Towards implementing the Mobile Transient Internet Architecture

by

Joud Said Khoury

B.E., Lebanese American University, 2003

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Electrical Engineering

The University of New Mexico

Albuquerque, New Mexico

December, 2006

Dedication

To the memory of my father, Said
To the hard work of my mother, Layla
To my dear nephews, Said and Serge
To my lovely twin sister, Jacky
and finally,
To my great brothers, Jack and John
For their endless love and support.

"The greater danger for most of us is not that our aim is too high and we miss it, but that it is too low and we reach it." – Michelangelo Buonarroti

Acknowledgments

I would like to thank my advisor and mentor, Professor Chaouki Abdallah for his continuous support, his help and patience throughout this work. His guidance makes my working and learning experience a very challenging and enjoyable one. Thank you Professor.

My sincere thanks for Dr. Henry Jerez who supervised this work and who guided me all the way. Henry worked out every detail with me. His great ideas inspired my work and motivated me to pursue this thesis. Thank you Henry for everything.

I would like to extend my thanks to the Corporation for National Research Initiatives CNRI who funded me and supported me throughout this work.

I would also like to dedicate this thesis to my mother for her endless love and support. Thank you Layla, Jack, John and Jacky.

Finally, I want to convey my gratitude to all my friends who helped me and supported me.

Towards implementing the Mobile Transient Internet Architecture

by

Joud Said Khoury

ABSTRACT OF THESIS

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

Electrical Engineering

The University of New Mexico

Albuquerque, New Mexico

December, 2006

Towards implementing the Mobile Transient Internet Architecture

by

Joud Said Khoury

B.E., Lebanese American University, 2003

M.S. in Electrical Engineering, University of New Mexico, 2006

Abstract

We introduce an application of a mobile transient network architecture on top of the current Internet. The mobile transient network architecture is a new network architecture destined to merge existing and future networks, communication implementations and protocol operations by introducing a new paradigm to data delivery and identification. This thesis explores some inevitable properties of the architecture and presents some application extensions to the architecture. It attempts to specifically reinforce some of the powerful notions exposed by the architecture from an application perspective. Of these notions, we explore the network expansion layer, an overlay of components and services, that enables a persistent identification network and other required services. The overlay abstraction introduces several benefits of which mobility and communication across heterogeneous network structures are of interest to this thesis. We present implementations of several components and protocols including gateways, Agents, Open Device Access Protocol (ODAP) and

the Handle-Session Initiation Protocol (H-SIP). Our present identification network implementation exploits the current implementation of the Handle System through the use of distributed, global and persistent identifiers called *handles*. Handles are used to identify and locate devices and services abstracting any physical location or network association from the communicating ends. Throughout this thesis, a communication framework is demonstrated that would allow for mobile devices on the public Internet to have persistent identifiers and thus be persistently accessible either directly or indirectly. This first application expands IP inter-operability beyond its current boundaries.

Another application is presented that allows for inter-domain SIP mobility by using handles to abstract SIP users. The Session Initiation Protocol (SIP) presents one approach towards supporting IP mobility. Additionally, SIP is increasingly gaining in popularity as the next generation multimedia signaling and session establishment protocol. It is anticipated that the SIP infrastructure will be extensively deployed all over the Internet. In this thesis, we explore an efficient approach to inter-domain SIP mobility in an attempt to improve personal and terminal mobility schemes. We succeed in applying our persistent identification framework to application level SIP addressing by introducing a level of indirection on top of the traditional SIP architecture. We refer to our approach as the Handle SIP (H-SIP). H-SIP leverages the current SIP architecture abstracting any domain binding from users. We show how this approach helps achieve efficient inter-domain authentication and call routing and we present evaluation and performance measurements for the implemented frameworks.

Contents

List of Figures

List of Tables

Glossary

MO	Mobile Object
ME	Mobile Entity
PIN	Persistent Identification Network
DO	Digital Object

Chapter 1

Introduction

1.1 Problem Statement

Several issues may be identified within the formal structure of the network over which mobile and transient devices must currently operate. One of those is the inherent dependence of the current Internet addressing, mainly IP, on location and attachment points. This hinders mobility in communication, which is an essential property that the network should support given the large proliferation of mobile devices. Another issue with the current Internet implementation is the static circuit-oriented communication paradigm. Communicating endpoints need to first establish a circuit and bind themselves to this static circuit during the course of a communication. Additionally, the current Internet fails to support off-line communication. A host can communicate with a another host only if both of them are connected. Secure communication is yet another issue that has no inherent support in the current implementation of the Internet. Consequently, mobility, transient and secure communication, and persistent identification are basic examples of essential needs of current and future networks that are difficult with the present Internet implementation.

Mobility is not only essential, but is continuously evolving and imposing additional burdens on the current networks. The advent of dual mode phones (WiFi-Cellular), for example, compels the network to support not only the ability of the device to change attachment points, but also its ability to use different communication environments and identification mechanisms. The large proliferation of mobile devices and services, and the nomadic computing lifestyles on the current networks necessitate the evolution of the Internet into one that inherently supports mobility and heterogeneous communication. To support mobility, a network entity (device, service) needs to maintain a persistent identity while moving. This identity should be independent of the entity's attachment point and particular instantiation. Hence, persistent identification is necessary towards supporting mobility.

This thesis is part of an ongoing research effort to address such problems. It discusses the concepts and current implementations of a possible future communication network that addresses many of the limitations of the current Internet.

1.2 Preliminaries

1.2.1 Persistent Identification

Persistent identity constitutes a key element of our research. To implement persistent identity, we make use of the currently established Handle System [45][44][46][5]. Our current identification network exploits the current implementation of the Handle System through the use of distributed, global and persistent identifiers called *handle*s. *Handle*s, which are identified later, are used to identify and locate devices and services while abstracting any physical location or network association from the communicating ends.

The Handle System is the largest global persistent identification network that

we are aware of at this time. It offers universal basic access to registered digital objects [27]. Besides, it provides a secure global name service for digital objects over the Internet. It also has most of the required characteristics in terms of security, scalability, and reliability to identify digital entities fostering inter-operability among heterogeneous networks.

Briefly, the Handle System [45, 44, 46, 5] is intended to be a means of universal basic access to registered digital objects [27]. It provides a distributed, secure and global name service for the administration and resolution of *handle*s over the Internet. A *handle* is a persistent name that can be associated with a set of attributes. Some of these attributes may describe location, permissions, administrators and state. The fact that *handle*s are defined independently of any of the attributes or public keys of the underlying objects, makes them persistent identifiers [43]. These identifiers are managed and resolved using a secure global name service that guarantees the association of the identifier with its respective attributes over distributed communication.

We have chosen the Handle System due to its intrinsic persistence, security and scalability. The fast resolution times and the distributed administration model that the Handle System offers, also suggest it as an identification system suitable for operating in highly mobile environments. Other identification systems are either not truly persistent, or constrained in terms of their update characteristics. With HIP [31], for example, the use of the attribute (public key in this case) as the actual identifier eliminates identity persistence across changes of that particular attribute. On the other hand, using domain names to identify endpoints does not scale efficiently. This is because the Domain Name System (DNS) [30] has significant lag time [21], does not naturally or securely implement distributed individual DNS entry administration, and its architecture includes a single point of entry. Although the current implementation of the Handle System includes a single point of failure at the

Global Registry, our final design calls for an evolution of the Handle System that uses a fully decentralized P2P architecture.

Security is a crucial attribute of the Handle System. The system acts as a certification authority assuring that attributes of the name/reference are securely transferred between the communicating ends. Hence, the Handle System allows for secure name resolution and administration in a distributed fashion making it highly scalable and suitable to operate in mobile environments.

1.3 Objective and Scope of this thesis

A postulated architecture, the "Mobile Transient Network Architecture" by Jerez [25], proposes a new paradigm of transient communication networks. The main goal of this architecture is to enable seamless end-to-end communication between mobile and stationary devices, across multiple networks, and through multiple communication environments. The architecture establishes a set of infrastructure components and protocols that set the ground for a Persistent Identification Network (PIN). The basis for the operation of PIN, is an identification space consisting of unique location independent identifiers similar to the ones implemented in the Handle System[5]. Persistent Identifiers are thus used to identify and locate Digital Entities which can include devices, services, users, and even traffic.

This thesis explores some basic properties of the architecture proposed by Jerez [25] and presents some application extensions to it. In chapter 2, we review some previous work. We summarize the logical network structure and abstraction, with the basic components and operations in chapter 3. We try to specifically reinforce the validity of Digital Object abstraction exposed by the architecture from an application perspective.

In chapter 4, we introduce an application of the mobile transient network architecture on top of the current Internet. Specifically, we explore an overlay of components and services, that enables a persistent identification network and other required services. The overlay abstraction introduces several benefits of which mobility and communication across heterogenous network structures are of interest to this thesis. We present implementations of several components and protocols including gateways, Agents and the Open Device Access Protocol (ODAP). We demonstrate *device and service mobility* as well as communication across heterogeneous networks. A communication framework is demonstrated to allow for mobile devices on the public Internet to have persistent identifiers and thus be persistently accessible either directly or indirectly. This first application expands IP inter-operability beyond its current boundaries.

Chapter 5 discusses another application of the persistent identification framework, the Handle-SIP (H-SIP). H-SIP is an application of our persistent identification framework to *user mobility* in the context of the Session Initiation Protocol SIP [37]. It allows for inter-domain SIP mobility by using *handle*s to abstract SIP users. The Session Initiation Protocol (SIP) presents one approach towards supporting IP mobility. Additionally, SIP is increasingly gaining in popularity as the next generation multimedia signaling and session establishment protocol. It is anticipated that the SIP infrastructure will be extensively deployed all over the Internet. We explore an efficient approach to inter-domain SIP mobility in an attempt to improve personal and terminal mobility schemes. We present a successful application of our persistent identification framework to application level SIP addressing by introducing a level of indirection on top of the traditional SIP architecture. We refer to our approach as the Handle SIP (H-SIP). H-SIP leverages the current SIP architecture, abstracting any domain binding from users. We show how this approach helps us achieve efficient inter-domain authentication and call routing, thus facilitating inter-domain mobility.

Chapter 2

Background & Related Work

In this chapter, we review previous work in the areas of naming systems and mobility. In later chapters, we go over specific concepts as the topic requires.

We will illustrate differences and intersections between our work and other prominent proposals for solving essential problems on the current Internet. Of these problems we specifically mention the need for a new Naming System, host identification and decoupling the host identity from the actual network attachment point, overlays, and rendezvous services. We try to focus on those properties that *technically* differentiate us from others.

2.1 Naming Systems

There are several proposals that address the current need for an additional naming system on the Internet as a result of the limitations of DNS and IP namespaces. The rigidity of these namespaces resulted in their failure to satisfy current architectural needs. They are either tied to a pre-existing infrastructure, they can not eliminate the need for middleboxes, or they are overloaded, etc. [10].

We will focus in this section on some proposals that advertise a global naming architecture, mainly the work by Balakrishnan, Shenker and Walfish [10] [47], TRIAD[18] and the Globe Project [11]. These proposals are of interest to us because they share the goals of the Handle System [44]. We note here that the Handle System was first conceived by Robert Kahn and Robert Wilensky [**?**, 27] in 1995.

- The naming system proposed in [10] is a very similar approach to the Handle System. Both proposals share the concepts of a global name service, identifier persistence and independence of identifier from attributes, delegation, high level identifiers, distributed administration etc. However, in contrast to the purely flat semantic-free names used in layered naming [10], SFR [47] and Globe Project [11], the identifiers in the Handle System have syntax and semantics and can resemble DNS names with hierarchies. The handle itself being a character string is human-friendly. However, it may or may not be understandable. We believe that adding a level of indirection (for mapping unfriendly names to friendly names), on top of a flat resolution infrastructure introduces unnecessary overhead. The separation between the semantic-free referencing infrastructure and the human-friendly names advertised in SFR [47] and [10] can be done within the same System. This is the approach that the Handle System takes. A particular meaningful handle can refer to another non-meaningful handle within the same system. Hence, multiple competing third-parties can manage the meaningful handle space. This would require some extensions to the current Handle System Namespace [45].

- Another difference is that the current Handle System service model is confederated with a global registry and local servers at sites related to naming authorities. This maintains some desirable features inherited from DNS like name uniqueness and fate sharing (local names can still be resolved if a site is

disconnected from the global system; this design property is directly related to the underlying "Area of Influence" aggregation in the transient network architecture).

- An essential concept that we share with TRIAD [18] is the coupling on the router of name identification and routing, as well as the participation of backbone nodes in the resolution. The technical implementation details of implementing this concept are however completely different between our approach and TRIAD. We propose our implementation in the context of the transient network architecture [25] and the details are beyond the scope of this thesis.

- The Handle System is a currently implemented and widely used identification system. The system is mature and satisfies most of the design concepts that [10] and [47] propose. So far, handles are used to identify digital objects in repositories where each digital object has its own handle allowing long-term control and flexibility in managing digital repositories. We have extended the application of handles to network devices and services as we show in chapter 4, as well as users as we show in chapter 5. An example of a system that uses handles is DSPace [40]. Other long time users of the handle system are the Library of Congress and Defense Technical Information Center (DTIC's) Defense Virtual Library (DVL) sponsored by DARPA. Besides, the two largest users of the Handle System are the International Digital Object Identifier Foundation (IDF) and CrossRef [3].

2.2 Host Identification and Mobility

Decoupling the host identity from its actual location has been the topic of extensive research. Some proposals address this issue by respecting the end-to-end argument [31, 32, 41], while others rely on an overlay[42, 16] etc.

The IP address performed well as a location identifier due to the embedding of topological location. When mobility is introduced, the IP can no longer be used for host identity anymore. So the IP looses any meaning of identity reference and degenerates into a pure routing identifier. Decoupling the identity from the attachment point is currently achieved by inserting a level of indirection on top of the network layer to manage the abstraction of host identities. This is essential for reasons of mobility, multi-homing, etc. In our case, we use handles to identify Digital Objects in contrast to using Host Identifiers (HI) in the case of HIP [31]. In this section, we will compare our work to HIP.

Comparing to Host Identity Protocol HIP [31]

Comparing our work to the HIP, we note several differences and intersections worth listing. We compare the two proposals based on namespace, service model, and functionality. Briefly, HIP uses the public key as the host identifier and introduces a new layer in the network for translation of the Host Identifier (HI) into an actual IP. Note then that we intersect with HIP in the context of host identification. It is likely that our Digital Entity abstraction of hosts, in particular, will have much in common with HIP. However the significant logical differences in the overall identification model and the broader scope of the transient architecture, differentiate us and leave us with different technical problems. We hereby list the most common differences:

Scope: HIP is intended to fill the gaps and solve some issues in the current IP and DNS namespaces. The scope of the namespace is mainly computing platforms (endpoints) and packets. The notions of services and service bindings are not present here.

The Handle System, on the other hand, is a general-purpose global name service for secure name resolution and administration. Handles are used to identify

and locate Digital Object abstractions. Digital Objects were originally defined as uniquely identified data abstractions that encapsulate, describe, and provide value-added services to data. We have extended this data abstraction to Digital Entities, which further includes every fine grained network component and piece of information. Hosts are an integral part of any network but are only a part the overall Digital Object abstraction. Devices, services, users, data, bitstreams, and people can all be abstracted as Digital Objects.

Persistence: The HIP proposal introduces the host identifier as the public key of a key pair to provide authentication, for example when used with IPsec [28]. Clearly, the HI is not persistent across changes of the public key. This is the consequence of using an attribute as an identifier.

A *handle* is a persistent name that may be associated with a set of attributes. Some of these attributes describe location, permissions, administrators and state. The fact that *handle*s are defined independently of any of the attributes or public keys of the underlying objects, makes them persistent identifiers. These identifiers are managed and resolved using a secure global name service that guarantees the association of the identifier with its respective attributes over distributed communication.

Administration: The HIP proposal does not mandate an administrative infrastructure but depends on a rendezvous infrastructure for node location. To gain the full benefit of HIP, the HIs need to be stored somewhere and thus a rendezvous mechanism needs to be deployed.

The Handle System provides an administrative and rendezvous infrastructure. The secure service model and the distributed administration are attractive properties in a future Internet. Security is an inherent property of a trustworthy system.

Trust model: Security in HIP is an end-to-end IPsec security. In this scenario, users

have to negotiate every aspect of their communication to avoid false attacks. This introduces more complexity and computational loads on the nodes.
All communication with the Handle System is secure. This translates into a new trust model where nodes trust each other because they trust the system. The end-to-end security model leverages the intermediate secure Handle System.

Mobility: Our design of host mobility in chapter 4 remains under development. Basically, we need to address the simultaneous mobility issue, connection migration and the resulting security threats. There is a large body of literature related to mobility, and we only mention some proposals here. We share a similar perspective with HIP in supporting host mobility using a rendezvous infrastructure. Other similar mobility proposals are the Location independent networking for IPv6 LIN6 [8] which separates the host identifier from the host location. LIN6 uses LIN6 ID and mapping agents to maintain unique node bindings. The TCP Migrate [41] is another proposal for particularly addressing mobility on the Internet using an end-to-end session layer approach. DNS is used to store address bindings of hosts and no indirection point is needed for routing. Migrate however does not solve simultaneous mobility, and requires modifying TCP stacks on all hosts.

Comparing to the Internet Indirection Infrastructure I3

Internet Indirection Infrastructure I3 [42] is a general purpose indirection overlay network that uses rendezvous servers where clients can register triggers. It decouples the sending and the receiving actions where clients send traffic to the overlay and the latter takes care of forwarding the traffic to interested clients that registered triggers in the system. Some of the major drawbacks of I3 is that all traffic has to go through the overlay server thus degrading the routing efficiency, in contrast to direct routing over IP. This causes triangle routing, doubling the network traffic, and

requires maintaining flow state within the I3 network. It also stands in contrast to the end-to-end argument for communication. The simple service model in I3 makes it vulnerable to a multitude of security attacks thus requiring greater attention and complication to solve. The proposal Hi3 [32] is an attempt at using an overlay (I3) in conjunction with a secure direct end to end approach (HIP). This would allow for efficient end-to-end traffic with HIP (mobility, multi-homing, denial-of-service resistance) that leverages an independent, secure, integrated rendezvous infrastructure (I3) as an overlay to route the HIP control traffic (stability).

2.3 Conclusion

We have reviewed previous work in the areas of naming systems and mobility. In the following chapters, we introduce a network mobility scheme using an indirection layer. We also introduce an application layer mobility scheme to support inter-domain SIP mobility.

Chapter 3

Mobile Transient Network Architecture

3.1 Introduction

The current Internet implementation is based on a location and association aware communication substrate. Virtual circuits must be established between communicating endpoints in the form of connections that are bound to physical attachment points (IP addresses), thus hindering the evolution of the network. Mobility, transient communication, and persistent identification are basic examples of essential needs of current networks that are rendered difficult with the present Internet implementation. These features should be inherent characteristics of any future proposed network architecture.

To overcome current limitations, a new transient architecture [25] was postulated. This architecture merges existing and future networks by introducing a new approach to identification and data delivery. The architecture aims at enabling seamless end-to-end communication between mobile devices over a transient substrate and across

heterogeneous network infrastructures. The architecture proposes implementing a persistent identification network on top of a distributed overlay, that is an aggregation of a globally coordinating set of gateways and agents. The architecture identifies all network entities with global and persistent identifiers that are location and association independent. Communication is based on routing pools and novel protocols for routing data across several abstraction levels of the network, regardless of the end-point's current association and state. The network is regarded as an aggregation of "Areas of Influence", local ad-hoc communities that implement their own communication technologies and protocols. These Areas coordinate globally to move data towards its final destination.

In this chapter, we briefly review some of the concepts and components postulated by the transient internet architecture [25]. Of these notions, and as far as this thesis is concerned, we focus on mobility, persistent identification, Digital Entities, Agents, and Gateways. Some concrete implementations are presented in chapters 4 and 5 to demonstrate these basic notions. This chapter incorporates a substantial part of [25].

3.2 The Digital Object Architecture Abstraction

The Digital Object Architecture (DOA) [**?**, 27] aims at managing information on the Internet rather than just communicating it. The architecture is comprised of "digital objects", a resolution mechanism that maps identifiers for digital objects into state information about the objects, repositories which may be static or mobile, and metadata registries that may be used to retain information about the objects for purposes of search and data mining. The Architecture received the 2003 Digital ID World Award for combining innovation with practical reality.

The major contribution of this thesis is to demonstrate the feasibility and the

usefulness of applying the DOA abstraction to networked entities, mainly devices and services. Each network resource is mapped into its digital abstraction, i.e. a digital object, that is uniquely identified by a persistent identifier. Such digital objects are considered persistent and independent of their current physical or geographic attributes. This abstraction allows persistent addressing and communication with these entities regardless of their current association, location or means of communication. For example, such an entity will move seamlessly from one network environment to another and still be seamlessly incorporated. Network entities may include end-points, users, applications, services, sessions, and backbone building blocks.

In our digital object abstraction, we have implemented two applications of these network entity abstractions. In chapter 4, actual network devices such as computers and PDAs were abstracted into their representation as digital objects within the architecture. In chapter 5, SIP-enabled users were abstracted into persistently identified entities.

This type of abstraction allows a significant level of independence in terms of communication mobility, and persistence. It allows users to roam amongst several independent domains, while maintaining their identities and their network connectivity. It also decouples the logical object abstraction from any particular instantiation of the particular object. By using persistent identifiers that are independently and ubiquitously discoverable, networked devices can migrate not only from one physical location to another, but may also change key attributes of their underlying communication protocols without affecting their identity and their ability to communicate within the network. Persistent identification allows these objects to move from one network to another, and to maintain communications (using a variety of protocols and interfaces, such as IPv4, IPv6, Bluetooth, GPRS and WiFi), without affecting the way in which they are addressed, perceived, or communicated with. These objects need not have ever been assigned an IP address of any kind.

The key to achieving this level of persistence is to determine a persistent identifier from some invariant attribute of the entity, such as its physical MAC address. An indirection level is then used to map that entity into the rest of the network. Indirection is provided by a resolution system, the Handle System in our case, and communication translation boxes which we call XCOM gateways (discussed in chapter 4). The gateways allow networked resources to be perceived during communication as persistent digital objects.

3.3 Mobility, Persistence, and Agents

Mobility is traditionally associated with physical network address association and re-association. Communication is based on the concept that a full virtual circuit between one end and the other is, at least during the course of a particular communication exchange, immutable. Therefore, communication is expected to occur within a set of formally identified and immutable devices associated with particular instantaneous connections. As a result, the communication is static in nature, especially because the routing mechanisms used by the current internet implementation are based on network level addressing. This communication additionally depends on the ability of a particular node to implement the communication protocol of the initiator and to be physically associated with an address (typically IP) in the currently deployed network. A different type of communication that is persistent in nature and oblivious to network address volatility is proposed by the architecture. Persistence is the result of an independent network identification mechanism that abstractly identifies each device and piece of data being transmitted regardless of its communication and interaction mechanisms. This identification system is essentially invariant across time and network association, which translates into communications that can survive not only network re-association and migration, but also physical

disconnections and relocations.

Persistence requires the presence of independent actuators that perform maintenance tasks and implement the overall architecture policies. A series of agents are employed by the architecture to perform these tasks. Such agents are responsible for updating and disseminating information inside the system. This information handling is destined to update the global persistent identifiers and to establish logical coordination. These agents are capable of interfacing directly with the operating system, and with the associated hardware of the particular hosting nodes. They are addressable regardless of their particular association through persistent identifiers, which enables them to be part of a set of flexible overlay networks. The globally signed and authenticated agents are capable of updating crucial global information, and routing traffic based on this logically persistent infrastructure. Hence, persistence, which was traditionally adversely affected by mobility, is achieved not only trough the existence of a logical persistent infrastructure, but also through a set of agents that contribute to the freshness and stability of the overall system.

3.4 Devices and services

In order to guarantee the stability and effectiveness of the persistent network, global and persistent indirection identifiers are used to abstract device and service identification from their particular network association and location. The Handle System [5] allows the use of a common infrastructure to identify particular devices and their services as well as the data flowing through them. Particular devices will map locally and in the context of their current means of communication to a particular network and physical address. At the same time, services will map locally to particular identifiers possibly inside the particular operating system and environment in which they are set to exist. Both types of addresses are bound to change over

time and are therefore prime candidates for persistent identification using a set of handle identifiers that abstract their locality, thus transforming them into globally addressable resources.

The maintenance and implementation of the persistent identifiers at the level of the devices and their services is traditionally handled by agents that reside inside the moving devices. There also exists a set of devices which are either incapable or unwilling to hold diverse or flexible enough agents to correctly implement the logical network structure. This is the case of devices with very limited hardware resources such as sensors, embedded systems, and some application-specific hardware devices such as Voice-over-IP phones. In order to enable these devices to participate in the persistently identified network, the architecture introduces Persistent Coordinated Translation (PCT) gateways. These gateways set the framework for the dynamic incorporation of multiple devices to the overall inter-network persistent overlay. They not only assume delegated agent functions for the devices, but also provide these agents with the necessary hardware resources enabling them to communicate and expand their associated device connectivity features, while implementing the global overall architecture.

3.5 PCT-gateways and Open device access protocol

Throughout its development, there has been a push to create an Internet that puts the least amount of intelligence and configurability in the underlying network infrastructure, and to push any intelligence to the edges of the network obeying the *end-to-end argument* [38]. The transient architecture postulates a system in which a third intermediate level that prolongs the actual reach of the Internet is introduced. It introduces an expansion layer that appears to the Internet as an edge service as well as an infrastructure component. The expansion layer uses the edge nodes to

provide expansion services that turn them into basic communication infrastructure components. This layer and its components relieve roaming and mobile devices, as well as heterogeneous networks, from the burdens of compatibility and constrained implementations of larger protocols and non-native stacks. The Persistent Coordinated Translation (PCT) gateway is intended, in essence, to provide a seamless implementation of an overlay persistent network to diverse, and often non-standard network implementations. It allows these networks to communicate not only with the overall backbone infrastructure, but also amongst themselves. The PCT gateway is responsible for empowering persistent identifier association and network protocol translation, to homogenize the communication space amongst heterogeneous networks. PCT gateways may lay in the midst of logical network junctions, or at the edges of homogenous network connectivity structures, in order to expand and reshape their reach onto other networks and devices. Two types of PCT Gateways are identified:

- Edge PCT Gateways or E-PCT Gateways: E-PCT gateways implement a novel protocol called the Open Device access protocol postulated by Jerez in [24]. The protocol is destined to discover and associate heterogeneous devices and the networks through them with the overall persistent infrastructure. These gateways serve two types of devices:

  - Devices that are capable of maintaining their own agents but need intermediate deployment and staging servers mainly due to resource constraints.
  - Devices that are not capable of deploying an actual agent but carry within them the means to authenticate and validate with a particular surrogate agent.

  These gateways are responsible for associating and updating device and service *handle* identifiers while providing basic network translation resources. E-PCT

gateways can provide intermediate or staging bases for agents to deploy and implement services with their associated devices. The agents can then discover such devices and associate them with the overall architecture. These agents and the E-PCT gateway in turn perform physical, network, and software level translation operations as well as regular overlay network functions destined to maintain the persistent identifier consistency. The E-PCT gateways also provide multiple "Area of Influence" inter connectivity and routing services.

- Interface PCT Gateways or I-PCT Gateways: I-PCT gateways handle the migration of non-persistent instantaneous traffic to transient persistent networks and vice versa. They are also responsible for interfacing current technologies with the persistent resources and facilities of the transient infrastructure. The agents residing in these gateways perform on-the-fly protocol translations such as DNS-to-Handle and vice versa as well as application specific implementations. This is precisely the type of gateway used by Khoury [**?**] for integration of DNS infrastructure with the Voice-over-IP roaming infrastructure as discussed in later chapters. The I-PCT gateways provide application level translation and service integration rather than formal traffic routing.

The effective combination of E-PCT and I-PCT Gateways enables the merging and adaptation of the basic transient network infrastructure behind the new routing protocols and content distribution in the transient network architecture. The resulting network is a basic bandwidth aggregation and propagation system that builds on persistence and indirection mechanisms to realize a mixed routing, dynamic and self organizing network.

3.6 Conclusion

In this chapter, we briefly reviewed some concepts proposed by the transient network architecture. Some concrete implementations are presented in chapters 4 and 5 to demonstrate the validity and usefulness of persistent identification, mobility, agents and gateways. Other concepts proposed by the transient architecture such as Areas of Influence, Green Networks, Data routing, etc. are beyond the scope of this thesis and were not presented in this chapter.

Chapter 4

IP Mobility and Inter-Operability

4.1 Introduction

In this chapter, we address the issue of host and service mobility as an application to the transient network architecture. The chapter is further described in [29].

Mobility is ever evolving and imposing additional burdens on the current networks. The advent of dual mode phones (WiFi/Cellular), for example, compels the network to support not only the ability of the device to change attachment points, but also its ability to use different communication environments and identification mechanisms. The mobility problem is solved by using an abstraction mechanism for the identification of hosts and services. The heterogeneous communication issue is addressed through the use of Persistent Coordinated Translation (PCT) gateways that enable protocol and service translation in addition to communication mechanism translation.

We show how to apply the Handle System identification framework to networked Digital Entities, mainly devices and services. Each Digital Entity is assigned a *handle*

i.e. a name that can be associated with a set of attributes. Some of these attributes may be location, permissions, administrators, and state. We assume all connections to be transient in nature but persistently identified.

Simply put, decoupling the host identity from its attributes is implemented as follows: Our framework identifies all network devices and services using high level identifiers that are unique, global, and persistent. The devices themselves, or their respective delegates, are expected to maintain a valid binding between the device identifier and its present attributes (for example IP address, administrators, etc...). The freshness of the binding is implemented by agents that reside on the devices or on their delegates. These agents make sure that the identifier-to-address mapping is never stale, thus rendering the device accessible even after it changes its attachment point. An indirection layer situated above the actual network layer, performs handle resolution into actual IPv6/v4 addresses (we are currently reusing DNS). Hence, a device is able to change its attachment point frequently, while remaining accessible through its persistent identifier. More details on this are presented in section 4.3. The abstraction and persistence of the identifiers, the scalability of the identification system and its inherent security are all attractive features to support mobility on the Internet.

The rest of the chapter is organized as follows: Section 4.2 introduces the problem of host mobility and identification, and presents some current proposed solutions to the problem. Section 4.3 describes the implementation of some of these components, essentially Agents and gateways. In addition, we present demonstrations conducted on top of the implemented framework to assert the usefulness of the underlying concepts in addressing some current Internet limitations. Finally, the chapter points out some future work in the area of IP mobility.

4.2 Mobility and Identification

Clearly, *mobility* and *host identity* are interrelated notions. Forwarding traffic to a mobile host can not be achieved within a network that identifies a host by a physical address. The IP address performed reasonably well as an identifier until truly mobile systems arrived. Mobile devices have caused IP to loose any meaning of identity reference and consequently, to degenerate into a pure routing identifier. Various approaches have been proposed for solving the problem of host mobility on the Internet. Most of these approaches focus on inserting a level of indirection, whether at the routing level (network core), or at the end hosts (network edge) or through an overlay infrastructure. Some of these proposals address the mobility issue by decoupling the host identity from the actual host location.

Since mobile entities tend to frequently change attributes and state, identifying them becomes a complex task, especially when the same mobile object travels along different heterogeneous network structures and systems. This highlights the issue of identifying and locating those digitally abstracted objects. The basic assumption here is that mobile objects maintain their identity as their attributes change. To further appreciate the identification problem of mobile devices, we present a brief review of some proposed solutions to host mobility.

On one hand, Mobile IP [34] and Nimrod [14, 35] are two purely routing solutions. Mobile IP assumes that a mobile host will maintain the same IP address on its home network. The home agent handles the routing and makes sure that all packets addressed to the mobile host's home IP address, will be able to reach it. This indirection elevates the home IP address from a physical identifier to a host identifier that is location independent. Nimrod [14], a next generation routing infrastructure, is another routing approach to mobility. It introduces the concept of global end point identifiers (EIDs). Nimrod uses EIDs to address packets where an EID is resolved

into an actual attachment point by the routing infrastructure itself. Mobility in Nimrod [35] is enabled using the Dynamic Association Module (DAM) that takes care of updating the EID mappings in the routing infrastructure. Peernet [16] is yet another routing approach that separates identity from location by providing an alternative to IP addressing. In Peernet, location information is based on binary address trees that simplify routing. Hosts are expected to update their location information into specific servers so that peer hosts can find them.

On the other hand, some proposals introduce end-to-end approaches to mobility. Migrate [41] is a session layer approach to mobility that involves modifying the network stacks at the end hosts. Migrate uses domain names (or other naming systems) to identify and track hosts. Domain names are translated into actual IP addresses at run time, hence, separating host identities from their physical attachment points. The Host Identity Protocol (HIP) [31] uses a cryptographic key as the host identifier (HI) and introduces a new layer at the end host stack (above the network layer) for the translation of the HI into an actual IP address.

Separating physical addresses from identifiers is also addressed through overlay networks on top of IP. The Internet Indirection Infrastructure (I3) [42] is an overlay network that utilizes rendezvous servers for clients to register triggers. I3 decouples the sending and receiving actions where clients send traffic to the overlay and the latter takes care of forwarding the traffic to other interested clients that registered triggers in the system. Mobility in I3 is the direct consequence of Identification abstraction. This is achieved using the trigger *id* [42] that acts as a rendezvous point between the sender's and receiver's traffic.

The Session Initiation Protocol (SIP) [37] architecture is another overlay approach that can be efficiently reused to provide mobility [48, 39] with a readily available infrastructure. This avoids the redundancy introduced by simultaneous deployment with Mobile IP [34]. The proposal Hi3 [32] is an attempt at making use of an

overlay (I3) in conjunction with a secure direct end-to-end approach (HIP). This would allow for end-to-end traffic with HIP [31] (mobility, multi-homing, denial of service resistance) that leverages an independent secure, and integrated rendezvous infrastructure (I3 [42]) as an overlay to route the HIP control traffic.

The aforementioned mobility overview is meant to show different paradigms to identification in mobile environments that attempt to decouple the host identity from its actual IP address. Home IP addresses, domain names, rendezvous ids, and cryptographic keys are examples of endpoint identifiers that are meant to abstract the actual attachment point of an endpoint. We contend that any identification framework meant to address the same issues and operate in highly mobile and transient environments should at least support *scalability*, *persistence*, *abstraction*, and *security*. In chapter 5, we apply this indirection approach to SIP [37] systems using the Handle system to successfully abstract SIP user identities.

4.3 Implemented Framework

This section details our implementation of some basic components that work together to deliver part of the transient network architecture functionality, mainly of the DOA abstraction. The application we are interested in relies mainly on host mobility and heterogeneous communications. For the remaining part of this section, we use the term MEs to refer to network devices and services. A complementary application, presented in chapter 5, shows how *users* become MEs in a *Voice-over-IP* framework while roaming across SIP [37] domains. The users are identified by *handles* leveraging identity persistence at the service level.

The Handle System acts as a global resolution/indirection platform that abstracts the IP specifics from devices. Our implementation associates general, non-IP (Bluetooth devices, robots,...) resources with global IPv6 or IPv4 addresses. This

makes them accessible over the internet just as traditional IP-enabled devices are. IP-enabled devices are directly accessed through their global IPv6/4 addresses, whereas, non IP devices make use of an Interface PCT gateway that associates global IPv6/4 addresses with them. Communication with the Handle System, mainly resolution and administration, uses the Handle System protocol [46]. The gateways deployment will therefore allow for communication between devices residing on heterogeneous networks, while the persistent identification framework will foster mobility as we describe later.

A sketch of the overall implemented framework is depicted in Figure 4.1. It shows a particular implementation that exploits the notions of IP mobility and communication across heterogenous networks. The implementation reflects an "Intelligent house" deployment of an E-PCT gateway that exposes home appliances, robots (non-IP devices) to the global Internet, by binding them to global IPv6 addresses. A client on the Internet is now able to communicate with these devices and access their services. We describe next the details of the components and protocols involved in this implementation.

4.3.1 Components and Protocols

Persistent Identification Network and Agents

We use the current implementation of the Handle System [45, 44, 46, 5] to identify MEs. MEs therefore acquire unique, global, and persistent *handle*s. Device *handle*s map locally depending on their communication medium into physical addresses. So, for example, the *handle hdl/laptop* of a laptop computer that is connected to the Internet, can map into a global IPv6/4 address. Service *handle*s, on the other hand, can map into service-specific identifiers which could reside inside the operating system they are set to operate in. For example, the *handle hdl/uuid* could map into a specific

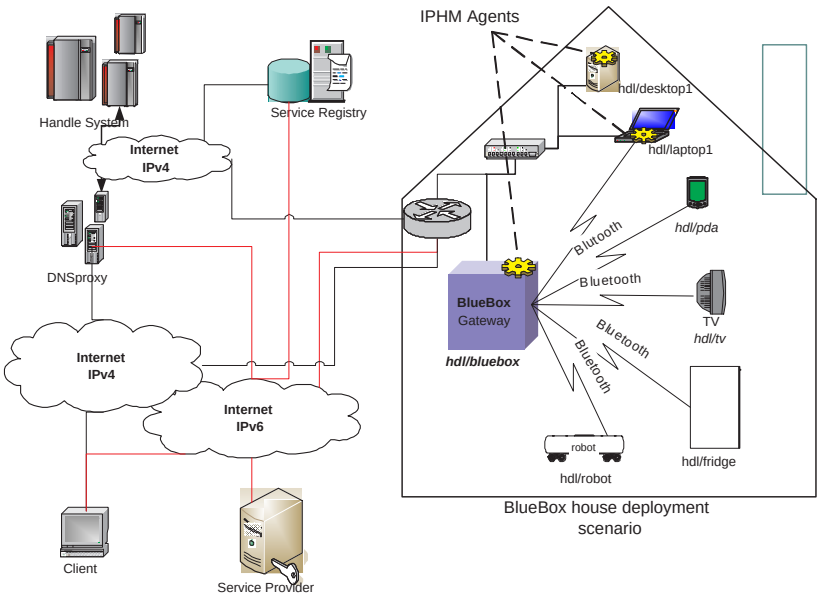

Figure 4.1: A reference sketch of the implemented framework.

service running on a Bluetooth device and identified with a certain universally unique identifier (UUID) [1]. In yet another example, the *handle hdl/sipuser* of a SIP user [37], can map into the actual SIP URL (domain binding) on which that user is reachable [**?**]. The attributes that the *handle*s map to are volatile in nature, but the *handle*s themselves are persistent.

The *handle* thus becomes independent of the resource's location, ownership and other state information. So, changing a resource's location, for example, will not break the *handle* as long as the new location is updated in the *handle*. This raises the issue of updating the *handle* attributes according to administrative privileges. One type of stationary agents that we implemented, the IP Handle Monitor (IPHM), makes sure that the *handle* bindings are never stale. In Figure 4.1, the IP devices *hdl/laptop1*, *hdl/desktop1* and the gateway *hdl/bluebox* run an IPHM agent allowing them to change their network attachment point while still being accessible through their *handle*s.

The implemented IPHM is a relatively simple software component that runs on network devices with abundant resources (laptops) or limited resources (PDA's), and monitors the device's attachment point. It is fed with the information about the particular device *handle*, as well as the certificates needed to administer the *handle*. The currently implemented IPHM detects any changes in the device's active IPv6/4 address and uses this information to update the respective attributes for the particular *handle* in the Handle System. Perhaps the best way to describe the IPHM functionality is through the use of a flowchart. We distinguish two fairly simple algorithms that the IPHM currently executes. The first, see Fig. 4.2, is responsible for device authentication with the Handle System. The second algorithm, depicted

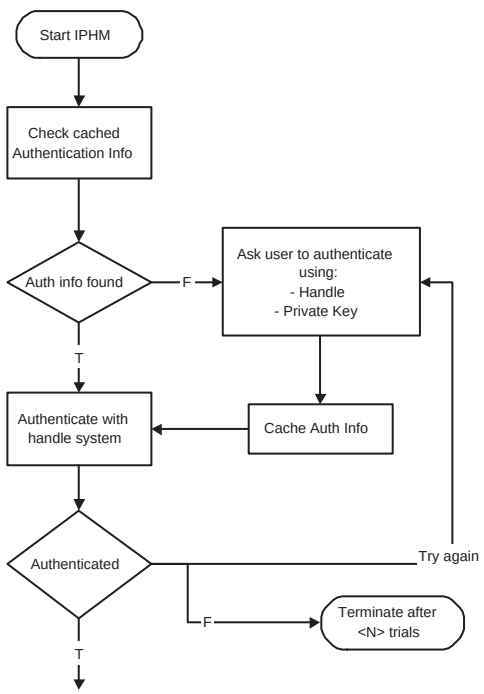

Figure 4.2: IPHM Device Authentication algorithm

in Figure 4.3, takes care of IP administration. The notation <> in the flowcharts refers to dynamic parameters. Currently the algorithm prefers to bind global IPv6 addresses with device *handle*s. In the case where no global IPv6 address is detected,

the *handle* is bound to the IPv4 address. As far as this implementation is concerned,

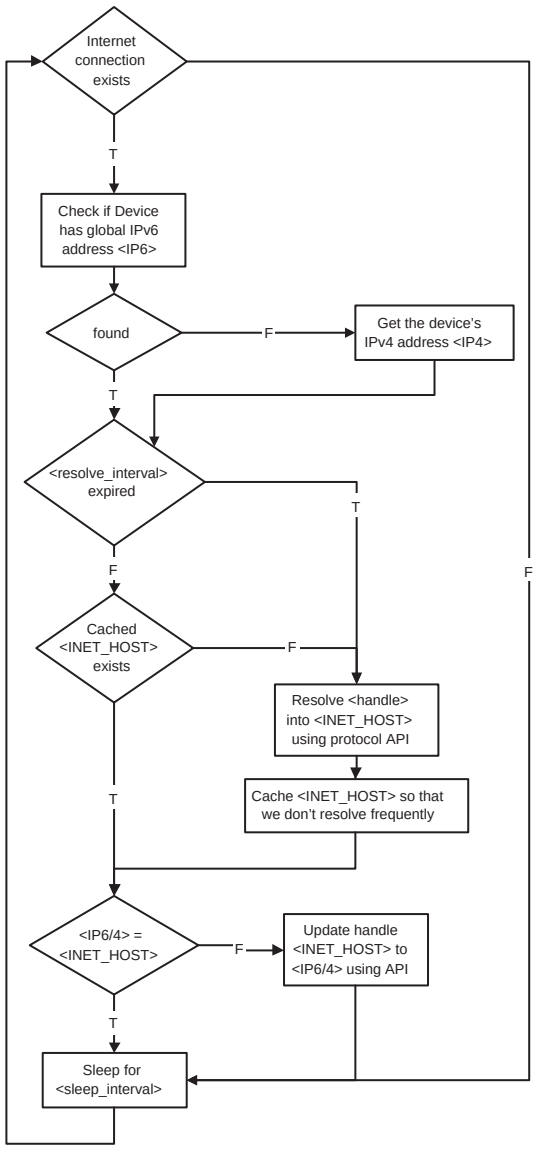

Figure 4.3: IPHM IP Admin algorithm

IPv4 addresses are assumed to be public.

The IPHM is implemented in JAVA. It runs on Windows and Linux Operating Systems. In addition, we have implemented a J2ME version (mobile edition) of the IPHM that runs on limited resource devices, mainly PDA's (Linux, Windows Mo-

bile 2003). The J2me version was tested on the Zaurus SL600 (Linux) PDA and the Siemens SX66 (Windows Mobile 2003) PDA. Note that dual stack (ipv4/ipv6) network connectivity was used during deployment allowing resources to be accessed either through a public ipv4 address or preferably through a global ipv6 address.

Open Device Access Protocol - ODAP

Briefly, ODAP [24] is a protocol targeted at the discovery of nearby network devices and the association of these discovered devices with an IP address and an indirection global address. The protocol operates in push and pull modes. In the pull mode, a gateway running the ODAP, polls devices in its vicinity for the services they wish to advertise. In the push mode, the devices will initiate service registration requests with a gateway in their vicinity. The protocol's interface currently allows for four distinct operations including *Device Discovery*, *Service Listing*, *Service Implementation*, and *Device handover preparation*. The behavior of these functions depends on the operational mode of the protocol.
We have implemented a basic version of the ODAP protocol interface operations on the *BlueBox* E-PCT gateway discussed in section 4.3.1.

PCT Gateways

E-PCT To illustrate the E-PCT type gateway, we have implemented a stand-alone product which we call the *BlueBox*. The *BlueBox* implements the Open Device Access Protocol. It allows devices residing on private networks that either have a public/private IP, or do not have an IP at all (e.g. Bluetooth devices, sensor network components, home appliances,...) to be exposed by associating the global IPv6/v4 addresses of these devices with their particular *handle*s. The actual IPv6/v4 addresses of these devices are associated with the *BlueBox* that may later on, either

route or translate the traffic to the devices.

The implemented gateway uses Linux as its operating system and communicates on two different interfaces. The Bluetooth interface is used for internal communication with Bluetooth devices. Externally, the gateway uses Ethernet/WiFi to connect to the Internet. Bluetooth is used to illustrate a non-IP environment. The ODAP is implemented in pull mode. The *BlueBox* runs an IPHM that monitors its IP address keeping its *handle* binding updated. A basic version of the ODAP is implemented on the *BlueBox* PCT Gateway in Java. The protocol interface operations are set to communicate over HTTP for demonstration purposes. The gateway receives client requests over HTTP and uses Bluetooth internally to communicate with the Bluetooth devices.

We next discuss the ODAP interface operations implemented on the *BlueBox* in light of the framework of Figure 4.1, where a client is trying to access the services provided by some devices residing behind the *BlueBox* gateway:

- *Device Discovery* As the gateway receives a client request for Device Discovery, it starts scanning for Bluetooth devices in its vicinity. It then authenticates these devices and associates each with a global *handle* partly constructed of the device's Bluetooth MAC address. Note that a special algorithm is used to compute the *handle* of a Bluetooth device and to verify it globally. This step is part of Mobile Device Authentication Protocol MDAP (under development), a protocol used to authenticate passive and active ODAP clients. It is worthwhile to note that presenting the authentication algorithm with a device (Bluetooth device in this case) will always yield the same unique *handle* for that device. A valid global IPv6/4 address from the gateway's pool of addresses is then associated with the computed *handle*. The *handle* is then either updated (*handle* already exists in the Handle System meaning that the device was previously registered with some gateway) to reflect the new IP address, or created (first

time a device registers with a gateway). At this point, the gateway can inform the requesting client of the online devices which are then listed as a set of handles.

- *Service Listing* The client then issues a Service Listing request for a specific Bluetooth device. The gateway in turn polls that device for the services/data the latter is advertising. The gateway generates the respective *handle*s of these services i.e. the gateway maps the services/data identifiers to their global *handle*s. Note that in the Bluetooth case, services are identified with unique UUIDs [1] that are part of the global service *handle*. The services for the device are then listed.

- *Service Implementation* If the service is not found locally by the gateway, the gateway resolves the service *handle* to locate the code required to implement the service. This last step involves the following sequence of events: The service handle is used to query a service registry. This query, which is contextualized to the particular characteristics of the gateway, is sent to the registry as an actionable request. The service registry points the gateway to a specific service provider site containing the actual signed code that fits the gateway's needs. This process involves secure implementation of the service which is itself identified by a persistent identifier.

  The mechanism depicted above for supporting service implementation is referred to as delegated implementation i.e. the gateway implements the service and it exposes a high level interface (web interface for example) to the client. We have implemented this approach on our *BlueBox*. On the other hand, service implementation can be done through encapsulation where the client encapsulates Bluetooth commands in IP, and the gateway decapsulates these commands without implementing the service locally.

- *Device Handover Preparation* The gateway sends keep-alive verification mes-

sages to the devices to verify their presence. Upon failure to verify device presence for a defined interval of time, the device IP is cleared and made available in the gateway's pool.

I-PCT We have implemented an I-PCT gateway responsible for protocol translation, specifically, Handle-to-DNS. We refer to this gateway as the Handle-DNS proxy (HDP). HDP is a modified DNS server that communicates using the bind protocol and implements extra functionality allowing it to associate canonical names and aliases inside its particular naming zone with *handle*s. HDP will therefore resolve canonical names inside its naming zone using the Handle System and will, in addition, allow any common DNS server to resolve DNS entries in the format: <handle>.[DNS proxy domain] to the actual value of the INET_HOST attribute of that particular *handle*. Figure 4.5 shows how the HDP is integrated with the current DNS infrastructure. Figure 4.4 shows how a client requests *handle* resolution just the same way it resolves traditional DNS names. The HDP checks for the canonical name of the DNS query in its zone files and if not found, assumes the name is a *handle* and tries to resolve it using the Handle System. The host names in the zone files can be associated with either static IP addresses (*www* in Fig. 4.4), or *handle*s (*userweb* in Fig. 4.4). In the latter case, an additional step is required to resolve the *handle*.

The significance of this approach becomes evident when the host IP changes frequently i.e. mobile host. If the host device is running an IPHM, the IP address of that device will be automatically reflected in the Handle System and no administrative changes need be implemented in the zone files. We have implemented a global DNS proxy which accepts requests from any client and acts as a mediator between DNS and the Handle System. A request to this server is a normal DNS query with query *name* having the following format: <name> = <hostname | handle> .proxy.domain. Note that the Global HDP will assume that the name part before

"proxy.domain" is a *handle* if it is not a host name in the zone files and if it obeys the handle naming syntax [45].

Possible values of DNS query names are "2118/resource1.dns.handle.net" or "user-web.dns.handle.net". The latter turns out to be the same as "100.1000/jweb.dns.handle.net" as depicted in Figure 4.4. DNS Clients (e.g. nslookup) should allow query names to contain the special characters used in the *handle* namespace such as ascii '/' (0x2F). Refer to [45] for a detailed description of the *handle* namespace. Since Internet hostnames can not contain the '/' character [19], we replaced it with the '∼' (0x7E) character for testing purposes. It is worthwhile to note that changing the IP address of the *handle* is done securely using public key authentication by the *handle* administrator (IPHM in this case). Consequently, the simultaneous functionality of the IPHM, discussed in the previous section, and the HDP allows ubiquitous access to

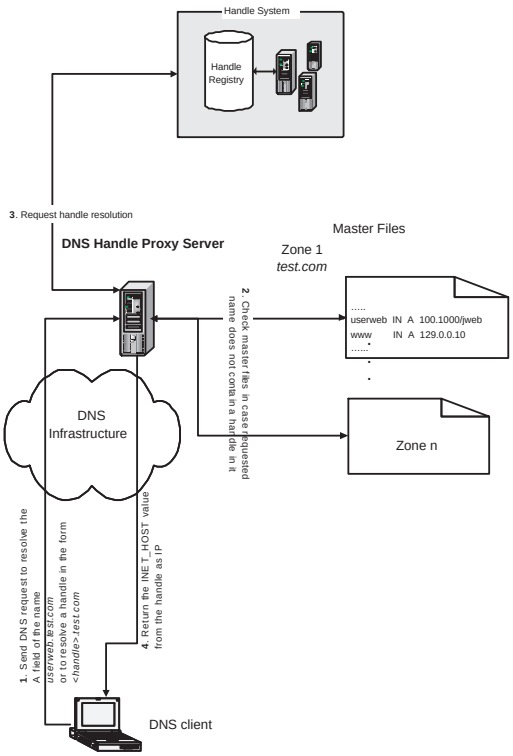

Figure 4.4: Handle-DNS Proxy

networked resources using the current network infrastructure. HDP is implemented in Java. It is currently running on a Fedora Core 4 OS.

4.3.2 Demonstrations

The following demonstrations clarify how mobility and heterogeneous communication are achieved.

IPHM and DHP

The functionality of the IPHM is demonstrated on two kinds of devices with different computing resources, laptops and PDA's. In both cases the IPHM is fed into the device with the respective authentication information. When the IPHM first starts, it tries to authenticate with the Handle System asking the user for the *handle* and the private key.
When the *handle* is successfully authenticated, the authentication information is encrypted and cached internally so that subsequent initiation of the IPHM (e.g. rebooting the device) uses the cached authentication info and does not require further user authentication. A similar scenario takes place when the IPHM starts on the PDA. For our demonstrations, a WiFi-enabled Zaurus SL6000 linux PDA is used. As the device connects to a different network, the IPHM automatically detects and updates the new IP address binding.

If the device supports dual stack (IPv6/IPv4) and obtains an IPV6 address, then the IPv6 address is reflected in the *handle*. However if the device is only IPv4 enabled, the IPv4 address is reflected by the *handle*. In the event that the IPv4 address is globally routable no further action is needed. If the IPv4 address is private then Network Address Translation (NAT) traversal is required. The NAT traversal, while also part of our overall research, is beyond the scope of this thesis. The IPHM is

dispatched as a light-weight process that runs in the background of the device and is transparent from the user. It is anticipated to become part of the operating systems.

Further, a web server was installed on both the laptop and the PDA. With the help of the Handle-DNS Proxy described earlier, we are able to access the web server using the *handle* of the device. Thus accessing `http://hdl~pda.proxy.domain/` in a browser always directs us to the web server running on the zaurus PDA (Note here that the zaurus PDA *handle* is hdl/pda). This eliminated the need to update DNS entries should the PDA connect to another network. Persistent identification is thus demonstrated. Most recent browsers (IE, FireFox) support IPv6 and prefer it by default. So, the browser will attempt to resolve the *handle* into an IPv6 address first through the HDP.
This demonstration aimed at presenting the mutual functionality of the IPHM and the HDP and their effective role in allowing IP mobility of connected devices over the current Internet.

BlueBox

This section demonstrates the functionality of the *BlueBox* as well as the Open Device Access Protocol ODAP. Figure 4.5 illustrates the communication flow and components involved in the implemented framework. The gateway or *BlueBox* is a linux machine. Recall that it has two communication interfaces, Bluetooth and WIFI/Ethernet interfaces. The gateway allows clients to access it through an HTTP interface. Note that clients are either IPv4 only, IPv6 only, or dual stack clients and so is the gateway. However, throughout this discussion, both clients and gateway are dual stack, and use IPv6. Note also that the demonstration version of ODAP runs on top of HTTP and consequently the gateway runs an HTTP server (tomcat 5.5.9 is used). Future versions of ODAP are anticipated to implement their own communication protocol.

After booting the gateway and connecting it to the Internet, some Bluetooth devices are placed in its vicinity. These devices include a mobile phone, a PDA and a Robot, all of which have their Bluetooth interfaces turned on and allow other Bluetooth devices to see them and access their services (Figure 4.5). A client browser sends an HTTP request to `http://hdl~bluebox.proxy.domain/` which is resolved by the browser into the IPv6 address of the gateway (A global HDP is running on the domain proxy.domain) as depicted in Figure 4.5 (arrows 1,2,3,4).

The client issues a *Device Discovery* request. The gateway in this case lists the discovered devices in the client's browser. At this point, the client sends a *Service Listing* request for the robot. The gateway sends back a list of the services exposed by the robot, mainly the robot control service. The client requests service implementation of the robot control service. Note that this service is made available using the LabVIEW software [7] which allows a web interface for controlling

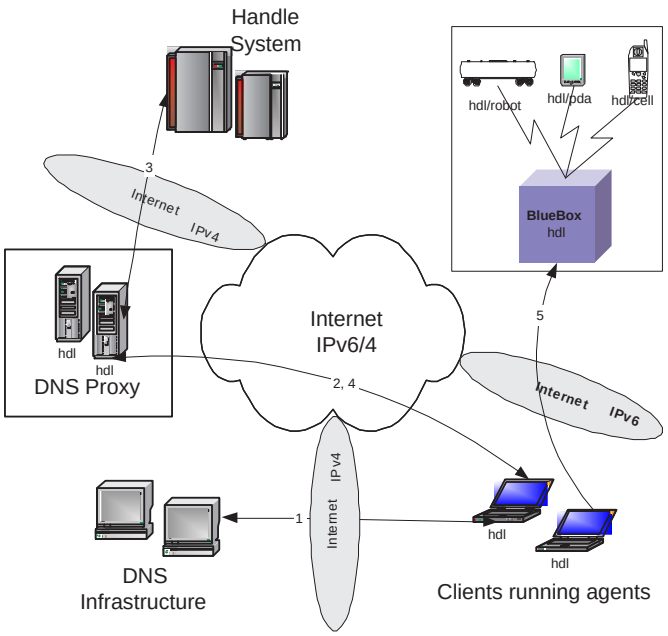

Figure 4.5: Implemented Framework

the robot. The gateway sends back a URL to client pointing the robot control web interface `http://hdl~robotcontrolservice.proxy.domain` allowing the client to control the robot. In the meantime, the gateway is always sending keep-alive messages to the devices to verify their presence and maintain their global IPv6 binding. We also demonstrated communication between an IPv6 device and another IPv4 device where the gateway handled the translation. The protocol itself is abstracted from the communicating ends.

4.4 Future Work and Enhancements

One of the limiting factors of the implemented framework is the overhead needed to resolve *handle*s. Since *handle*s can identify highly mobile devices, caching is not a viable solution. This encourages future implementations of the Handle System that would allow for significantly faster resolutions. The transient internet architecture postulates a fully Distributed overlay of the Persistent Identification Network (D-PIN) and proposes an advanced implementation of the Handle System in a structured P2P fashion that would make it more reliable. Mobile agents that handle routing based on persistent identification is one of the topics we are currently studying.

This chapter has presented a mobility application of the transient network architecture. Future work will address additional paradigms such as the Distributed PIN implementation, persistent identifier routing, etc. Our current research is focusing on implementations of the Green Network, a new network that would introduce persistence at the network level to all transient communications. Additional work is underway on the service implementation side, to enable mobile signed code implementation. This feature will allow the PCT gateways to resolve the code necessary to implement a particular service once it has been mapped into its particular persistence identifier. This feature goes beyond the postulations of CORBA [2], and would

enable full mobility and automatic service functionality updates.

4.5 Conclusion

In this chapter we have demonstrated an application implementation of a transient network architecture. We have established the usefulness and applicability of the persistent identification network PIN to identify and locate network devices and services. Demonstrations used the current implementation of the Handle System. We have also illustrated the notions of mobility and persistence in communication across heterogeneous networks using the indirection framework. The chapter has also explored the network IP layer expansion, which enabled a persistent identification network and other essential services. We have applied our concepts of transient connectivity and active persistence validation and translation to enable seamless mobility for IP devices like servers, laptops, and PDAs, as well as non-IP devices, such as Bluetooth enabled terminals. Our approach allows clients running regular IPv6 or IPv4 to directly communicate with Mobile Entities as such MEs roam. Communication is achieved regardless of the MEs current IP association or even their ability to establish IP connections. We have effectively enabled IPv4 systems and Bluetooth enabled devices to have an IPv6 presence that is persistent in nature. Mobility and communication across heterogenous network structures are thus major benefits of this work.

Chapter 5

Inter-Domain SIP Mobility

5.1 Introduction

Low-cost, broader data services and deployment of high speed access networks are major drivers pushing service providers and enterprizes to adopt packet switched multimedia communication, as opposed to current circuit switched and cellular alternatives. As an example, we have recently witnessed a rapid increase in the popularity and deployment of Voice over IP services. Enterprizes and mobile operators are currently promoting simultaneous Cellphone/Wi-Fi access by introducing dual-mode phones that can switch between the cellular network and the IP packet-switched network. Obviously, significant benefits follow for both the consumer, who can save money and get broader coverage, and the operators who can widen their service base for a lower cost. However, identity persistence issues become obvious and need to be addressed in this context.

The Session Initiation Protocol (SIP) [37] and H.323 [4] are among the most widely adopted protocols for IP telephony. The fact that both SIP and H.323 have different architectural components with merits and demerits, the ability of these

two protocols to coexist on the same network, and the simpler implementation and open collaboration of SIP, are all factors that drive this chapter to focus on SIP and introduce the proposed notions and implementations in the context of SIP. Additionally, SIP has been accepted by the 3rd Generation Partnership Project (3GPP) as a signaling protocol for establishing real-time multimedia sessions. The protocol is continuously gaining in popularity, and has been adopted by service providers such as Verizon and Sprint to provide IP telephony, instant messaging, and other data services. It is anticipated that SIP will be widely deployed by operators and enterprizes, thus populating the Internet with SIP infrastructure components. The widespread deployment of SIP is the premise of this chapter, as we will leverage this idea to propose an efficient inter-domain mobility scheme for SIP environments.

The session initiation protocol (SIP) [37] is a signaling and control protocol for handling multimedia sessions, allowing the establishment and termination of media streams between two or more participants. SIP works in concert with other multimedia protocols due to the independence of the protocol from the underlying transport mechanisms and session types. The SIP architecture allows for its deployment either as a centralized or distributed system, or even as a combination of both.

The SIP architecture is also proposed as an efficient candidate that may be reused to provide personal, terminal, and session mobility [48, 39, 33, 15] with a readily available infrastructure. This avoids the redundancy introduced by simultaneous deployment with Mobile IP [34]. The successful reuse of SIP to simultaneously support multimedia communications and mobility leverages the issues emanating from SIP users *roaming*[1] across multiple SIP domains. These issues are highlighted through a brief overview of the SIP protocol functionality.

SIP handles user location through the use of a Proxy/Location server [2] that ac-

[1]Throughout this chapter, roaming is defined as the SIP inter-domain roaming i.e. the migration of a user between different SIP domains.

[2]We will use the terms SIP proxy, SIP server, SIP registrar interchangeably.

cepts user registration requests and updates the respective user location in a location repository. The protocol inherently implements location independence through the use of the uniform resource identifiers (URI) [13], which directly offers personal mobility. A URI acts as a location independent identifier abstracting the actual physical location of a user with respect to the system. So, SIP allows for personal mobility whether through the use of a Proxy that sets up the session between the calling parties or through the use of redirection servers. However, the protocol defines a user only within the domain boundaries of the service provider. A user must associate with a specific proxy server that handles user authentication as well as initial traffic routing. The proxy maintains a unique account for the user, who in turn, is expected to coordinate with that same proxy irrespective of the user's location. This requirement translates into undue loads on the SIP server and on a particular domain. Additionally, it complicates the coordination of *roaming* users who must communicate with a central proxy server while roaming. Despite the possible presence of Firewalls and other network restrictions on the foreign domain, roaming users are required to use the central home server instead of using the available local servers.

Consequently, while URIs solve the location binding issue, they introduce the domain binding issue. Inefficient traffic routing is a direct consequence of such binding. Besides, the URI identification translates into users needing to be aware of each others' current domain associations. It also brings up the complexity of satisfying calls when initiated from regular keypad terminals.

This chapter addresses the inter-domain mobility issue by introducing an abstraction framework based on a unique and persistent identification mechanism. The chapter is an effort first introduced in [**?**] [**?**]. As far the this work is concerned, it only provides an approach that can enhance personal and terminal mobility [33] in current SIP architectures. As to session mobility, the readily available approaches like mid-call mobility [39] or its enhancements [12] may be used. The framework we

propose, which we refer to as the Handle-SIP (H-SIP), can seamlessly fit into the current SIP architecture allowing SIP users to transparently roam across different SIP domains. H-SIP can then be gradually deployed and can coexist with the traditional SIP infrastructure. User location and association are abstracted through the use of globally unique and persistent identifiers, namely the *handle*s discussed earlier. Using the Handle System as an intermediate layer on top of multiple distributed SIP implementations, thus allows us to implement seamless multi-domain authentication and call routing.

The remainder of this chapter details our proposed approach. Section 5.2 shows how H-SIP is efficiently used to enhance inter-domain SIP mobility. We also present a detailed explanation of the proposed inter-domain authentication, registration, and call routing mechanisms. In section 5.3, we develop the implementation test-bed and discuss the experimental model and performance measurements of H-SIP. Future work is discussed in section 5.4.

5.2 SIP Inter-Domain Mobility

5.2.1 Sessions and Mobility

To clarify the SIP inter-domain mobility problem, we present a simple example. Recall that SIP defines a user as an entity that associates with a particular domain. Figure 5.1 depicts a simple scenario of a roaming user *r_user* who has a valid association with his home domain *hdomain* but is currently present in a foreign domain *fdomain*. SIP signaling traffic originating from (REGISTER) or terminating at *r_user* must inefficiently pass through his home proxy server. Figure 5.1 identifies this traffic as traditional traffic flow (arrows 1,2,3: arbitrary SIP user trying to INVITE the roaming user). There are several ways in which roaming issues can be

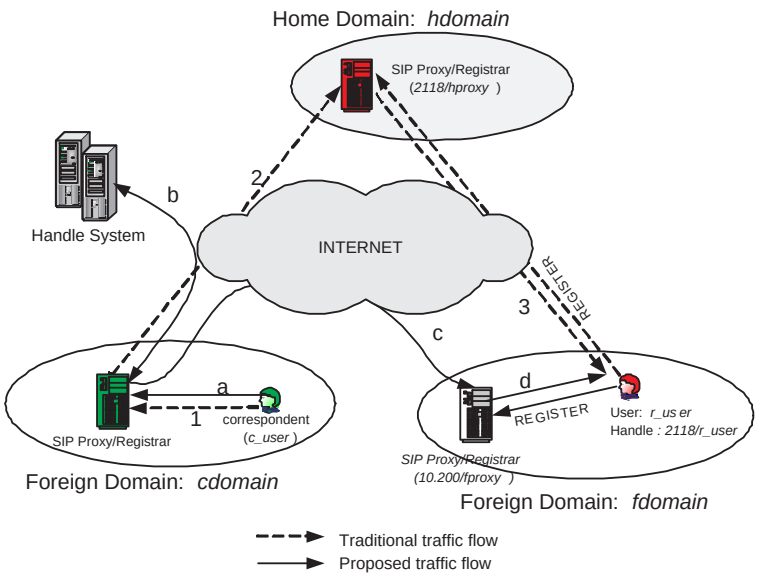

Figure 5.1: A Reference inter-domain roaming scenario

addressed, depending on whether the SIP architecture is roaming-unaware or modified to become roaming-aware. We study these issues and we present our approach by showing a typical flow for INVITE and REGISTER requests. We also compare the different approaches and illustrate the different scenarios in Figure 5.2. A more elaborate description of these scenarios is presented in section 5.3.

- The first scenario shows how SIP naturally handles a call flow for a roaming user. A data flow is presented in Figure 5.2.A. In this case, no roaming logic is injected into the system (system is roaming-unaware). All requests to/from the roaming user must go through the central home proxy server. The home proxy thus treats both roaming and non-roaming users equally and portrays a roaming user as merely a home domain user registering with a foreign contact address. Clearly, if the user is present in another country, the user's traffic would still have to go through his central home proxy (triangle routing) as depicted in Figure 5.2.A, despite the availability of a local proxy server in the

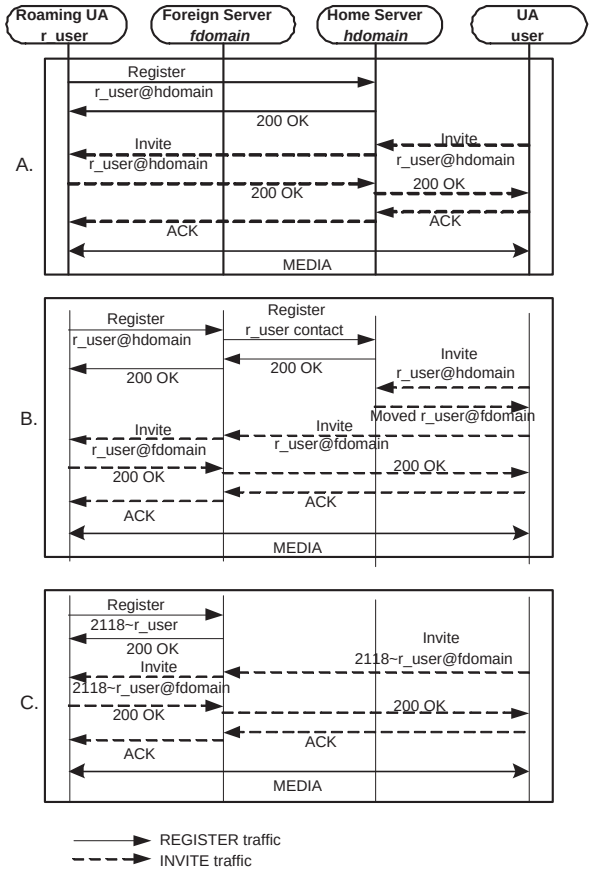

Figure 5.2: SIP traffic flow A. With no roaming logic B. With traditional roaming logic C. With proposed roaming logic

foreign domain (Foreign Server). This results into significant delays that are not accepted for time sensitive applications. Even with SIP mobility management (SIPMM) [48, 39] support (personal, terminal and session mobility) enabled, the same scenario occurs.

SIP Mobility allows a user to roam between sub-nets and domains maintaining accessibility and session continuation using pre-call and mid-call mobility signaling. With pre-call signaling, the mobile user will re-REGISTER with the home proxy anytime the user's IP address changes. With mid-call signaling, the mobile user will negotiate an address change with the correspondent user

while the session is in progress using re-INVITE messages. Mid-call mobility assumes a session is already in progress between the calling parties. Inefficient pre-call traffic routing, and service centralization, are obvious limitations that users roaming in these traditional and Mobile SIP environments have to suffer from. This is the same case for Mobile IP with Location Registers (MIP-LR) [23, 22], whereas here the SIP proxy servers are replaced with location registers. We argue that our proposed approach to roaming and inter-domain mobility can in general, significantly enhance the SIP personal and terminal mobility performance. Additionally, since our approach addresses SIP personal and terminal mobility, we can improve the pre-call portion of any SIP session mobility scheme while other features like mid-call mobility can remain unchanged. For mid-call mobility, current proposals like MIP-LR, SIPMM, or a combination of these two [50] can be used. These approaches implement mid-call mobility by sending binding updates directly to correspondent nodes without going through Home Agents. Mobile IP (MIP) [34], however, uses Home Agents to forward traffic which creates triangular routing issues. An enhanced version of MIP is MIPv6 [26] that avoids triangular routing and implements route optimization. As to the simultaneous mobility issue, discussed lately in [49], it is left for a future work to offer a secure framework for simultaneous mobility in the context of H-SIP.

- A second scenario is that of a SIP roaming-aware approach such as the one proposed by Double User Agent Servers [20], that mimics the roaming solution employed in the telecommunication environments. In other words, a user who is roaming outside his home domain, registers with a foreign server. The latter consults the user's home server for redirection, authentication and billing, and proceeds to process the user's transactions. Correspondent users trying to communicate with the roaming user will have to go through his home proxy server which in turn redirects them to the foreign proxy where the user is

currently located. Hence, significant signaling overhead results primarily due to the nature of the SIP Uniform Resource Identifier (URI). The URI is composed of a domain part, like in *r_ user@hdomain*, thus forcing the calls directed to this user to go through the *hdomain* proxy server first. The data flow for this scenario is presented in Figure 5.2.B. We argue that this approach is inefficient as it introduces unnecessary overhead and load on the original server .

In the two scenarios above, the use of URIs to identify users and the inherent dependence of the URI on a particular domain, complicates message routing. One solution is to abstract the actual identifier eliminating per-call coordination to minimize the signaling traffic in highly mobile environments.

5.2.2 H-SIP: Abstraction layer

Our proposed approach uses *handles* as globally unique identifiers to locate and identify SIP architectural elements. This abstraction allows the system to route calls independent of user location and domain association. We refer to the modified SIP framework as the Handle-SIP or H-SIP. Note that we have also exploited this abstraction approach at the level of network devices and services in chapter 4. In

handle	Field Type:index	Value
2118/r_user	HS_ADMIN:100	rwr:0.NA/2118:300
	HS_ADMIN:101	rwr:2118/r_user:200
	HS_VLIST:200	rwr:2118/r_user:300
		rwr:2118/hproxy:300
		rwr:10.200/fproxy:300
	SIP_URL:250	sip:2118.r_user@x.y.z.w
	SIP_PWD:251	password
	HS_PUBKEY:300	00BE0034……..

Figure 5.3: Sample user *handle* structure

our approach, elements of the SIP architecture, SIP users and proxy servers, are

identified with *handle*s abstracting any domain binding. Users will identify each other, and identify the SIP servers they associate with using *handle*s instead of URIs and domain names respectively. In Figure 5.1, the roaming user *r_user* will have his own *handle 2118/r_user* with the necessary administrative privileges over the *handle*. Additionally, the home proxy server has a *handle*[3] *2118/hproxy*, and the foreign proxy server has a *handle 10.200/fproxy*. Note that a *handle* has the form "*prefix/suffix*". The prefix represents the naming authority (NA) while the suffix represents a unique local name under the NA namespace [45], thus rendering the *handle* globally unique. A possible realization of the *handle 2118/r_user* inside the Handle System is depicted in Figure 5.3.

The *handle* has several fields. The HS_ADMIN and HS_VLIST fields determine the administrators of the *handle* who are the naming authority (*0.NA/2118*), the *handle* itself (*2118/r_user*) and the two proxy servers in the HS_VLIST field. Any of these administrators has the privilege to modify the fields inside the *handle* provided the administrator succeeds to authenticate with the Handle System using his private key.

5.2.3 Authentication and Registration

Currently, the most common authentication mechanism employed by SIP is the digest authentication [17] used by HTTP. When a user associates with a domain proxy server, he obtains an account on that server with a username and password which he uses to authenticate himself to the server if asked to. The digest authentication depicted here is domain dependent i.e. the user's credentials are valid for a particular domain. Briefly, digest authentication proceeds as follows:

[3]Please note that a direct mapping between domains and *handle*s exists and is enabled by a handle/DNS proxy approach.

1. User sends a REGISTER request to a SIP proxy/registrar server.
2. The server replies with a 401 unauthorized response message challenging the user to authenticate himself for the requested service (realm) through a user and password prompt.
3. The user sends back a message digest of his credentials, which include his username, password etc.
4. The same message digest is computed internally using the server's internal user information and compared to the one sent by the user.
5. Authentication is granted if the two digests match.
6. User registers with the SIP proxy/registrar server.

In our approach, we still use digest authentication for the SIP users due to its wide support by current SIP servers and user agents, although a better authentication mechanism can be designed to leverage the inherent security that *handle*s expose.

Access to the authentication information is controlled inside the Handle System by the users. Recall that each user owns and administers his own *handle*. As part of this process, the user specifies in the HS_VLIST field, the set of *handle*s that have administrative rights over his *handle*. Among these *handle*s, the user should include *handle*s of any SIP proxy server that he wishes to register with, which could be any foreign server(s) that he trusts.

Two approaches can be exploited to implement the logic needed by the current SIP architecture for supporting *handle* authentication and registration. The first is to modify the actual SIP servers by extending their functionality through a server plug-in. This approach requires absolutely no changes to the current User Agent devices whether hardphones or softphones. The devices will adapt seamlessly to the

system. Alternatively, a second approach would be to modify the User Agent devices instead, which is a more cumbersome task that would require software upgrades for all existing User Agents.

We have implemented the first approach that deals with extending the functionality of the proxy/registrar servers. We present the proposed solution in light of the reference example of Figure 5.1. In Figure 5.3, the roaming user *2118/r_user* has granted both SIP proxy servers *2118/hproxy* and *10.200/fproxy* administrative rights over his *handle*. Note that the HS_VLIST could refer to another *handle* containing a list of globally trusted servers. For the roaming user *r_user* present in the foreign domain *fdomain*, the authentication/registration process with the foreign proxy server *10.200/fproxy*, depicted in Figure 5.2.C and Figure 5.1 (proposed traffic flow, arrows a,b,c,d), proceeds as follows:

1. *r_user*, after including the *handle 10.200/fproxy* in his *handle* HS_VLIST field, sends a REGISTER request to *fproxy*.

2. *fproxy* challenges *r_user* to authenticate himself.

3. *r_user* uses same digest authentication with username as the *handle 2118~r_user* and password as the value of the SIP_PWD field that he created in his *handle* as shown in Figure 5.3.

4. *fproxy* uses the Handle Protocol [46] to resolve the *handle 2118/r_user* into the SIP_PWD field. The server then computes a message digest over the obtained credentials.

5. Authentication is granted if the two digests match.

6. After authenticating *2118/r_user*, the foreign proxy *fproxy* proceeds to create an internal account for *r_user* to be able to use the SIP services on *fproxy*.

The internal user account will have a username identical to the *handle* of the registering user with the '~' replaced by '.' i.e *2118.r_ user* in this case.

7. Registration of the user follows. This requires that *fproxy* modifies the *handle 2118/r_ user* updating the field SIP_URL to point to the internal account, *2118.r_ user@x.y.z.w* in this case, as shown in Figure 5.3. This means that *r_ user* is currently associated with *fproxy*.

Obviously, our modified authentication algorithm is domain independent. In other words, the user's credentials are valid for all realms provided the correct administrative privileges are set in the Handle System. This property is essential, as it allows a particular authenticated SIP message to traverse multiple domains instead of requiring re-authentication for each domain on the path of the message. Since all communication between the Proxy and the Handle System is secure [46], the proxy can be reasonably certain that the roaming user is indeed who he claims to be by validating his credentials against the secure *handle*. Internally, the proxy server monitors the user accounts created and removes an account (also updating the *handle*) due to unregister requests or account expiration. A sample *handle* for the foreign proxy is shown in Figure 5.4. Devices, whether hardphones and softphones are treated

handle	Field Type:index	Value
10.200/fproxy	HS_ADMIN:100	rwr:0.NA/10.200:300
	HS_ADMIN:101	rwr:10.200/fproxy:300
	INET_HOST:240	x.y.z.w
	HS_PUBKEY:300	00BE00445……..

Figure 5.4: Sample proxy *handle* structure

similarly. This depends on the ability of the device owner to present the SIP proxy with a username (could be the *handle*) and password for authentication.

With this approach, a user does not need to register with a home proxy server as would otherwise be required by pre-call mobility [39]. After registering with the foreign server, the user's handle-to-URI mapping remains fresh allowing correspondent users to reach him simply by addressing his *handle* as we show in section 5.2.4.

5.2.4 Routing

After abstracting any domain binding from users and allowing seamless authentication and registration with local proxy servers, the next step is to permit the user to initiate and receive calls by addressing a particular *handle* with no explicit reference to domain bindings (URIs). In this sense, a SIP user can INVITE any other SIP user provided he knows the latter's *handle*. From a user's perspective, all other users seem to belong to one local domain and abstraction is complete. To explain how call routing is achieved, we go through the steps where an arbitrary SIP user *c_ user* (caller) tries to INVITE the roaming user *r_ user* (callee) using the latter's *handle 2118/r_ user* as shown in Figure 5.1. The call routing process, presented in Figure 5.2.C, proceeds as follows:

1. Caller *c_ user* sends an INVITE request to *r_ user*. The invite request reaches the caller's SIP proxy/registrar containing the following header fields:

   ```
   INVITE sip:2118~r_user@somedomain SIP/2.0
   To:<sip:2118~r_user@somedomain> .........
   ```

   Note: In this message, the domain *somedomain* is irrelevant to our approach. We are only concerned with the *handle* part of the Request-URI. To distinguish between *handle* and non-*handle* requests, we resort to the '/' character[4] in the host name.

[4]Since Internet hostnames can not contain the '/' character [19] ascii (0x2F) (essential

2. Proxy checks if the *handle 2118~r_user* is a locally registered user. If not, the server resolves the *handle* into the SIP_URL field which is *2118.r_user@x.y.z.w* in this case as shown in Figure 5.3.

3. The server then rewrites the target URI of the message to the resolved URI.

4. From this point on, the natural SIP call flow is leveraged and the traditional SIP architecture [37] is utilized for efficient call routing. Note that other proxy servers on the call path treat the request as a normal request i.e. no *handle* resolution is required.

Again, with our approach, correspondent users trying to communicate with the mobile user need not go through a home proxy for session setup or redirection. This renders the call route more efficient eliminating unnecessary overhead and significant round-trip times.

One last point worth mentioning is the ability of a user to register with multiple servers from different devices simultaneously using the same *handle*. In our implementation, the *SIP_URL* field of a particular *handle* can contain a list of bindings (URIs) to enable this attractive property. Exploiting this property is left for future work.

5.3 Implementation

5.3.1 Test-bed

We have implemented the aforementioned functionality as an extension to two open source SIP servers, the JAIN-SIP Proxy [6] and the SIP Express Router (SER) [36].

character in the *handle* Namespace [45]), we replaced it with the '~' ascii (0x7E) character in the examples above for implementation purposes. We also allow the '#' ascii (0x23) character for compatibility with hard IP phones.

The JAIN-SIP proxy server is an open source JAVA-based SIP proxy built on top of the JAIN-SIP-1.1 API. SER is an open source, configurable SIP server that is widely deployed in the research community. We have implemented a JAVA based H-SIP API that can be easily called from both proxy servers to expose the H-SIP interface operations. The operations mainly enable inter-domain authentication, registration and call routing using *handle*s. All results depicted hereafter are based on the JAIN-SIP proxy.

Our test-bed is a realization of the framework depicted in Figure 5.1. We are running three modified SIP servers on three separate domains:

1. *ece.unm.edu* located at the University of New Mexico, Albuquerque, New Mexico. [Server IP: 129.24.24.106]

2. *cnri.reston.va.us* located in Reston, Virginia. [Server IP: 132.151.9.104]

3. *istec.org* located in Panama. [Server IP: 168.77.202.59]

A roaming user is allowed to move across the domains while establishing connectivity within each domain using the respective local server. The three servers are running Fedora Core 4 kernels and are identical in terms of work load and processing speed (AMD Athlon 1.1 GHz processors).

To use the framework, users are expected to be able to manage their own *handle*s. The Handle System provides a free administration tool [5] for this purpose. This tool is currently implemented in JAVA.

Currently, users wishing to associate with a proxy server are required to specify the latter's IP address or domain name. Since our approach exploits *handle*s instead of domain names, we have implemented a specialized gateway that translates between DNS and *handle* protocols as discussed in section 4.3.1.

5.3.2 Experimental Model

In this section, we will compare the performance of roaming and non-roaming environments in terms of registration and call establishment delays. The non-roaming case represents traditional SIP signaling between a user and his home SIP server irrespective of roaming while the former case represents our proposed approach. Both cases were tested under the same conditions regarding UA/SIP processor speed and work loads. As UA, we used the Java SIP Communicator [9]. We also tested with the Cisco 7940/7960 IP phones.

Registration

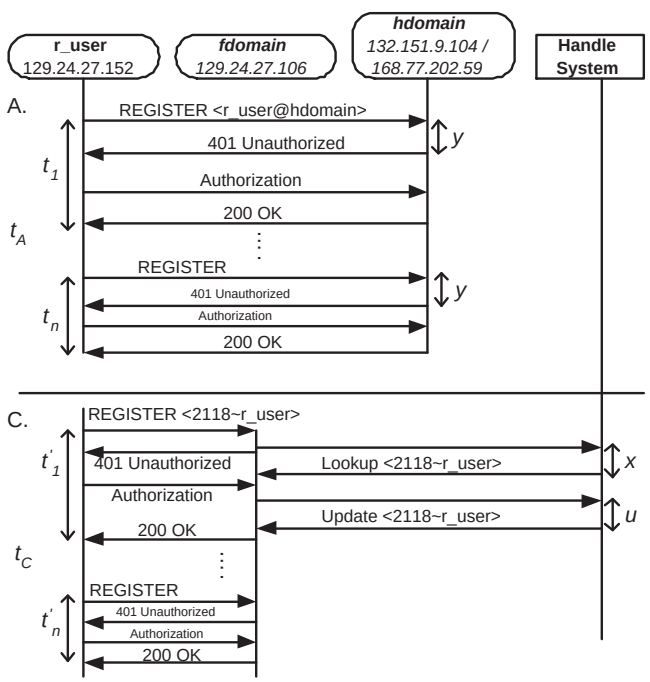

Figure 5.5: REGISTER message flow A. With no roaming logic C. With proposed roaming logic H-SIP

Figure 5.5 is a magnified image of Figure 5.2 that focuses on the registration

process according to the test-bed. We show a comparison of the registration process for roaming (H-SIP) and non roaming (no H-SIP) environments. *r_ user* is located in *fdomain*, *ece.unm.edu*. For the no H-SIP case, we tested with *hdomain* being either *cnri.reston.va.us* (Reston, Virginia) or *istec.org* (Panama). Again, registration here assumes the basic digest authentication with the proxy server. In our model, the UA refreshes its registration with the proxy/registrar continuously (we used a registration TTL of 1 minute).

If we estimate the server's average processing time of the REGISTER request including digest authentication by α , then from Figure 5.5.A, the average registration time t_A as seen by *r_ user* is given by,

$$t_A = \frac{1}{n} \sum_{1 \leq i \leq n} t_i = t_1 \tag{5.1a}$$

since,

$$t_1 = t_2 = \ldots = t_n \approx \alpha + 2y \tag{5.1b}$$

where t_i is the time consumed by the i^{th} REGISTER, and y is the average round-trip communication delay between *r_ user* and the registering SIP server. Obviously, t_A includes a relatively long communication delay as a result of the presumably large geographical separation between the roaming user and the home SIP server.

Now, if we consider H-SIP, then from Figure 5.5.C, the average registration time t_C as seen by *r_ user* is given by,

$$t_C = \frac{1}{n} \sum_{1 \leq i \leq n} t_i^{'} \tag{5.2a}$$

and,

$$t_1^{'} \approx \alpha + x + u \tag{5.2b}$$

$$t_2^{'} = \ldots = t_n^{'} \approx \alpha \tag{5.2c}$$

where $t_i^{'}$ is time to perform the i^{th} REGISTER, x is the average *handle* resolution delay and u is the average *handle* update delay. First, we note that the round-trip delay y is negligible in this case due to the existence of r_user and the SIP server on the same local network. In addition, note here that the server will issue one *handle* resolution and one update for the first REGISTER request only. Then, the first *handle* resolution is always cached internally on the server. Unless r_user moves to another domain or un-registers, no *handle* resolution/update is further required. This means that subsequent REGISTER requests will read the cached value and thus, for $i > 1$, $x = u = 0$. This also means that x and u can be discarded for n sufficiently large.

Here, t_A and t_C are measured by r_user as the average time between sending the REGISTER request and receiving the 200 OK response. In Figure 5.5, r_user performs a DNS lookup or a Handle-DNS lookup in scenarios A and C respectively. These lookups are excluded from the performance metrics i.e. t_1 and t_2 do not include the DNS lookups for the SIP servers. However, we closely examine and compare the two lookup times and we show the performance results in section 5.3.3.

Call establishment

Figure 5.6 is a magnified image of Figure 5.2 that focuses on the call establishment process according to the test-bed, where c_user tries to establish a call with the roaming user r_user. For this section, *cdomain* is *ece.unm.edu* [New Mexico]. The *hdomain* is *istec.org* [Panama] and the *fdomain* is *cnri.reston.va.us* [Virginia]. For brevity, the 100 TRYING messages are excluded from Figure 5.6. We focus on the INVITE message flow and the servers require no authentication.

We denote by $rt_{(m,n)}$ the round-trip communication delay between nodes m and n. For example, in Figure 5.6, $rt_{(cdomain,hdomain)}$ is the round-trip delay between

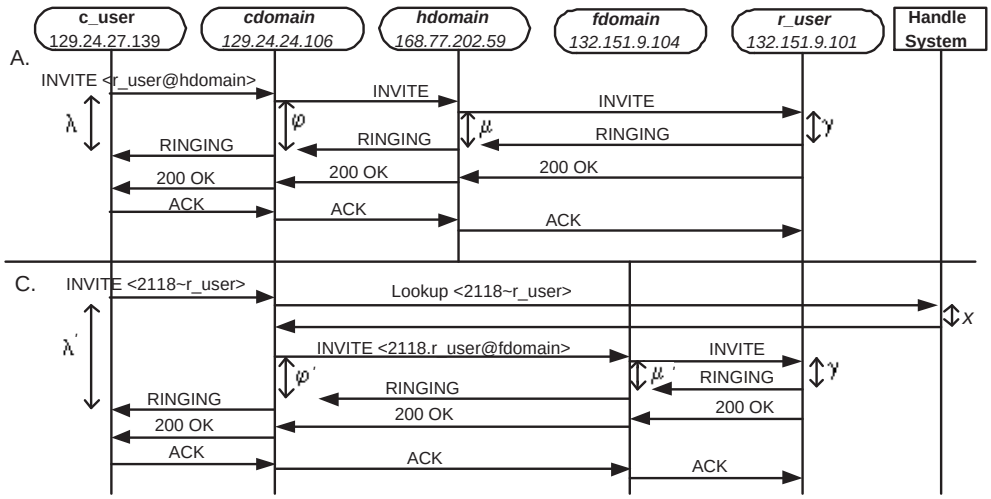

Figure 5.6: INVITE message flow A. With no roaming logic C. With proposed roaming logic H-SIP

the *cdomain* SIP server and that of *hdomain*. If we estimate the server's average processing time of the INVITE request by β, and the UA's average processing time of the INVITE request by γ, then again from Figure 5.6.A,

$$\lambda \approx rt_{(c\_user,cdomain)} + \beta + \phi \tag{5.3a}$$

$$\phi \approx rt_{(cdomain,hdomain)} + \beta + \mu \tag{5.3b}$$

$$\mu \approx rt_{(hdomain,r\_user)} + \gamma \tag{5.3c}$$

and therefore,

$$\lambda \approx rt_{(c\_user,cdomain)} + rt_{(cdomain,hdomain)} + rt_{(hdomain,r\_user)} + 2\beta + \gamma \tag{5.3d}$$

where λ is the average call establishment time (the INVITE/RINGING round-trip) as seen by c_user, ϕ is the average INVITE/RINGING round-trip time between the *cdomain* and *hdomain* SIP servers and μ is the average INVITE/RINGING round-trip time between the *hdomain* SIP server and r_user.

Now, if we consider H-SIP, then from Figure 5.6.C,

$$\lambda^{'} \approx rt_{(c\_user,cdomain)} + \beta + x + \phi^{'} \tag{5.4a}$$

$$\phi^{'} \approx rt_{(cdomain,fdomain)} + \beta + \mu^{'} \tag{5.4b}$$

$$\mu^{'} \approx rt_{(fdomain,r\_user)} + \gamma \tag{5.4c}$$

and therefore,

$$\lambda^{'} \approx rt_{(c\_user,cdomain)} + rt_{(cdomain,fdomain)} + rt_{(fdomain,r\_user)} + 2\beta + x + \gamma \tag{5.4d}$$

where $\lambda^{'}$ is again the average call establishment time as seen by c_user, $\phi^{'}$ is the average INVITE/RINGING round-trip time between the *cdomain* and *fdomain* SIP servers, $\mu^{'}$ is the average INVITE/RINGING round-trip communication time between the *fdomain* SIP server and r_user, and x is the average *handle* resolution delay. Note here that each INVITE request will require the server to issue one fresh *handle* resolution.
Both λ and $\lambda^{'}$ are measured by c_user as the time between sending the INVITE request and receiving the RINGING response.

5.3.3 Performance Measurements

The measurements of the registration times, call establishment times and communication delays in this section were all averaged from 10,000 samples dispersed over a 10 day period i.e. $n = 1000$ samples a day.

Registration

Examining equations (5.1a) and (5.2a), we deduce that $t_C \leq t_A$ for sufficiently large n. In the general case of a roaming user, the round-trip delay $2y$ is expensive.

Table 5.1: Average Registration delays with and without H-SIP (as shown in Fig. 5.5), $n = 1000$, transport=UDP

		Registration Delays [ms]			
		$x,\ u$	y	t_A	t_C
No H-SIP	hdomain: Virginia	NA	86	209	NA
	hdomain: Panama	NA	116	275	NA
H-SIP		84, 260	0	NA	40

Our real-time measurements are listed in table 5.1. Clearly our measurements show that equations (5.1a) and (5.2a) hold for an $\alpha \approx 39ms$. We see here that t_A is approximately $5t_C$ or $7t_C$ for the Virginia and Panama cases respectively. Obviously, the H-SIP approach outperforms the traditional approach as long as y is significant, which is often the case for roaming subscribers. The value of α directly depends on the implementation of the SIP server which is the JAIN-SIP Proxy server [6] in this case. Besides, x is random and it directly depends on the location of the Local Handle Server (LHS) [44] storing the particular *handle*.

Call Establishment

In comparing the call establishment time given by equations (5.3d) and (5.4d), we are interested in computing the performance enhancement or degradation $\Delta\lambda = \lambda - \lambda^{'}$. We consider the following variables to verify our model:

$$\tau_1 = rt_{(hdomain,r\_user)} - rt_{(fdomain,r\_user)} \gg 0 \tag{5.5a}$$

$$\tau_2 = rt_{(cdomain,hdomain)} - rt_{(cdomain,fdomain)} \tag{5.5b}$$

$$\sigma = \tau_1 + \tau_2 > 0 \tag{5.5c}$$

where σ is the difference in the cumulative round-trip delay between the no H-SIP case and the H-SIP case respectively. Equation (5.5a) is true in general due to the

Table 5.2: (a) Comparison of average call setup delays (as shown in Fig. 5.6) (b) Average Round-trip communication delays , transport=UDP

Call Setup Delays [ms]		
$\Delta\lambda$	$\Delta\phi$	$\Delta\mu$
88	168	106

(A)

Round-trip Delays [ms]		
τ_1	τ_2	x
111	47	85

(B)

presumably large geographical separation between the roaming user and his home server versus using a local server. We also argue that equation (5.5c) holds in general based on our model i.e. the cumulative round-trip delays for *c_ user* to reach *r_ user* is smaller in the H-SIP scenario. However, τ_2 in 5.5b can be positive or negative since it directly depends on the *cdomain-hdomain* separation versus that of *cdomain-fdomain*. For our particular setup, $\tau_2 > 0$.

It follows from equations (5.3) and (5.4) that,

$$\Delta\mu = \mu - \mu^{'} \approx \tau_1 \tag{5.6a}$$

$$\Delta\phi = \phi - \phi^{'} \approx \tau_2 + \Delta\mu \tag{5.6b}$$

$$\Delta\lambda \approx \Delta\phi - x \tag{5.6c}$$

Notice here that,

$$\Delta\lambda \approx \tau_1 + \tau_2 - x = \sigma - x \tag{5.6d}$$

Consequently, H-SIP outperforms the traditional SIP approach, in general, as long as $\sigma > x$. Our conducted real-time measurements are listed in Table 5.2. In Table 5.2, we verify the validity of our model by separately comparing the real-time call setup measurements to the round-trip delays, thus verifying equations (5.6).

DNS vs Handle-DNS resolution

Obviously, the only performance degradation introduced by our approach is the *handle* resolution overhead. This overhead has been extensively measured by the

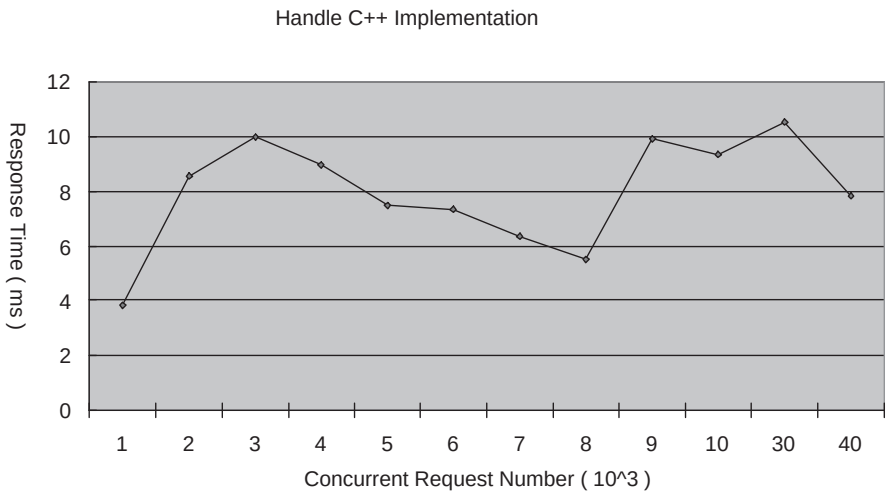

Figure 5.7: *Handle* implementation performance measurement as of August 2005. Acquired through the courtesy of Mr. Sam Sun and CN-NIC.

corporation for national research initiatives (CNRI) and its partners and is illustrated in Figure 5.7. The current performance of the system that oscillates from 3 to 10 ms makes it comparable to the Bind implementation of the DNS protocol. Additionally, due to its intrinsic fully distributed administration and resolution, it avoids the pitfalls that plage the current DNS implementation where DNS resolution times can extend to over 100 milliseconds [21]. Overall robustness of the handle-DNS implementation has also been extensively tested along with load assessment. The results shown in Figure 5.8, are very encouraging and show the Handle System as a potential replacement of the DNS system in general. On top of addressing the domain resolution itself, our approach minimizes the signaling traffic needed by a roaming user to join the SIP infrastructure and be ready to initiate calls. The user is efficiently utilizing the services of a local server with no need for per-call coordination with his home server. If the home server is located in another continent, the

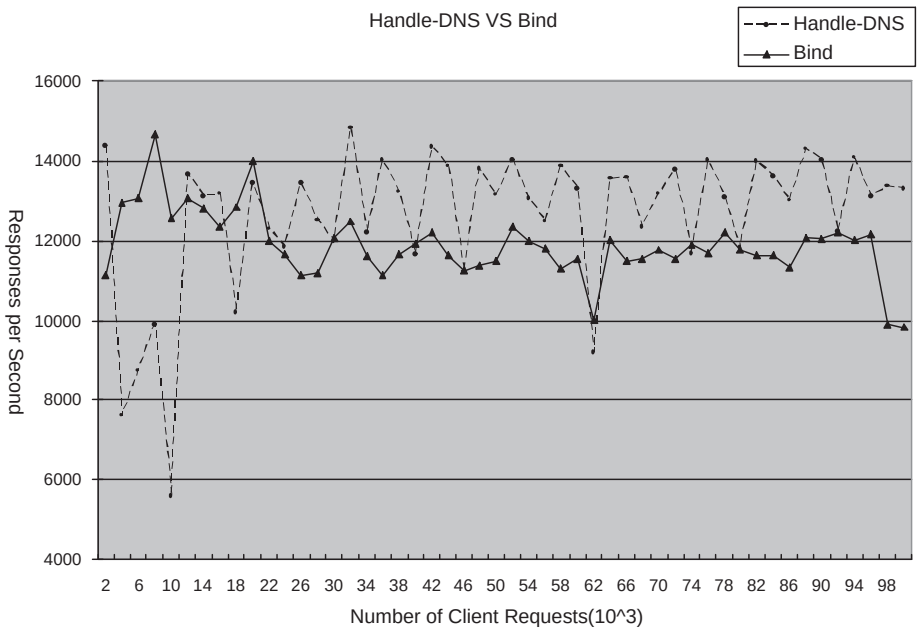

Figure 5.8: *Handle* load performance measurement as of August 2005. Acquired through the courtesy of Mr. Sam Sun and CN-NIC.

round trip times for registration/redirect messages from the roaming user to home proxy/correspondent user become significant. The proposed approach optimizes the association, authentication, and call routing times for the roaming user by guaranteeing that the mobile user will always be addressed through the nearest available server in his vicinity.

5.4 Future Work

This application has assumed the use of globally unique *handle*s to identify SIP users. It is also possible to use a set of aliases of such *handle*s that can in turn translate into *handle*s. Such a service would be administered by yet another service provider using the *handle* protocol to administer and coordinate the general use of aliases in

the system. This may be done using registry-like features for particular *handle*s, and allows the provider to service users that opt into such services. Since *handle*s can also point to other *handle*s, certain intermediate *handle*s may be provided to systems that choose to run private *handle* servers in a way similar to a private branch exchange (PBX). The advantage of having a distributed resolution infrastructure that is also domain independent, translates into users being able to run smaller SIP servers that communicate with the Handle System to expedite routing. We may even envision a cellular-like behavior for SIP systems in which users are able to use the resources of many smaller SIP servers along a user's roaming path. These ideas, as well as possible ways to expedite routing and *handle* resolution, are possible future research paths. One can think of the current implementation of the Handle System as a global Location Registry that has a single point of failure. Part of our current research is focused on implementing a structured peer-to-peer form of the Handle System to expedite lookups and resolutions, and to eliminate single points of failure.

5.5 Conclusion

In this chapter, we outlined the use of an indirection architecture based on the Handle System to address SIP inter-domain mobility. Our approach not only enables roaming controlled by the users rather than organizations, but also provides a faster implementation than traditional currently deployed approaches. Through this work, users will be able to dynamically enable their own mobility and benefit from the advantages of a secure distributed persistent identifier network. By disassociating users from DNS domains, while still providing the means to interact with traditional SIP systems, we provide a scalable interchangeable enhancement to the SIP infrastructure.

Chapter 6

Conclusions & Future Work

We have reviewed in this thesis the basic concepts of a mobile transient network and its components. This heterogeneous network architecture is based on persistently identified abstractions which map, at any point in time, to their current implementation, location, and other attributes required for network communication. Each resource on the network, including network end-points, network components, users, applications, and backbone building blocks, is abstracted as a digital object and assigned a unique identifier within a distributed persistent identification system. It is that identification system which provides the mapping of an abstraction to the current state of the resource. The transient network architecture begins by assuming a mobile world with transient devices communicating across a fully distributed environment. This characterization not only enables a native answer to the requirements of these devices, which represent the fastest growing set of components for the Internet, but also provides a flexible structure to welcome future devices, services, and users.

We have demonstrated an application implementation of a transient network architecture. We have established the usefulness and applicability of the Persistent

identification network PIN to identify and locate network devices and services. Our demonstrations used the current implementation of the Handle System. We have also proved the notions of mobility and persistence in communication across heterogeneous networks using the indirection framework. Our design of host mobility is still under development, and the problems of simultaneous mobility, connection migration, and the resulting security threats need to be addressed.

Additionally, we have outlined the use of an indirection architecture based on the Handle System to address SIP inter-domain mobility. Our approach not only enables roaming controlled by the users rather than organizations, but also provides a faster implementation than currently deployed approaches. Throughout our work, users are able to dynamically enable their own mobility and benefit from the advantages of a secure distributed persistent identifier network. By disassociating users from DNS domains, while still providing the means to interact with traditional SIP systems, we provide a scalable enhancement to the SIP infrastructure.

Possible ways to expedite routing and *handle* resolution are future research paths. The current implementation of the Handle System has a global Location Registry that with a single point of failure. Part of our current research is addresses the implementation of a structured peer-to-peer form of the Handle System to expedite lookups and resolutions and eliminate single points of failure.

References

[1] Bluetooth technology. http://www.bluetooth.org.

[2] The common object request broker: Architecture and specification. http://www.omg.org/library/c2indx.html.

[3] Dois for research content. http://www.crossref.org.

[4] H.323 : Packet-based multimedia communications systems. http://www.itu.int/rec/T-REC-H.323-200307-I/en.

[5] The handle system. http://www.handle.net.

[6] Jain-sip proxy (built on jain-sip 1.1 api). https://jain-sip-presence-proxy.dev.java.net/.

[7] Labview® software from national instruments. http://www.ni.com/labview/.

[8] Location independent networking for ipv6. www.lin6.net.

[9] Sip-communicator 1.0. https://sip-communicator.dev.java.net/.

[10] Hari Balakrishnan, Karthik Lakshminarayanan, Sylvia Ratnasamy, Scott Shenker, Ion Stoica, and Michael Walfish. A layered naming architecture for the internet. In *Proceedings of ACM SIGCOMM 2004*, pages 343–352, Portland, Oregon, USA, August 2004. ACM.

[11] G. Ballintijn, M. van Steen, and A.S. Tanenbaum. Scalable human-friendly resource names. *Internet Computing*, 5(5):20–27, Sep 2001.

[12] Nilanjan Banerjee, Sajal K. Das, and Arup Acharya. SIP-based mobility architecture for next generation wireless networks. In *PerCom*, pages 181–190. IEEE Computer Society, 2005.

[13] T. Berners-Lee, R. Fielding, and L. Masinter. RFC 2396:uniform resource identifiers (URI): Generic syntax, 1998.

[14] I. Castineyra, N. Chiappa, and M. Steenstrup. RFC 1992: The nimrod routing architecture, August 1996.

[15] Ashutosh Dutta, Faramak Vakil, Jyh cheng Chen, Miriam Tauil, Shinichi Baba, Nobuyasu Nakajima, and Henning Schulzrinne. Application layer mobility management scheme for wireless internet, April 15 2001.

[16] Eriksson, Faloutsos, and Krishnamurthy. Peernet: Pushing peer-to-peer down the stack. In *International Workshop on Peer-to-Peer Systems (IPTPS), LNCS*, volume 2, 2003.

[17] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP authentication: Basic and digest access authentication. RFC 2617, June 1999.

[18] Mark Gritter and David R. Cheriton. An architecture for content routing support in the internet. In *USITS'01: Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems*, pages 4–4, Berkeley, CA, USA, 2001. USENIX Association.

[19] K. Harrenstien, M. K. Stahl, and E. J. Feinler. RFC 952: DoD Internet host table specification, October 1985.

[20] Chen Hongtao, Yang Fangchun, and Xu Peng. Analysis on sip mobility of double user agent servers. In *Communications and Information Technology*, volume 1, pages 87–90. ISCIT, IEEE, October 2005.

[21] C. Huitema and S. Weerahandi. Internet measurements: the rising tide and the dns snag. In *Proceedings of the 13th ITC Specialist Seminar on IP Traffic Measurement Modeling and Management*, IPseminar, Monterrey, CA, USA, September 18-20 2000. ITC.

[22] Ravi Jain, Thomas Raleigh, Charles Graff, Michael Bereschinsky, and Mitesh Patel. Mobile internet access and qos guarantees using mobile ip and rsvp with location registers. volume 3, pages 1690 – 1695. ICC International Conference on Communications, June 1998.

[23] Ravi Jain, Thomas Raleigh, Danny Yang, Li-Fung Chang, Charles Graff, Michael Bereschinsky, and Mitesh Patel. Enhancing survivability of mobile internet access using mobile IP with location registers. In *INFOCOM*, pages 3–11, 1999.

[24] H. Jerez and J. Khoury. Open device access protocol (odap)- working whitepaper. http://hdl.handle.net/4263537/5023, May 2006.

[25] Henry Jerez, Joud Khoury, and Chaouki Abdallah. A mobile transient network architecture, 2006. Pre-print available at https://dspace.istec.org/handle/1812/55.

[26] D. Johnson, C. Perkins, and J. Arkko. RFC 3775: Mobility support in ipv6, June 2004.

[27] Robert Kahn and Robert Wilensky. A framework for distributed digital object services. Internet Whitepaper http://www.cnri.reston.va.us/k-w.html, January 1995.

[28] S. Kent and R. Atkinson. RFC 2401: Security architecture for the internet protocol, November 1998.

[29] J. Khoury, H. Jerez, N. Nehme, and C. Abdallah. An application of the mobile transient network architecture: Ip mobility and inter-operability. 2006. pre-print available at http://hdl.handle.net/2118/jk_transapp_06.

[30] P. Mockapetris. RFC 1035: Domain names implementation and specification, November 1987.

[31] R. Moskowitz, P. Nikander, and P. Jokela. Host identity protocol architecture. RFC 4423, May 2006.

[32] P. Nikander, J. Arkko, and B. Ohlman. Host identity indirection infrastructure (hi3). In *The Second Swedish National Computer Networking Workshop*, November 2004.

[33] R. Pandya. Emerging mobile and personal communication systems. *IEEE Communications Magazine*, 33:44–52, June 1995.

[34] Charles E. Perkins. RFC 3220: Ip mobility support for ipv4, January 2002.

[35] Ram Ramanathan. RFC 2103: Nimrod mobility support, February 1997.

[36] Y. Rebahi, D. Sisalem, J. Kuthan, A. Pelinescu-Onicicul, B. Iancu, J. Janak, and D. Mierla. The sip express router, an open source sip platform. http://www.iptel.org/ser.

[37] J. Rosenberg, H. Schulzrinne, and etal. RFC 3261: Session initiation protocol, June 2002.

[38] J. H. Saltzer, D. P. Reed, and D. D. CLark. End-to-end arguments in system design. *ACM TOCS*, 2(4):277–288, November 1984.

[39] Henning Schulzrinne and Elin Wedlund. Application-layer mobility using sip. *SIGMOBILE Mob. Comput. Commun. Rev.*, 4(3):47–57, 2000.

[40] M. Smith, M. Bass, G. McClellan, and R. Tansley. Dspace: An open source dynamic digital repository. *D-LIB Magazine*, 9(1), January 2003.

[41] Alex C. Snoeren and Hari Balakrishnan. An end-to-end approach to host mobility. In *Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, August 2000.

[42] Ion Stoica, Daniel Adkins, Shelley Zhuang, Scott Shenker, and Sonesh Surana. Internet indirection infrastructure. *IEEE/ACM Transactions on Networking*, 12(2):205–218, April 2004.

[43] S. Sun. Establishing persistent identity using the handle system. Tenth International World Wide Web Conference, May 2001.

[44] S. Sun, L. Lannom, and B.Boesch. Handle system overview. RFC 3650, November 2003.

[45] S. Sun, L. Lannom, and B. Boesch. Handle system namespace and service definition. RFC 3651, November 2003.

[46] S. Sun, S. Reilly, L. Lannom, and J. Petrone. Handle system protocol (ver2.1) specification. RFC 3652, November 2003.

[47] Michael Walfish and Hari Balakrishnan. Untangling the web from DNS. In *NSDI*, pages 225–238. USENIX, 2004.

[48] Elin Wedlund and Henning Schulzrinne. Mobility support using sip. In *WOWMOM '99: Proceedings of the 2nd ACM international workshop on Wireless mobile multimedia*, pages 76–82, New York, NY, USA, 1999. ACM Press.

[49] K. Wong, Ashutosh Dutta, Henning Schulzrinne, and Ken Young. Simultaneous mobility: analytical framework, theorems, and solutions. *Wireless Communication and Mobile Computing*, June 2006.

[50] K.D. Wong, A. Dutta, J. Burns, R. Jain, K. Young, and H. Schulzrinne. A multilayered mobility management scheme for auto-configured wireless ip networks. *Wireless Communications*, 10(5):62–69, October 2003.