P2

Publication 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).

END-POINT IDENTIFIERS IN SECURE MULTI-HOMED MOBILITY

Jukka Ylitalo, Petri Jokela, Jorma Wall, Pekka Nikander*

Abstract: *Currently IP addresses are used both for node identifiers and topological location names in the Internet. The semantic overloading and non-cryptographic nature of IP addresses makes it impossible to use them as identifiers from the security point of view. The problem becomes even worse with multi-homed mobile nodes. Multi-homed mobile nodes have several interfaces bound to dynamically changing IP addresses. When a node changes its point of attachment to the network or it reroutes traffic from one interface to another, the connection identifiers are changed. A peer node cannot verify the validity of the new identifiers without a naming trust relationship between the identifiers and the identity of the node. The peer must have evidence that an identifier belongs to a specific identity. Currently, there are no way for a node, using traditional IP addresses, to prove that it owns a specific address, i.e., an identifier. We present in this paper the philosophy behind separation of end-point identifiers from location names, which is an essential part in designing secure multi-homed mobility architectures.*

Keywords: *End-point identifier, location name, locator, identity, mobility, multi-homing, security*

1 Introduction

In the 70's, after the Internet Protocol (IP) was taken in use, there was motivation to develop a connection-oriented, reliable end-to-end protocol, later called TCP. However, at that time computers were connected to one network with one interface without any mobility support. Any computer could be easily identified with its single IP address. The location directly identified the node in the network.

Unfortunately, even today the design of many existing protocols has been based on that restricted approach. This has led to a situation where a connection can only be identified with an IP address at an end-point. Because of that, the existing general mobility support solutions in the IP world have tried to hide the dynamic change of IP addresses from the transport layer protocols. In practice, these solutions implement the so called host NAT idea [Bel98] in such a way that an actual topological location name is bound to a static identifier. Thus, used the static identifiers are typically also location names. However, it has been found out to be a difficult task to prevent misuse of those non-cryptographic static identifiers.

The main principles of the current node identification practice carry this historical load, i.e., the node identifiers are tightly bound to routing information. On the other hand, we can say that the most important step towards separation of node identifiers from location naming is already taking place. An example of this is the home address, care-of address (CoA) pair used in Mobile IP [Per96] [DBJA02]. The home address acts as the identifier of a node and the care-of-address is the name of the topological location. Furthermore, a node cannot in the truest sence "own" [Nik01b] a location name, because bits can be duplicated and they cannot be said to be the property of anyone. The only way to "own" something in the Internet is to keep it secret. Therefore, the node identification must be based on cryptographic methods. This cannot be done with traditional IP address based identification.

We have to find out a logical and sound method for separating verifiable end-point [Chi99] identifiers from unverifiable location names. Further, if we can dynamically bind an identifier to a group of location names we obtain multi-homed mobility properties.

2 Definitions

To make the discussion in this paper definite, we define the relevant terminology briefly:

- A logical communication *end-point* is a computer node (hardware+kernel) or an application.
- A *subject* is an end-point that initiates a request for a resource that some node owns, and utilizes the resource to complete some task.

*Ericsson Research Finland. Address: Oy L M Ericsson Ab, Telecom R&D, FIN-02420 Jorvas, Finland. Phone: +358 9 2991. Fax: +358 9 299 3535. E-mail:{jukka.ylitalo, petri.jokela, jorma.wall, pekka.nikander}@ericsson.fi

- An *object* is an end-point that offers a resource and response to a request initiated by a subject related to the resource.
- An *identity* is defined in section 4.
- An *identifier* identifies an identity. An identifier can be bound to an identity in an unconditional way (naming trust relationship).
- A *name* is bound indirectly to an identity. There is no naming trust relationship between a name and an identity.
- A *location* name defines the topological point of an end-point in the network.
- A *connection* is a communication link between two end-points.

3 Problem Description

The problem field is related to the identity and identifier paradox and semantic overloading of IP addresses. In existing Internet architecture, there are several names that are used to identify an identity, e.g., FQDN (Fully Qualified Domain Name) and IP address. Unfortunately, those names cannot be bound definitely to an identity. The names are not based on cryptography and secondly the identity is quite an elusive thing from the naming point of view. What is actually the identity, the target of the identification process?

Currently IP addresses are used both for end-point identification and location naming purposes. This has led to several security problems, including the address ownership problem [Nik01a]. In other words, how can a node prove it owns the IP address, and the IP address is an identifier of its identity?

Further, the dynamic binding between a multi-homed mobile node and IP addresses makes things more difficult. The name of the mobile node changes dynamically when node changes its topological location in the network. How does the peer node know that the packets are coming from the correct node? How can connections be identified when a node makes a hand-off between access networks?

We have to analyze the true nature of the identity to be able to define what actually is an identifier and how an identifier is related to location names.

4 Identity

Psychology, philosophy, theology, biochemistry and legal science have different kind of approach to personal identity. If we ask, “Who is Bob?”, we will get different answers depending on the branch of science. In an open environment, like in Internet, we are finally not interested in knowing who Bob actually is, but can we trust him, i.e. “Can we trust Bob?”. The problem is not any more philosophical or theological, but more or less mathematical, psychological, biometrical and legal. Actually, we do not trust the name (identifier) of the person, but the identity that consists of a group of factors. Analyzing these factors over some time interval, we can create a trust-profile for a person. Another possibility is that we indirectly trust a person (Bob) by using a third party’s announcement about the person.

To establish a trust relationship, we must be able to definitely identify an identity (identification), and also be able to analyse the behavior of the identity to authorize it to use some service(s). The identification can be based on something known, something possessed or something embodied (biometric). The authorization, in turn, is based on factors related to trust for the identity. Traditionally, the electric identity (in smart cards) has been directly connected to a natural person. Therefore, psychological methods have been adequate tools in examining a person’s identity and trustworthiness. However, this is not a suitable method when we extend the term “identity” to comprise also computer and software agent identities.

To form a trust-profile of a subject’s identity in the network on account of its behavior is a demanding task. Thus, in the silicon world, the only possible way to define the data subject’s identity factors is to use quantitative parameters. Therefore, the nature of the natural identity in the network changes.

4.1 Natural Identity

We will discuss, in short, the identity from a psychological point of view. According to Sam Vaknin [Vak], Person’s *self-identity* is “different at different stages of [a person’s] life (Erikson) and it constantly evolves in accordance with his innate nature (Jung), past history (Adler), drives (Freud), cultural milieu (Horney), upbringing (Klein, Winnicott), needs (Murray), or the interplay with his genetic makeup”. Vaknin continues that we often confuse *habits* with identity, but habits are not part of a person’s identity in the truest sense. The removal of some habit does not change the person’s identity or the answer to the question “Who is Bob?”. It is our *personality* that allows us to adapt to

dynamically changing situations, like loosing a habit. If a personality is unable to adapt to changing circumstances, it is in a disordered state, also called *personality disorder*. We will analyse later, in section 4.3, the common features in a person's natural and electric identity, and how a personality disorder relates to electric identity.

4.2 Legality

From the legal point of view it is important to bind actions in the Internet made by a subject to its identity. Typically a natural person is identified with his name or some identification number. However, the new EU directive, on the protection of individuals with regard to the processing of personal data [Par95], takes into account also factors related to person's identity.

"*'Personal data' shall mean any information relating to an identified or identifiable natural person ('data subject'); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity.*"

There have been considered, in the directive, that a data subject may have several sub-identities at the same time. However, data subject's self-identity inherits all the factors of the sub-identities. Self-identity is considered to define a data subject in a unique way, in other words, a data subject is the same as its self-identity. We can say that a group of persons may have the same cultural identity (group-identity), but each of them has own mental identity. Thus, two persons cannot have exactly the same self-identity or they are called duplicates of each other. The current security architectures are based on this assumption. Even, twins share identical physical identity they finally have different self-identities and are therefore separate data subjects.

The presented EU directive is a corner stone in legislation in all union countries. The identity, in the directive, is bound directly to a natural person. However, the binding between personal data and electronical identity is not a straight issue if we allow own identities, e.g., for intelligent software agents. This raises a question "Who owns the identity?". Is it the software agent itself or the person who implemented the agent or the person on behalf of whom the agent is working? Can we call a software agent a data subject from the legal point of view if it has its own identity. In other words, is it legal to gather information (personal data) about an agent with an identity in the network when the agent uses services in the Internet?

4.3 Electric Identity

There are infinite amount of factors that must be taken into account in defining natural identity. Therefore, it is impossible to divide a person's identity into autonomous parts in practice. However, the *electric identity* is a combination of several natural identities' trust factors involving: the person using a software, the implementor(s) of the software and the implementor(s) of the environment where the software is running.

In the silicon world, unlike with natural persons, it is possible to divide hardware (brain) and software (mind) from each others. The hardware is only a habit to software that consists of applications, kernel and microcode. The hardware may change during time, but the software adapts into its new environment. The actual code defines a software's personality, the ability to adapt to new circumstances. If software cannot adapt to a new environment, it suffers from a personality disorder.

An interesting case is a moving software agent whose environment changes when the agent migrates from one computer to another. There may appear new services for the agent in the computer. This is in line with the definition of constantly evolving identity in psychology. The actual code of the agent does not change, but it adapts to a new environment and situation. Is the identity still the same or a new one? It depends on how the object discerns an agent's identity. The main aspect is that the object accepts a subject having a specific identity. Otherwise, the subject's identity has been changed.

4.3.1 Definition

Basically, a natural identity is always bound to a block of code (software) in the network. Figure 1 presents an UML model of electric identity. From the object point of view a subject has a unique electric identity if:

1. The subject represents a real world legal subject defined by law (e.g. company or a person) and inherits the latter's trust factors.

2. subject consist of software block (application, kernel, microcode) made by a natural identity and inherits the natural identity's trust factors.

3. subject runs in an environment made by a natural identity and inherits the natural identity's trust factors.

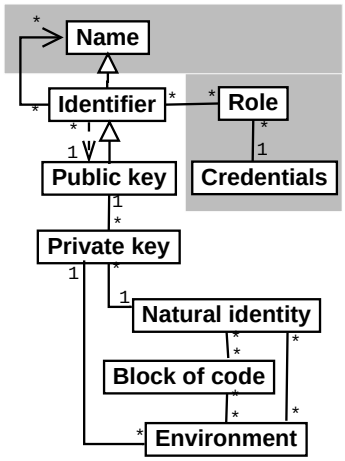

Figure 1: Model of subject's electric identity

4. subject has a secret that only it and its environment know, and the subject can prove that it owns the secret (public key pair) to object.

5. subject has one or more identifiers that are in a provable way bound to the secret. Otherwise, the identifier is called a name (habit) of subject. We discuss later, in section 5.2, more about the issue.

6. subject does not have an electric identity if any of the previous points are missing.

The electronic identity consists of different types of trust relationships. The block of code, in figure 1, trusts the environment to faithfully execute the program code. In the second form of trust, the subject trusts the environment not to misuse the secret. We can say that the subject authorizes the environment to store its secret. In the third form of trust the object believes that the identifier is bound to the secret of the subject. This form of trust is based on the cryptographic methods and is called naming trust. The last form of trust is called delegation. Actually, a natural identity delegates its rights to a block of code that represents him in the network. The presented four types of trust [Nik99] related to the electric identity are: *execution, naming, authorization* and *delegation*.

4.3.2 Analysis

An environment itself can be a subject having its own environment. For example hardware is an environment for a kernel, and kernel is an environment for a process that presents some legal subject. Actually, a data subject is an environment if it can access the other subject's secret. An environment does not own the other subject's secret, even if it knows it. Two subjects cannot share the same secret. In the same way, a subject represents some natural identity if the subject can access a natural identity's secret. All electronic identity trust factors are bound to the secret, and it can also be called the soul of the subject.

The subject is always identified by an identifier that is directly bound to the secret. If two subjects have different secrets, but are otherwise identical, they must be kept as different subjects. For example, a person may use different secrets at work and free-time to identify his laptop. He has two, time and location dependent, subjects that represent him in the network. The hardware (habit) is same, but the identity of the laptop is different at home and at work. It is important to notice that if a subject owns two secrets at the same time it is scattered and it suffers from personality disorder, because the subject may present the wrong secret in the wrong location, and therefore cause *misidentification*.

The identity is bound to a *role* [Lup98] via an identifier. Role is not a part of an electronic identity, but it is a habit. Thus, an electronic identity may have several, time and location dependent, roles. The electric identity's trust factors defines the role of the subject. Roles are further bound to authorization credits. Finally, the relationship between name, identifier and secret is analyzed in section 5.2.

4.4 Identity Types

We divide the electric identities into four main groups on account of mobility types: (1) *user*, (2) *software agent*, (3) *node* and (4) *group identity* (vs. network mobility).

In user identity, the secret is typically stored in a smart card. The smart card is an environment for the user identity. Thus, an application represents the user in the Internet and uses the person's secret to identify him on the network.

A software agent may also have an own electric identity. We discussed earlier, in section 4.2, problems that are related to legal issues of the agent identity. There are also problems involved in authorization trust between the environment and the agent. If the agent moves between nodes, it must trust the environment (node's identity) not to misuse the agent's secret, i.e. *identity theft*.

The third form of identity, node identity, consists of microcode, kernel and applications running on it. A node may represent all its processes on the network. In other words, users do not need to have their own electric identities. However, in a multi-user environment, every user should have a secret of their own. Still, in a single-user environment, the node (e.g. laptop) may represent the user. It is important to see the logical separation between the user and node identities.

In addition to the previous ones, we define an abstract identity called *group identity*. Threshold cryptography [Sch96] provides a method to divide the secret key of the public key pair into n parts. It is possible to define the count of secret keys that are required to make a valid signature. If we generate n secret keys and allow that any of the keys can be used to make a valid signature we obtain a group identity. Any of the group key owners may identify himself in the network with the same identifier. The group member may be user, software agent or a node.

5 Identity and Trust

Existing security solutions focus on finding effective methods to identify an identity, not to define the factors of identity. However, from the authorization point of view, all but the factors related to trust are meaningless. The trust factors are typically presented as credentials in access-control lists (ACLs) or in delegation certificates [EFL+99] [BFIK99]. The main principle is to define rights for subjects to access objects. The granting and delegation of rights between data subjects follow some set of rules. The Bell-LaPadula disclosure [BL], Biba integrity [Bib75] and Clark-Wilson integrity [CW87] models are well-known frameworks for such security rules. Finally, they all assign trust to some identity that is presented by some identifier in the security system. The identity can be of any electric identity type presented in section 4.4.

5.1 Establishing Trust

The main objective in establishing trust is how an object figures out the data subject's identity, and not how the subject itself defines its own identity. However, the subject can influence the generation of its identity profile at the peer end-point, but it cannot send profile information of its identity without some evidence. The object bases its trust on the evidence that it gains from the network in electric form. The trust can be established directly or indirectly. Anyhow, it is difficult to establish a trust relationship directly between subject and object in an open environment.

Basically, there is a chain of trust between a subject and an object. The chain consists of subjects that trust each other. The trust relationship can be established in two ways.

5.1.1 Third Party

In the first case, all objects trust some third party which issues qualified responses to objects on request. The objects send information about a subject's behavior to the centralized third party that maintains a trust database. Usually, information is sent only when some conflict happens between peers. In democratic countries, typically only the police has the right to maintain such a database of citizens. The law sets limits to the attributes that can be gathered about the subjects. For example in EU, there is a directive which defines such restrictions [Par95].

To obtain a scalable security solution based on subjects' identities we cannot base the architectures on any specific law or directive, because the Internet is an open environment that cross the borders of different countries. Currently, the public certification authorizes (CAs), e.g. Verisign, bind a subject to a public key pair, but CAs do not give announcements of a subject's background or how trustworthy one is. There is a clear separation of the roles between traditional CAs and third parties answering requests for responses.

5.1.2 Peer-to-Peer

In a typical trust model an object directly trusts some subject. An object creates a trust profile of a subject on account of the subject's behavior during some period of time. The behavior can be based on social and/or electric events. The object may trust some subject to delegate given rights to other subjects. This forms a flow of trust which starts from and finally reaches the same object. In this case, none of the subjects ask requests for responses about trustworthiness from other subjects. A subject only grants rights on account of its own evidence of the peer subjects.

5.2 Identification

An object must identify a subject to be able to trust it. The *identification* is a stronger method than *authentication*. The authentication ensures that a role has *something known* or *something possessed*. The secrets may be bound only to a role, but not directly to an identity. Thus, several subjects with own identities may play the same role.

The identification process, in turn, identifies an identity, not a role. The same methods, something known or possessed, can be used to identify a person, but the secrets are related to an identity, not a role. The last form of identification, *something embodied*, directly identifies a natural identity and cannot be in the truest sense a role identifier.

In the identification process the object forms a naming trust relationship with the subject. The object ensures that the identifier belongs to the identity. Therefore, the identifier must be bound directly to the secret of the identity. If there is no naming trust relationship between the identifier and the secret the identifier is called a *name* (habit) of the subject. This is a very essential issue in analyzing the differences between identifiers and names from the mobility and multi-homing points of view.

Public key pair cryptography and zero knowledge protocols can be used to form a secret for the subject. A public key of the key pair and any cryptographic derivative of it, like hash, are identifiers of the secret. Therefore, an IP address is only a name of an identity and not an identifier in truest sense, because it does not have a direct naming trust relationship with the secret.

6 Location Names

A topological location point has a name, i.e., an IP-address in the Internet. A location name is further bound to an interface at a node. Currently, the location name is used both for end-point identification and locator purposes which leads to *semantic overloading*. Any received packet is identified with its location name, i.e., source address.

However, a node does not have any means to prove that it really "owns" the location name. Therefore, the peer cannot be sure that a node is actually the one it claims to be. In addition, without any mobility support, connections are directly bound to location names and when a location name (IP address) changes, TCP connections will be broken.

6.1 Mobility

For the purposes of this paper, we define mobility to denote the phenomenon where an entity moves while keeping its communication context active (see e.g. [Chi99]). When discussing mobility, it is often desirable to differentiate between *user mobility, code or application mobility*, and *node or end-host mobility*. Recently it has become apparent that this list needs to be augmented with *network mobility*, which refers to a situation where a whole subnetwork moves from one location to another. The mobility types can be bound to identity types presented in section 4.4.

User mobility denotes functionality that allows users to move from one host to another and continue their tasks. In the extreme form it requires full process migration together with communication session migration, combined with adaptive user interfaces that allow the migrated process to adapt to the new execution environment. Code mobility, on the other hand, refers to functionality that is needed to support mobile agents and migrating processes. In a limited form, it just allows an existing process to be migrated to a new node, and some other mechanism is needed to re-establish the communications context.

Node mobility denotes functionality that allows a communications node to change its topological location in a network. In a typical case, a wireless node changes the access point it uses to communicate with a fixed network.

In this paper we concentrate on end-host mobility. With that we mean that an end-host, i.e., a computational unit hosting a number of communicating processes, changes its topological location. At the same time, however, we want to make sure that all active communication contexts remain active. In other words, we want that the communicating processes can continue as unaffected as possible.

To reflect reality, we assume that there are a number of mobile nodes that attach to a relatively fixed network. Whenever a node moves, its location name necessarily changes. Thus, in order to continue to communicate, the node must be able to signal the changes in its network names to its active peers. Furthermore, this signaling must be secure since unsecured signaling can lead to a unauthorized traffic diversion and denial-of-service attacks (see e.g. [ea01a]).

To really understand why the current situation makes end-host mobility unnecessarily hard, we have to consider the structure of the current IP architecture. Today there are exactly two name spaces that are related to mobility. Firstly, we have network layer addresses. As stated above, these addresses are determined by the network topology. Secondly, we have Transport Layer Identifiers (TLIs, e.g., ports in TCP and UDP). Of these, the network addresses usually have a global scope, and the TLIs are unique within the scope of a single address. Since we have only these two name spaces, the communicating processes must be named with $\{$address, TLI$\}$ pairs, binding the names effectively to the topological locations in the network. This situation makes it difficult to give unique names to processes that

are mobile, e.g. hosted on a mobile node. That is, since the addresses must change due to mobility, the names of the end-points also must change. This is sometimes called the mobility problem [BPT96].

There are the two fundamental approaches to solving the mobility problem: packet forwarding and dynamic updates of end-point bindings. Packet forwarding suffers from intrinsic performance penalty: the hosts cannot use optimal routes since they do not know the topological locations of their peers. Dynamic updating of bindings suffers from a number of security problems, as well as some other smaller problems [BPT96][ea01a].

On the dynamic update side, there are two basic reasons behind the security problems. Firstly, unsecured binding updates would allow various man-in-the-middle, masquerade, and denial-of-service attacks (see e.g. [BPT96],[Nik01b]). Thus, one must employ a way to somehow secure them. This, in turn, requires creating authorization relationships between the parties involved. Secondly, even though in theory it would be possible to create a global authorization infrastructure that would allow the update messages to be secured, such an infrastructure does not exist, and creating one would be extremely difficult or impossible in practice.

The existing mobility solutions in the Internet increases the semantic overloading of location names from before. For example, the Mobile IP (MIP) protocols [DBJA02] [Per96] use a static long term IP-address, called home address, to identify a node. The home address is further bound to a temporary, location dependent address, called care-of address. An interface is bound to a new care-of-address every time the access network changes (horizontal hand-off). Therefore, the node is identified using the home address, but the actual location is named with the care-of address. The mobile node informs its peers about the new location names with location name update packets.

This easily raises several problems with the IP based identifiers. Any node in the network, may reroute any other nodes packets to anywhere, if security aspects are not taken care of.

When a mobile node moves, the peer cannot be sure that the mobile node actually has been moved. In addition, the home address can be faked before the communication starts. Current protocols assume that the node remains the same during roaming, but they cannot prove that the node is the one it claims to be.

6.2 Multi-homing

Multi-homing refers to a situation where an end-point has several parallel communication paths that it can use. Usually multi-homing is a result of either the node having several network interfaces (end-host multi-homing) or due to a network between the node and the rest of the network having redundant paths (site multi-homing).

From our theoretical point of view, a multi-homed end-host is a node that has two or more points-of-attachment with the rest of the network. That is, a multi-homed mobile node has several interfaces connected to the Internet. This situation can be characterized as the node being reachable through several topological paths; the node is simultaneously present at several topological locations. As a consequence, it also has several location names. In the general case, the addresses are completely independent of each other.

The local routing mechanism tries to handle the routing between interfaces in a flexible manner, to maintain connection active during movement. When some access network disappears, a node may need to move the traffic from one interface to another (vertical hand-off). As a consequence, the location name changes.

This raises the same problem as with horizontal hand-offs. The multi-homed node must assure the peer node that the node communicates from a new location name. Therefore, the mobile and multi-homing problems are very similar from identification point of view.

7 Secure Multi-homing and Mobility

The IP-address and an identifier of the node play completely different roles in the Internet. The separation of an end-point identifier and location names from each other makes it easy to implement secure multi-homed mobility. Figure 2 defines the new logic in socket binding, i.e. connection association information. The protocol and process classes are left out of the UML model for clarity. A *connection* is established between sockets at peer end-points.

7.1 Identifier

When a socket is created, it is bound to a subject's and object's identifiers. The identifiers are further tied to the secrets and are used in establishing a *security association* (SA) between end-points. Key exchange protocols, like HIP [Mos01a], can be used to identify an end-point with its identifier. The key exchange protocol identifies the end-points to each other. The trust on the identities (see section 4.4) is established using the principles presented in section 5.1.

An identifier is put in every packet and is used to identify the SA during communication. A security protocol, like IPSec [KA98], uses the established SA to encrypt and authenticate the transmitted data flow. The SA is bound to the identifiers and not to location names.

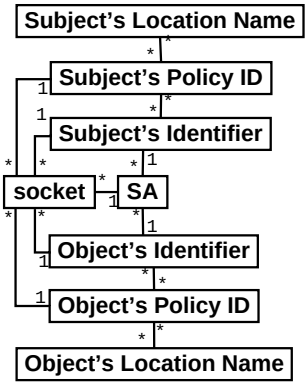

Figure 2: New logic in socket binding

7.2 Location Name

A multi-homed node is connected to different access networks through interfaces. Each interface, at different topological locations, has its own location name bound to it. The location names are used in actual routing.

When a mobile node moves geographically, the access networks may dynamically become available and disappear. This means the mobile node changes its topological location in a network and the location names are changed. Therefore, to be able to maintain active connections the mobile node must make vertical hand-offs and horizontal hand-offs during the movement. The node must have some kind of policy and mechanism to handle local routing during hand-offs.

The local policy id in figure 2 defines such a policy. The author has analyzed the usage of presented policies in [YJK+03]. The policy defines rules for local routing, in other words, priorities to use different interfaces. The socket is bound at initialization to some local policy that defines rules for source address selection. The connection association information (port, protocol) together with the QoS (Quality of Service) information defines the policy to be used for a specific connection. The policies should be defined beforehand by the application or node identity.

The foreign policy, in turn, defines rules for destination address selection. Further, the foreign policy is received typically from the peer-node (object) after establishing a SA between nodes. Of course, the key exchange protocol needs one location name to be used in the hand-shake. This initial location name is typically got from the DNS. Altogether, subject's and object's policies are bound dynamically to corresponding identifiers, and location names are bound dynamically to policies.

7.3 Binding between Identifier and Location Name

A complete data packet contains both the subject's identifier and and the object's location name. The subject's location name is actually meaningless from the object point of view, because the packet is identified with the subject's identifier. It also meaningless from the routing point of view, because the end-points should inform the peer about its location update with a separate location name update packet. However, the only reason to include the source IP address into the packet is ingress filtering.

There is no direct naming trust relationship between the identifier and the local names. However, when a packet is created, it is encrypted with a SA related to an identifier and the location name is added to the packet for routing purposes. In addition, when the subject sends a location name update message (encrypted or signed) to the object, the object binds identifier and location names together.

8 Implementation effort

We have implemented a prototype that fulfills the requirement of separating identifier from location names (see http://www.hip4inter.net). We use cryptographically generated identifiers for connection identification and IPv4/IPv6 addresses properly for routing purposes. The implementation is based on the Host Identity Payload and Protocol (HIP) [Mos01a][Mos01b][Mos01c] [JWYN02] presented by Robert Moskowitz. HIP protocol defines a new, cryptographically generated, namespace. The actual protocol locates logically between the transport and network layers. The implementation currently consists of a full HIP 4-way handshake which will be integrated with IPSec. Our next goal is to implement working end-host multi-homing and mobility [NYW03] in HIP, and evaluate the protype against known security vulnerabilities.

9 Related work

There are also other approaches where the node identifiers and address location names are essentially separated. In this section, we give a brief overview of the most relevant related work, starting from SCTP, Mobile IPv6, and other well established approaches, and proceeding to more adventurous proposals.

9.1 SCTP

Stream Control Transport Protocol (SCTP) [SXM+00] is an IETF proposed standard transport protocol, which is designed to eventually replace TCP and perhaps also UDP. In it, each communication process is associated with several IP addresses. Thus, instead of naming a communication process as {address, TLI}, the process is named as {{address1, ..., addressN}, TLI}. Furthermore, there is a proposed extension to SCTP that allows the addresses in the address set to be dynamically updated, thereby extending SCTP to address mobility in addition to multi-homing.

While the SCTP approach is sound as such, the proposed mobility extensions are bound to be plagued with the same security problems that Mobile IPv6 was recently hit with(see below). Since SCTP does not include explicit end-point identifiers, solving the security issues in a scalable way may be even harder than with Mobile IPv6.

9.2 Mobile IPv6

In Mobile IP [DBJA02], a static address is assigned to each node. This home address is then used to name communication end-points, allowing the communicating nodes to use the stable addresses independent of movements. A home agent forwards any packets sent to the home address. The current topologically correct address, i.e. the care-of-address, is signaled to the home agent, and optionally to peers. Mobile IP does not currently address end-host multi-homing, but there are informal proposals floating around how a single mobile node could use multiple home addresses and multiple care-of-addresses at the same time [MNKL02] [YJK+03].

Until recently, the largest unsolved problem in Mobile IPv6 was achieving a scalable security solution. The currently proposed solution is based on the ideas of relying on the routing infrastructure to check that a mobile node is reachable both at its claimed home address and its claimed current address (care-of-address) (Return Routability, RR). This approach is not very secure, even though it is claimed to be (almost) as secure as the current IPv4 internet. Thus, there are discussions going on about better proposals, e.g. hashing a public key and other information to the low order bits of an IPv6 address (Cryptographically Generated Addresses, CGA) [DBJA02][OR01][MC02].

The Cryptographically Generated Address replaces the suffix part of the address with a cryptographic hash generated from a public key. The peer nodes can verify that the suffix is actually generated from the public key of the node and that it belongs to the sender on account of the valid signature. Therefore, the suffix part of the address fulfills the requirements of an identifier presented in section 5.2. We can say that a CGA is a mix of a location name and an identifier at the same time. The CGA method fulfills the requirement of an electric identity presented in section 4.3.1.

9.3 LIN6

LIN6 [IKT01][ea02] is an approach somewhat similar to the late 8+8 or GSE [ea99]. The basic idea is that each node has a 64 bit globally unique identifier, called LIN6 ID, which is present in the IPv6 interface identifier portion of all IP addresses used by the node. In addition to this, LIN6 reserves a special IPv6 prefix, called a LIN6 prefix, which is not routable. A node can be uniquely named by prefixing its LIN6 ID with the LIN6 prefix, resulting in what is called a LIN6 Generalized ID (GI). These Generalized IDs are then stored into the DNS, together with the address of a Mapping Agent. Since the GIs are globally unique and permanent, the communicating nodes use them as end-point names. The Mapping Agent is queried for the mobile node's current addresses. The nodes then dynamically translate the prefixes on outgoing and incoming packets, making it possible to use GIs on sockets and real addresses in routing. LIN6 also supports multi-homing through allowing a single GI to be associated with several real addresses.

At this writing, the largest unsolved problems in LIN6 are related to the scalability aspects on the security side. The address update messages are protected with IPsec AH, thereby requiring some kind of global infrastructure in order to establish the required security associations. However, it is possible to generate a LIN6 ID in the same way as CGA, and therefore apply the security principles presented in this paper also to LIN6.

9.4 Homeless Mobile IPv6

Homeless Mobile IPv6 [NCL01][ea01b] was an idea by Nikander et. al. of adding end-host multi-homing to Mobile IPv6, and at the same time getting rid of home addresses and much of the extension header overhead. The basic idea was fairly similar to SCTP, but the implementation was placed on the network layer instead of the transport layer. Each node was represented at the kernel level as a set of IP addresses instead of a single address. The sockets were bound

to the address sets (instead of single addresses), and the applications were not aware of the actual addresses changing underneath. Extended Mobile IPv6 Binding Updates were used to signal changes in the address sets.

The project did not properly address the involved security problems; instead, the security considerations lead to the definition of the address ownership problem [Nik01b][Nik01a] and thereby paved the way for the Mobile IPv6 security solutions.

9.5 TCP Migrate

Snoeren and Balakrishnan [SB00a][SB00b] propose an extension to the TCP protocol that allows the TCP end-points to migrate from an address to another. Being structurally fairly similar to Huitema's Multi-Homed TCP, the approach solves the security issue through using the unauthenticated Elliptic Curve (EC) Diffie-Hellman protocol to generate a session key, separately for each TCP session. However, they do not solve the double jump problem, and rely on Dynamic DNS for initial contact.

In a later paper [SBmFK01], the work is continued towards a more generic mobile session layer concept. The authors concentrate on dealing with long-lasting but transient outages in connections, and suggest that in addition to solving the initial connection and mobility tracking problems, a proper mobility architecture should also address graceful disconnections, recovery from peer hibernation, and need for fast reconnection.

10 Conclusions

A subject has an identity which is identified using an identifier not a name. The identity must contain a secret that is known only by the subject. In addition, the subject must be able to prove to object that it owns the secret. The identification is based on cryptographically achieved naming trust relationship between an identifier and the identity. Basically, a public key pair is suitable for fulfilling the requirements of identity and identifier. The public key of the key pair, and any hash of it, are called identifiers. A name cannot be called an identifier if it is not a secure derivative of a public key. A normal IP address cannot therefore be an end-point identifier. Instead, an IP address is called a location name of a subject.

Identifiers are directly bound to the socket, and location names (i.e. IP addresses) are further dynamically bound to the identifiers. The dynamic binding between identifiers and location names makes it possible to implement secure multi-homed mobility. The identifiers are used to create a security association between end-points and to identify a connection. The location names are used properly for routing purposes.

There are currently several end-host multi-homing protocol proposals. However, all of them, except HIP and Mobile IPv6 with CGA support, use names as end-point identifiers which leads to insecure mobility solutions. The authors believe that it is possible to re-facture some of the existing mobility and multi-homing protocols to take advantage of cryptographically generated identifiers as end-point primary identifiers. This leads to an architecture where security is an integral part of a protocol and not an extension.

11 Acknowledgements

We want to thank our colleagues Goran Schultz and Yuri Ismailov for their constructive comments of this paper. We also want to thank Tiina Huurinainen for her feedback related to psychological viewpoints.

References

[Bel98] S. Bellowin. EIDs, IPsec and HostNAT. http://www.research.att.com/s̃mb/talks/hostnat.pdf, March 1998. Presentation give at 41st IETF in Los Angeles, California.

[BFIK99] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. RFC 2704:The KeyNote Trust-Management System Version 2, September 1999. Proposed Standard.

[Bib75] K. Biba. Integrity Considerations for Secure Computer Systems, 1975. MTR-3153, Mitre Corporation.

[BL] D. E. Bell and L. LaPadula. Secure Computer Systems. ESD-TR-73-278, Mitre Corporation; v I and II (Nov 1973), v III (Apr 1974).

[BPT96] P. Bhagwat, C. Perkins, and S. Tripathi. Network Layer Mobility: an Architecture and Survey. IEEE Personal Communications Magazine, June 1996.

[Chi99] J. N. Chiappa. Endpoints and Endpoint Names: A Proposed Enchancement to the Internet Architecture. text, http://users.exis.net/ jnc/tech/endpoints.txt, 1999. unpublished note.

[CW87] D. Clark and D. Wilson. A Comparison of Commercial and Military Computer Security Policies. In *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, pages 184–194, 1987.

[DBJA02] C. E. Perkins D. B. Johnson and J. Arkko. Mobility Support in IPv6, June 2002. Internet Draft, work in progress, draft-ietf-mobileip-ipv6-18.txt.

[ea99] M. Crawfor et. al. Separating Identifiers and Locators in Addresses: An Analysis of the GSE Proposal for IPv6. http://www.ietf.org/proceedings/99nov/I-D/draft-ietf-ipngwg-esd-analysis-05.txt, October 1999. Internet Draft, work in progress, expired.

[ea01a] A. Mankin et. al. Threat Models Introduced by Mobile IPv6 and Requirements for Security in Mobile IPv6, November 2001. Internet Draft, work in progress, expired, draft-ietf-mobileip-mipv6-scrty-reqts-02.txt.

[ea01b] P. Nikander et. al. Homeless Mobile IPv6. http://www.tml.hut.fi/ pnr/publications/draft-nikander-mobileip-homelessv6-01.txt, February 2001. Internet Draft, work in progress, expired.

[ea02] F. Teraoka et. al. LIN6: A Solution to Mobility and Multi-Homing in IPv6, August 2002. Internet Draft, work in progress, expired, draft-teraoka-ipng-lin6-01.txt.

[EFL+99] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. RFC 2693: SPKI Certificate Theory, September 1999. Proposed Standard.

[IKT01] M. Ishiyama, M. Kunishi, and F. Teraoka. An Analysis of Mobility Handling in LIN6. pages 1 – 1652, September 2001. The Fourth International Symposium on Wireless Personal Multimedia Communications (WPMC '01), Conference Proceedings Volumes 1 - 3.

[JWYN02] P. Jokela, J. Wall, J. Ylitalo, and P. Nikander. Optimized Packet Structure for HIP, June 2002. Internet Draft, work in progress, draft-jokela-hip-packets-00.txt.

[KA98] S. Kent and R. Atkinson. RFC 2401: Security Architecture for the Internet Protocol, November 1998. Proposed Standard.

[Lup98] E.C. Lupu. A Role-Based Framework for Distributed Systems Management. Dissertation, University of London, Department of Computing, July 1998.

[MC02] G. Montenegro and C. Castelluccia. SUCV Identifiers and Addresses, July 2002. Internet Draft, work in progress, draft-montenegro-sucv-03.txt.

[MNKL02] N. Montavont, T. Noel, and M. Kassi-Lahlou. MIPv6 for Multiple Interfaces, July 2002. Internet Draft, work in progress, raft-montavont-mobileip-mmi-00.txt.

[Mos01a] R. Moskowitz. Host Identity Payload And Protocol. http://homebase.htt-consult.com/draft-moskowitz-hip-05.txt, November 2001. Internet Draft, work in progress, expired.

[Mos01b] R. Moskowitz. Host Identity Payload Architecture. http://homebase.htt-consult.com/draft-moskowitz-hip-arch-02.txt, February 2001. Internet Draft, work in progress, expired.

[Mos01c] R. Moskowitz. Host Identity Payload Implementation. http://homebase.htt-consult.com/draft-moskowitz-hip-impl-01.txt, February 2001. Internet Draft, work in progress, expired.

[NCL01] P. Nikander, C. Candolin, and J. Lundberg. From address orientation to host orientation, December 2001. Roseaux et systemes repartis, calculateurs paralleles, ISSN 1260-3198, Special Issue on Mobility and Internet, Volume 13, Nr:o ?, Hermes Science Publications, Paris, France.

[Nik99] P. Nikander. An Architecture for Authorization and Delegation in Distributed Object-Oriented Systems. Dissertation, Helsinki University of Technology, Department of Computer Science, TML laboratory, March 1999.

[Nik01a] P. Nikander. An Address Ownership Problem in IPv6. http://www.tml.hut.fi/ pnr/publications/draft-nikander-ipng-address-ownership-00.txt, February 2001. Internet Draft, work in progress, expired.

[Nik01b] P. Nikander. Denial-of-Service, Address Ownership, and Early Authentication in the IPv6 World. Published in the workshop proceedings at the LNCS series, April 2001. presented at Cambridge Security Protocols Workshop 2001, April 25-27, Cambridge University.

[NYW03] P. Nikander, J. Ylitalo, and J. Wall. Integrating Security, Mobility, and Multi-Homing in a HIP way, February 2003. The 10th Annual Network and Distributed System Security Symposium, Catamaran Resort Hotel, San Diego, California. To be published.

[OR01] G. O'Shea and M. Roe. Child-proof Authentication for MIPv6 (CAM). In *ACM Computer Communications Review, Volume 31, Number 2, ISSN 0146-4833*. ACM, April 2001.

[Par95] European Parliament. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, November 1995.

[Per96] C. Perkins. RFC2002: IP Mobility Support, October 1996. Proposed Standard.

[SB00a] A.C. Snoeren and H. Balakrishnan. An End-to-End Approach to Host Mobility, August 2000. Proc. of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking.

[SB00b] A.C. Snoeren and H. Balakrishnan. TCP Connection Migration. http://nms.lcs.mit.edu/papers/draft-snoeren-tcp-migrate-00.txt, November 2000. Internet Draft, work in progress, expired.

[SBmFK01] A.C. Snoeren, H. Balakrishnan, and m. Frans Kaashoek. Reconsidering Internet Mobility, May 2001. Proc. of the 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII).

[Sch96] B. Schneier. Applied Cryptography, 2nd Edition: Protocols, Algorithms, and Source Code in C, 1996.

[SXM+00] R.R. Stewart, Q. Xie, K. Morneault, C. Sharp, H.J. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Pax son. RFC2960: Stream Control Transmission Protocol, October 2000. Proposed Standard.

[Vak] S. Vaknin. The Shattered Identity. http://samvak.tripod.com/identity.html; referred Oct. 2002.

[YJK+03] J. Ylitalo, T. Jokikyyny, T. Kauppinen, A.J. Tuominen, and J. Laine. Dynamic Network Interface Selection in Multihomed Mobile Hosts, January 2003. HICSS-36, Hilton Waikoloa Village, Island of Hawaii, (Big Island), to be published.