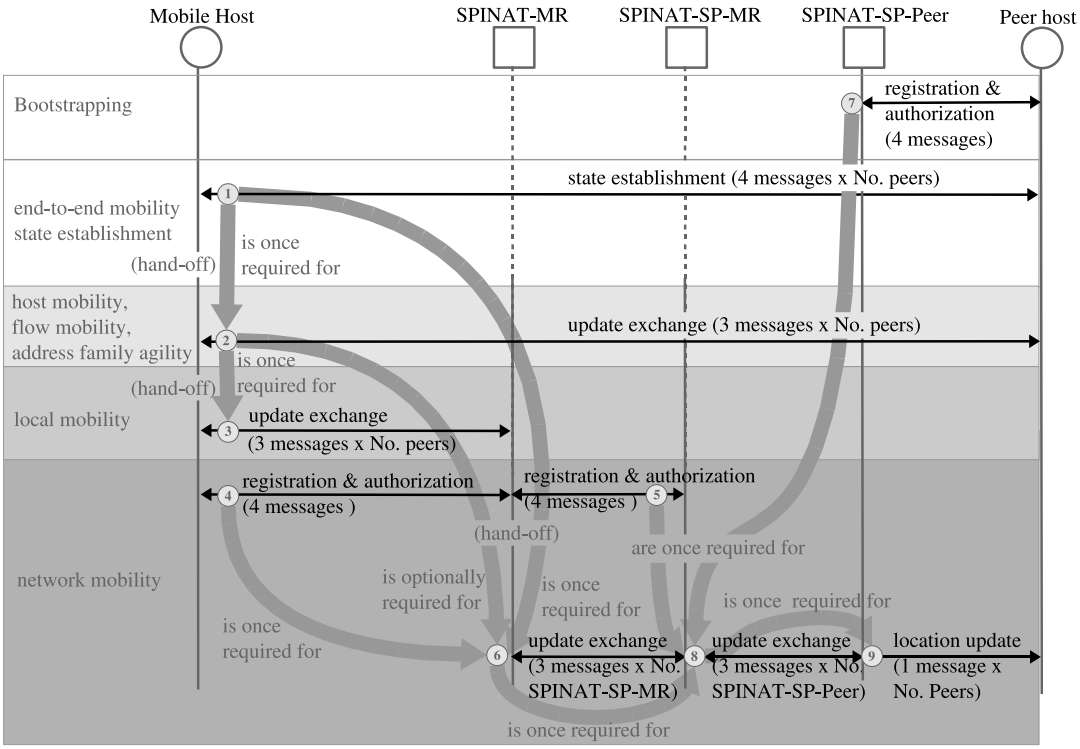

Figure 7.8: Relationships between the control-plane exchanges.

between nodes.

The mobile host runs once the end-to-end update exchange (1 or 2 in Figure 7.8) with each peer host to establish SPINAT states for local and/or network mobility. The end-to-end update exchange (2 in Figure 7.8) consists of three messages that are sent directly between end hosts [P3][P4]. The main difference from Mobile IPv6 [45] is that part of the binding update messages are triangularly routed via a home agent in Mobile IPv6. In addition, the end hosts do not establish SAs between them in Mobile IPv6.

The SPINAT nodes are able to support local hand-offs (3 in Figure 7.8) after the end-to-end exchanges between the hosts. The mobile host and the closest SPINAT node, i.e. a mobile anchor point, establish an SA between each other during the update exchanges (section 6.3). Due to the integrated key-sharing mechanism, the amount of signalling is not increased during local hand-offs between the mobile host and mobile anchor points. Basically, the three-way update exchange is optimal at any granularity level of mobility. In practice, it is not possible to verify a location

with fewer than two messages. The local hand-off latency, excluding the link layer exchanges, consists of the integrated three-way update and key sharing exchange that is run between the mobile host and the local SPINAT node.

The mobile hosts can also authorize (4 in Figure 7.8) the closest SPINAT node, i.e. a mobile router, to run update exchanges on behalf of the mobile host to minimize signalling inside the moving network. Furthermore, the mobile router can delegate the signalling rights to another SPINAT node, i.e a signalling proxy, located in the fixed network (5 in Figure 7.8), to minimize over-the-air signalling. The authorization exchanges do not depend on the number of peer hosts. However, the mobile host needs to run an end-to-end exchange (1 or 2 in Figure 7.8) with each peer host via the SPINAT node to establish states for the end-to-end connections at the SPINAT nodes.

Based on the authorization between the mobile host and the mobile router (4 in Figure 7.8), the mobile router runs an update exchange (6 in Figure 7.8) with its signalling proxies after its hand-off. Furthermore, the authorized (5 in Figure 7.8) signalling proxies run update exchanges (8 in Figure 7.8) directly with peer hosts or with other proxies at the peer side.

Also the peer hosts can authorize signalling proxies, i.e. SPINAT nodes, to run the reachability test on behalf of the peer hosts (7 in Figure 7.8). Thus, the mobile router hand-off results in update exchanges between signalling proxies (8 in Figure 7.8). The amount of update exchanges is bound to the number of signalling proxies, not to the number of end hosts. Finally, the peer hosts are informed of the mobile hosts current location with a location update message (9 in Figure 7.8). Other kinds of proxy based approaches that completely hide the mobility from the peer hosts are out of scope for this thesis.

The delegation of signalling rights, and the distribution of control signalling between signalling proxies in the fixed network reduces the over-the-air bandwidth consumption and results in optimized packet forwarding paths between end hosts. In addition, the multiple mobility state machines running in parallel at different signalling proxies reduce the message processing time, thus minimizing the hand-off latency. The analysis of the hand-off times and related measurement results are presented in publication [P9].

7.5 Compact Implementation

Figures 7.9 (page 163) and 7.10 (page 164) illustrate the HIP specific lines of C-code per week in the user-space daemon and in the FreeBSD6 kernel. The figures are based on the most important C-files in the CVS repository including also the comment lines. The required changes in libc, test suites, administration tools, crypto libraries and header files are not included in the charts.

The end-to-end host mobility implementation requires around 13 000 lines of code for the user-space daemon and 3500 lines of code in the kernel. Together, the 16500 lines of code support secure host and flow-based hand-offs, address-family agility at end hosts, rendezvous functionality and the BEET mode.

Additional granularity levels require another 18 500 lines of code for the user-space daemon and 1000 lines of code in the kernel. Together, the 19 500 lines of code add support for transparent state establishment and update at SPINAT nodes, address-family agility at SPINAT, BEETIN mode, flow mobility, transparent registration with mobile routers and minimal signalling proxy functionality; i.e., it is used by the mobile router. The SPINAT implementation is around 11 000 lines of code, while the basic mobile router functionality requires around 7 500 lines of additional code. It is good to notice that the current code does not contain privacy nor nested moving network support. In addition, the earlier implemented local mobility protocol [P4] is not supported in the CVS HEAD.

The current HIP based prototype consists of 36 000 lines code. For example, a Mobile IPv6 implementation[18] requires around 23 000 lines of code in user space and a couple of thousand lines of code in the kernel. Moreover, the Mobile IPv6 requires an Internet Key Exchange (IKE) daemon implementation. A user-space IKE daemon implementation[19] requires around 58 000 lines of code. Together, an IPv6 based host mobility requires over 80 000 lines of C code. The lines of code in the presented examples are counted in the same way as in Figures 7.9 (page 163) and 7.10 (page 164) .

It seems that the presented namespace bindings and the design choices made in this thesis result in a compact implementation, compared to the other IP based mobility and security protocols.

[18]MIPL Mobile IPv6 for Linux code, version 2.0.2
[19]Racoon in IPsec-tools 0.6.7

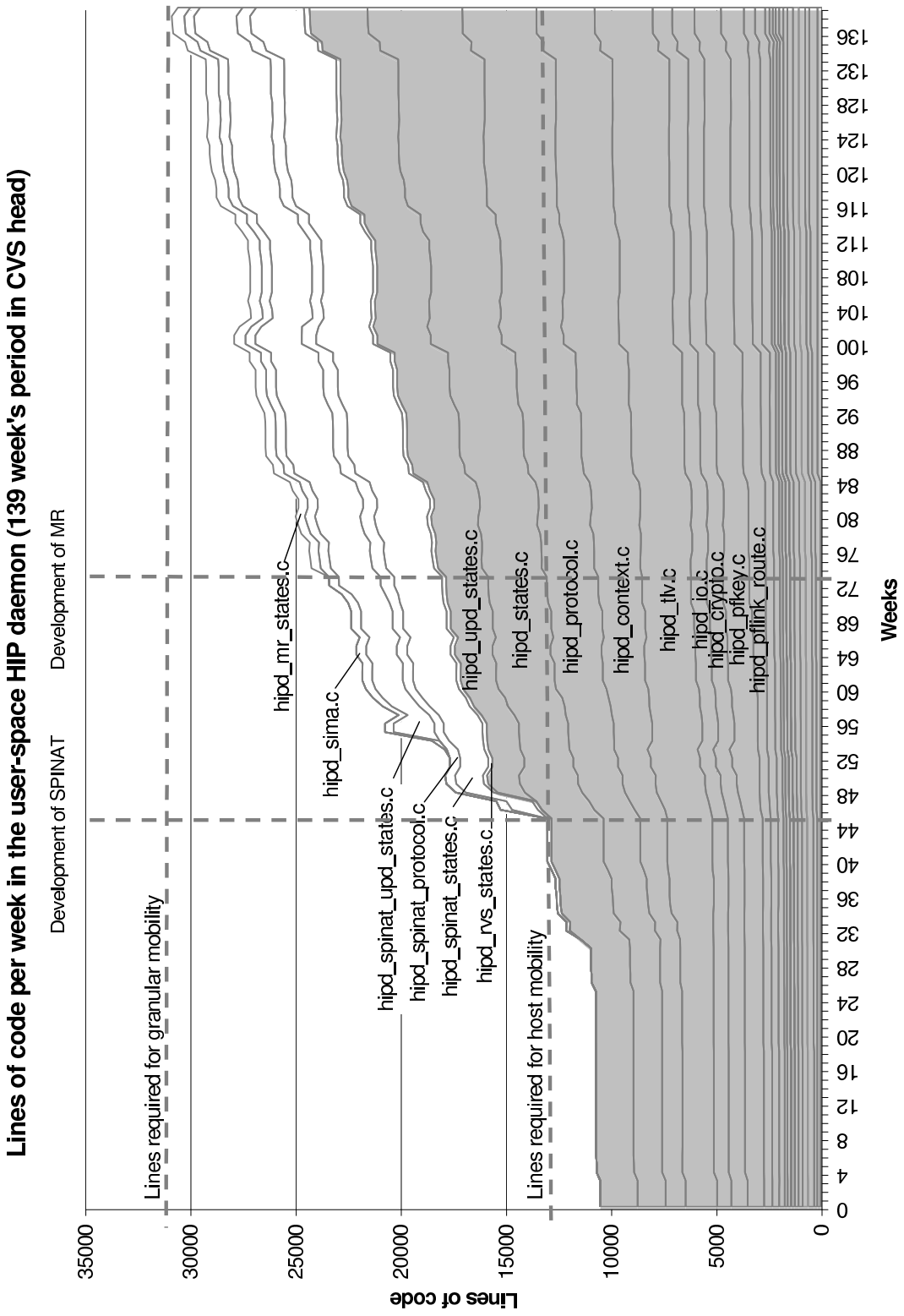

Figure 7.9: Lines of code per week in the user-space daemon.

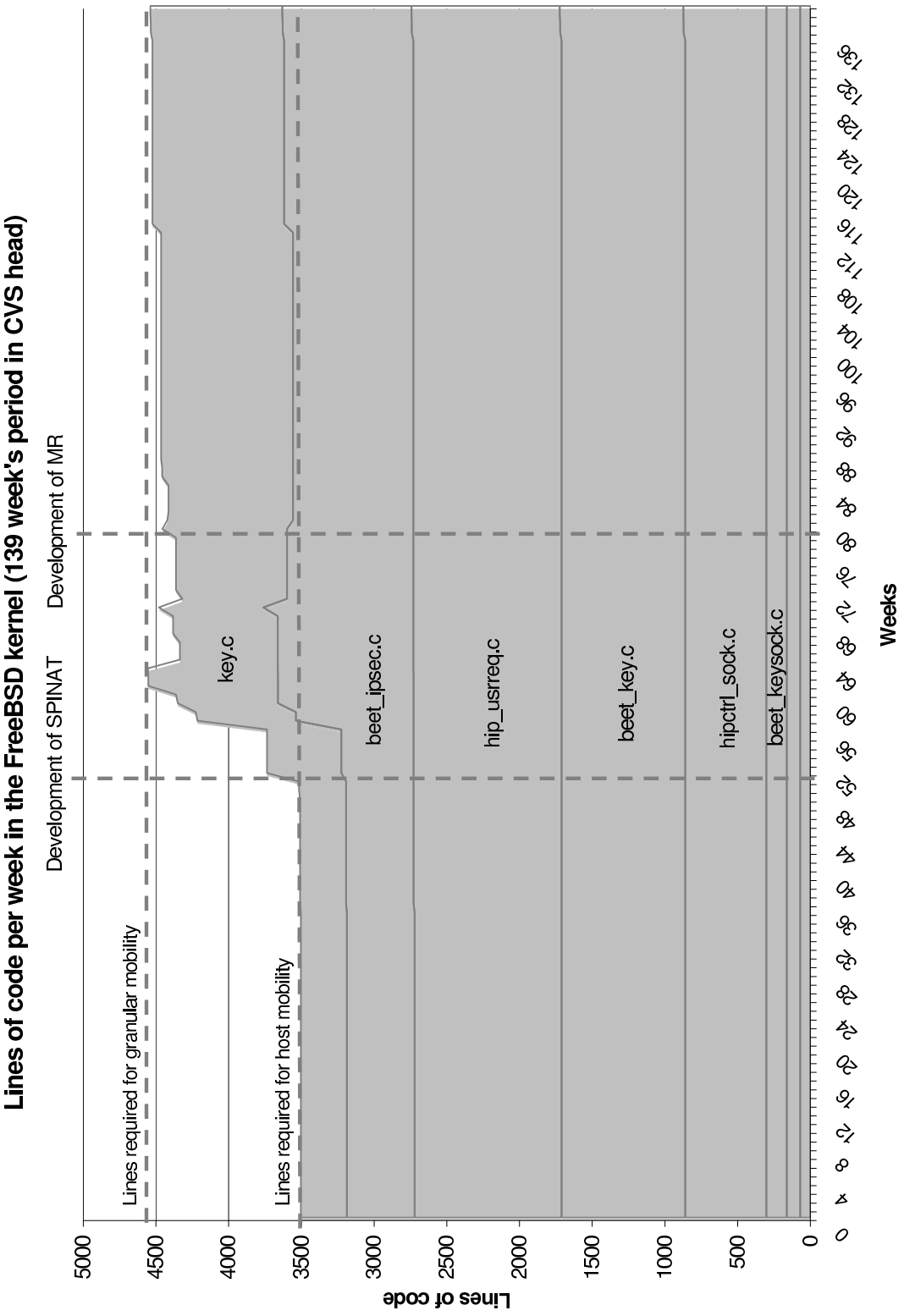

Figure 7.10: Lines of code per week in the kernel.

8 Conclusions

The existing Internet architecture can be seen as a framework that consists of IP transport, a host stack, and applications. The standard interfaces between these elements make it possible to deploy new kinds of services in the Internet without changing the underlying architecture. In this work, the host stack is changed in a radical way, eventually leading to new kinds of business models. Still, the untouched IP transport and applications are able to use the old interfaces providing backward compatibility for the existing services.

During the work, the author has incrementally introduced new pieces to the basic HIP-based identifier-locator split architecture. Based on these contributions, the enhanced HIP architecture covers now a rich set of new granularity levels of mobility. This thesis defines the relationships between the presented mobility dimensions.

While it is possible to achieve most of the presented granularity levels using other approaches, the main advantages of the approach taken in this thesis are visible in the security mechanisms and resulting mobility optimizations. The approach integrates host authentication, key-sharing, and mobility mechanisms together in a novel and seamless way.

The author has defined a set of new dynamic bindings for the host stack, and the relationships between them to support multiple granularity levels of mobility. The presented approaches for flow mobility, local mobility, network mobility, and address-family agility solutions are based on utilizing these bindings. The solutions are independent of the link-layer access technology and IP version. Depending on the solution, the new bindings are located at the end hosts or also at the intermediate nodes.

The authentication, authorization and delegation mechanisms are integrated with the mobility-related signalling. To overcome a number of authentication-related problems, the approach taken in this thesis relies on cryptographic host identifiers for both host naming and authentication purposes. The solutions integrate the cryptographic host identifiers with mobility-management signalling to protect the dynamic bindings from traffic redirection and DoS security vulnerabilities. The security associations between the nodes provide integrity and confidentiality protection, and optimized forwarding paths for payload packets.

The presented hash-chain-based weak authentication techniques are computationally efficient and scalable compared to using a PKI or manual keying. On the flip side, they are more vulnerable to MitM attacks than public-key-based authentication schemes. In addition, the author provides a solution for identity protection for end hosts. This thesis presents the first two-round-trip authenticated Diffie-Hellman key exchange that is able to protect the identities of the communicating hosts from passive and active attackers when the attackers do not have prior knowledge of the public keys or the destination of the packets.

In this thesis, delegation of signalling rights and aggregation of signalling are applied to host-identifier-based network mobility. This reduces the amount of control signalling in a moving network, between the moving network and the Internet, and between the peer hosts and the Internet.

To optimize local binding updates between mobile hosts and intermediate nodes, the author also defines a three-way update exchange that is used both for local and macro-mobility management, and for key sharing. It is the first three way exchange that simultaneously updates a location of a mobile host and establishes a security association between the mobile host and a mobility anchor point without requiring a priori knowledge of the mobile host. The solution protects the nodes from MitM attackers that are located between the mobile host and the anchor point. The optimized key sharing minimizes the local hand-off latency without increasing the amount of security-related signalling.

The globally unique and IP-version-independent host identifiers make it possible to support address-family agility both at the application and network layers. The author provides the first solution supporting secure communication between different versions of socket APIs over different IP versions. In addition, backward compatibility with most of the existing applications and the integrated implementation makes it easy to support simultaneously multiple granularity levels of mobility.

The presented flow-mobility approach is the first IP-based solution for dividing single traffic flows dynamically between network interfaces with user, peer and third party defined policy rules. The introduced virtual-interface identifiers are used for identifying traffic flows at the end hosts and at the intermediate nodes. The identifiers are used for mitigating synchronization problems between communication states and for defining static flow-policy rules in certificates.

From the semantic point of view, the required changes in the host stack are larger than, e.g., in the MIPv6 based solutions. However, the actual implementation size of the approach taken in this thesis is considerable smaller than size of the comparable MIPv6 approach. This diminishes the complexity of the implementation and results in fewer bugs in the code. Furthermore, it speeds up the development process and reduces implementation costs.

The author believes that the deployment of the presented architecture strongly depends on the end-users' awareness of security risks in data communication, and on the service providers' concern of losing good reputation as a consequence of security flaws. From the operators' viewpoint, the user requirements result in reactive deployment of new technologies. The service providers can also act in a proactive way to protect their clients from losing time or money due to security flaws. The author believes that, in both cases, the operators will benefit from the presented transparent security and mobility mechanisms that do not cause trade-offs in the service usability but increase flexibility in data communication. The end users alike benefit from private and resilient communication, independent of the available access technologies, IP versions, and the location of the users.

List of Publications

This thesis consists of the following 9 publications that are listed below.

P1 Jukka Ylitalo, Tony Jokikyyny, Tero Kauppinen, Antti J. Tuominen, and Jaakko Laine, "Dynamic Network Interface Selection in Multihomed Mobile Hosts", In Proc. of the Thirty-Sixth Hawai'i International conference on System Sciences (HICSS-36), published on cd-rom, abstracts p.315, Big Island, Hawai'i, US, January 6-9, 2003, ISBN 0-7695-1874-5, Publisher: IEEE Computer Society (Ed: Ralph H. Jr. Sprague).

P2 Jukka Ylitalo, Petri Jokela, Jorma Wall, and Pekka Nikander, "End-point Identifiers in Secure Multi-Homed Mobility", in Proc. of the 6th International Conference On Principles Of DIstributed Systems (OPODIS'02), pp. 17-28, Reims, France, December 11-13, 2002. Studia Informatica Universalis, Vol. 3, Suger, Saint-Denis, rue Catulienne, France 2002, ISBN 2-912590-26-4, Publisher: Université de Reims Champagne-Ardenne (Eds: Alain Bui, Hacene Fouchal).

P3 Pekka Nikander, Jukka Ylitalo, and Jorma Wall, "Integrating Security, Mobility, and Multi-Homing in a HIP Way", in Proc. of Network and Distributed Systems Security Symposium (NDSS'03), pp. 87-99, San Diego, CA, US, February 6-7, 2003, ISBN 1-891562-16-9, Publisher: Internet Society

P4 Jukka Ylitalo, Jan Melén, Pekka Nikander, and Vesa Torvinen, "Re-thinking Security in IP based Micro-Mobility", in Proc. of the 7th International Conference on Information Security Conference (ISC'04) , pp. 318-329, Palo Alto, CA, USA, September 27-29, 2004, LNCS 3225, ISSN 0302-9743, ISBN 3-540-23208-7, Publisher: Springer (Eds: Kan Zhang, and Yuliang Zheng)

P5 Vesa Torvinen, and Jukka Ylitalo, "Weak Context Establishment Procedure for Mobility Management and Multi-Homing", in Proc. of the Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS'04), pp. 111-123, Windermere, UK, September 15-18, 2004, ISBN 0-387-24485-9, Publisher: Springer (Eds: David Chadwick, and Bart Preneel)

P6 Jukka Ylitalo, and Pekka Nikander, "BLIND: A Complete Identity Protection Framework for End-points", in Proc. of the 12th International Workshop on Security Protocols, Cambridge, UK, April 26-28, 2004, LNCS 3957, ISSN 0302-9743, ISBN 3-540-40925-4, Publisher: Springer (Eds: Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe).

P7 Jukka Ylitalo, Patrik Salmela, and Hannes Tschofenig, "SPINAT: Integrating IPsec into Overlay Routing", in Proc. of the First International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm'05), pp. 315-326, Athens, Greece, September 5-9, 2005, ISBN 0-7695-2369-2, Publisher: IEEE Computer Society (Ed: JD Cantarella).

P8 Jukka Ylitalo, and Pekka Nikander, "A new Name Space for End-Points: Implementing secure Mobility and Multi-homing across the two versions of IP", in Proc. of the Fifth European Wireless Conference, Mobile and Wireless Systems beyond 3G, pp. 435-441, Barcelona, Spain, February 24-27, 2004, ISBN 84-7653-846-4, Publisher: SCI UPC (Eds: Olga Casals, Jorge Carcia-Vidal, Jose M Barcelo, and Llorenc Cerda).

P9 Jukka Ylitalo, Jan Melén, Patrik Salmela, and Henrik Petander, "An Experimental Evaluation of a HIP based Network Mobility Scheme", in Proc. of the 6th International Conference on Wired/Wireless Internet Communications (WWIC 2008), pp. 139-151, Tampere, Finland, May 28-30, 2008, LNCS 5031, ISSN 0302-9743, ISBN 978-3-540-68805-1, Publisher: Springer (Eds: J. Harju et al). Based on the current author's invention of applying the delegation of signalling rights scheme to the moving network context: Jari Arkko, Jukka Ylitalo, and Pekka Nikander, "Addressing mechanisms in mobile IP", United States Patent 20030084293, May, 2003. `http://www.freepatentsonline.com/20030084293.html`

Author's contribution

The author's contributions are visible in the four main areas. First, in the delegation of signalling rights scheme. Second, in the flow mobility scheme. Third, in the one-way hash chain schemes. Fourth, in the identifier-locator translation schemes. The detailed contributions are presented in the following.

Delegation of Signalling Rights

The author has defined a way to implement a moving network architecture using the delegation of signalling rights scheme [P9], discussed earlier in section 6.4.

Flow Mobility

The author has analyzed the namespace problem field from the flow based mobility point of view. He has been a member of a design and implementation team that has defined a simultaneous multi-access architecture [P1]. The author's main contribution in the team has been following. He has defined a policy based mechanism for selecting source and destination interfaces for outgoing flows ([P1], chapters III (not III-A)& VII), discussed earlier in section 6.1. The author has also defined a model of policy based socket binding ([P2], Figure 2; [P1], Figure 8; [P3], Figures 9 & 12), discussed earlier in section 6.1.

One-way Hash Chain

The author has analyzed how to secure the namespace bindings using weak authentication techniques, mainly one-way hash chains and secret splitting techniques. He has defined one-way hash chain schemes to protect mobility and multi-homing signalling.

The author has defined a way to secure a local and macro mobility protocol using one-way hash chains, and secret splitting techniques [P4], discussed earlier in section 6.3.

The author has defined a way to use one-way hash chains between mobile and multi-homed end hosts to secure a context establishment protocol [P5], discussed earlier in sections 5.2.9-5.2.11.

The author has defined a context establishment protocol that provides identity protection for end points [P6], discussed earlier in section 5.2.6-5.2.8.

Identifier-Locator Translation

The author has analyzed the namespace binding problems from the address-family's point of view at end hosts and intermediate nodes. He has been a member of a design and implementation team that has defined a way to implement mobility and multi-homing across different address families. The communicating legacy applications may use different legacy socket APIs. The author has defined a new IPsec mode for intermediate nodes. The new mode implements address-family translation between different addressing domains (see: [P8][P4][P9]) that is discussed earlier in sections 5.3.5 and 6.2.

The author has defined a Security Parameter Index (SPI) multiplexed Network Address Translation (NAT), called SPINAT. This new intermediate node transparently establishes HIP state during end-to-end communication (see: [P7] [P6] [P4]; [P3], chapters 4.3 & 5.3.). This is discussed earlier in sections 5.3 and 5.4.

References

[1] A. Conta and S. Deering and M. Gupta. 2006. Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443.

[2] A. Marine and J. Reynolds and G. Malkin. 1994. FYI on Questions and Answers, Answers to Commonly asked "New Internet User" Questions. RFC 1594.

[3] A. Perrig and D. Song and R. Canetti and J. Tygar and B. Briscoe. 2005. Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction. RFC 4082.

[4] M. Abadi and C. Fournet. 2004. Private authentication. Theoretical Computer Science 322, pages 427–476.

[5] D Adkins, K. Lakshminarayanan, A. Perrig, and I. Stoica. 2003. Towards a More Functional and Secure Network Infrastructure. Technical Report UCB Tech. Rep. UCB/CSD-03-1242, Univ. California, Berkeley.

[6] B. Ahlgren, J. Arkko, L. Eggert, and J. Rajahalme. 2006. A Node Identity Internetworking Architecture. In: Proc. of the IEEE INFOCOM 2006 Global Internet Workshop. Barcelona, Spain.

[7] W. Aiello, S. Bellovin, M. Blaze, R. Canetti, J. Ionnadis, A. Keromytis, and o. Reingold. 2002. Efficient, dos-resistant, secure key exchange for internet protocols. Washington, DC, USA.

[8] R. Anderson, F. Bergadano, B. Crispo, J.-H. Lee, C. Manifavas, and R. Needham. 1998. A new family of authentication protocols. ACM SIGOPS Operating Systems Review 32, pages 9–20.

[9] A.S. Tanenbaum. 1996. Computer Networks (Third edition). Prentice Hall.

[10] S. Aust, D. Proetel, N. A. Fikouras, C. Pampu, and C. Görg. 2003. Policy based Mobile IP Handoff Decision (POLIMAND) using Generic Link Layer Information. In: Proc. of the 5th IEEE International Conference on Mobile and Wireless Communication Networks (MWCN 2003). Singapore.

[11] B. Aboba and J. Carlson and S. Cheshire. 2006. Detecting Network Attachment in IPv4 (DNAv4). RFC 4436.

[12] B. Carpenter. 1996. Architectural Principles of the Internet. RFC 1958.

[13] B. Carpenter. 2000. Internet Transparency. RFC 2775.

[14] B. Schneier. 1996. Applied Cryptography (Second Edition). Wiley.

[15] B.A. Forouzan. 2004. Data Communications and Networking (third edition). Mc Graw Hill.

[16] A. Bakre and B. R. Badrinath. 1995. I-TCP: indirect TCP for mobile hosts. In: Proc. of the International Conference on Distributed Computing Systems. IEEE, Piscataway, NJ, USA, Vancouver, Canada.

[17] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish. 2004. A Layered Naming Architecture for the Internet. In: Proc. of the Sigcomm'04. Portland, OR, USA.

[18] H. Ballani and P. Francis. 2005. Towards a global IP Anycast service. In: Proc. of the Sigcomm'05. Philadelphia, PA, USA.

[19] S. Bellovin and M. Merritt. 1994. An attack on the Interlock Protocol when used for authentication. IEEE Transactions 40, pages 273–275.

[20] A. Biggadike, D. Ferullo, G. Wilson, and A. Perrig. 2005. NATBLASTER: Establishing TCP Connections Between Hosts Behind NA Ts. In: Proc. of the SIGCOMM'05 Asia Workshop. Beijing, China.

[21] C. Ellison and B. Frantz and B. Lampson and R. Rivest and B. Thomas and T. Ylonen. 1999. SPKI Certificate Theory. RFC 2693.

[22] C. Huitema. 1995. Multi-homed TCP. Internet-Draft, work in progress.

[23] C. Huitema. 2006. Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs). RFC 4380.

[24] C. Larsson and H. Levkowetz and H. Mahkonen and T. Kauppinen. 2006. A Filter Rule Mechanism for Multi-access Mobile IPv6. Internet-Draft, work in progress.

[25] C. Molina-Jimenez abd L. Marshall. 2001. True anonymity without mixes. In: Proc. of the IEEE Workshop on Internet Applications. San Jose, CA, USA.

[26] C. Perkins. 2002. IP Mobility Support for IPv4. RFC 3344.

[27] R. Caceres and L. Iftode. 1994. The Effects of Mobility on Reliable Transport Protocols. In: Proc. of the International Conference on Distributed Computing Systems.

[28] M. Caesar, M. Castro, E. B. Nightingale, G. O, and A. Rowstron. 2006. Virtual Ring Routing: Network Routing Inspired by DHTs. In: Proc. of the Sigcomm'06. Pisa, Italy.

[29] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker. 2006. ROFL: Routing on Flat Labels. In: Proc. of the Sigcomm'06. Pisa, Italy.

[30] K. L. Calvert, J. Griffioen, and S. Wen. 2002. Lightweight Network Support for Scalable End-to-End Services. In: Proc. of the Sigcomm'02. Pittsburgh, PA, USA.

[31] A. T. Campbell and J. Gomez-Castellanos. 2000. IP micro-mobility protocols. ACM SIGMOBILE Mobile Computing and Communications Review 4, pages 45–53.

[32] C. Carter, R. Kravets, and J. Tourrilhes. 2005. Contact Networking: A Localized Mobility System. In: Proc. of the 3rd international conference on Mobile systems, applications, and services (Mobisys'05). Seattle, Washington, USA.

[33] A. Carzaniga and A. L. Wolf. 2003. Forwarding in a Content-Based Network. In: Proc. of the Sigcomm'03. Karlsruhe, Germany.

[34] C. Castelluccia. 2000. HMIPv6: A Hierarchical Mobile IPv6 Proposal. ACM Mobile Computing and Communication Review (MC2R) 4, pages 48 – 59.

[35] V. Cerf and R. Kahn. 1974. A Protocol for Packet Network Intercommunication. In: Proc. of the IEEE Trans on Comm, COM-22, No. 5.

[36] V. Cerf and P. Kirstein. 1978. Issues in Packet Network Interconnection. In: Proc. of the IEEE, v.66, 11.

[37] C. Chau. 2006. Policy-based Routing with Non-strict Preferences. In: Proc. of the Sigcomm'06. Pisa, Italy.

[38] D. Clark. 1988. The Design Philosophy of the DARPA Internet Protocols. In: Proc. of the SIGCOMM. Published also in ACM SIGCOMM, Computer Communication Review, volume 25, number 1, January 1995.

[39] D. Clark, R. Braden, A. Falk, and V. Pingali. 2003. FARA: Reorganizing the Addressing Architecture. In: Proc. of the ACM SIGCOMM'03 Workshop on Future directions in network architecture, pages 313–321. Karlsruhe, Germany.

[40] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden. 2002. Tussle in Cyberspace: Defining Tomorrow's Internet. In: Proc. of the Sigcomm'02. Pittsburgh, PA, USA.

[41] D. Clark and D. Tennenhouse. 1990. Architectural Considerations for a New Generation of Protocols. In: Proc. of the ACM SIGCOMM, pages 200–208.

[42] D. Clark and L. Chapin and V. Cerf and R. Braden and R. Hobby. 1991. Towards the Future Internet Architecture. RFC 1287.

[43] D. Cohen. 1996. Introduction to Computer Theory. Wiley.

[44] D. Harkins and D. Carrel. 1998. The Internet Key Exchange (IKE). RFC 2409.

[45] D. Johnson and C. Perkins and J. Arkko. 2004. Mobility Support in IPv6. RFC 3775.

[46] D. Meyer and L. Zhang and K. Fall. 2007. Report from the IAB Workshop on Routing and Addressing. RFC 4984.

[47] D. Plummer. 1982. An Ethernet Address Resolution Protocol – or – Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. RFC 826.

[48] D.E. Comer. 2000. Internetworking with TCP/IP, principles, protocols, and architectures (Volume 1, fourth edition). Prentice Hall.

[49] C. de Launois and M. Bagnulo. 2006. The Paths Towards IPv6 Multihoming. IEEE Communications Surveys and Tutorials 8.

[50] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert. 2005. Network Mobility (NEMO) Basic Support Protocol. RFC 3963.

[51] S. Dharmapurikar, P. Krishnamurthy, and D. E. Taylor. 2003. Longest Prefix Matching Using Bloom Filters. In: Proc. of the Sigcomm'03. Karlsruhe, Germany.

[52] E. Gray. 2008. The Architecture of an RBridge Solution to TRILL. Internet-Draft, work in progress.

[53] E. Lear. 2002. What's In A Name: Report from the Name Space Research Group. Internet-Draft, work in progress.

[54] C. Ellison. 1999. The nature of a usable PKI. Computer Networks 31, pages 823–830.

[55] J. Eriksson, M. Faloutsos, and S. Krishnamurthy. 2003. PeerNet: Pushing Peer-to-Peer Down the Stack. In: Proc. of the Peer-to-Peer Systems II, Second International Works hop (IPTPS'03), pages 268–277. Berkeley, CA, USA.

[56] F. Johansson. 2004. The Medici Effect. Harward Business School Press.

[57] K. Fall. 2003. A Delay-Tolerant Network Architecture for Challenged Internets. In: Proc. of the Sigcomm'03. Karlsruhe, Germany.

[58] A. Festag, H. Karl, and A. Wolisz. 2007. Investigation of Multicast-Based Mobility Support in All-IP Cellular Networks. Journal on Wireless Communications and Mobile Computing 7, pages 319–339.

[59] N. Fikouras, K. Kuladinithi, C. Görg, C. Bormann, and A. Timm-Giel. 2004. Multiple Access Interface Management and Flow Mobility. In: Proc. of the 13th IST Mobile and Wireless Communications Summit 2004. Lyon, France.

[60] B. Ford. 2004. Unmanaged Internet Protocol: taming the edge network management crisis. Computer Communication Review 34, no. 1, pages 93–98.

[61] P. Francis and R. Gummadi. 2001. IPNL: A NAT-Extended Internet Architecture. In: Proc. of the Sigcomm'01. San Diego, California, USA.

[62] P. Francis and R. Gummadi. 2001. IPNL: A NAT-extended Internet architecture. ACM SIGCOMM Computer Communication Review 31, no. 4, pages 69–80.

[63] G. R. Wright and W. R. Stevens. 1995. TCP/IP Illustrated, Volume 2, the Implementation. Addison-Wesley.

[64] P. B. Godfrey, S. Shenker, and I. Stoica. 2006. Minimizing Churn in Distributed Systems. In: Proc. of the Sigcomm'06. Pisa, Italy.

[65] A. Grilo, P. Estrela, and M. Nunes. 2001. Terminal Independent Mobility for IP (TIMIP). IEEE Communication Magazine 39, pages 34–41.

[66] S. Guha, Y. Takeda, and P. Francis. 2004. NUTSS: A SIP based Approach to UDP and TCP Network Connectivity. In: Proc. of the SIGCOMM'04 Workshops. Portland, Oregon, USA.

[67] R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. 2003. The Impact of DHT Routing Geometry on Resilience and Proximity. In: Proc. of the Sigcomm'03. Karlsruhe, Germany.

[68] H. Alvestrand. 2004. A Mission Statement for the IETF. RFC 3935.

[69] H. Krawczyk and T.J. Watson. 1996. SKEME: a versatile secure key exchange mechanism for Internet. In: Proc. of the Symposium on Network and Distributed System Security (SNDSS '96) . NY, USA.

[70] H. Petander and E. Perera and A. Seneviratne and Y. Ismailov. 2007. An Experimental Evaluation of Mobile Node based versus Infrastructure based Handoff Schemes. In: In Proc. of the World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007). Helsinki, Finland.

[71] H. Soliman. 2007. Mobile IPv6 support for dual stack Hosts and Routers (DSMIPv6). Internet-Draft, work in progress.

[72] H. Soliman and C. Castelluccia and K. El Malki and L. Bellier. 2005. Hierarchical Mobile IPv6 Mobility Management (HMIPv6). RFC 4140.

[73] H. Soliman and N. Montavont and N. Fikouras and K. Kuladinithi. 2007. Flow Bindings in Mobile IPv6 and Nemo Basic Support. Internet-Draft, work in progress.

[74] J. Hasan and T. N. Vijaykumar. 2005. Dynamic Pipelining: Making IP Lookup Truly Scalable. In: Proc. of the Sigcomm'05. Philadelphia, PA, USA.

[75] R. Hsieh and A. Seneviratne. 2003. A Comparison of Mechanisms for Improving Mobile IP Handoff Latency for End-to-End TCP. In: Proc. of the Mobicom'03. San Diego, California, USA.

[76] S-M. Huang, Q. Wu, Y-B. Lin, and C-H. Yeh. 2006. SIP mobility and IPv4/IPv6 dual-stack supports in 3G IP multimedia subsystem. Journal on Wireless Communications and Mobile Computing 6, pages 585–599.

[77] G. Huston. 2005. Architectural Approaches to Multi-homing for IPv6. RFC 4177.

[78] I. Castineyra and N. Chiappa and M. Steenstrup. 1996. The Nimrod Routing Architecture. RFC 1992.

[79] IEEE Computer Society. 2003. IEEE Std 802.11g-2003. IEEE Standard for Information Technology.

[80] IEEE Computer Society. 2005. IEEE Std 802.3-2005. IEEE Standard for Information Technology.

[81] Information Sciences Institute, University of Southern California. 1980. DOD STANDARD, INTERNET PROTOCOL. RFC 760.

[82] Y. Ismailov, J. Holler, S. Herborn, and A. Seneviratne. 2006. Internet Mobility: An Approach to Mobile End-System Design. In: Proc. of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL'06).

[83] A. Iyer and C. Rosenberg. 2006. Understanding the key performance issues with MAC protocols for multi-hop wireless networks. Journal on Wireless Communications and Mobile Computing 6, pages 745–760.

[84] J. Abbate. 1999. Inventing the Internet. Electronic Book, The MIT Press.

[85] J. Arkko and C. Vogt and W. Haddad. 2007. Enhanced Route Optimization for Mobile IPv6. RFC 4866.

[86] J. Arkko and J. Kempf and B. Zill and P. Nikander. 2005. SEcure Neighbor Discovery (SEND). RFC 3971.

[87] J. Arkko and P. Eronen and H. Tschofenig and S. Heikkinen and A. Prasad. 2006. Quick NAP – Secure and Efficient Network Access Protocol. In: Proc. of the The 6th International Workshop on Applications and Services in Wireless Networks (ASWN 2006), pages 163–170. Berlin, Germany.

[88] J. Arkko and T. Aura and J. Kempf and V-M. Mäntylä, P. Nikander and M. Roe. 2002. Securing IPv6 neighbor discovery and router discovery. In: In Proc. of the 2002 ACM Workshop on Wireless Security (WiSe), pages 77–86. Atlanta, GA USA.

[89] J. Day. 2007. Patterns in Network Architecture. A Return to Fundamentals. Prentice Hall.

[90] J. Laganier and L. Eggert. 2008. Host Identity Protocol (HIP) Rendezvous Extension. RFC 5204.

[91] J. Molsa. 2006. Mitigating Denial of Service Attacks in Computer Networks. Doctoral Dissertation, Helsinki University of Technology.

[92] J. Postel. 1981. INTERNET CONTROL MESSAGE PROTOCOL. RFC 792.

[93] J. Postel. 1981. Transmission Control Protocol. RFC 793.

[94] J. Rosenberg and R. Mahy and P. Matthews. 2008. Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN). Internet-Draft, work in progress.

[95] J. Saltzer. 1978. Naming and Binding of Objects. Operating Systems, Lecture notes in Computer Science, Vol. 60, Edited by R. Bayer, New York, Springer-Verlag.

[96] J. Saltzer. 1993. On the Naming and Binding of Network Destinations. RFC 1498.

[97] J. Wroclawski. 1997. The Metanet: White Paper. In: Proc. of the Workshop on Research Directions for the Next Generation Internet. Vienna, VA. URL `http://www.cra.org/Policy/NGI/papers/wroklawWP`.

[98] J. Ylitalo and V. Torvinen and E. Nordmark. 2004. Weak Identifier Multihoming Protocol (WIMP). Internet-Draft, work in progress.

[99] J. P. Jeong, K. Lee, J. Park, and H. Kim. 2004. ND-Proxy based Route and DNS Optimizations for Mobile Nodes in Mobil e Network. Internet-Draft, work in progress.

[100] JH. Choi and G. Daley. 2005. Goals of Detecting Network Attachment in IPv6. RFC 4135.

[101] J.N. Chiappa. 1999. Endpoints and Endpoint Names: A Proposed Enhancement to the Internet Architecture. unpublished note. URL `http://ana.lcs.mit.edu/~jnc/tech/endpoints.txt`.

[102] K. Beck. 1999. Extreme Programming Explained: Embrace Change. Addison Wesley Professional.

[103] K. Egevang and P. Francis. 1994. The IP Network Address Translator (NAT). RFC 1631.

[104] K. Hiltunen. 2006. Using RF Repeaters to Improve WCDMA HSDPA Coverage and Capacity Inside Buildings. In: Proc. of the 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2006). Helsinki, Finland.

[105] H. Kang, K. Kim, S. Han, K-J. Lee, and J-S. Park. 2003. Route Optimization for Mobile Network by Using Bi-directional Betwee n Home Agent and Top Level Mobile Router. Internet-Draft, work in progress.

[106] T. Karagiannis, D. Papagiannaki, and M. Faloutsos. 2005. BLINC: Multilevel Traffic Classification in the Dark. In: Proc. of the Sigcomm'05. Philadelphia, PA, USA.

[107] A. Keromytis, V. Misra, and D. Rubenstein. 2002. SOS: Secure Overlay Services. In: Proc. of the Sigcomm'02. Pittsburgh, PA, USA.

[108] J.I. Khan and A. Wierzbicki. 2008. Foundation of peer-to-peer computing. Computer Communications 31, pages 187–189.

[109] T. Koponen, M. Chawla, B-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. 2007. A Data-Oriented (and Beyond) Network Architecture. In: Proc. of the Sigcomm'07. Kyoto, Japan.

[110] L. Lamport. 1981. Password authentication with insecure communication. Communication Magazine of ACM 24, pages 770–772.

[111] L. Lessig. 1999. Code and Other Laws of Cyberspace. Basic Books.

[112] B. Leiner, V. Cerf, D. Clark, R. Kahn, L. Kleinrock, D. Lynch, J. Postel, L. Roberts, and S. Wolff. 1997. A Brief History of the Internet, Part I. Electronic Journal: e-OTI.

[113] L. Li, D. Alderson, W. Willinger, and J. Doyle. 2004. A First-Principles Approach to Understanding the Internet's Router-level Topology. In: Proc. of the Sigcomm'04. Portlan, Oregon, USA.

[114] M. Blaze J. Feigenbaum J. Ioannidis A. Keromytis . 1999. The KeyNote Trust-Management System Version 2. RFC 2704.

[115] M. Kunishi and M. Ishiyama and K. Uehara and H. Esaki and F. Teraoka. 2000. LIN6: A New Approach to Mobility Support in IPv6. In: Proc. of the International Symposium on Wireless Personal Multimedia Communication. Thailand.

[116] L. Ma, F. Yu, and V.C.M. Leung. 2007. Performance Improvements of Mobile SCTP in Integrated Heterogeneous Wireless Networks. IEEE Trans. Wireless Communications .

[117] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjalmtysson, and A. Greenberg. 2004. Routing Design in Operational Networks: A Look from the Inside. In: Proc. of the Sigcomm'04. Portlan, Oregon, USA.

[118] J. Manner and M. Kojo. 2004. Mobility Related Terminology. RFC 3753.

[119] A. Matos, J. Santos, J. Girao, , M. Liebsch, S. Sargent, and R. Aguiar. 2006. HIP Location Privacy Framework. In: Proc. of the MobiArch'06. San Francisco, California, USA.

[120] N. Montavont and T. Noel. 2006. Fast movement detection in IEEE 802.11 networks. Journal on Wireless Communications and Mobile Computing 6, pages 651–671.

[121] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. 2008. Host Identity Protocol. RFC 5201.

[122] A. Nakao, L. Peterson, and A. Bavier. 2003. A Routing Underlay for Overlay Networks. In: Proc. of the Sigcomm'03. Karlsruhe, Germany.

[123] P. Nikander. 2001. "Denial-of-Service, Address Ownership, and Early Authentication in the IPv6 World. In: Proc. of the Security Protocols, 9th International Workshop. Cambridge, UK. LNCS 2467, pp. 12-26, Springer 2002.

[124] P. Nikander. 2001. From Address Orientation to Host Orientation. In: Proc. of the IP-Based Cellular Networks 2001. Paris, France.

[125] P. Nikander and J. Arkko. 2003. Delegation of Signalling Rights. In: Proc. of the 10th International Security Workshop, pages 203–212. Cambridge, UK.

[126] P. Nikander, J. Arkko, and B. Ohlman. 2004. Host Indentity Indirection Infrastructure (Hi3). In: Proc. of the Second Swedish National Computer Networking Workshop (SNCNW'04). Karlstad, Sweden.

[127] S. Nováczki, L. Bokor, G. Jeney, and S. Imre. 2008. Design and Evaluation of a Novel HIP-Based Network Mobility Protocol . JOURNAL OF NETWORKS 3, no. 1.

[128] H. Ohnishi, K. Sakitani, and Y. Takagi. 2003. HMIP based Route optimization method in a mobile network. Internet-Draft, work in progress.

[129] P. Ferguson and H. Berkowitz. 1997. Network Renumbering Overview: Why would I want it and what is it anyway? RFC 2071.

[130] P. Jokela and J. Melen and J. Ylitalo. 2006. HIP Service Discovery. Internet-Draft, work in progress.

[131] P. Nikander. 1999. An Architecture for Authorization and Delegation in Distributed Object-Oriented Agent Systems. Doctoral Dissertation, Helsinki University of Technology.

[132] P. Nikander and J. Laganier. 2008. Host Identity Protocol (HIP) Domain Name System (DNS) Extensions. RFC 5205.

[133] P. Nikander and J. Laganier and F. Dupont. 2007. An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers (ORCHID). RFC 4843.

[134] P. Nikander and J. Melen. 2006. A Bound End-to-End Tunnel (BEET) mode for ESP. Internet-Draft, work in progress.

[135] P. Nikander and T. Henderson and C. Vogt and J. Arkko. 2008. End-Host Mobility and Multihoming with the Host Identity Protocol. RFC 5206.

[136] P. Srisuresh and K. Egevang. 2001. Traditional IP Network Address Translator (Traditional NAT). RFC 3022.

[137] P. Thubert and C. Bontoux and N. Montavont. 2006. Nested Nemo Tree Discovery. Internet-Draft, work in progress.

[138] P. Vixie and S. Thomson and Y. Rekhter and J. Bound. 1997. Dynamic Updates in the Domain Name System (DNS UPDATE). RFC 2136.

[139] T. Pagtzis, J. Crowcroft, and K. Clark. 2006. On the performance of proactive mobile IPv6 for context-aware all-IP wireless access networks. Journal on Wireless Communications and Mobile Computing 6, pages 559–583.

[140] N. Passas, S. Paskalis, A. Kaloxylos, F. Bader, R. Narcisi, E. Tsontsis, A. S. Jahan, H. Aghvami, M. O'Droma, and I. Ganchev. 2006. Enabling technologies for the 'always best connected' concept. Journal on Wireless Communications and Mobile Computing 6, pages 523–540.

[141] H. Petander, E. Perera, K. Lan, and A. Seneviratne. 2006. Measuring and Improving Performance of Network Mobility Management i n IPv6 Networks. In: IEEE Journal on Selected Areas of Communications, Special Issue on Mobile Routers and Network Mobility.

[142] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. 2003. On Selfish Routing in Internet-Like Environments. In: Proc. of the Sigcomm'03. Karlsruhe, Germany.

[143] R. Droms. 1997. Dynamic Host Configuration Protocol. RFC 2131.

[144] R. Jaksa and C. Williams and B. Sarikaya. 2006. Method to specify optimal specification of service ports for flow bindings. Internet-Draft, work in progress.

[145] R. Koodli. 2005. Fast Handovers for Mobile IPv6. RFC 4068.

[146] R. Moskowitz and P. Nikander. 2006. Host Identity Protocol (HIP) Architecture. RFC 4423.

[147] R. Pastor-Satorras and A. Vespignani. 2004. Evolution and Structure of the Internet: A Statistical Physics Approach. Electronic Book, Cambridge University Press, ISBN 052182698.

[148] R. Shirey. 2007. Internet Security Glossary, Version 2. RFC 4949.

[149] R. Stewart and Q. Xie and K. Morneault and C. Sharp and H. Schwarzbauer and T. Taylor and I. Rytina and M. Kalla and L. Zhang and V. Paxson. 2000. Stream Control Transmission Protocol. RFC 2960.

[150] B. Raghavan and A. C. Snoeren. 2004. A System for Authenticated Policy-Compliant Routing. In: Proc. of the Sigcomm'04. Portlan, Oregon, USA.

[151] V. Ramasubramanian and E. G. Sirer. 2004. The Design and Implementation of a Next Generation Name Service for the Internet. In: Proc. of the Sigcomm'04. Portlan, Oregon, USA.

[152] R. Ramjee, L. Li, T. L. Porta, and S. Kasera. 2001. IP Paging Service for Mobile Hosts. In: Proc. of the Mobicom'01. Rome, Italy.

[153] R. Ramjee, T. Porta, L. Salgarelli, S. Thuel, and K. Varadhan. 2000. IP-based Access Network Infrastructure for next Generation Wireless Data Networks. IEEE Personal Communication Magazine 7, pages 34–41.

[154] A. Rao and I. Stoica. 2005. An overlay MAC layer for 802.11 networks. In: Proc. of the 3rd international conference on Mobile systems, applications, and services (Mobisys'05). Seattle, Washington, USA.

[155] S. Ratnasamy, S. Shenker, and S. McCanne. 2005. Towards an Evolvable Internet Architecture. In: Proc. of the Sigcomm'05. Philadelphia, PA, USA.

[156] A. A. Reaz, P. Chowdhury, M. Atiquzzaman, and W. Ivancic. 2006. Signalling Cost Analysis of SINEMO: Seamless End-to-End Network Mobility. In: Proc. of the MobiArch'06. San Francisco, California, USA.

[157] M.G. Reed, P.F. Syverson, and D.M. Goldschlag. 1998. Anonymous Connections and Onion Routing. IEEE Journal on Selected Areas in Communication Special Issue on Copyright and Privacy Protection 16, pages 482–494.

[158] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu. 2005. OpenDHT: A Public DHT Service and Its Uses. In: Proc. of the Sigcomm'05. Philadelphia, PA, USA.

[159] P. Roberts and J. Kemph. 2006. Mobility Architecture for the Global Internet. In: Proc. of the MobiArch'06. San Francisco, California, USA.

[160] T. L. Rodeheffer, C. A. Thekkath, and D. Anderson. 2000. SmartBridge: A Scalable Bridge Architecture. In: Proc. of the Sigcomm 2000. Stockholm, Sweden.

[161] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy. 2003. STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). RFC 3489.

[162] S. Bellowin. 1998. EIDs, IPsec and HostNAT. a presentation give at 41st IETF in Los Angeles, California. URL `http://www.cs.columbia.edu/~smb/talks/hostnat.pdf`.

[163] S. Deering and R. Hinden. 1998. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460.

[164] S. Gundavelli and K. Leung and V. Devarapalli and K. Chowdhury and B. Patil. 2008. Proxy Mobile IPv6. Internet-Draft, work in progress.

[165] S. Kent and R. Atkinson. 1998. IP Encapsulating Security Payload (ESP). RFC 2406.

[166] S. Kent and R. Atkinson. 1998. Security Architecture for the Internet Protocol. RFC 2401.

[167] S. Krishnan and N. Montavont and E. Njedjou and S. Veerepalli and A. Yegin. 2007. Link-Layer Event Notifications for Detecting Network Attachments. RFC 4957.

[168] S. O'Malley and L. Peterson. 1992. A Dynamic Network Architecture. In: Proc. of the ACM Transactions on Computer Systems, vol. 10, no. 2, pages 110–143.

[169] S. Pierrel and P. Jokela and J. Melen. 2006. Simultaneous Multi-Access extension to the Host Identity Protocol. Internet-Draft, work in progress.

[170] U. Saif and J. M. Paluska. 2005. Service-Oriented Network Sockets . In: Proc. of the 3rd international conference on Mobile systems, applications, and services (Mobisys'05). Seattle, Washington, USA.

[171] J. Saltzer, D.P. Reed, and D.D. Clark. 1981. End-to-End Arguments in System Design. In: Proc. of the Second International Conference on Distributed Computing Systems (April 1981) pages 509-512. Published with minor changes in ACM Transactions in Computer Systems 2, 4, November 1984, pages 277-288. Reprinted in Craig Partridge, editor Innovations in internetworking. Artech House, Norwood, MA, 1988, pages 195-206. ISBN 0-89006-337-0. Also scheduled to be reprinted in Amit Bhargava, editor. Integrated broadband networks. Artech House, Boston, 1991. ISBN 0-89006-483-0. Paris, France.

[172] S. Schmid, L. Eggert, M. Brunner, and J. Quittek. 2004. TurfNet: An Architecture for Dynamically Composable Networks. In: Proc. of the First IFIP International Workshop on Autonomic Communication (WAC 2004). Berlin, Germany.

[173] A. C. Snoeren and H. Balakrishnan. 2000. An End-to-End Approach to Host Mobility. In: Proc. of the Mobicom 2000. Boston, Massachusetts, USA.

[174] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. 2002. Internet Indirection Infrastructure. In: Proc. of the Sigcomm'02, pages 73–88. Pittsburgh, PA, USA.

[175] T. Aura. 2005. Cryptographically Generated Addresses (CGA). RFC 3972.

[176] T. Aura and M. Roe and J. Arkko. 2002. Security of Internet Location Management. In: In Proc. of the 18th Annual Computer Security Applications Conference (ACSAC), pages 78–87. Las Vegas, NV USA.

[177] T. Heer. 2006. LHIP: Lightweight Authentication for the Host Identity Protocol (HIP). Diploma thesis, University of Tubingen.

[178] T. Kauppinen and H. Mahkonen and M. Kuparinen and C. Larsson and H. Levkowetz. 2006. Filter Interface Identifier Binding in Mobile IPv6. Internet-Draft, work in progress.

[179] T. Koponen and A. Gurtov and P. Nikander. 2005. Application Mobility with HIP. In: Proc. of the 12th International Conference on Telecommunications (ICT'05). Cape Town, South Africa.

[180] T. Narten and E. Nordmark and W. Simpson. 1998. Neighbor Discovery for IP Version 6 (IPv6). RFC 2461.

[181] P. Thubert and M. Molteni. 2007. IPv6 Reverse Routing Header and its application to Mobile Networks. Internet-Draft, work in progress.

[182] J. D. Touch and V. K. Pingali. 2003. DataRouter: A Network-Layer Service for Application-Layer Forwar ding. In: Proc. of the International Workshop on Active Networks (IWAN'03) . Kyoto, Japan.

[183] V. Fuller and T. Li and J. Yu and K. Varadhan. 1993. Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. RFC 1519.

[184] L. Subramanian V. N. Padmanabhan. 2001. An Investigation of Geographic Mapping Techniques for Internet Hosts. In: Proc. of the Sigcomm'01. San Diego, California, USA.

[185] W. R. Stevens. 1994. TCP/IP Illustrated, Volume 1, the Protocols. Addison-Wesley.

[186] N. Wakamiya and M. Murata. 2005. Toward overlay network symbiosis. In: Proc. of the Fifth International Peer-to-Peer Conference (P2P2005). Konstanz, Germany.

[187] R. Wakikawa, S. Koshiba, K. Uehara, and J. Murai. 2003. ORC: Optimized Route Cache Management Protocol for Network Mobility. In: Proc. of the 10th International Conference on Telecommunication s (ICT 2003), pages 1194–1200. Papeete, French Polynesia.

[188] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. M orris, and S. Shenker. 2004. Middleboxes No Longer Considered Harmful. In: Proc. of the USENIX OSDI. San Francisco, CA, USA.

[189] M. Walfish, M. Vutukuru, H. Balakrishnan, D. Karger, and S. Shenker. 2006. DDoS Defense by Offense. In: Proc. of the Sigcomm'06. Pisa, Italy.

[190] Joel M. Winett. 1971. The Definition of a Socket. RFC 147.

[191] B. Wong, A. Slivkins, and E. G. Sire. 2005. Meridian: A Lightweight Network Location Service without Virtual Coordinates. In: Proc. of the Sigcomm'05. Philadelphia, PA, USA.

[192] W.R. Stevens. 1998. Unix Network Programming, Networking APIs: Sockets and XTI, Volume 1 (second edition). PTR PH.

[193] W.R. Stevens. 1999. Unix Network Programming, Interprocess Communications, Volume 2 (second edition). PTR PH.

[194] Y. Rekhter and B. Moskowitz and D. Karrenberg and G. J. de Groot and E. Lear. 1996. Address Allocation for Private Internets. RFC 1918.

[195] Y-S. Yen, C-C. Hsu, and H-C. Chao. 2006. Global dynamic home agent discovery on mobile IPv6. Journal on Wireless Communications and Mobile Computing 6, pages 617–628.

[196] J. Ylitalo. 2005. Re-thinking Security in Network Mobility. In: Proc. of the NDSS'05 Wireless and Mobile Security Workshop (Extended abstract). San Diego, CA, USA.

[197] V. C. Zandy and B. P. Miller. 2002. Reliable Network Connections. In: Proc. of the Mobicom'02. Atlanta, Georgia, USA.

[198] S. Q. Zhuang, K. Lai, I. Stoica, R. H. Katz, and S. Shenker. 2003. Host Mobility using an Internet Indirection Infrastructure. In: Proc. of the First International Conference on Mobile Systems, Applications, and Services (Mobisys'03).

Appendix A State Establishment at SPINAT Nodes

This Appendix describes the locator translation functionality of a SPINAT node in more detail than what is presented in the publications. The purpose of the Appendix is to keep the discussion consistent and to enlarge the viewpoint also to cover the implementation of the SPINAT. The author and his colleagues have publicly demonstrated the implementation of the SPINAT address-family agility functionality at amongst others, IST'05 [20] , Mocca'05 [21] , ECIW'06, [22] and PIMRC'06 [23] conferences. In section A.1, a state at the SPINAT node is established using the end-to-end state establishment exchange. Section A.2 describes the state establishment from the state update exchange viewpoint. Section A.3 presents the rendezvous functionality of the SPINAT nodes.

A.1 Using End-to-end State Establishment Exchange

The following discussion is based on a scenario where a mobile host inside a private IPv4 locator domain communicates with a peer host having only an IPv6 attachment to the Internet. The on-the-path SPINAT node transparently establishes a state during the initial end-to-end mobility state establishment signalling. The scenario is illustrated in Figure A.1. The example can also be applied vice versa with private IPv6 locator domains. The implementation uses site local IPv6 locators for private IPv6 locator domains. A scenario where the peer host is also located in a private locator domain is discussed later in section A.3.

In the approach taken in this thesis, the peer host has initially registered to a rendezvous server that is located in the Internet. The peer host has leased an IPv4 locator from the server to be reachable from the IPv4 networks. Initially, the peer

[20]Oy L M Ericsson Ab, The first public demo of the SPINAT and address agility implementation, Demonstrated at the 14th IST Mobile & Wireless Communications Summit, Dresden, Germany, June 2005.

[21]Oy L M Ericsson Ab, The second public demo of the SPINAT and address agility implementation, Demonstrated at Mocca WWI Symposium, Paris, France, December 2005.

[22]Oy L M Ericsson Ab, The first public demo of the HIP Mobile Router and address agility implementation, Demonstrated at the 5th European Conference on Information Warfare and Security (ECIW'06), Helsinki, Finland, June 2006.

[23]Oy L M Ericsson Ab, The second public demo of the HIP Mobile Router and address agility implementation (MERCoNe Project), Demonstrated at the 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'06), Helsinki, Finland, September 2006.

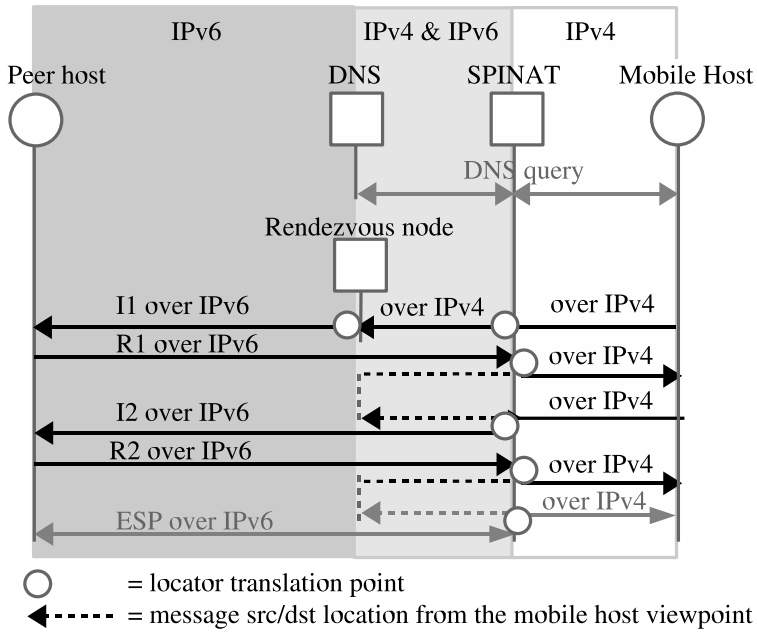

Figure A.1: Using State Establishment Exchange to Initialize Locator Translation between the private domain and the Internet.

host is located in the Internet. However, once the peer host moves into a private network, the SPINAT node between the private network and the Internet also works as a rendezvous point for the peer host (discussed later). Furthermore, the mobile host is attached to a private IPv4 locator domain behind a SPINAT node. The rendezvous server is reachable both from the peer's and the mobile host's location. The peer host has stored its leased IPv4 locator to the DNS together with its HI.

The mobile host receives the leased IPv4 locator of the peer host from the DNS, and sends the I1 message to that location. The on-the-path SPINAT node establishes a soft state for the HITs, before forwarding the I1 message to the rendezvous server. It also translates the private IPv4 source locator to the public multiplexed IPv4 locator of the SPINAT node. The SPINAT node adds its self-signed public locator set information to the end of the message. The locator set should contain at least a locator per supported address-family. As earlier mentioned, the author assumes that the Internet provides both global IPv4 and IPv6 connectivity. In addition, the SPINAT node adds an encrypted 'echo request' parameter to the I1 message that is echoed back by the peer host in R1 message.

In the presented scenario, the mobile host and the SPINAT node do not explicitly establish a security association. Therefore, the approach is vulnerable for the following attack. An attacker on the forwarding path may replace the SPINAT node's self-signed locator set with its own. Typically, the peer host does not have prior knowledge of the SPINAT node's identity. However, it is good to notice that an attacker is also able to achieve the same effect by changing the source locator of the IP header in the unsigned I1 message. Thus, adding a self-signed locator set to the I1 message does not change the security level of the protocol. [24]

When the I1 message arrives to the rendezvous server, the server replaces the IPv4 header with an IPv6 header. The destination IPv6 address is set to the current location of the peer host. The source locator is translated to one of the IPv6 locators of the rendezvous server. The server adds also the received IPv4 source locator (that is bound to the SPINAT node) to the I1 message before forwarding it. It is good to notice that the peer host has an earlier established security association with the rendezvous server. Therefore, the rendezvous server is able to protect the added locator parameter using HMAC.

After receiving the I1 message, the peer host replies with the R1 message. It also verifies the signed locator set if the packet contains authorization information, i.e. a certificate. The peer host adds its locator set to the R1 message. The locator set is learned later by the SPINAT node and the mobile host. Instead of sending the packet back to the rendezvous server, the peer host uses one of the received SPINAT node's locators. The packet is sent directly, i.e. using optimized routing, to one of the SPINAT node's IPv6 locators. The output of the LPM algorithm is a global IPv6 locator pair between the peer and the SPINAT node.

Once the SPINAT node receives the R1 message it marks the initial IPv4 locator as verified. The peer host was reachable behind the initial IPv4 locator. The SPINAT node also verifies that the received 'echo reply' parameter contains the same secret that was included in the I1 message. As mentioned above, the SPINAT node stores the received locator set of the peer host. Furthermore, the SPINAT node translates

[24]The peer host stays stateless and sends the R1 message back to the locator received in the I1 message. Thus, the on-the-path attacker can cause a DoS situation only for the initiator. However, according to the HIP specification the retransmitted I1 messages should be sent to alternative locations if possible. Therefore, the peer host should be reachable in multiple, topologically scattered, rendezvous locations. The SPINAT node can be authorized to send location updates on behalf of the mobile host (see section 6.4). From the peer host point view, the authorization can be used to verify the added SPINAT node's locator set.

both the source and the destination IPv6 locators to IPv4 locators. The idea of the SPINAT node is similar to that of the NAT devices. The mobile host inside the private network does not know the private locators of the SPINAT node. For security reasons, the SPINAT node does not use its private IPv4 locators but sets the source locator to point to the IPv4 rendezvous location of the peer host. In other words, the IPv4 source locator is translated to the same locator as to where the I1 message was sent, and the destination locator is translated to the private IPv4 locator of the mobile host.

After receiving the R1 message, the mobile host learns the peer host's locators carried in the message. The base exchange continues in a normal way from the mobile host's point of view. The I2 message is sent back to the IPv4 rendezvous locator of the peer host. However, the SPINAT node does not forward the packet to the rendezvous server but runs the LPM algorithm using its public locator set and the earlier stored peer host's locator set. The output of the algorithm is most probably an IPv6 locator pair, because the locators contain longer prefix and thus in several cases more common bits. Thus, the SPINAT node translates the IPv4 header to an IPv6 header containing the source IPv6 locator of the SPINAT node and the destination IPv6 locator of the peer host. It is good to notice that only the first I1 message was routed via the rendezvous server and the rest of the messages are sent directly between the SPINAT node and the peer host.

Finally, after receiving the I2 message the peer host creates a state for the mobile host. From the peer host's point of view, the mobile host is reachable at the SPINAT node's locator carried in the I2 message's IP header. The peer host flips the locators in the header and replies with an R2 message. At this point, the SPINAT node marks the destination locator of the earlier sent I2 message as verified. The source locator of the R2 message and the destination locator of the earlier sent I2 message may differ from each other. It is possible that an on-the-path attacker has changed the source locator of the R2 message. The only thing the SPINAT node knows is that the peer host was reachable from the destination location where the I2 message was sent. Before forwarding the R2 message to mobile host, the SPINAT node implements the same address translation that took place with the R1 message. The soft state is also changed to normal state and BEETIN SAs are created at the SPINAT node (discussed earlier in section 5.3.5).

It is good to notice that the previous scenario is vulnerable to a redirection attack

when the peer host directly uses the source locator in the I2 message header. The attack is possible in the basic HIP exchange, and it is not caused by the SPINAT approach. An on-the-path attacker can change the source locator of the I2 message before forwarding the packet to the peer host. The peer host sends the R2 reply message back to that altered location, where another on-the-path attacker waits for the packet. The attacker translates the destination locator to the same locator as to where the SPINAT node sent the original I2 message from. Furthermore, the attackers forward the first of the ESP payload packets in both directions using the triangular route. From the peer host point of view the R2 message works as a challenge message, and the first incoming ESP packet verifies the destination locator.

Finally, the second attacker moves away from the victim's network where all the peer host's traffic ends up. The attack requires that the attacker is located simultaneously in two locations. To prevent this kind of attack, the I2 message should contain the signed mobile host's and SPINAT node's locator sets. It is good to notice that the peer host receives the SPINAT node's public key in the I1 message that is routed via a different path from the rest of the messages.

A.2 Using End-to-end State Update Exchange

The previous section described how the state establishment works in a basic scenario where the hosts are located at different locator domains and the peer host is directly attached to the Internet. This section briefly discusses a scenario where the end-to-end mobility state update exchange is used to establish states at on-the-path SPINAT nodes. The scenario is illustrated in Figure A.2.

The mobile host runs the state establishment exchange with the peer host only once. After that the mobile host updates its location to the peer host by running an update exchange. From the SPINAT node viewpoint, the intermediate state establishment using the update exchange differs from the state establishment exchange case. The update exchange is only a three way hand shake, while the state establishment exchange consists of four messages. However, we can assume that the mobile host has learned the peer host's current locator set during the state establishment exchange. This observation makes it a bit easier to implement a reachability test towards the peer host from the SPINAT node's point of view. It is good to notice that while the end-to-end update exchange is used for updating a state at the end host, it is used

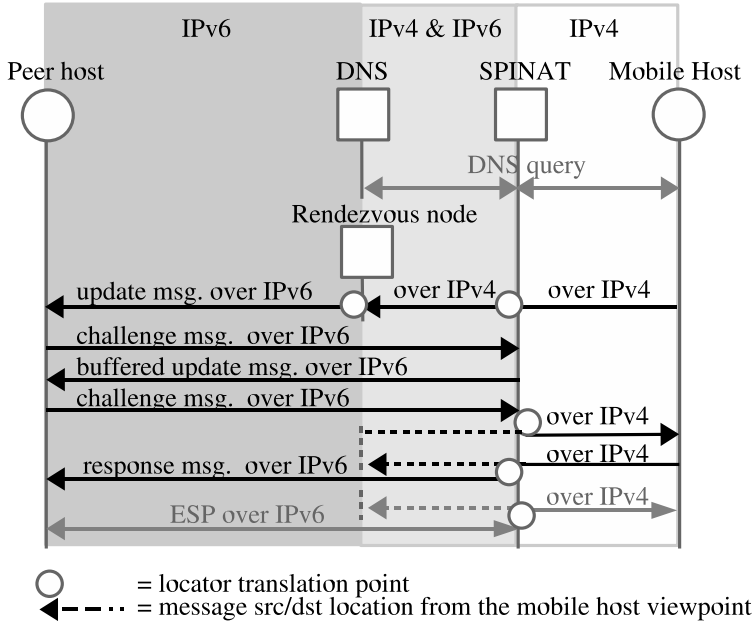

Figure A.2: Using Update Exchange to Initialize Locator Translation between the private domain and the Internet.

for creating states at the SPINAT nodes.

The mobile host should sign the peer's locator set with its own public key. The signed peer host's locator set and the mobile host's public key are added to the first update message. Once the SPINAT node receives the update message, it creates a soft state and verifies the signature. The SPINAT node selects a locator pair of the peer's and its own locator sets based on the LPM algorithm. The SPINAT node also adds its own self-signed locator set to the update message before forwarding the message to the peer host. The earlier presented I1 message's security considerations also apply to this update message. In addition, the SPINAT node buffers the first update message to be able to retransmit the message later.

The peer host replies with the challenge message and sends it directly to the SPINAT node. The peer host should use one of the locators of the locator set included in the first update message. The SPINAT node verifies that the source locator in the challenge message sent by the peer host belongs to the locator set that was carried in the first update message. In addition, the SPINAT node compares locators in the buffered message and the received challenge message. If the destination locator in

the buffered packet and the source locator in the challenge packet differ from each other the packet has been routed via a rendezvous node. In that case, the SPINAT node cannot directly start using the received source locator with the peer host. Instead, the SPINAT node drops the received challenge message and retransmits the buffered update message to the location where the challenge message was sent and adds 'echo request' parameter to the message. In other words, the SPINAT node uses the first round trip of the update exchange as a reachability test. This mechanism can be also applied with I1 messages.

If the peer host was reachable from its claimed location, the SPINAT node receives the challenge message from the peer host containing correct 'echo reply' parameter. The SPINAT node marks the peer host's location verified and forwards the packet to the mobile host. If the challenge message was spoofed, the mobile host drops the message, and the SPINAT node releases its soft state. Otherwise, the mobile host finalizes the update exchange by sending the response message to the peer host. The SPINAT node uses the already verified destination locator towards the peer host.

A.3 Transparent Rendezvous State establishment

The registration to the SPINAT node provides a way to implement overlay routing based on the host identifiers. Once the I1 (or update) message arrives to the multiplexed locator of the SPINAT node, the SPINAT node checks whether it has a rendezvous state for the corresponding destination HIT. If it finds a state, the SPINAT node performs the required locator translation. The locator translation between domains can be divided into two cases.

In the first case, the locators in both domains belong to the same locator family. The SPINAT node translates only the destination locator of the packet and forwards the packet to the peer host inside the private domain. The SPINAT node creates a soft state for the end-to-end HIP connection only after receiving the R1 reply message from the peer host.

In the second case, the SPINAT may receive an I1 (or update) packet with locators that belong to a different family from the private domain locators. As a result, the SPINAT node must translate both locators in the packet. The situation is illustrated in Figure A.3. In the legacy HIP case, this is a problem because the initial messages do not contain the reachability information of the sender. To overcome

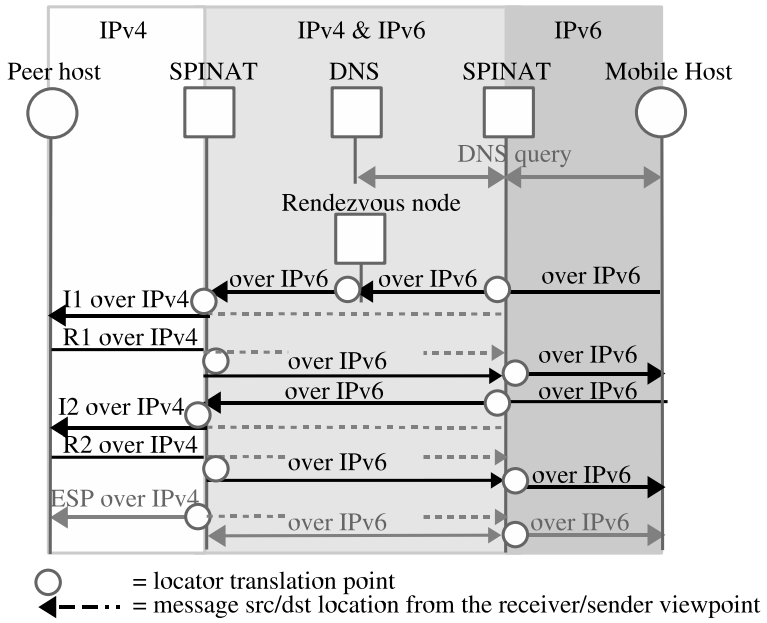

Figure A.3: An example of initializing locator translation between private domains.

the problem, the initiator should add its globally reachable locator set to the I1 and update messages. When the initiator is inside a private domain, it does not know the SPINAT node's public locator. Therefore, in the approach taken in this thesis the initiator adds the locators leased from the public rendezvous nodes to the I1 (or UPDATE) message. Before the message leaves the private domain on the initiator's side, the SPINAT node adds its own publicly reachable locator set to the I1 (or UPDATE) message as described earlier. The locator set should contain at least one locator per locator family where the initiator is reachable.

Once the SPINAT node on the responder's side receives the initial message, it can select one of the locators carried in the message to be used for the source locator translation if needed. (It is good to notice that the SPINAT node does not create a state and stores the locator set carried in the I1 message.) After receiving the I1 message, the peer host selects the destination locator for R1 message. The SPINAT node establishes a state for the end host pair, and forwards the R1 packet to the destination locator defined in the R1 message header. The destination locator points either to the initiator's SPINAT node or to one of its public rendezvous nodes.

Once the initiator receives the R1 message (possibly routed via a public rendezvous

node), it selects the destination locator for the I2 message. In this scenario, the R1 message contains two locator sets, one added by the responder and one by the responder's side SPINAT node. The initiator and initiator's side SPINAT node include their locator sets also to the I2 message because the responder's side SPINAT node does not establish a state with the I1 message. The I2 and R2 are routed directly between the initiator's and responder's SPINAT nodes.

The SPINAT nodes' rendezvous functionality provides a way for the hosts inside a private domain to be reachable also from the Internet. It is good to notice that the mobile host may dynamically change its location from one private network to another. As a result, the lifetime of the rendezvous state at the SPINAT node is the lifetime of the end-to-end security association. Moreover, the lifetime of the security associations should be at most the estimated stay at the host's current location. To achieve efficient garbage collection for dead security associations, the SPINAT nodes may require fast rekeying sequences. Thus, the SA lifetimes at end-nodes are bound to intermediate nodes' state lifetimes.

A.4 Locator Family Translation for ESP packets

The creation of BEETIN SAs was presented earlier in Section 5.3.5. This section presents a scenario where the mobile host is located in a private IPv4 domain behind a SPINAT node, and the peer host has a direct IPv6 attachment to the Internet (see Figure 5.1 on page 98).

The outgoing and incoming public side SAs are bound to IPv6 locators. The private SAs are bound to IPv4 locators. It is good to notice that the SAs are not bound to the SPINAT node's private locators. As earlier mentioned, the SPINAT node is transparent to the mobile host from the SA establishment point of view.

Once the mobile host sends the first ESP protected packet to the peer host, the SPINAT node intercepts the ESP packets. All the ESP packets having an SA are subject to IPsec processing. The incoming SA is identified using both the destination locator and the SPI value carried in the packet. Without the destination locator information, the incoming private side and outgoing public side SA would match with the SPI value.

For the basic case, the BEETIN mode skips the ESP packet authentication and

decryption. However, as mentioned in [P7], it is possible to send integrity and/or confidentiality protection keying material to the SPINAT node through a protected channel. This provides a way to implement lawful interception at SPINAT nodes. Such a channel may be established during a rendezvous state establishment exchange between the host and the SPINAT node.

The BEETIN mode works like the BEET mode from the identifier-locator translation viewpoint. In the example scenario, the IPv4 locators in the IP header are replaced between the peer's and mobile host's HITs. The difference with the two IPsec modes is that the BEET mode sends the authenticated and decrypted packet back to IPv6 input handling, while the BEETIN mode sends the ESP packet to the IPv6 output handling (see Figure 7.3 on page 150).

The HITs and SPI values in the outgoing ESP packet match with an IPsec policy that is bound to the outgoing public SA. The BEETIN mode translates the HITs in the IPv6 header to the IPv6 locators of the SA. The same mechanism takes place in the opposite direction when the peer host sends ESP traffic back to the mobile host. In practice, the BEETIN mode offers an overlay routing mechanism for ESP protected traffic [P7]. The identifier-locator mappings are stored to the IPsec policies and SAs.

ISBN 978-951-22-9530-2
ISBN 978-951-22-9531-9 (PDF)
ISSN 1795-2239
ISSN 1795-4584 (PDF)