ENRICHING COMMUNICATION METHODS FOR COMPOSABLE MOBILE SYSTEMS

BY TAM N. VU

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Marco Gruteser and Dipankar Raychaudhuri

and approved by

____Jennifer Rexford

____Richard Martin

____Marco Gruteser

______________________________________Dipankar Raychaudhuri

New Brunswick, New Jersey

October, 2013

ABSTRACT OF THE DISSERTATION

Enriching Communication Methods for Composable Mobile Systems

by Tam N. Vu

Dissertation Directors: Marco Gruteser and Dipankar Raychaudhuri

We are now surrounded by a variety of devices, simple and complex, and embedded in an infrastructure of computing, sensing and communication that spans across the globe. This ever-large computing, communication and sensing ecosystem provides a unique opportunity to dynamically and opportunistically compose logical mobile systems from the best set of wireless components available locally and globally. Such composable mobile systems allow users to easily and seamlessly extend the capability of their device, overcoming the basic design limitations of traditional monolithic mobile devices in terms of screen size, weight, processing power, sensing power, and battery life. However, existing communication methods and the current Internet architecture, designed in the era of large and trustworthy stationary computers, fail to meet the demands of the emerging composable computing era. In this work, we focus on developing new communication methods between devices in the vicinity and designing a new network architecture to facilitate seamless communication at global scale for composable computing.

To enable secure and convenient communication between physically co-located devices, we designed and evaluated a technique that allows a device to transmit information through the screen of touchscreen-enabled devices, called *capacitive touch communication (CTC)*. The key idea is to exploit the pervasive capacitive touchscreen as a receiver for a bitstring to be transmitted. At the network layer, we revisited two conerstones of the Internet architecture to provide better support for composable computing: naming and routing. We proposed an in-network scalable name resolution service, called DMap that lays the foundation for a fast and global-scale name resolution service necessary to provide seamless connectivity between billions of network-connected objects. To provide reliable and efficient inter-network connectivity, especially in the presence of high mobility, we propose a clean-slate edge-aware inter-domain routing protocol, called EIR. The protocol provides enhanced information about network topology and edge network properties in order to enable networks across the Internet to make better routing decisions than currently possible with BGP. This is accomplished with a telescopic network state dissemination protocol which makes the entire network graph visible while keeping the routing overhead within limits.

Preliminary designs and implementations of CTC, DMap and EIR demonstrated their features and benefits towards composable computing. Our evaluation showed that CTC is potentially a secure and convenient communication method for touchscreen-enabled devices, albeit at low bit rates. DMap evenly balances storage load across the global network while achieving lookup latency of $\sim$100 ms, considered adequate for support of dynamic mobility across the global Internet. The EIR interdomain routing protocol provides good performance in highly dynamic environments with frequent migration of clients across network domains.

Acknowledgements

I would like to thank my two advisers Marco Gruteser and Dipankar Raychaudhuri for leading me along the way to this point and educating me on both academic and practical research. Being their student is one of the luckiest experiences in my life. Marco has taught me how to find research topics from real world problems, how to look at problems from different perspectives, how to always keep the bar high and come up with the best possible solutions, and how to deliver research results to different audiences effectively. I am grateful to his constant guidance and invaluable intellectual insight, and thankful to him for giving me just enough rope to satisfy my ambitions to explore disruptive ideas without managing to hang myself. Ray has always been very encouraging and has let me to choose my own path for my research. His invaluable experience in industrial research and passion on building real systems with real world impact served as a model for me to look up to and strive after. I can't thank him enough for his constant advice on various aspects of a Ph.D. student's life, both inside and outside academia.–Marco and Ray, you are the best advisers any Ph.D. student could hope to have. You will always have my best wishes and deepest gratitude.

I am grateful to Thu Nguyen for serving as my academic adviser, my qualifying committee member, and collaborator. Thu has been a continuous source of support and encouragement during my time at Rutgers. I also wish to thank Richard Martin for always making himself available and agreeing to help me with just anything I have asked for; from being my CS adviser for my first first year at WINLAB, serving in my dissertation committee, to collaborating with me on many of my research topics.

I have learnt a lot from my summer internship in Princeton University. Many thanks to Jennifer Rexford and Mike J. Freedman who spent much of their time helping me to define a research problem, to find solutions, and to teach me how to write my very first research paper. Special thank to Jennifer Rexford for her inspiring discussions and thoughtful comments on my work on GNRS and EIR, which is part of this thesis. I also thank her for serving as my dissertation committee member and giving me great feedback.

The design and implementation of CTC, GNRS and EIR were collaborative efforts from many collaborators inside and outside WINLAB. I would like to acknowledge the fine work of individuals who have contributed to this dissertation. Special thanks to Akash Baid who worked closely with me in designing and presenting our work on GNRS. He generously donated his time and expertise on presenting CTC and EIR to help me meet publication deadlines. Hai Nguyen helped with implementation of EIR on Click router and conducted experiments on Orbit testbed.

I have been very lucky to have opportunity to work with and learn from Richard Howard. I would like to thank him for being so generous to me with his time and expertise. He is a source of technical knowledge and inspiration with his infectious enthusiasm and an incredible outlook on life. I also wish to thank many collaborators, colleagues and friends, who have influenced my views and efforts during my time here at Rutgers, including Ivan Seskar, Jeff Walling, Predrag Spasojevic, Wade Trape, Roy Yates, Yanyong Zhang, Kiran Nagaraja, Samuel Nelson, Junichiro Fukuyamaz, Yingying Chen, Janne Lindqvist, Ashwin Ashok, Chenren Xu, Sangho Oh, Baik Hoh, Sanjit Kaul, Gayathri Chandrasekaran, Kien Le, Thang Le, and Tuan Phan. I would also like to thank all my wonderful friends inside and outside Rutgers University for bringing happiness and wisdom to my life.

Finally, I would like to thank my parents, Chieu Vu and Bac Nguyen, and my brother, Thanh Vu, for their love and support during these years. My parents have made countless sacrifices for me, and have provided me with steady guidance and encouragement. I know that they miss me very much and would rather have me home, but they still encourage me to follow my own path. This dissertation is dedicated to them.

Table of Contents

Abstract . . . ii

Acknowledgements . . . iv

1. Introduction . . . 1

1.1. Composable Mobile Systems and Their Communication Challenges . . . 2

1.2. Existing Literature on Composable Mobile Systems . . . 5

1.3. Identifier-based Internet Architecture for Highly Dynamic Environments . . . 6

1.4. Research Areas and Contributions . . . 8

2. Capacitive Touch Communication . . . 11

2.1. Introduction . . . 12

2.2. Background . . . 14

2.2.1. Capacitive Touchscreen Technology . . . 14

2.2.2. Related Work . . . 15

2.3. Capacitive Touch Communication . . . 18

2.3.1. Creating Artificial Touch Events . . . 19

2.3.2. Communication System Overview . . . 20

2.4. Decoder design . . . 23

2.4.1. Determination of Expected Number of Events for $ones$ and $zeros$ . . . 25

2.4.2. Minimum Distance Demodulation . . . 28

2.4.3. Ring Detection for Indirect Communication . . . 31

2.5. Exploring the parameter space . . . 33

2.5.1. Triggering Touch Events . . . 34

2.5.2. Bitrate vs. Detection Rate Trade-off . . . 37

2.5.3. Indirect Communication Results . . . 38

2.6. Ring Prototype and Evaluation 39
2.6.1. Hardware Prototype 40
2.6.2. Preliminary Prototype Performance 41
2.7. Discussion 43
2.8. Conclusion 46

3. Global Naming Resolution Service 48
3.1. Introduction 48
3.2. Background and Motivation 52
3.2.1. Identifier and Locator Separation 52
3.2.2. Requirements of Host-Based Identifiers 52
3.2.3. Incentive for Shared Hosting 53
3.3. Direct Mapping (DMap) 54
3.3.1. Overview of DMap 54
3.3.2. Handling Unallocated Network Addresses 56
3.3.3. Spatial Locality and Local Replication 58
3.3.4. Inconsistent GUID→NA Mappings 59
3.4. Evaluation 60
3.4.1. Storage and Traffic Overhead 60
3.4.2. Query Response Time and Load 61
3.4.2.1. Simulation Setup and Input Workloads 61
3.4.2.2. Evaluation Results 63
3.5. Analytical Model for Query Response Time 66
3.5.1. The Jellyfish Model 67
3.5.2. Upper Bound for Query Response Times 67
3.5.3. Analytical Results 69
3.6. Related Work 70
3.7. Concluding Remarks 71

4. Edge-aware Interdomain Routing Protocol 73
4.1. Introduction 74
4.2. Key Principles and Techniques 76
4.2.1. Design Principles 76
4.2.2. Enabling Techniques 78
4.3. EIR Protocol Design 79
4.3.1. Building Blocks 81
4.3.1.1. Aggregated node (aNode) 81
4.3.1.2. Virtual link (vLink) 81
4.3.2. Route Dissemination 82
4.3.3. Policy Handling 84
4.3.4. Route selection 86
4.4. Case Studies 86
4.4.1. Dynamic Mobility Support 87
4.4.2. Cross-domain multi-homing 88
4.4.3. Wireless edge peering 90
4.5. Evaluation 91
4.5.1. Implementation 91
4.5.1.1. Prototype Design 91
4.5.1.2. Evaluation Methodology 92
4.5.1.3. Results 93
4.5.2. Internet-scale Simulation 95
4.5.2.1. Methodology 95
4.5.2.2. Routing update latency and overhead 98
4.6. Concluding Remarks 99

5. Future Directions and Conclusion 100
5.1. Extracting Information from Shifting Contexts 100
5.2. Minimizing User Attention and Effort in Human-System Interactions 101

5.3. Support Mobile Communication at Wireless Edge Networks 102

5.4. Protecting User Privacy and Securing Context Information 102

References 104

Chapter 1

Introduction

For a long time, mobile applications have been designed with strict constraints on the amount of resources that the applications could utilize, such as small-sized screen and keyboard, limited energy, and inadequate local sensing information. As a result, personal mobile computing experience and the variety of applications currently available, though improved compared to the desktop-computing era, remain rather confined. Given that we are now surrounded by a variety of devices, simple and complex, and embedded in an infrastructure of computing, sensing and communication that spans across the globe, a question facing mobile computing community is how to eliminate these resource constraints to enable new applications and services using all of these surrounding facilities.

Fortunately, today's fast-growing computing, communication and sensing ecosystem provide a unique opportunity to dynamically and opportunistically compose advanced mobile services from the best set of wireless components available locally and globally. Such composable mobile systems allow users to easily and seamlessly extend the capability of their device, overcoming the basic design limitations of traditional monolithic mobile device in terms of screen size, weight, processing power, sensing power, battery life and more. However, existing communication methods and the current Internet architecture, designed in the day of large and trustworthy stationary computers, fail to meet the demands of the emerging composable computing era. This dissertation focuses on (i) developing a new communication method between devices in the vicinity and (ii) designing a new network architecture to facilitate seamless communication at the global scale for composable computing.

1.1 Composable Mobile Systems and Their Communication Challenges

A composable mobile system is a logical system that is opportunistically assembled from a set of local and remote discrete elements, and able to easily and seamlessly extend its capability by interacting with and utilizing computing, communications and sensing resources available. Consider the following scenario to illustrate the features and benefits of composable computing:

It is 8 o'clock in the morning and Bill is heading out the door going to work. On the way out, he grabs his matchbox-sized computer and a jacket. The computer is a powerful device with a large storage capacity and equipped with multiple wireless interfaces. It does not have any built-in display or keyboard. Instead, it only has a single LED to indicate that it is on and its wireless connections are properly working. Since the jacket is made of fabric that could harvest energy from light and human movement, it charges the matchbox computer when the box is placed inside the jacket's pocket. Content from the computer wirelessly flows to and appears on the see-through display embedded on Bill's eyeglasses. Bill controls the computer using his voice which is recorded by the microphone-like device embedded on the jacket's top button. He quickly skims through his email inbox while walking to his garage to get into the car.

As Bill enters his car, the computer connects to the audio system of the car via Bluetooth, tunes the radio to his favorite morning news channel and outputs it to the surround sound system inside the car. Meanwhile, the in-pocket computer shows the estimated arrival time given the traffic condition, alerts Bill with slowdowns ahead, and optionally assists him to write an email saying he might be late for a meeting due to bad traffic. At all time, the computer uses the frame stream from the eyeglass-mounted camera to monitor the distance from Bill's car to its front cars to assure a safety distance between the two.

Once Bill arrives at his office, the computer detects the presence of his desktop and laptop computers. It then transfers latest states of all applications and services it is currently running to those devices and then goes into a energy-conserving mode. Bill can now seamlessly continue to work on what he had been working on previously on the matchbox computer, e.g. finish reading a report to prepare for a meeting in 45 minutes.

After 45 minutes, he reaches for a tablet and walks to the conference room for the meeting.

The tablet screen is apparently a better choice for both reading a report and taking notes in a semi-stationary environment such as a meeting compared to the see-through screen and an embedded microphone. Bill flips through a couple of slides using his tablet while listening to a presentation of his colleague. It is now half an hour into the meeting, Bill is going to present his ideas to his group. As he walks up to the speaker spot, the matchbox wakes up and sends Bill's presentation streams to the projector. He starts his presentation as soon as he reaches the dais without having to carry the tablet or to manually connect it to the projector...

Using this scenario, we imply that rather than being a monolithic device with a predetermined set of components manufactured into a single physical device as found in today's mobile devices, the next-generation mobile systems should be opportunistically composed from surrounding discrete elements, and be able to easily and seamlessly extend its capability by interacting with and utilizing computing, communications and sensing resources nearby. That would not only bring computing experience of user to a new high but also help addressing many challenges that mobile computing have to deal with today including energy limitations and limited context sensing information.

However, the abovementioned scenario is not yet practical due to a number of limitations of existing technology. Acquiring complex context information is the first hurdle. In order to seamlessly blend in, to be a part of and finally take advantage of the surrounding environment, mobile devices first must be able to learn the surrounding context and discover nearby elements in the environment. Examples of such context information include the identification of the nearby networked display panels, the speed at which the whole environment is moving, or something as simple as who is interacting with the device itself. While the increasing availability of sensors integrated in mobile devices provides a rich set of context information, not all of it can be directly inferred from those sensors' output. Even when the information is potentially available, direct measurement might not be practical or possible. For example, sensing mobility using a GPS receiver might be either too power hungry or inaccurate due to the environment, e.g being indoors. Addressing this context acquisition challenge is the first step in realizing composable computing. While we have been addressing a few aspects of this challenges [1, 2, 3, 4], it is not the focus of this dissertation.

Once we have an adequate view of our surrounding elements, which is provided by context sensing, the next step in creating these composite systems, applications and services is to bring surrounding computing resources together via wireless communication. The key challenge lies in providing resilient, secured, yet efficient and convenient methods to allow these discrete elements to smoothly and seamlessly interact with each other locally and globally. It is challenging because wireless-enabled devices do not inherently have a link-layer (layer-2) connection with each other across a variety of wireless standards. As a result, it requires either (i) a discovery-like service that uses the layer-3-and-up protocol in addition to layer-2 discovery (beacons or probes) prior to making connection at layer-2 or (ii) a new communication method that is back-ward compatible with off-the-shelf devices today.

Communication method applied in composable mobile systems must be able to operate in highly dynamic setups. Users of the systems tend to quickly switch from one device to another or between environments. In addition, a single physical device or component could be shared among multiple users at different points in time or even simultaneously. For instance, Bill, in the previous example, associates with the matchbox computer when he is highly mobile, with the desktop when he is in his office, and with the tablet when he attends a meeting. Also, the projector is used by Bill for only a short period of time during his presentation and is then utilized by the next presenter. The dynamism caused by device, human and network mobility is even more prevalent. Despite the presence of mobility and dynamism, users expect their devices to "*just work*,", providing convenient, reliable, and responsive services and applications. However, the protocols of the current stationary-oriented Internet is not up to this task. This dissertation takes concrete steps towards addressing these challenges by reexamining the key elements of the Internet architecture in response to the needs of composable mobile systems and their users. After introducing the common theme of an identifier-based internet architecture, which many components of this dissertation are based on, we present in later chapters each component in detail.

1.2 Existing Literature on Composable Mobile Systems

In his influential *Scientific American* article in 1991 Mark Weiser, then a Principle Scientis at Xerox Palo Alto Research Center (PARC), painted his vision of ‘ubiquitous computing’ through the fictional world of ‘Sal’ [5]. For the first time, the notions of computer use ‘beyond the desktop’ and the computers that ‘weave themselves into the fabric of everyday life until they are indistinguishable from it’ were described which spawned a new research arena of ubiquitous computing. As Weiser pointed out in his article, providing a network infrastructure that ties all discrete computing components and applications is one of the key challenges in realizing the vision of ubiquitous computing. Since then, many systems has been proposed to provide framework for a varieties of devices compositions. The iRoom project at Stanford [6] connects various information appliances together to create an interactive workspace using a centralized infrastructure. Another early project from Microsoft, called EasyLiving [7], linked devices in a home environment, focused on sensing people using vision and interference in a indoor infrastructure to dynamically decide which connection should be established. These two projects shares the same constraints of requiring fixed infrastructure composition.

To support connectivity establishment in a more dynamic and ad hoc maner, dynamic composable computing (DCC) frame work [8] was proposed by Roy Want and his colleagues in Intel Research Lab. Rather than trying to solve the problem of establishing *connecting* between users and individual devices, DCC aims to provide users with *composition* capability, which allows users to operate the whole computing systems at a higher level to connect multiple local entities together into one logical operation. However, the frame work was mainly designed for a seamless computing experience between *local* discrete elements. Global composition was not considered which is a focus of this thesis.

There are many other research efforts in connecting a device to other devices within its vicinity. Internet-Suspend-Resume (ISR) [9] project shows how a user’s state can be migrate from a local machine to a remote host. It shows how to maintain an uninterrupted computing experience that mobile users could enjoy by suspending then migrating from one physical machine to another. ABC et. al, with Pebbles project [10], explores how to use a set of hand-held devices in conjunction with a public display to create multi-user collaborative working environment. It allows multiple participants to jointly interact with the shared surface; which serves as an example of what can potentially be done with composable systems in order to utilize physical computing resources more efficiently. However, it does not answer the question of how a flexible platform can support composition, including how device discovery and connection occur.

MobiUS [11], a work from Microsoft research, shows how two mobile PDA devices could wirelessly combine their screens to coorperatively display a video across the two screens. This work focuses the challenges at the application layer – video decoder – rather than generalizing the composition process. Another piece of work from Intel Research on the Personal Server [12] allows devices to project their data on a remote screen, but it does not support building a computer system from an ad hoc set of wireless components.

1.3 Identifier-based Internet Architecture for Highly Dynamic Environments

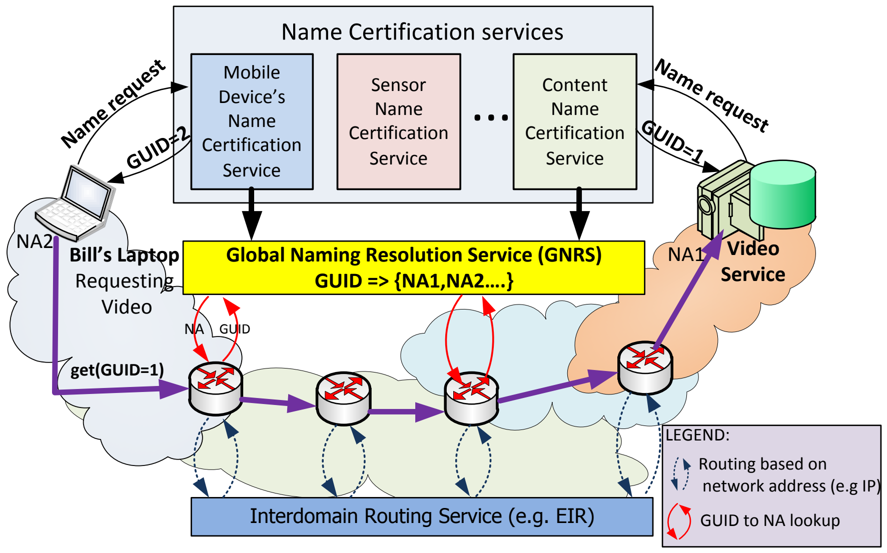

Name Certification services
Mobile Device's Name Certification Service
Sensor Name Certification Service
Content Name Certification Service
Name request
GUID=2
Name request
GUID=1
NA2
Bill's Laptop Requesting Video
Global Naming Resolution Service (GNRS)
GUID => {NA1,NA2….}
NA1
Video Service
NA
GUID
get(GUID=1)
Interdomain Routing Service (e.g. EIR)
LEGEND:
Routing based on network address (e.g IP)
GUID to NA lookup

Figure 1.1: MobilityFirst Architecture Overview

Current approaches for handling mobility involve a combination of cellular network protocols (e.g. 3GPP) [13] and mobile IP [14] but it is widely recognized that these solutions have serious limitations in terms of scale and service flexibility. This has motivated a number of ”clean-slate” future Internet architecture projects aimed at investigating fundamentally new approaches to meeting anticipated needs such as large-scale mobility, security/privacy or content support [15].

The MobilityFirst project [16] represents one of these architectural efforts with a particular focus on supporting large-scale, efficient and robust mobility services in the future Internet. The MobilityFirst architecture is based on a clean separation between the “identifier” of end-users or other network-connected objects, and their routable addresses or “locators”. Separation of names and addresses makes it possible for mobile devices to have a permanent, location independent name or globally unique identifier (GUID) which can then be mapped to a set of routable network addresses (NA) corresponding to the current point(s) of attachment. This concept of separating identifiers from routable addresses or locators has been advocated by a number of authors in the networking community to address mobility [17, 18, 19, 20]. It provides many advantages, including simplified implementation session management, multi-homing, mobility, disconnection, authentication and security. Figure 1.1 shows the layering of functionality in the proposed MobilityFirst architecture.

In this approach, a human-readable name such as ”Bill’s Laptop” is mapped to a GUID through one of many possible application level services, called Name Certification Service, deployed by network providers or independent third-party providers. The GUID is then assigned to the mobile device (or other network-connected object) and entered into the network-level GNRS service shown in the figure. The GNRS is a distributed in-network service which is responsible for maintaining the current bindings between the GUID and network address(es) (NA’s). Mobile devices (or routers at their point of attachments) update the GNRS with current NA values resulting in a table entry such as <GUID: NA1, NA2, NA3, optional properties>. During the communication between endpoints, network addresses (NA) are used for routing purposes.

A technical problem we address here is that of *realizing a scalable GNRS service with ∼100 Billion GUID entries (i.e. network-attached objects) with lookup latencies fast enough to support anticipated mobility speeds and application usage patterns*. We emphasize that since the GNRS can also be queried by the in-network routers for dynamic GUID:NA resolution for in-transit packets, low latency in the query response procedure is a critical requirement for the design.

The MobilityFirst project has a particular emphasis in designing both intra- and inter-domain routing protocols to efficiently support mobility requirements at the edge. *In recognition of the fact that wireless access and mobility at the edge also have implications for routing between networks and some of the resulting requirements cannot be met by current BGP solutions, we propose an Edge-aware Interdomain Routing (EIR) protocol*. Our objective for this work is to explore and understand Internet scale routing mechanisms in view of emerging mobility services and use the results to influence evolving standards rather than to change BGP in a single step.

In order to address the problem of mapping user identifier to its routable NA in the cases where the user frequently switches between devices, we propose a solution that is based on a combination of a new communication technique, that is backward compatible with off-the-shelf capacitive touchscreen-enabled devices – called Capacitive Touch Communication, and the Global Name Resolution Service (GNRS).

1.4 Research Areas and Contributions

Enriching communication methods for composable mobile systems provides opportunities for research in many broad fields. This section provides a brief overview of three specific pieces of work in this dissertation.

Managing user identifier through capacitive touch communication [21]: In this work, we present a capacitive communication method through which a device can recognize who is interacting with it. This method exploits the capacitive touchscreens, which are now used in laptops, phones, and tablets, as a signal receiver. The signal that identifies the user can be generated by a small transmitter embedded into a ring, watch, or other artifact carried on the human body. We explore two example system designs with a low-power continuous transmitter that communicates through the skin and a signet ring that needs to be touched to the screen. Experiments with our prototype transmitter and tablet receiver show that capacitive communication through a touchscreen is possible, even without hardware or firmware modifications on a receiver. This latter approach imposes severe limitations on the data rate, but the rate is sufficient for differentiating users in multiplayer tablet games or parental control applications. Controlled experiments with a signal generator also indicate that future designs may be able to achieve datarates that are useful for providing less obtrusive authentication with similar assurance as PIN codes or swipe patterns commonly used on smartphones today.

Global Naming Resolution Service with DMap [22]: This work presents the design and evaluation of a novel distributed shared hosting approach, DMap, for managing dynamic identifier to locator mappings in the global Internet. DMap is the foundation for a fast global name resolution service necessary to enable emerging Internet services such as seamless mobility support, content delivery and cloud computing, and composable mobile systems. Our approach distributes identifier to locator mappings among Autonomous Systems (ASs) by directly applying $K>1$ consistent hash functions on the identifier to produce network addresses of the AS gateway routers at which the mapping will be stored. This direct mapping technique leverages the reachability information of the underlying routing mechanism that is already available at the network layer, and achieves low lookup latencies through a single overlay hop without additional maintenance overheads. The proposed DMap technique is described in detail and specific design problems such as address space fragmentation, reducing latency through replication, taking advantage of spatial locality, as well as coping with inconsistent entries are addressed. Evaluation results are presented from a large-scale discrete event simulation of the Internet with $\sim$26,000 ASs using real-world traffic traces from the DIMES repository. The results show that the proposed method evenly balances storage load across the global network while achieving lookup latencies with a mean value of $\sim$50 ms and 95^{th} percentile value of $\sim$100 ms, considered adequate for support of dynamic mobility across the global Internet.

Edge-aware Interdomain Routing Protocol with EIR [23]: This component of the MobilityFirst architecture is a clean-slate inter-domain routing protocol designed to meet the needs of the future mobile Internet. In particular, we propose EIR (edge-aware inter-domain routing) as a general solution for a range of mobility-related requirements including device and network migration, cross-domain end-host multi-homing, global roaming agreement setup and wireless edge peering. The EIR protocol provides enhanced information about network topology and edge network properties in order to enable networks across the Internet to make better routing decisions than currently possible with BGP. This is accomplished with a telescopic network state dissemination protocol which makes the entire network graph visible while keeping the routing overhead within limits. EIR enables autonomous systems to optionally expose internal network topology and aggregate properties such as bandwidth, availability and variability, thus enabling corresponding networks to select paths which take into account both service requirements (such as dual-homing or multicast/anycast) and edge network constraints (e.g. LTE vs. WiFi). Further, EIR is designed to work in conjunction with late binding of names to addresses and in-network storage in order to provide robust services in environments with dynamic mobility and disconnection. The design of the EIR is given along with sample use cases (i.e. host mobility, multi-homing, multicast and edge peering) to further explain the benefits of the protocol. This is followed by a two-stage evaluation of EIR including an Internet scale simulation model to verify scalability, and a 200-node experimental ORBIT emulation to validate the protocol design and provide experimental results for selected usage scenarios.

The remainder of this dissertation is organized as follows: Chapter 2 presents CTC, the novel communication method to manage user identifiers. Chapter 3 describes DMap, the globally distributed and scalable naming resolution service. Chapter 4 shows EIR, the interdomain routing protocol that satisfies a range of mobility-related requirement in composable mobile systems. Chapter 5 concludes the dissertation with future research directions.

Chapter 2

Capacitive Touch Communication

As we move into the era of composable computing, we increasingly rely on a variety of devices. We tend to quickly switch between them and temporarily share them with others. Each of these devices should quickly and accurately recognize the current user not only for authentication and identification but also to personalize the services and information it provides. The device should then follow a simple protocol to update the global naming resolution service (GNRS) with the new UserID:deviceID mapping so that others, who wish to communicate with the user, know to which device they should designate the communication channel to. For example, if Alice wishes to call Bill for streaming voice or video, it would be more convenient for her to be able to use human-readable, short and simple name as "Bill" instead of having to manually find out which device Bill currently associated with, then look up for the device's NA, and lastly open a socket connection between her device and the network address of Bill's device. However, existing technologies, at best, address these needs through slow, error prone, and cumbersome procedures. Bill would need to first authenticate himself to devices every time he switches from one to another. He then has to actively update a name resolution service elsewhere about his association with the current device. We explored a new form of "wireless" communications, that we termed *capacitive touch communications* that allows touchscreen-enabled devices to quickly and unobtrusively identify and authenticate their users. The key idea is to exploit the pervasive capacitive touchscreen and touchpad input devices as receivers for an identification bitstring transmitted by a hardware identification token worn by users. The token transmits electrical signals as it makes direct contact with the screen, or indirect contact through human skin. Since this communication technique has never been explored before in literature, realizing it takes paramount efforts. The first challenge comes from the fact that hardware and firmware of the touch devices are proprietary; hence we do not have the exact understanding of the internal electronic and circuitry of the devices. As the result, getting the basic understanding of capacitive touch technologies is a long experimental process. We then faced challenges that are common in wireless communications, such as bit and frame synchronizations, and having a noisy channel. However, traditional techniques in wireless communication cannot be directly applied to address these challenges due to the physical properties of the system including limited symbol rate, or uncontrolled parasitic capacitance. Instead, we devised our own calibration and demodulation algorithms to address theses challenges. We tested the system on an off-the-self device and showed the feasibility of capacitive touch communication. While the current data rate is limited, it is enough to allow new way of managing user identifiers and enable new classes of applications such as parental control, multi-users games, and weak authentications.

2.1 Introduction

Mobile devices now provide us ubiquitous access to a vast array of media content and digital services. They can access our emails and personal photos, open our cars [24] or our garage doors [25], pay bills and transfer funds between our bank accounts, order merchandise, as well as control our homes [26]. Arguably, they now provide the de-facto single-sign on access to all our content and services, which has proven so elusive on the web.

As we increasingly rely on a variety of such devices, we tend to quickly switch between them and temporarily share them with others [27]. We may let our children play games on our smartphones or share a tablet with colleagues or family members. Sometimes a device may be used by several persons simultaneously, as when playing a multi-player game on a tablet, and occasionally, a device might fall into the hands of strangers.

In all these situations, it would be of great benefit for the device to know who is interacting with it and occasionally to authenticate the user. We may want to limit access to age-appropriate games and media for our children or prevent them from charging our credit card.[1] We desire to hide sensitive personal information from strangers, colleagues, or perhaps even a curious

[1]Apple is facing a law suit over children's in-app credit card purchases [28].

spouse [29, 27]. Or, we may simply want to enjoy an enhanced user experience from the multi-player game that can tell who touched the screen.

Unfortunately, user identification and authentication mechanisms available on today's mobile devices have been largely adopted from PC software and have not followed the versatility of the usage and sharing possibilities. For example, several mobile devices (e.g. iPad or iOS devices) do allow to restrict access to device functions, but the devices do not provide any easy way to quickly change, let alone authenticate users. They provide PIN codes, passwords, for authentication, and a number of other techniques have been proposed by researchers [30]. Yet they remain cumbersome and very few people enable these security features on their phones.

In this work, we will explore a form of "wireless" communication, that we term *capacitive touch communication* to address this issue. The key idea is to exploit the pervasive capacitive touch screen and touchpad input devices as receivers for an identification code transmitted by a hardware identification token. While the token can take many forms, we consider here an example realization as a ring, inspired by the signet rings used since ancient times. The token transmits electrical signals on contact with the screen, either direct contact or indirect contact through the human skin.

The major contributions of this work are as follows:

- Painting a vision to use the near-ubiquitous capacitative touch sensors to distinguish and possibly authenticate users.
- Introducing and exploring the concept of capacitive touch communication as one mechanism to distinguish users.
- Showing how the output of an off-the-shelf touchscreen system can be affected by electrical signals generated in a token that is in contact with the screen. We also show how such signals can be transmitted through the human skin.
- Designing and implementing a prototype transmitter in the form of a signet ring and receiver software for communicating short codes through an off-the-shelf capacitative touch screen

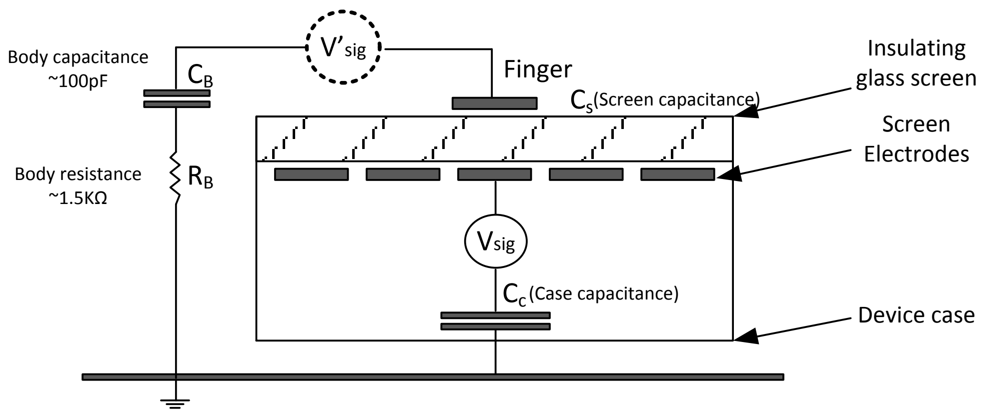


Figure 2.1: Schematic of a basic capacitive touchscreen

2.2 Background

Touchscreen technology was first developed in the 1960's for air traffic control systems [31] and is now a popular user interface technology on devices ranging from ATMs and self-service terminals in grocery stores or airports, to cars, smartphones, and tablets. Even the touchpads used in laptops are based on similar technology. These products employ different touchscreen implementations, including analog resistive, surface capacitive, projected capacitive, surface acoustic wave, infrared and optical technology to mention a few. On mobile devices, however, capacitive touchscreens have emerged as the main technology and we focus our work on those.

2.2.1 Capacitive Touchscreen Technology

A capacitive screen in most commercial tablets and smart phones consists of an array of conducting electrodes behind a transparent, insulating glass layer which detects a touch by measuring the additional capacitance of a human body in the circuit. Figure 2.1 shows a schematic of one possible realization of such a system [32]. When a user touches the screen, her finger acts as the second electrode in a capacitor with the screen as the dielectric. The touchscreen electrodes are driven by an AC signal (V_{sig}) which sends a current through the screen capacitance C_s passing through the body capacitance C_B, and then back into the tablet through the case capacitance C_c. This change in voltage measured at one or more screen electrodes is then passed to the screen controller for processing. Because all of the relevant capacitance values

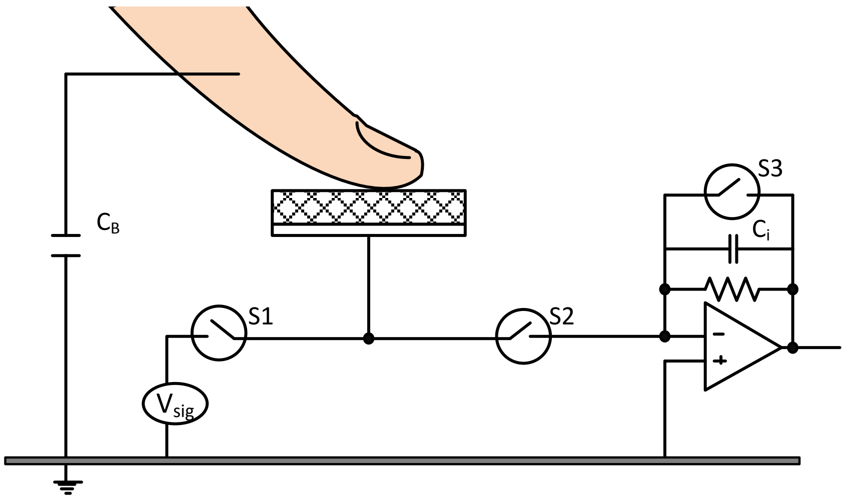


Figure 2.2: Internal touch detection circuit

are small (hundreds of picofarads [33]) environmental noise makes direct measurement of this current impractical. Instead, the charge integration circuitry in Figure 2.2 is used to measure the excess capacitance associated with a finger touch. In this case, a digital signal, V_{sig}, is synchronized with a pair of switches and a charge integrator. Switch S_3 is first closed to discharge capacitor C_i and then opened. Next, switch S_1 is closed and S_2 opened while V_{sig} is high. This charges the series combination of the C_B, C_c, and C_s. Then S_1 is opened and S_2 closed, transferring this charge to C_i. This is commonly known as sample and hold operation. After a fixed number of cycles, the voltage on C_i is directly proportional to the ratio between C_i and the series combination of C_B, C_c, and C_s. This voltage is then used to detect touch and, through the matrix addressing of the electrodes, position of the touch. Hence, even when a finger moved across the screen surface without lifting it, the finger triggers this detection at different positions on the electrode array.

2.2.2 Related Work

The most closely related projects to our work are Touché [34], DiamondTouch [35], Signet [36], IR Ring [37], Magkey/Mickey [38]. Proposed in 2001 as one of the first efforts toward differentiating touches of different users interacting with the same surface, DiamondTouch uses a physical table to transmit capacitively coupled signals through users, chairs, and finally to the receiver. Along the line of using human body as a medium of communication Braun *et* *al.* [39] proposed a technique to detect the presence of human body using capacitive proximity sensing. However, these approaches require extensive hardware infrastructure which makes it impossible to apply to mobile scenarios. Our work seeks for solutions that do not require any modification or addition of hardware to existing touchscreen-enabled devices.

Touché proposes a technique, called Swept Frequency Capacitive Sensing, that can recognize human hand and body configurations. While the technique could enable a new way of human computer interaction, it would require additional special hardware component to be manufactured onto the devices. Signet[36] uses physical patterns of conductive material as unique inputs for authentication through a capacitive touch screen. In contrast, our work focuses on using arbitrary programmable sequences of bits through direct use of the user's fingers. As such, it makes the solution non-intrusive and applicable to wider classes of applications.

There are several ways to authenticate a user, which in general can be divided into 1) what you know, 2) what you have, and 3) who you are. PINs, passwords and swipe patterns are the most widely spread authentication mechanism for mobile phones [30, 40]. These methods are easy to implement and require no special hardware, but are easily observable by an adversary and usually have very low information entropy. For example the usual 4 bit numeric PINs used in most phones have a theoretical potential entropy of $\log_2(10^4) = 13.3$ bits. Practical entropy for 4-digit PINs is likely to be much lower, as is the case with passwords [41]. The second type of authentication mechanisms ("what you have") are often also referred to as authentication tokens, examples include Magkey/Mickey [38], RFID or other wireless tokens such as transient authentication [42], and IR Ring [37]. Magkey and Mickey are tokens that use magnetic fields and acoustic signals that are received by the phone's compass and microphone respectively. RFID, NFC and other wireless-based techniques are prone to eavesdropping and suffer from interference among multiple radio signal sources. One example technique belonging to that category is RingBow [43], a wearable hardware token in the form of a ring, which communicates with the mobile device using Bluetooth. This type of communication is insecure during the pairing period and does not allow touchscreen-enabled devices to associate touch events to their users. And finally, IR Ring demonstrated the possibility to use infra-red and IR video cameras to authenticate users on a multitouch display, which is not directly applicable to today's mobile devices due to its additional hardware requirement.

Examples of “who you are” include iris recognition, face recognition and voice recognition all of which are being actively prototyped and tested on mobile devices. Motorola Atrix claims to be the first phone in the western market to have a fingerprint sensor [44] while Sony is developing a novel finger-vein pattern matching technique [45]. Both these techniques require specialized hardware which adds to the cost and form-factor of handheld devices and are prone to known vulnerabilities [46, 47]. On the other hand, face, iris and voice recognition utilizes the in-built sensors and most of the feature set required are already implemented in mobile devices for other applications [48, 49]. While these techniques can leverage the abundance of past research in the respective fields, they also suffer from the well known spoofing mechanisms [50, 51]. For example both high-quality photograph of the eye and printed contact lenses have been used to achieve close to 100% spoof acceptance rates for iris recognition systems [52].[2] Similar results hold for face detection and voice detection although large strides are also being made for spoof detection in biometric authentication systems (see Jain et al. [54] and the references therein). More recently, innovative uses of the various sensors available in most smart phones have led to a number of *unconventional techniques*. For example, there are proposals [55, 56] for in-air gesture based authentication mechanism which uses the accelerometer sensors of the mobile device. Being easily visible to an adversary, such a scheme suffers from an unpleasant tradeoff between coming up with complex gestures and being susceptible to copy attacks, and can also be socially awkward. Implicit authentication is a similar approach which aims to authenticate mobile users based on everyday actions such as number/duration of calls, location, connectivity pattern, etc. and keeps a multi-variable continuous authentication score of the user. As obvious as it is, this requires a continuous modelling and logging of data from a variety of sensors and has a high energy cost.

Today’s consumer electronic devices often include some form of parental control mechanisms, which are usually limited to locking out some functionalities of the device or service, e.g. adult content. Parental control mechanisms are an overlooked area of research, however, recent studies indicate that there would be demand for flexible access control mechanisms at home [57]. Our presented work can be seen as an easy to use enabler for parental access control

[2]The face recognition system available in the new Google Android based Galaxy Nexus platform can be compromised just by showing a picture taken with another smartphone [53].

mechanisms.

The problem of device pairing is also closely related to secure authentication and solution approaches often overlap. The general objective in this case is to enable two devices with no prior context to securely associate with each other in the presence of man-in-the-middle adversary. The short-range and frequency hopping nature of Bluetooth makes it a robust authentication mechanism, however several recent works expose a key vulnerability - passive sniffing of the PIN during the pairing process [58]. Similarly, for near-field communications (NFC) [59] based pairing, eavesdropping using directional antennas has been shown to be a critical security threat [60]. Novel use of the accelerometer sensor in mobile devices has recently been shown to provide a secure method of device pairing [61]. While robust for two equipped mobile devices, the requirement of shaking prevents its use from cases which require pairing of a mobile device with a fixed device. Further, replication of the movement by an adversary is possible through careful observation of the pairing process. Finally, a recent approach uses public RF signals such as TV and FM broadcasts to derive cryptographic keys for secure pairing between close-by devices [62].

Auxiliary channels to establish shared secrets have been studied extensively in the domain of secure pairing since the resurrecting duckling model [63]. Examples include using infrared [64] or humans [65]. More recently secure pairing efforts have focused on using the same channel for authentication and data, and deriving the keying material based on the local environment, e.g [62]. In contrast, our approach provides a seamless way to both securely pair the device and authenticate later.

2.3 Capacitive Touch Communication

To allow mobile devices to identify their users in a less obtrusive manner, we explore a novel form of "wireless" communication in which a touch panel acts as a receiver and a small ring-like device worn by the user serves as the transmitter. This type of communication, which we term *capacitive touch communication*, could have wide applicability since touch panels are now ubiquitous.

While it would be interesting to also consider modifications to the touch sensor hardware

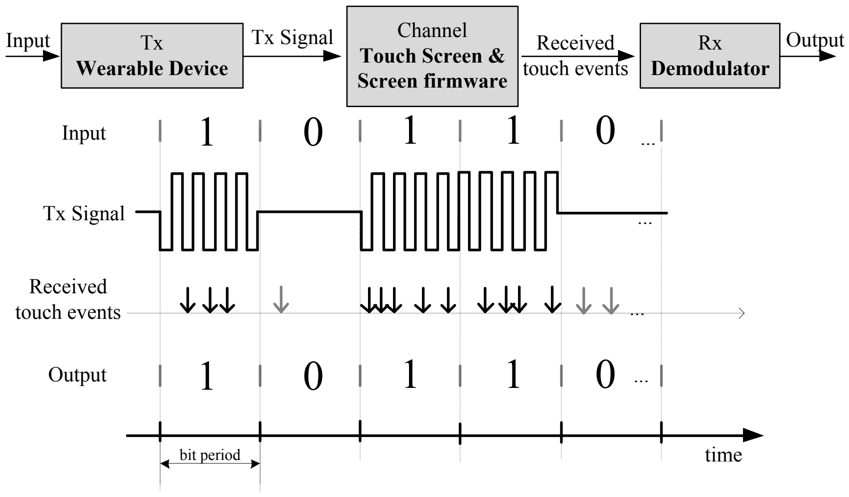


Figure 2.3: Capacitive touch screen communication showing OOK modulation and variations in number and timing of generated events

and firmware to facilitate such communication, we focus this first study on exploring to what extent the communication can be achieved with off-the-shelf touch sensor systems. This means we will only have access to the touch events exported by the screen's driver, not the raw voltages measurements. It imposes very stringent requirements on the communication protocols, as we will see in the next sections. We believe, however, that this is a useful point solution within the design space of capacitive touch communication, since this approach would allow more rapid deployment on existing devices.

2.3.1 Creating Artificial Touch Events

Motivated by this goal, we discovered a technique for "spoofing" the screen detection algorithm by causing the system to alternately register touch/no touch conditions even when the finger is not moving. This allows us to send a digital signal into the touchscreen.

Referring again to Figure 2.1, one possible method for artificially creating touch events is by injecting a synchronized signal (V'_{sig}) into the circuit with the proper amplitude and phase to increase or decrease the charge integrated on C_i. Unfortunately, the signal in the device, V_{sig}, is not available to the external user, so such synchronization would be extremely difficult. As such, we use an unsynchronized lower frequency signal of high amplitude which charges and discharges C_i asynchronously, leading to repetitive, but irregular, touch/no touch events

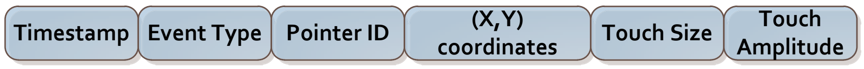


Figure 2.4: Touch event structure retrieved by touch screen controller

captured by the touchscreen controller. This process essentially "spoofs" the touch detection mechanism by injecting high level repetitive signals and introduces a technique to send a low bit rate signal into the tablet. With precise knowledge of the proprietary touch sensor systems it should be possible to create much more fine-grained signaling methods. For the purpose of this feasibility study, however, we will now consider how this coarse technique can be leveraged for designing a user identification system.

2.3.2 Communication System Overview

The communication scheme we are proposing can be modeled as a classical communication system with a transmitter, a receiver and a complex channel connecting the two, as shown in Figure 2.3.

Transmitter: The transmitter in our system is a wearable battery-powered hardware token. One possible form that such a token could take is that of a ring, essentially a digital version of the signet rings carried by nobility in earlier times[3]. While many other forms of tokens are possible, we will use the *ring* concept as a running example throughout this chapter.

The ring would contain a small flash memory that stores a bit sequence or a message, which could be a user identifier or a secret key that authenticates a user. It also has a simple processor that reads the bit sequence and generates an On-Off keying (OOK) [66] modulated signal. That is, bit one is represented by turning *on* a carrier signal; and bit $zero$ by switching *off* that carrier signal, as the *Tx Signal* shown in Figure 2.3. When the ring is pressed against the screen, it acts as a voltage source(V'_{sig} in Figure 2.1) which creates a set of touch events with timestamps following the bit sequence being transmitted.

Channel: Since the events generated follow the bit sequence being transmitted, these events can be used to reconstruct the original bit sequence, which is unknown to the screen otherwise.

[3]A finger ring bearing a hard-to-fake engraved pattern, which serves as a seal of authority, a signet.

Thus, in this setting, the channel can be thought of as the combination of all hardware and software components that affect the relationship between the transmitted bit sequence and the events registered: (i) the series of capacitances, (ii) the firmware that comes with the screen, and (iii) the proprietary driver that is a part of the device's operating system.

Unfortunately, due to the internal switching frequency inside the touch panel, non-deterministic amount of charge accumulation and the firmware/driver artifacts, the number and the timing of the events do not directly follow the input sequence. For example, in Figure 2.3, when the first bit one is transmitted, three touch events are triggered, while in the succeeding $ones$ five and four events are produced. Furthermore, even though transmission of a $zero$ should not trigger touch events, one and two events are registered in the two $zeros$ presented in this example respectively. In addition, the channel adds a variable and unknown delay between the transmitted sequence and the touch event registered.

Receiver: The *Tx Signal* transmitted by the ring generates touch events represented by the 6-tuple structure depicted in Figure 2.4 (a detailed description of this structure is presented in Section 2.5). Because the only information we can use is the timestamps of the events registered by the screen driver, the system requires an unconventional receiver design. Instead of the usual practice of looking at the amplitude (*Touch Amplitude* field) of the received signal, which in this case is not related to the transmitted data, we use the *number of events registered* for demodulating. That is, the software component receives a bit 1 if the number of events which appeared in that bit period is greater than a certain threshold and receives a bit 0 otherwise.

We note that there is a variable delay from the moment that touch events were registered to the kernel until it is handed to the application-level software, which in our case is the application-level demodulator. This delay makes demodulating less accurate. The time variance, we suspect, is due to the queueing and processing delays incurred when the event information travels up the software stack, from the touch-event handler in the Android kernel to the application level. To mitigate this inaccuracy, our demodulator looks at the touch event timestamps at the kernel level (using a few *printk* commands in the touchscreen driver of our prototype).

The key challenge is to handle the variance in the number and timing of the events that is introduced by the channel. To address this issue, we characterize the expected behaviour of the

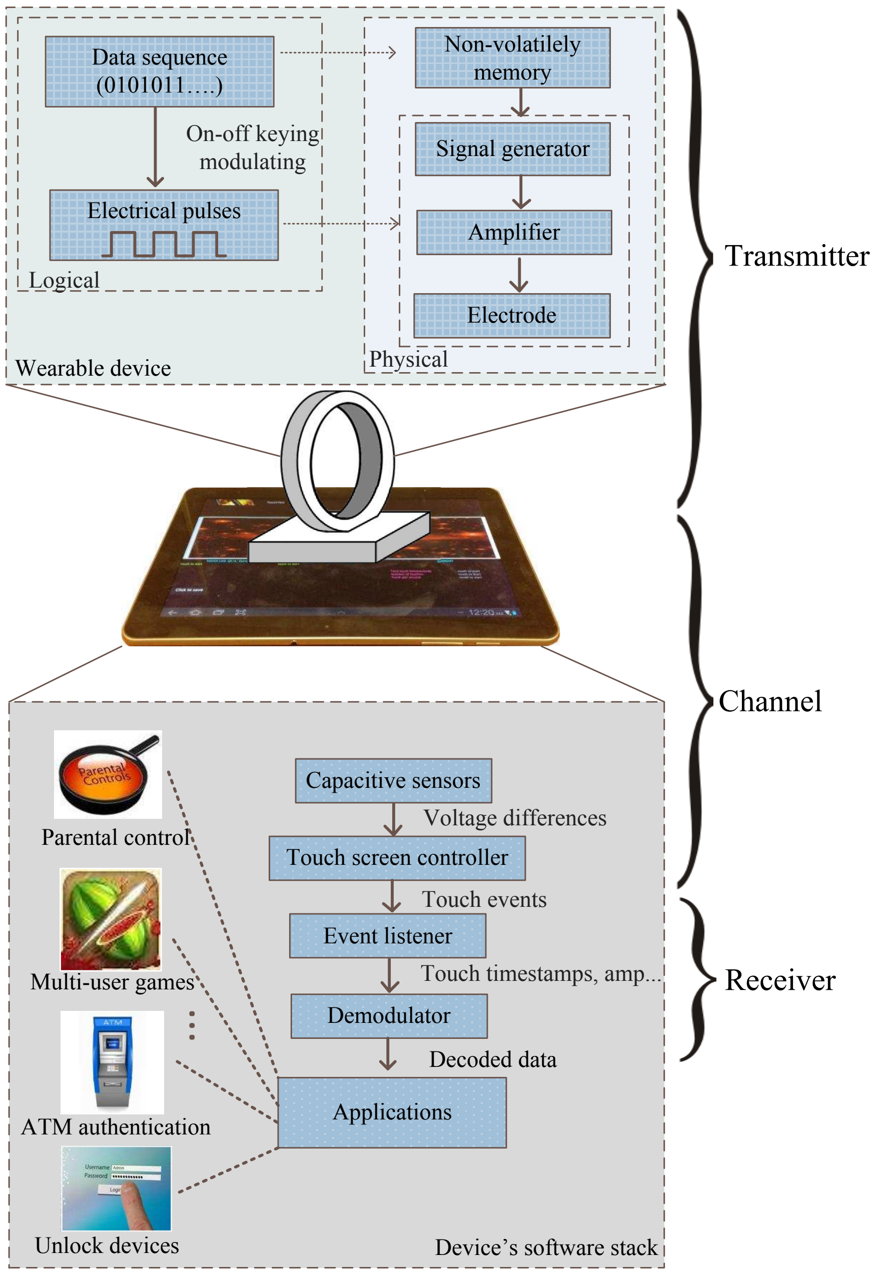


Figure 2.5: Overall architecture of the capacitive touch communication system

channel, reflected in terms of event counts, for decoding of the received sequence, as described in Section 2.4.1. Specifically, we apply a joint decoding-synchronization technique that uses a threshold-based and distance-based method to simultaneously synchronize and decode the received sequence.

Figure 2.5 shows the high level architecture of the system and how components interact.

Note that since we do not have access to the touchscreen controller, the touch events, not the underlying physical voltage differences, are the input to the receiver.

Indirect Communication: Even without direct contact with the screen, the ring can communicate with the touchscreen device as long as the ring bearing finger is in contact with the screen. In particular, the electrical pulses that are transmitted through a human finger's skin from the ring create the same effect of changing the screen capacitance to register artificial touch events. However, we found that due to the skin resistance, the number of events generated through this type of indirect contact is only enough for detecting the presence of the ring, but not stable and regular enough for reliably decoding the data being transmitted. We can leverage this capability of the communication system to enable a novel technique to differentiate two users simultaneously interacting with the same touchscreen, for example in a shared-screen two player game. The detection algorithm used for this mode of communication is described in Section 2.4.3.

2.4 Decoder design

The proposed capacitive touch communication system allows users to send messages to the application layer of the device. This unconventional use of the touchscreen, especially under the constraint of using commercial off-the-shelf devices without lower layer access, poses a number of challenges:

1. We observed that the receiver responds differently to the same input following a different bit pattern; this could be due either to the physical layer or the software that is optimized for detecting touch events from a human finger. For example, the number of events registered to the screen when bit 1 is sent after a long sequence of 0s is different from that of a bit 1 that comes after a sequence of 1s. The normal solution is to code the data to avoid this pattern dependent effect. Rather than adopting a typical bit-by-bit decoding solution, our data rate is already so limited that we developed our own code optimized specifically for the observed pattern dependence.

2. There is a variable delay between the transmission of a symbol and its reception at the

Algorithm 1: Threshold selection algorithm

```
input  : $E_{discrete}$ - Event sequence in time domain
         $TxBitSeq$ - Original transmitted bit sequence
         $BitRate$ - Transmission bit rate (bps)
output: $1e$ and $0e$ - Expected number of events in $ones$ and $zeros$
1  bitPeriod $\leftarrow \frac{1000}{BitRate}$
2  oneC $\leftarrow 0$ // Event counter for all ones
3  zeroC $\leftarrow 0$ // Event counter for all zeros
4  //Convert discrete events to event vector in time series
5  for $i = 1 \rightarrow max(E_{discrete})$ do
6      if exist $E_{discrete}[j] == i$ then $E_t[i]$= 1
7      else $E_t[i]$ = 0
8  //Find the starting position that gives the max $1e0e$ Ratio
9  for startPos =1 $\rightarrow$bitPeriod do
10     for j =1 $\rightarrow$ length($TxBitSeq$) do
11         eCount = sum($E_t[\text{startPos} + (\text{j} - 1) * \text{bitPeriod}, \text{startPos} + \text{j} * \text{bitPeriod}]$)
12         if $TxBitSeq(j) == 1$ then
13         oneC = oneC + eCount
14         else zeroC = zeroC + eCount
15     // Update $1e0e$ Ratio
16     current$1e$ = oneC/$no.\ bit\ 1\ in\ TxBitSeq$
17     current$0e$ = zeroC/$no.\ bit\ 0\ in\ TxBitSeq$
18     if current$1e$/current$0e$ > maxRatio then
19     maxRatio = current$1e$/current$0e$
20     $1e$ =current$1e$ ; $0e$= current$0e$
21 return $1e$ and $0e$;
```

receiver after processing through all layers of firmware and software. This jitter significantly increases the difficulty of detection. Since the communication channel has low bandwidth and high jitter, no traditional symbol synchronization schemes can be directly applied. We overcome the bit synchronization challenge by simultaneously synchronizing and demodulating the signal.

3. The channel adds an unknown delay between receiver and transmitter; this problem is classically solved using a frame synchronization which requires using a preamble. Since we have a low bandwidth channel and would like to transmit the message in only a few seconds, the message can only include limited number of bits. Thus, we can not afford to add the preamble. Instead, we use constrained bit patterns that are unique under cyclic shifts caused by unsynchronized frames.

The conversion from touch events to a sequence of binary digits is based on the principle of On-Off keying; the touchscreen driver produces several events when a binary one is transmitted and only a few events when a $zero$ is transmitted. The key challenge is to handle the variance in the number of events associated with $ones$ and $zeros$. In the coming sections, we describe an off-line calibration procedure to characterize the expected behaviour of the channel, which is then used in the online phase to classify touch responses as $zero$ or one transmissions. Once a sequence of bits is decoded, we use a “closeness” metric to determine the distance of the received message from the set of all possible messages of the same length. This process corrects for uncertainty in timing and event number. Details about the design of the closeness metric and the decoding process are presented in the next sections.

2.4.1 Determination of Expected Number of Events for $ones$ and $zeros$

To determine the number of touch events associated with a one or $zero$, it is necessary to calibrate the device at each data rate before use. This calibration to determine thresholds only needs to be performed once per device, at initialization; thereafter it can be stored in a lookup table and adjusted during self calibration depending on an estimate of the data rate of the incoming data sequence or fetched as an input from applications. To determine the counting threshold for each data rate, a sequence of $ones$ and $zeros$ is repeatedly transmitted in a prescribed pattern.

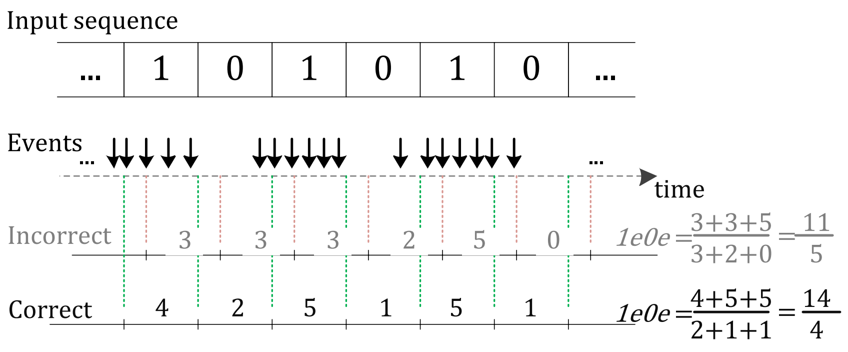


Figure 2.6: Offline calibration process searching for the correct bit synchronization point at which the *1e0e* ratio reaches the maximum value

On the receiver side, event sequence is detected and recorded to a log file. Threshold selection algorithm, algorithm 1, takes the log file and the prescribed pattern as input to compute the two expected counter thresholds $1e$ and $0e$. We devise Algorithm 1 to simultaneously demodulate the received event sequence and find the bit starting point. The intuition behind the algorithm is that the correct bit synchronization maximizes total number of events in all $ones$ and minimize number of event in all $zeros$. We define *1e0e* ratio as being the normalized ratio between the *total number of events in all* $ones$ and *total number of events in all* $zeros$:

$$1e0e\ Ratio = \frac{\frac{\Sigma(Event\ Counters\ in\ Ones)}{Number\ of\ Ones}}{\frac{\Sigma(Event\ Counters\ in\ Zeros)}{Number\ of\ Zeros}}$$

This ratio is maximized when bit synchronization is correct. The ideal synchronization, for example, should have total number of events in all $zeros$ close to 0, and number of events in all $ones$ close to the total number of events in the whole event sequence, in which case *1e0e* ratio reaches its maximum. Illustrated in Figure 2.6, in which the transmitter repeatedly transmits a sequence of alternating 0 and 1, the incorrect synchronization misaligns many events of bit $ones$ in to bit $zeros$ making the *1e0e* lower compared to that of the correct synchronization. Algorithm 1 first converts the discrete timing event information to a event/no-event time series data. That is, if the received sequence of event is $E_{discrete} = \{E_1, E_2, ..., E_m\}$ in which E_i is i^{th} event, it will be represented by a vector in the form: $Et = [Et_1, Et_2, ...Et_{t_{max}}]$ where $Et_i = 1$ if there exists an event E_k such that $E_k = i$, and 0 otherwise. In the second step, the algorithm tries all possible bit starting points within the first bit period, with each trial involving a counting of the number of events in all bit periods of the sequence. The starting point that

Bit Rate (bps)	**4**	**5**	**8**	**10**	**12**	**15**
Expected no. of events in *ones* (1e)	11.3	9.2	5.8	4.5	3.6	3.3
Expected no. of events in *zeros* (0e)	2.0	1.6	1.2	1.0	0.7	0.7
One-Zero threshold	7	6	4	4	2	2

Table 2.1: One-Zero threshold and expected number of events in bit *one* and *zero* for different bit rates

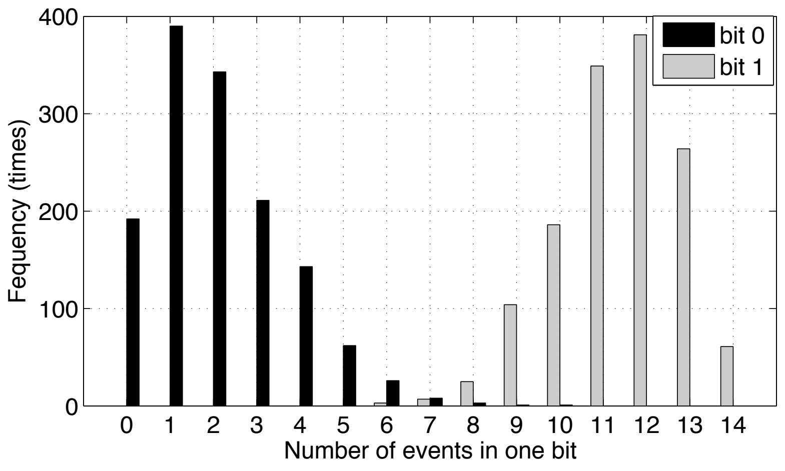
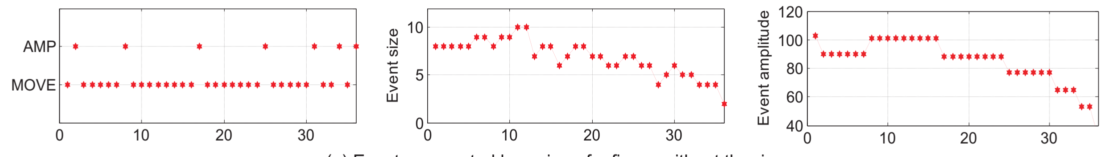
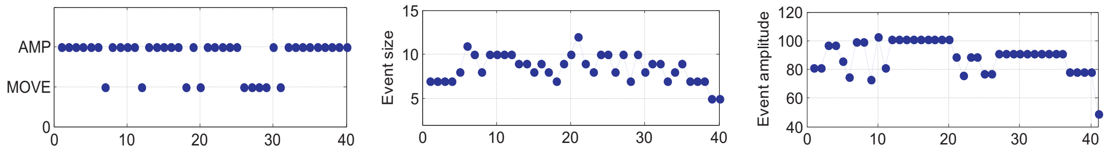


Figure 2.7: Number of events in bit *one* and bit *zero* for transmissions at 4 bits/s

leads to the highest ratio is considered the correct bit sync position, while the bit sequence corresponding to that starting point is the demodulated result of the event sequence. At the end of this process, since the total number of events in all *ones* and total number of events in all *zeros* is found, the expected number of events in *ones* and *zeros*, $1e$ and $0e$ can be derived and stored in memory for future demodulation. Figure 2.7 shows the distribution of the number of touch events registered corresponding to the transmissions of *zero* and *one* evaluated by using algorithm 1 for a 3000 bit sequence of alternating *zeros* and *ones*. The variations due to the transmission bit rate is recorded in table 2.1, which shows that the event count threshold required for decoding varies from 7 events for 4 bits/s to 2 events for 15 bits/s.



(a) Events generated by swipe of a finger without the ring



(b) Events generated by swipe of a finger with the ring

Figure 2.8: *Type, Size* and *Apmlitude* values generated from finger swipes with and without the ring. A ring bearing finger produces many *AMP* events while swipe without ring induces correlation between *Size* and *Amplitude*

2.4.2 Minimum Distance Demodulation

Using the counter thresholds from the previous section, Algorithm 2 demodulates the timing event sequence to get the data sequence sent by the transmitter. Sharing the same synchronization challenge with the threshold detection algorithm, this algorithm has to detect the point in time at which the data is transmitted. At the same time, it demodulates the event sequence to get the information that has been transmitted. Note that simply relying on the first event to determine the starting point is not enough since there is a fair amount of timing uncertainty in the communication channel. Intuitively, the algorithm traverses the sequence to try all possible starting points. At each point, it gauges the “distance” between the event sequence and all messages. It then ranks the positions with “similarity” value and selects the one that has highest “similarity” index. The message corresponding to that index will be the decoded value of the event sequence.

So the question remains as to how to measure the similarity between two sequences. We define a distance metric as following: let $D(i,j)$ be the distance between an event sequence that has a starting point at point i and the message, K_j, with $j = 1..number\ of\ messages$. Using the same notations as defined in the previous algorithm, in which $Et = [Et_1, Et_2, ...Et_{t_{max}}]$ is the event vector re-sampled along the time domain, an event counter, eC_p, for bit at p_{th} position

Algorithm 2: Min Distance Demodulation Algorithm

```
input  : $E_t$ - Event sequence in time domain
         $BitRate$ - Transmission bit rate (bps)
         $MessageLength$ - Original message length
         $PosMessageVec$ - Possible message vector
         $1e$ and $0e$ - Expected number of events in ones and zeros
output:  $RxBitSeq$ - Received bit sequence
1  bitPeriod $\leftarrow \frac{1000}{BitRate}$
2  minDistance $\leftarrow$ MAX-INT
3  for startPos $=1 \rightarrow$bitPeriod do
4      foreach message in $PosMessageVec$ do
5          rotatedMesgVec =getAllCyclicVersions($message$)
6          for j $=1 \rightarrow MessageLength$ do
7              eCount[j ] = sum($E_t$[startPos $+$ (j $-$ 1) $*$ bitPeriod, startPos $+$ j $*$ bitPeriod])
8          foreach rotatedInstance in rotatedMesgVec do
9              currentDist = 0;
10             foreach $i^{th}$ bit in $rotatedInstance$ do
11                 if $i^{th}$ bit == 1 then
12                 currentDist = currentDist + max(0, $1e$ - eCount[$i$])
13                 else
14                 currentDist = currentDist + max(0,eCount[$i$] - $0e$)
15             // Update Min distance
16             if currentDistance $<$ minDistance then
17                 minDistance = currentDist;
18                 RxCandidate = rotatedInstance;
19 return RxCandidate
```

from the starting point can be computed by:

$$eC_p = \sum_{q=(p-1)*bit\ period}^{p*bit\ period} Et_q$$

Then distance $D(i, j)$ can be derived as:

$$D(i, j) = \sum_{k=1}^{message\ length} d_k$$

with

$$d_k = \left\{ \begin{array}{l} max(0, eC_p - 0e), \text{if the } k^{th} \text{ bit on message } K_j \text{ is } 0 \\ max(0, 1e - eC_p), \text{if the } k^{th} \text{ bit on message } K_j \text{ is } 1 \end{array} \right\}.$$

We note that since messages are cyclically transmitted, the algorithm does not only compute the distance of a sequence to a message but it does so for all unique *rotated* version of that message.

The intuition behind this metric is that it rewards starting points that make the decoded sequence look similar to one of the messages in the message vector. The smaller the distance, the closer the decoded sequence to the message. Hence, smallest $D(i, j)$ will tell which position on the sequence is correct synchronization position and which message is the event sequence representing.

We note that when the number of possible messages is small (order of hundreds), it is feasible to apply Algorithm 2 to exhaustively search through the whole message space to demodulate. However, when the number of possible messages is large, the above exhaustive algorithm can become prohibitively expensive or impossible. In such cases, more efficient algorithm assuming no knowledge of the message becomes handy. That algorithm shares the same intuition with Algorithm 1, in that it tries all possible starting points. However, at each possible position, it directly converts the sequence to data bit sequence by counting the number of events in each bit period and select the one that yields the highest $1e0e$ ratio.

Other demodulation schemes. In the process of finding the most suitable demodulation scheme, we experimented with several other demodulations schemes such as *Non-thresholding modulation*, $1e0e$ *ratio demodulation* and *maximum key correlation*. Non-thresholding modulation scheme does not require any training to learn expected number of events in zeros (*0e*) and ones (*1e*). It instead looks at all possible starting positions and compares them with all possible keys to find the best match. The comparison is done by counting the number of touch events in bit $ones$ and bit $zeros$. The ratio between the two counters is used as the correlation metric. The algorithm simultaneously picks the synchronization point and decodes the sequence of events by selecting the starting point that gives the highest correlation with one of the possible keys. The maximum key correlation method takes an approach that is similar to the minimum distance modulation but has a different evaluation function. For that we defined another correlation coefficient function to take the noisy channel into account. Specifically, the function gives one point to a bit that is equal to the bit at the same position on the correct key and gives partial point to the bit that is not correctly decoded but has a number of events close to the $One - Zero$ threshold. Lastly, by relaxing the requirement about the prior knowledge of the possible message space, we have the third alternative algorithm, $1e0e$ ratio demodulation algorithm. It becomes useful when the possible message space is unknown or so large that it is prohibitively expensive to conduct an exhaustive search to find minimum distance or maximum correlation. All three alternative algorithms however do not perform as well as the Min Distance Algorithm presented earlier after the calibration process under the assumption of a manageable and known message space.

2.4.3 Ring Detection for Indirect Communication

As mentioned in Section 2.3, an indirect mode of communication is enabled when instead of the ring, a ring bearing finger is in direct contact with the touchscreen. In such cases, only the presence of a ring needs to be detected. However detecting the ring in the presence of finger movements (or finger swipes) is challenging since the events generated due to the movement of the finger and those by the ring cannot be easily distinguished.

Figure 2.8 shows three fields of the touch event outputs: *Type*, *Size* and *Amplitude*, generated when a user swipes a finger across the screen with and without the ring. We leverage two key observations from the patterns observed for designing a robust detection algorithm: (i) events generated by the finger movement without the ring are mostly of type *MOVE* while those generated by the ring are mostly of type *AMP*, however due to the excess pressure exerted from the drag force of the finger on the touchscreen, a few *AMP* events can also be generated

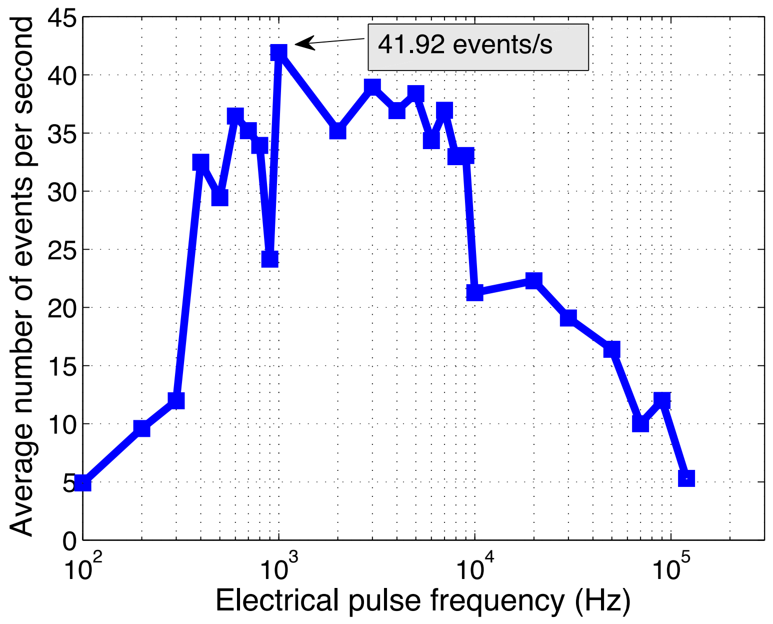


Figure 2.9: Touchsreen responses to 10 Volt peak-to-peak square wave signals with different frequencies ranging from 100Hz to 120KHz (log scale)

during finger swipe movements without the ring; (ii) in the absence of the ring, the sequence of *Size* and *Amplitude* values are correlated since increasing the pressure brings more surface area of the finger in contact with the screen. We confirm these observations by collecting data from a large number of swipe movements, both with and without the ring from 5 different users.

Since both the presence of a large number of *AMP* events and the absence of correlation between *Size* and *Amplitude* indicate the presence of a ring, we define a metric p_{ring}, which relates to the normalized number of *AMP* events registered (n_{amp}) and the correlation coefficient between the *Size* and *Amplitude* values (c_{SA}) as:

$$p_{ring} = \alpha \times n_{amp} + (1 - \alpha) \times (1 - c_{SA})$$

where $\alpha \in [0, 1]$ is parameter which signifies the relative contributions of n_{amp} and c_{SA} in determining the p_{ring} value. Given a set of generated events, a detection threshold λ_{th} is then used on the p_{ring} value to classify the presence or absence of the ring. We determined the values of the two parameters α and λ_{th} through a training set consisting of 1000 swipes from 3 different users, using traditional least square minimization. After the training, α and λ_{th} were determined to be 0.83 and 0.5 respectively.

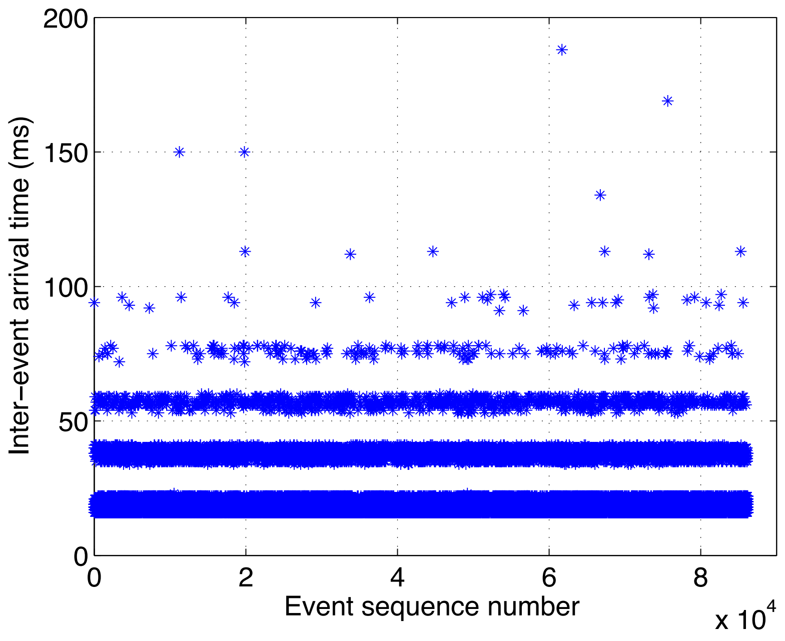


Figure 2.10: Distribution of inter-event arrival times of events generated by a 10 Volt peak-to-peak 1KHz square wave signal captured in kernel level log files

2.5 Exploring the parameter space

Data transmission using capacitive touchscreen communication is an unexplored mode of communication. In this section, we explore the dynamic ranges of frequency, voltage and signal types that can be used for triggering usable events through the screen driver. Having picked the most suitable set of parameters, we then study the performance of the communication system for different use cases.

In order to conveniently vary the input signal parameters, as required in this analysis, we placed a flat, rounded copper piece on the screen surface of a Samsung Galaxy Tab 10.1, which uses a Atmel maxTouch touch panel [67], and attached it to the output of an AFG 3000 Series function generator [68]. This setup simulates the touch of the ring on the screen surface while offering two main benefits over the battery-powered prototype described in Section 2.6: (i) it alleviates the microcontroller's limitation in generating arbitrary waveforms and (ii) it greatly expands the scale of repeated experiments (order of tens of thousands of logging runs) which would be otherwise limited by time and human effort.

2.5.1 Triggering Touch Events

The inner-workings of the touch screen are proprietary and not available for use in designing either our hardware or software. A main task is to determine what type of electrical signal will be interpreted as a touch event when it is injected into the touchscreen. To answer this, we inject different signals from a function generator through an attached electrode approximately the size of a finger to the surface of the touchscreen. Note that we are not only interested in reliably creating artificial touch events but also trying to create those event at maximum rate. Since the transmitter modulates the signal using an OOK scheme, the higher the event rate is, the faster it can transmit.

The touch events retrieved by the tablet's operating system, Android 3.2, are represented in a 6-tuple structure depicted in Figure 2.4. Indicated through *Event Type* field, touches are classified into one of the following types: *MOVE*, *AMP*, *MOVEAMP*, *PRESS*, *RELEASE* and *SUPPRESS*. For example, a *MOVEAMP* event is registered when both touch pressure and X,Y-coordinates change at the same time; and a *SUPPRESS* event happens when the touch pressure exceeds a predefined threshold. Note that such touch events are triggered when a finger first touches the panel, when the position of the finger on the screen changes, when the pressure changes, and when the finger leaves the screen. *Touch Size* and *Touch Amplitude* specify the size and amplitude of the touch respectively. *Pointer ID* is used to differentiate the presence of two or more points of contact at the same time, or multi-touch. A physical touch causes voltage changes at many different electrodes, but the firmware and driver aggregate them to output a single touch event to the operating system. Since the aggregation algorithm is proprietary, the conversion from electrical signals of our interest to such touch events can only be empirically learnt.

An important aspect of the system is the maximum possible data rate through the screen, which depends on two key characteristics of the screen: (i) the highest rate at which the driver and firmware allows touch events to be registered and (ii) the internal switching frequency of the sensing hardware. Atmel mXT1386's datasheet specifies a maximum of 150 *raw touch events* per second [67]. However, due to the driver in Android's software stack, the maximum rate is significantly reduced. We conducted many experiments to gauge the actual maximum event

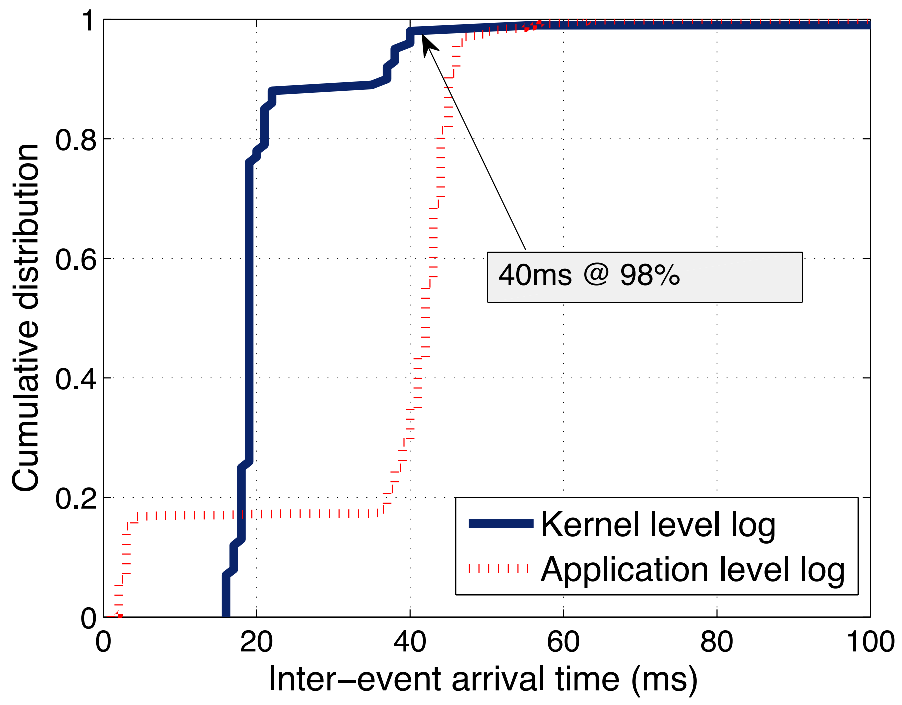


Figure 2.11: CDF of inter-event arrival times of events captured at application level and kernel level log files with 10 Volt peak-to-peak 1KHz square wave input signal

detection rate. We transmitted signals with different waveforms, at different frequencies and voltage levels to a screen. With frequencies ranging from 100 Hz to 120 KHz, we observed that a 10 Volt peak-to-peak square wave signal at a frequency of 1 KHz can register the maximum rate of 41 events/second (i.e. average inter event arrival time of $\frac{1}{41} = 24$ ms).

In particular, we began with finding the frequency to which the touch-screen was most responsive. To do so, we set the Tektronix digital function generator to generate square wave of different frequencies at 10 Volt peak-to-peak amplitude. The frequency was varied from 100 Hz to 1 KHz with 100 Hz difference, from 1 KHz to 10 KHz with 1KHz difference, and from 10 KHz to 120 KHz with 10s KHz difference. To collect the signals, we wired the output from the function generator to a flat soldered electrode, then placed the electrode on the surface of the Samsung Galaxy Tab 10.1 touchscreen. To make the electrode stable on the surface, we taped it tightly to the touchscreen to avoid unintended movement. For each frequency, we collected the data for 200 seconds. Then recorded the number of events collected from the kernel. The average number of events is shown in Figure 2.9, which suggests that the screen best responds to a signal at 1 KHz.

Figure 2.11 shows the CDF of inter-event arrival times at the kernel and application level log files. While almost 90% of the times, two consecutive events captured by the kernel log happen within 20 ms with very little variation, that number widely varies from 3 ms to 48 ms in the case of the application level log. That observation indicates that using the timing information from kernel level log would could improve the demodulation results which mainly relies on event timestamps. In addition, we also observed that sinusoidal or triangular signals do not register any events. With such waveforms, the rate of signal change is so slow that the voltage amplitude sensed is not high enough to be considered a touch event. We further tested with signals with different amplitudes and noticed that if the voltage amplitude of the signal is too high, the screen blocks all subsequent touch events for a short period of time and sends an error event to the operating system, which is the SUPPRESS event mentioned above. If the voltage is too low, the signal is either not detected or detected at a very low rate by the touch screen.

A scatter plot of 86,200 events collected over 1850 seconds, Figure 2.10, illustrates the distribution of inter-arrival times, i.e. the time difference between two consecutive events, captured in kernel level log. An interesting pattern can be observed in Figure 2.10 is: most of the inter arrival times fall into specific narrow bands which we believe to be due to firmware throttling. Its cumulative distribution shows that 98% of the time, the inter-event arrival time is less than 40 ms. Note that this event detection rate is more than 7 times lower than the rate of 150 raw touch events per second specified by the manufacturer [67]. Without access to the physical layer and the proprietary driver, we cannot determine the origin of this discrepancy. We suspect that the touch panel device driver imposes a practical limit on how many events the operating system can see under certain assumption on the maximum possible event rate that can generated by human being.The data rate could be at least 7 times faster than what we currently achieve with access to the driver; and even higher data rate might be possible with direct access to the lower physical layer.

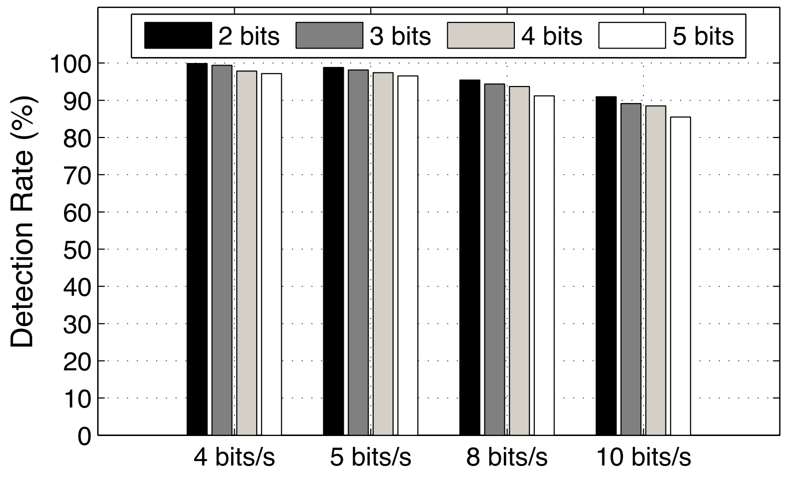


Figure 2.12: Detection rate for different message lengths and bit rates

2.5.2 Bitrate vs. Detection Rate Trade-off

The main performance metrics here are the detection rate and the false acceptance rate. The detection rate signifies the probability of correct decoding of a message while the false acceptance rate characterizes the probability of a wrong message being incorrectly decoded as the original message. As explained in Section 2.4 and shown in Table 2.1, there exists a trade-off between the detection rate and the bit rate at which messages can be decoded from the touchscreen event logs. Correspondingly, since there are higher chances of incorrect decoding at higher bit rates, the number of false positives increase as the bit rate increases. In order to quantify this phenomenon, we use the setup described above to repeatedly transmit messages of different length at different bit rates. Figure 2.12 and Figure 2.13 show the detection and false acceptance rates observed. To derive the detection rate for each message length and bit rate, we transmit each message of that length 5000 times and present the average percentage of messages that are correctly decoded. Similarly, the false acceptance rate is derived by sequentially fixing each message as the correct message and transmitting all other messages of the same length 5000 times.

The trends in Figure 2.12 and Figure 2.13 indicate a gradual decline in the detection rate with the increase in either the transmission bit rate or message length. We note that for simple

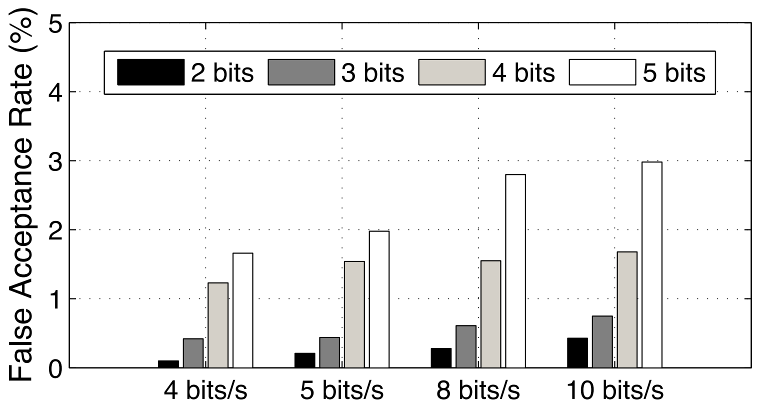


Figure 2.13: False acceptance rate for different key lengths and bit rates

parental control applications, a 99% detection rate can be achieved by using 2 or 3 bit messages at 4 bits/s. For applications that have a less stringent detection rate requirement, a much higher bit rate can be used to speed up the required data transmission time. For authentication applications, transmitters typically need to transmit a longer bit sequence. To get the similar entropy level that the 4-number PIN code has, for example, transmitters have to send a 14-bit long sequence into the screen, which could take about 3 seconds at 5 bits/s.

2.5.3 Indirect Communication Results

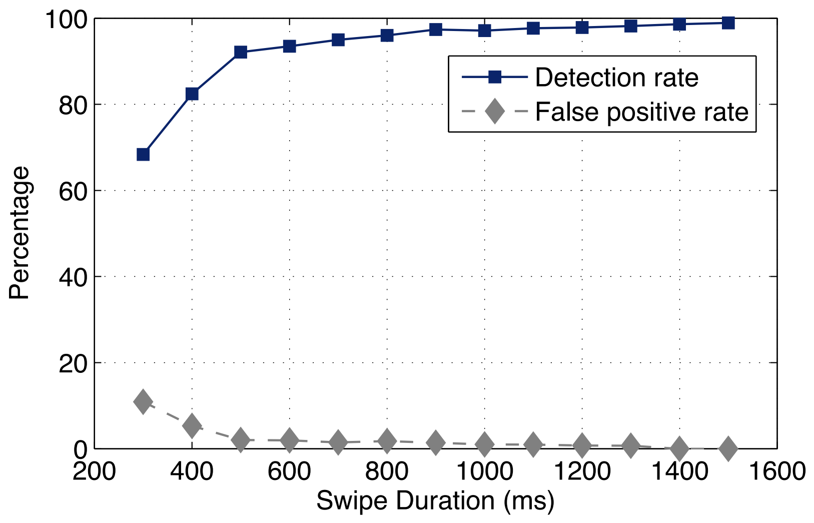


Figure 2.14: Multi-user games: Swipe detection rate

The next set of results are targeted towards detection of individual users in an indirect communication scenario. In this scenario, while the bit rate required is not very high, touching the ring to the touchscreen would hinder in the game-playing process. As such we leverage the fact that even if the finger-tip of the ring bearing finger touches the screen, the patterns in the registered event logs can be used to differentiate between a user with a signet ring and the one without it.

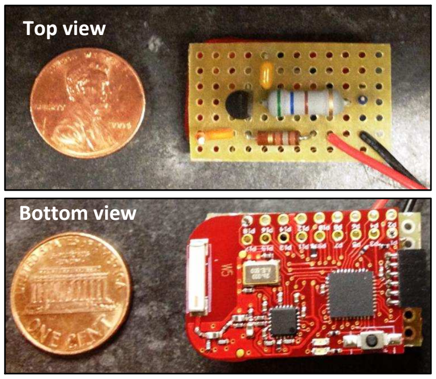
In order to quantify the performance of this algorithm, we collected a total of 6,000 swipes from 3 different users with half the swipes with a ring on. We asked the users to vary the swipe duration between 300ms to 1.5 seconds but since making a swipe last for precisely a given time is difficult, we bucketed the collected swipes into 100ms durations starting from 250ms to 1550ms and discard swipes outside of this range. The swipe duration of all swipes within a bin are approximated by the mean value of the bin. Using the move events registered in this dataset, we calculated the detection rate of ring bearing users and the percentage of swipes without rings which were wrongly classified as one with rings, i.e., the false acceptance rate.

The resulting values shown in Figure 2.14 shows that the detection rate increases with the duration of the swipe, at first sharply from $\sim$68% for 300ms swipes to $\sim$92% for 500ms swipes and then gradually after that. Thus if the duration of the swipes used in a multi-player game is longer than 700ms, the users can be classified correctly with a 95% confidence level.

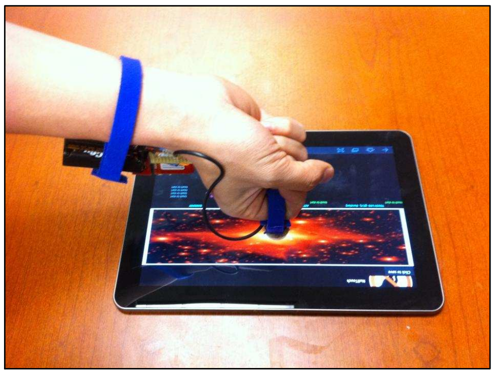
We note that the use of the ring and this communication technique in general has minimal impact on the screen's operational performance (i.e. power consumption, touch event parameters) and to the running applications. Because it generates multiple touch events which are handled by the screen's firmware and the operating system, the ring introduces a small processing overhead to the mobile device. That overhead however is negligible compared to the legacy load of the mobile device. On the other hand, the effects of the ring on touch events' amplitude and size is observable. That would affect legacy applications that operates on the two parameters.

2.6 Ring Prototype and Evaluation

The use of this communication channel with touchscreen device requires a hardware token to generate the appropriate electrical signal and inject it into the touch sensing circuitry. In this



(a) Prototype circuit board



(b) Usage of the prototype ring

Figure 2.15: The prototype ring and its usage for transmitting short messages from the ring to a touchpad

section, we describe our prototype of the ring which uses off-the-shelf components.

2.6.1 Hardware Prototype

The core of the token is a low cost, low power microprocessor, TI-MSP430F2722 [69] that was programmed to generate modulated 3 Volt square waves at a frequency of 1KHz. Figure 2.16 shows the schematic of the custom-built ring. This square wave is modulated with On-Off keying to trigger artificial touch events in the screen's firmware. The microprocessor is mounted on a 18 mm x 30 mm off-the-shelf board, part of TI-MSP430 eZ430 development kit, as shown on Figure 2.15(a)-bottom view. We specify the transmission data rate and data sequence by programming the microprocessor through the USB interface that comes with the kit. The square wave and its parameters were selected through experiments with a function generator, as described in section 2.5. Since we found that 3 Volt was not adequate for generating touch events, we amplify the output of the microprocessor using a single bipolar transistor, BC548B [70], with the supply voltage of 9 Volt (Figure 2.15(a)-top view). One of the most challenging parts of the prototype was to design the electrode configuration that would allow the signal to be injected in series with the touchscreen and the body capacitance of the user. The best point in the circuit to inject the signal, V'_{sig} in figure 2.1, would be in series with the finger and the rest of the body at a point close to the screen. This has obvious anatomical difficulties and the low internal resistance of the body makes injection between two closely spaced electrodes,

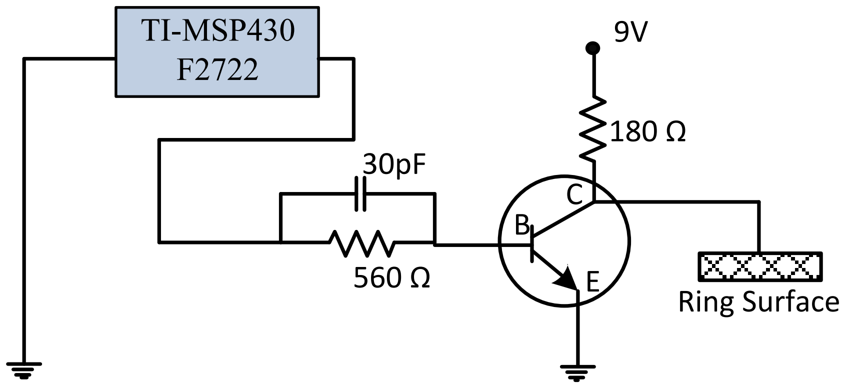


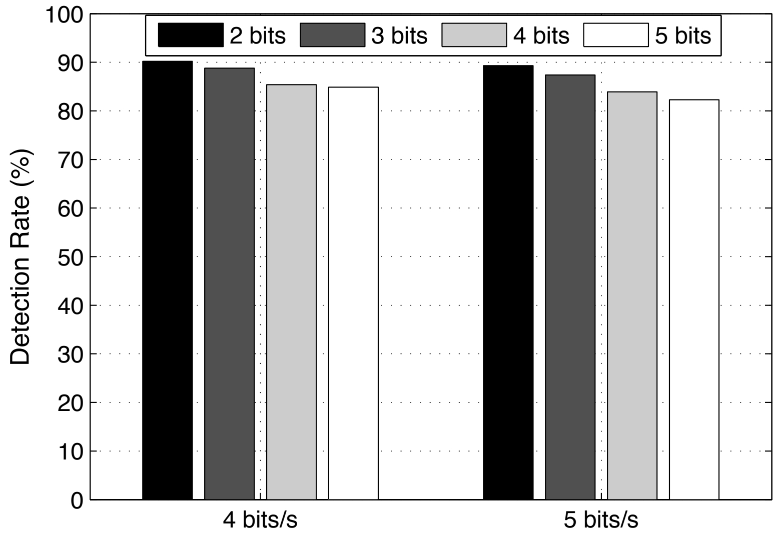
Figure 2.16: Schematic of the custom-built ring

as on the inside surface of a ring, impractical. We opted for a system where the user would wear an insulating ring where V'_{sig} was injected between electrodes on the inside and outside of the dielectric band. The inner electrode was connected with the finger and, through the body capacitance C_B and case capacitance C_c (as described in section 2.2), to the internal circuitry in the tablet. The outer electrode on the ring was directly pressed on the screen, forming C_s to complete the circuit.

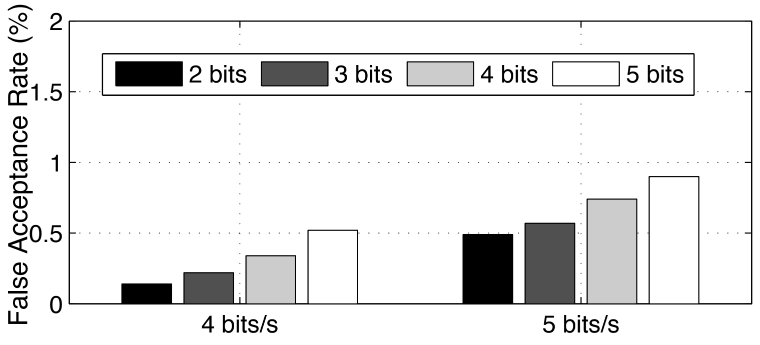
Because a uniform and reproducible contact between the touchscreen and the ring is essential to minimize the error rate, we choose to use a flexible conductive material to make the electrode and design the face of the ring to control the compression of that material. If the pressure is too high, the screen bends and its capacitance, C_s, increases which in turns can introduce errors. We control this pressure by surrounding the electrode with an insulating spacer of the correct thickness to properly control the compression of the flexible electrode.

2.6.2 Preliminary Prototype Performance

Using the prototype ring, we experimented with injecting messages through the Samsung Galaxy Tablet 10.1 touchscreen. We implemented an Android application that mimics common login authentication procedures. The application decodes the key carried by and transmitted from the ring. Depending on which key its receives, the application will load the profile of the corresponding user that associates with that key. While conducting the experiment, we noticed that at times no events were triggered during the transmission of a 1 bit or vice versa. This lead to unreliable decoding of messages but we were still able to distinguish two codes with a larger hamming distance. One user carries a ring with the key "1110" and another user carries a ring with the key "1000". Each users touched the ring on to the tablet's display 50 times. A simple threshold-based algorithm that uses the number of touchscreen events generated as input was able to identify the first ring correctly 44 times and the second ring 43 times, leading to an overall detection rate of 87%. We suspect that the quality of the contact between the ring and the touch panel plays a critical role in these experiments. To eliminate this variance due to



(a) Detection rate



(b) False acceptance rate

Figure 2.17: Detection rate and False acceptance rate using ring prototype for different message lengths and bit rates

contact differences from touch to touch, we experimented with transmitting multiple messages while the ring was held steady on the display. Here, we used message lengths between 2 and 5 bits transmitted at the rates of 4 bits/s and 5 bits/s from which the detection rate (DR) and false acceptance rate (FAR) are evaluated. For each message at each data rate, we put the ring down onto the screen 3 times and keep it there long enough so that 200 repetitions of the message are transmitted from the ring to the screen. We show in Figure 2.17 the DR and FAR results over the 200 repetitions from best case (presumably best contact) of these three trials. Each bar represents the average rates over different data rates and message lengths. We observed that the detection rate decreases with the increase of both the message length and bit rate. Note, however, that the overall detection rate could be improved through retransmissions of the message. Therefore, even the lower detection rate of 82% may still be adequate for some of our targeted applications. For the user identification application, for example, up to 3 seconds of continuous repeated message transmission would results in less than 6 errors per 1000 uses. These results illustrate what can be achieved with this transmitter if the reliability issues are worked out.

We believe that another source of error in this prototype stems from the relatively long rise time of the square wave since the touchscreen events appear to be triggered by the edges in the input signal. It is also important to note that both the electronics and the firmware of the screen, which we do not have access to, are optimized for the relatively slow movement of a human finger. Thus, the screen driver deliberately throttles the maximum rate of touch events, which reduces touch error in normal use but limits our system to very low bit rate transmission. We believe that the transmission rate could be improved substantially with access to the touchscreen controller firmware, which should allow processing internal touchscreen measurements, e.g. physical voltage differences.

2.7 Discussion

Let us briefly consider remaining issues related to the energy consumption and security applications of this technique.

Energy Consumption. The current prototype implementation is based on a 3 Volt microprocessor driving a 9 Volt high speed bipolar transistor amplifier to generate a continuous signal. Energy consumption and some of the synchronization issues in our prototype could be significantly reduced by incorporating a switch under the contact surface that powers up the ring when pressed against the touch screen. To estimate the cost and battery life of such a ring version, we use the smallest readily available lithium primary battery, the CR2025 which is 10 mm in diameter and supplies 3.0 volts with a 30 mA-h capacity. The typical current drain in standby with RAM-retention of a modern microprocessor (e.g. the TI MSP430 family) is about 0.1 microamps. Even with this small battery, this would provide over 3 decades of standby lifetime for the ring electronics. Once awake, the processor will use significantly more current, but the minimal computing requirements result in this being low, also. The smaller MSP-430 processors typically use about 220 microamp at 1 MHz, so even if shifting out the short code takes 100 cycles of the CPU, this battery will still provide enough energy for over 5000 uses.

Since the capacitances are very small, the current will also be low and a simple buck-boost dc-dc converter with one miniature inductor will be quite adequate to supply the 9 Volt [71]. Assuming only a 10 $\%$ charge conversion efficiency for the converter, this circuit still uses only about 2 nanocouloumbs/charge-discharge cycle. Modulating at 1 KHz and sending 10 bits/second, this allows the battery to supply over 50 million bits, far in excess of any of the other limits in the system. The cost of such a system will be dominated by the processor, several tens of cents, but in high volume that can be replaced by a simple sequence generator, either read-only or flash, for only a few cents.

Security considerations. The current limits on data rate only allow transmission of very short codes and thus allow only weak authentication at best. Improvements in data rate through modifications in the touchscreen firmware could alleviate these limits, however. The low carrier frequency of our system, between 5-10 kHz, would then also offer additional protection against eavesdropping. Since antenna size should be proportional to the wavelength of the signal, transmission of this signal into the RF domain would require an antenna much larger than the size of the human body. While we cannot rule out that some signal can be received with customized resonant antennas, however, the level of effort required would be much higher than for picking up a e.g. 2.4 GHz signal used in WiFi and Bluetooth. If such eavesdropping ever were an issue, it could also be addressed by transmitting a noise signal from the receiving device.

Another security consideration is the concern of unauthorized use of the hardware token. It however can be addressed by integrating bio-metric signature techniques [72] with the token, activating its transmission capability only when the token recognizes the owner's signature. Note that the referred biometric signature techniques cannot be directly used in replacement of our techniques for authentication due to its required infrastructure support.

Alternative hardware designs. The current design could be enhanced with a feedback channel using a photodetector. The ring could receive information from the mobile device through this visual channel, where the device encodes the information in the pixel intensities. This would enable a challenge response protocol, which could greatly enhance the security of an authentication system. In addition to challenge response security enhancement, the photodetector could receive acknowledgement signals from the tablet to ensure the reliability of the transmission. One way to use this feedback information would be for the signet ring to optimize detection by the tablet by varying the frequency and phase of the electrical pulse pattern.

An alternative physical layer approach could be to vary the effective capacitance between the ring and screen. This could be done by inserting another capacitor between the ring surface and the screen whose area or thickness could be modulated. Done properly, this could generate touch events with even less power than the current hardware design. Using the form factor of the ring surface that creates multiple contact points with the screen taking advantage of the multi-touch capabilities could further improve the data rate for any of the physical layer technique we discussed.

Error correction and control coding schemes. Under the current data rate, we design our code to reduce the false positive and improve detection rate by first studying the pattern of error when a short data sequence is transmitted. We choose the code words that are more distinguishable given the observed error patterns. In particular, we use codes which have frequent changes in values and, hence can be easily synchronized. In addition, we require that beginning of the code sequence is easily distinguishable by avoiding non-cyclic patterns.

When able to achieve a higher data rate and target applications that require the transmission of a long data sequence, we plan to exercise the same process to design an appropriate coding scheme. A specific application would determine the length of the code being transmitted and the characteristics of error pattern would indicate which coding scheme is best suited.

Applications. As alluded to in the introduction, there are several applications that could make use of our capacitive touch communication technique. With the current performance, the proposed technique can be directly applied to parental control applications, multi-user games and weak authentication for mobile devices. Further improvement in transmission rate and reliability would open up many other of applications.

User identification and authentication in many cellular networks has so far been based on SIM cards, essentially tokens directly inserted into a cellular phone. This was an adequate solution when people access the network through a single device. With access to diverse devices such as smart phones, laptops, tablets, and cars that may be shared among multiple users - who may be constantly on the move - it is becoming more important to understand which user is interacting with them at any given time. In addition, with future shared data plans (shared across devices) data usage from any device could be charged toward user account instead of charging toward devices. That billing model can be realized by our proposed techniques in which the signet ring is used as a separate identification token, a portable SIM, worn by users.

The ring can be used as a replacement for credit card (i.e. credit ring) for authenticating monetary transactions on mobile phones and ATM machines. At the same time, thanks to the pervasiveness of capacitive touch technology, the same ring could be used to access a smart-home where it would not only unlock the door but could also authorize access to and load user-specific preferences on all the user's devices in the house such as entertainment systems, home appliances.

2.8 Conclusion

We have presented the design and implementation of a technique to transmit information through a capacitive touchscreen. Our method triggers touch events in the touchscreen device by injecting an electric signal that affects the capacitance measurements of the screen. Our experiments show that this is feasible even with an off-the-shelf touchscreen system, albeit at very low bitrates. Controlled experiments with a signal generator demonstrates data rates of 5-10 bps. While some reliability challenges remains, we also achieved up to 4-5 bps with a wearable transmitter token in the form of a small signet ring and demonstrated that some signals can be transmitted through the human skin. Transmission of information via small physical tokens can be used to distinguish who is interacting with a mobile device, and can be useful for parental control, multiuser games (particularly when played on a single device), and possibly play a role in authentication solutions. It differs from other short-range communication systems in that it requires physical touch for communication, which can be an advantage if multiple potential users are so close that they cannot be differentiated with the other short-range systems. The technique could also be used to distinguish different devices touching the screen such as styluses or boardgame tokens. We believe that significantly higher data rates could be achieved by designing receiver capabilities into touch screens and few this work as a first step towards exploring how this touch sensor can participate in the exchange of information between mobile devices.

Chapter 3

Global Naming Resolution Service

DMap is the foundation for a fast global name resolution service necessary to enable emerging Internet services such as seamless mobility support, content delivery and cloud computing. In the context of composable mobile systems, DMap allows devices to lookup network locators of other endpoints efficiently at scale. To address the most challenging problems in global naming resolution – scalablity and latency – DMap distributes identifier to locator mappings among Autonomous Systems (ASes) by directly applying consistent hashing functions on the identifier to produce network addresses of the AS gateway routers at which the mapping is stored. The novelty comes from leveraging the reachability information of the underlying routing mechanism that is already available at the network layer, and achieves low lookup latencies through a single overlay hop without additional maintenance overheads. The evaluation results, using a large-scale custom-built discrete event simulation of the Internet with $\sim$26,000 ASs and real-world traffic traces, show that the proposed method evenly balances storage load across the global network (hence scalable) while achieves lookup latency with a mean value of $\sim$50 ms and 95^{th} percentile value of $\sim$100 ms, considered suitable to support dynamic mobility across the global Internet.

3.1 Introduction

The concept of separating identifiers from routable addresses or locators has been advocated by a number of authors in the networking community [17, 18, 19, 20]. Separation of names from addresses makes it possible to avoid implicit or explicit binding of sources and destinations to the network's actual topology. Using existing terminology, the *identifier* names a communicating object, such as a particular mobile phone, while the *locator* identifies an address the network can use to route messages. For example, a phone connecting to different 3G, 4G, and

WiFi networks would get a separate locator for each network. However, the identifier, which in this case could be the International Mobile Subscriber Identity (IMSI) number, would remain the same. The goal of this work is to explore the feasibility of identifier based communication under the assumption of large-scale dynamic mobility of named objects. In the above example, programmers should be able to send messages to a particular phone based on its IMSI number rather than to an IP address. We take the position that identifiers can also be used to name abstract entities and services; they need not to be tied to a particular device.

Identifier based communication has many advantages, including simplified implementation session management, multi-homing, mobility, disconnection, authentication and security [17, 18, 19, 20]. When there is a high degree of dynamism between the communicating entities and the network (as in most mobile service, content retrieval and cloud computing scenarios), using identifiers to define network-attached objects is more appropriate than using locators. Intuitively, it is easier to work with networking primitives based on identifiers when the locator changes faster than the timescales of the communication session. For example, a voice call may last 30 minutes, but a mobile device in a vehicle may change its network attachment points many times during this period.

Realizing an identifier based protocol stack has several challenging aspects; the key design issue we address in this work is the dynamic binding of identifiers to locators. That is, when the user presents the networking stack with an identifier, the networking subsystem must quickly return a set of locators, or *network addresses* (NAs) back to the user. We address the challenge of providing a fast global name resolution service at Internet scale in this chapter, and describe and evaluate a specific *Direct Mapping (**DMap**)* scheme for achieving a good balance between scalability, low update/query latency, consistency, availability and incremental deployment.

We take note of two trends in the Internet community that have significant bearing on the design of a global name resolution scheme. First, a flat identifier space is preferred to the hierarchical domain names currently used in the Internet. The use of flat, location independent identifiers is a central tenet of a number of clean slate proposals such as AIP [73], HIP [19], ROFL [74] and MobilityFirst [16]. The key advantage of flat labels lies in their use for direct verification of the binding between the name and an associated object (see [75] for a detailed discussion). As a result, name resolution schemes that rely on the hierarchical structure of

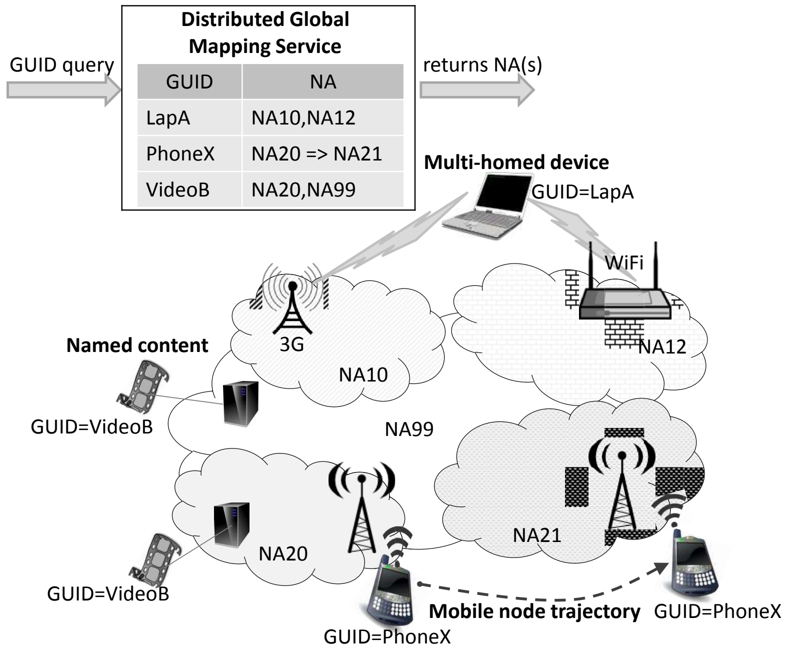


Figure 3.1: Distributed global identifier to locator mapping service

the name such as the Domain Name System (DNS) or LISP-TREE [76] are not suitable for supporting such a flat identifier space.

The second trend is that due to its separation from the network attachment point, global names or identifiers will tend to belong to end-users or application providers rather than to the network, as is currently the case with IP. Hosts and other network-attached objects (content, computing services, etc.) are not owned by any Internet Service Provider (ISP), but they just happen to be connected to a particular Autonomous System (AS). Most name resolution schemes propose to store the mappings of the identifiers belonging to an AS inside that AS only [77]. The same rationale, however, does not work for a host-based identifier space because hosts (or content) do not belong to any particular AS, especially with the increasing number of mobile hosts which often have multiple simultaneous points of network attachment. Thus we challenge the assumed constraint of ownership based storage and design a scheme with network-wide sharing of the identifier-locator mappings independent of the AS boundaries.

Motivated by these trends, we propose a dynamic identifier to locator mapping management scheme called DMap which supports a flat space of a identifiers, referred to as *Globally Unique Identifiers*, or GUIDs. A GUID is a long bit sequence, such as a public key, that is globally unique and long enough that the chance of a collision is infinitesimally small. Each end host, such as laptops, mobile phones, servers and virtual machines can have a GUID. In addition, even abstract objects, such as a piece of content or a particular context, can have GUIDs. Each GUID is associated with one or more network addresses (NAs) that it attaches or belongs to. For example, the NAs of a multi-homed laptop in Figure 3.1 includes the NA of its 3G service provider and the NA of the network that its WiFi interface attaches to. We denote the identifier to locator mapping as the GUID$\rightarrow$NA mapping.

To perform the mapping service for a given GUID, DMap applies $K(K > 1)$ hashing functions onto it to produce a list of K network addresses, which are IP addresses in today's Internet, and stores the GUID$\rightarrow$NA mapping in the ASs that announce those network addresses. By doing so, DMap spreads the GUID$\rightarrow$NA mappings amongst ASs, such that an AS will host mappings of other ASs, as well as have its mappings hosted by others. A key advantage of this *shared hosting* approach is that it allows the hosting ASs to be deterministically and locally derived from the identifier by any network entity. DMap is simple yet efficient. It leverages the routing infrastructure to reach the hosting AS in a single overlay hop; it does not require a home agent, unlike mobile IP and existing cellular networks. Further, the potential shortcoming of the direct mapping scheme, the lack of locality, is addressed by having multiple copies of the mappings that are stored in multiple locations. We further improve the design by including a local copy of the mapping within the AS that the GUID is residing in (this AS may change as the host moves).

Through detailed simulation studies, we show DMap achieves a 95^{th} percentile round trip query response time of below 100ms, which is important to support the fast growing class of mobile devices connected to the Internet. Our results also show that DMap can proportionally distribute GUID$\rightarrow$NA mappings among ASs, which is critical to scale our system to support billions of GUIDs and NAs associated with a global scale network.

The rest of this chapter is organized as follows. In Section 3.2, we provide the background and motivation for DMap. The working of DMap and how DMap addresses several technical challenges are discussed in Section 3.3. We present detailed simulation evaluation results in Section 3.4, and an analytical model in Section 3.5. Finally, we have the related work in Section 3.6 and the concluding remarks in Section 2.8.

3.2 Background and Motivation

3.2.1 Identifier and Locator Separation

While there is broad agreement on identifier locator separation [17, 78], implementation proposals vary widely along two main design dimensions: (i) what does an identifier correspond to? and (ii) how is it mapped to a locator? There are two main approaches. In the *router-based* proposals such as LISP [79], Six/One [80] and APT [81], the identifiers identify the network endpoints, and hosts can be reached by specifying the endpoint through which they are connected to the network. Thus a host has to acquire a different endpoint identifier every time it changes its network attachment point (though patches to work around this problem have been proposed [82]). In contrast, there is an alternative *host-based* approach in which identifiers are designated to end hosts, resulting in each host maintaining its identifier irrespective of changes in its point of attachment to the network. HIP [19], MILSA [20] and MobilityFirst [16] proposals have shown the distinct benefits of having host-based identifiers in terms of mobility support, multi-homing support and security, evidently at the cost of requiring changes in the host-side protocol stack.

3.2.2 Requirements of Host-Based Identifiers

We believe that a host-based mapping scheme must meet the following requirements:

- **Flat Identifiers:** The mapping architecture needs to support structureless, flat identifiers.
- **Low Latency:** Since mobility is directly handled using dynamic identifier to locator mapping, latency requirements are much stricter in host-based schemes.
- **Low Staleness:** Fast mobility support also requires that the identifier-locator mappings be updated at a time-scale smaller than the inter-query time.
- **Storage Scalability:** Since flat identifiers would lead to substantially more number of identifier to locator entries, the mapping scheme needs to scale to the order of billions of entries instead of thousands [83].

The above requirements call for a fundamental shift from traditional mechanisms such as MobileIP, DNS and DHT. While it is applicable at small scale, the mapping scheme of MobileIP incurs high overhead since all mappings are resolved by the home agent regardless of its distance to correspondents. A home agent acting as a relaying node on the data plane in tunnelling mode makes MobileIP not scalable to global Internet scale. On the other hand, since it relies on extensive caching, DNS cannot deal with fast updates. In addition, to store the mappings of billions of hosts and handle their updates/queries, a much larger dedicated infrastructure than the current DNS would be required. Traditional Distributed Hash Table (DHT) schemes and their optimized variations, e.g., [84, 85], aim to solve the problems of centralized solutions but invariably introduce a fundamental tradeoff between service latency and table/maintenance overhead. A detailed discussion of other existing mechanisms along with their pros and cons are presented in Section 3.6.

3.2.3 Incentive for Shared Hosting

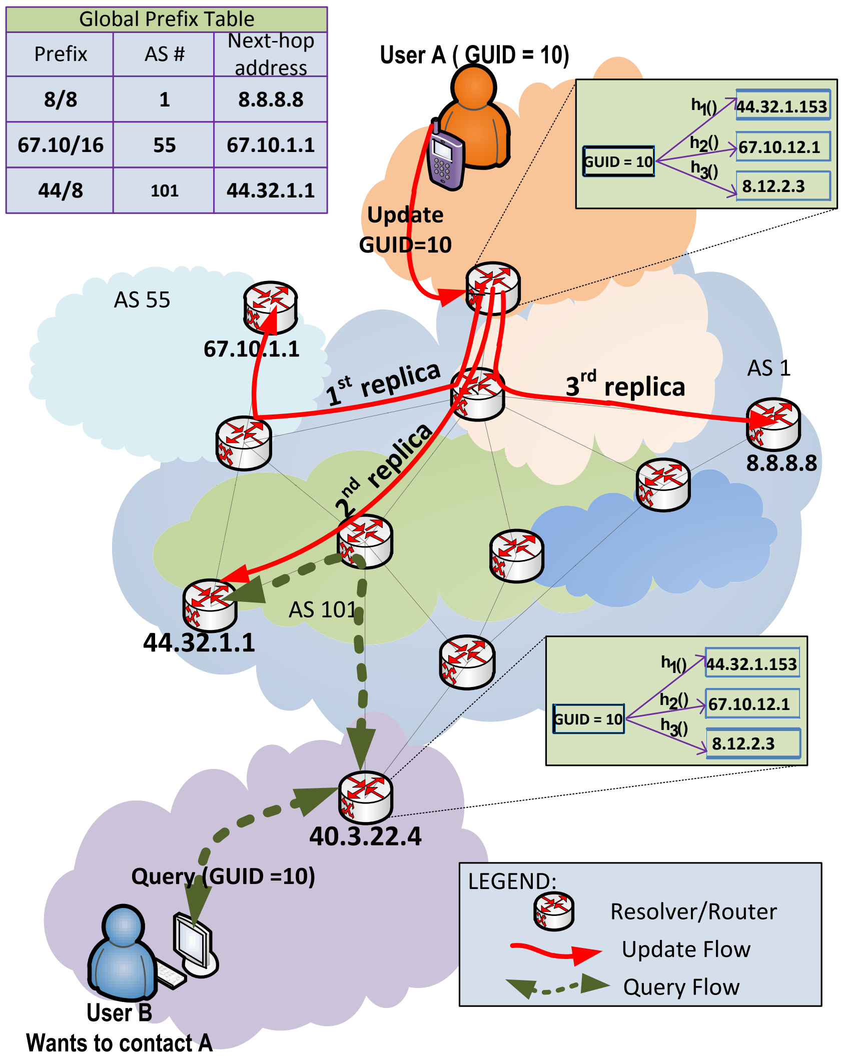
To address the problems above, in this work, we propose DMap which is built on the principle of shared hosting of the locator to identifier mappings among all the ASs in the network. A concern that may arise naturally with shared hosting is incentive: *why would Network Operator A store and manage Network Operator B's identifiers?* As we argued above, with host-based identifiers, the concept of site-dependent or provider-dependent identifiers are diluted specially in the case of mobile hosts. For fixed legacy hosts, we assert that just like peer-to-peer file sharing systems and TCP congestion control, cooperative schemes that result in a common good with a small individual cost have a natural incentive mechanism for deployment as long as the individual cost of participation is reasonable. In particular, the incentives for foul-play, i.e., not storing or answering mapping requests in this case, would depend on the benefit vs. possible penalty of non-compliance. Both technical solutions (such as reputation management in peer-to-peer systems) and non-technical policy bindings (analogous to Network Neutrality arguments) can be invoked to force/persuade ASs to fairly participate in the scheme. However, in this work, we focus on the architectural and performance aspects of the scheme and leave the design of the incentive mechanisms open.

3.3 Direct Mapping (DMap)

In DMap, each GUID$\rightarrow$NA mapping is stored in a set of ASs. Each GUID is directly hashed to existing network addresses and its mapping is thus stored within the ASes corresponding to these network addresses.

3.3.1 Overview of DMap

In designing our mapping method, we strive to minimize update/lookup latencies as well as the amount of state information that needs to be maintained. We achieve these goals by leveraging the globally available BGP reachability information to distribute the GUID$\rightarrow$NA mappings among all the participating ASs. In our scheme, DMap first hashes a GUID to an existing network address, and then stores its GUID$\rightarrow$NA mapping within the AS that announces this network address. This results in exactly a single overlay hop for all the update/lookup requests without introducing any additional state information on each router. Next, we look at an example to illustrate this approach. In this example, we assume the usage of the existing IP address space, but we note that the same technique can be easily extended to any future addressing scheme such as IPv6, AIP [73] or HIP [19].



Global Prefix Table		
Prefix	AS #	Next-hop address
8/8	1	8.8.8.8
67.10/16	55	67.10.1.1
44/8	101	44.32.1.1

Figure 3.2: DMap with K=3 independent hash functions

Let us suppose host X, with GUID G_x, is attached to NA N_x. X first sends out a *GUID Insert* request, which is captured by the border gateway router in its AS. The border gateway router then applies a predefined consistent hash function on G_x and maps it to a value IP_x in the IP space. Based upon the IP prefix announcements from its BGP table, the border gateway router finds out which AS owns IP_x and sends the $G_x \rightarrow N_x$ mapping to that AS. Later, suppose host Y wishes to look up the current locator for GUID G_x. Y sends out a *GUID Lookup* request. After the request reaches Y's border gateway router, the border gateway runs the same hash function to identify the AS that stores the mapping. Every time when X changes its association and connects to a different AS, it needs to update its mapping by sending out a *GUID Update* request. Update requests are processed similarly as insert and lookup requests.

Using the above approach, a GUID's mapping is hashed to a random AS, without considering the locality between the GUID and its lookup requests. This lack of locality may potentially lead to unnecessarily long lookup latencies. Thus, instead of storing a mapping at only one AS, we consider having K replicas of the same mapping stored at K random ASs. Having K replicas can significantly reduce the lookup latency as the requesting node can choose the closest replica (e.g., based upon the hop count between itself and the hosting ASs). Meanwhile, it will not have a big impact on the update latency as we can update the replicas in parallel. With K mapping replicas, the lookup latency becomes the shortest latency among the K ASs, while the update latency becomes the largest among the K ASs. Figure 3.2 illustrates an example update and lookup process with $K = 3$. Finally, we note that important DMap parameters, such as which hash functions to use and the value of K, will be agreed and distributed before hand among the Internet routers.

Compared to other mapping schemes, one distinct feature of DMap is the direct participation of network routers in storing GUID$\rightarrow$NA mappings and in responding to updates and lookups. DMap does not require any additional state information as the IP reachability information is already made available by the BGP routing protocol. In addition, we note that unlike many recent proposals [86, 87, 76, 77], DMap does not distribute GUID mappings based on the assumption of the aggregate-ability of the GUID space. Our scheme is suitable for flat address spaces, which has been pointed out as a desirable feature for the Future Internet [73, 19].

3.3.2 Handling Unallocated Network Addresses

DMap hashes a GUID to an IP address, and stores the GUID mapping in the AS that announces this IP address. Due to fragmentation in the IP address space, it is possible that the hashed IP address is not announced by any AS. This problem is referred to as the *IP hole problem*. To understand the extent of this problem, we take a close look at today's IP address space. At present, 86% of the 2^{32} IP addresses available in IPv4 are allocated to various entities [88]; the rest are reserved for other purposes including multicast, limited multicast, loopback address, broadcast, etc. Among the allocated addresses, 63.7% of them are announced by one of the

ASs. This leads to an overall 55% announcement ratio over the entire IPv4 address space, which results in a 45% chance that a randomly hashed IP_x will belong to the set of unannounced addresses.

We address the IP hole problem by finding a deputy AS through rehashing if the IP address after the first hash falls into a hole. After $M - 1$ rehashes, if the resulting address still falls into an IP hole, we pick the deputy AS as the one that announces the IP address that has the minimum *IP distance* to the current hashed value. Given two k-bit addresses, A and B, their IP distance is defined as:

$$IP\_distance_{[A,B]} = \sum_{i=0}^{k-1} |A_i - B_i| * 2^i.$$

We further define the IP distance between an address and an address block as the minimum IP distance between that address to all addresses in the block. In this way, we can guarantee that a deputy AS can always be found. There is a concern that the above method may introduce

Algorithm 3: Hashing GUID to address space

```
input  : GUID - the $GUID$ to be hashed
         $M$ - maximum number of rehashing
output: An address guaranteed to be found in prefix table
1  number_of_tries ← 0;
2  result ← hash($GUID$);
3  while (number_of_tries $< M$) do
4      if Longest_Prefix_Matching(result) $> 0$ then
5          return result; //ended here if found
6      // no prefix was found
7      result ← hash(result);
8      number_of_tries ← number_of_tries + 1;
9  //No match found after M hashes
10 nearestPrefixID = findNearestPrefix(result);
11 return An address in nearestPrefixID;
```

load imbalance among ASs: the AS that announces an IP address that is adjacent to a large set of reserved addresses (thus unannounced) may become a popular deputy AS and needs to store a large number of mappings. Fortunately, the probability of reaching an IP hole after M

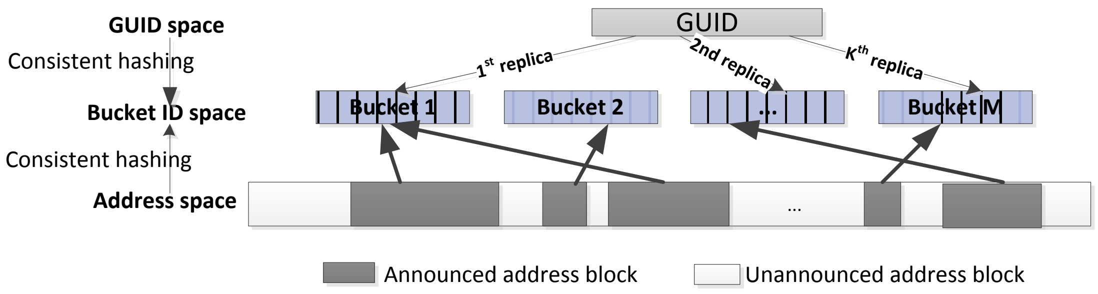


Figure 3.3: Bucketing scheme handling non-contiguous address space issue

hashes decreases rapidly with increasing M. For instance, this probability is as low as 0.034% for $M = 10$. As a result, the chances that we need to resort to the ASs that announce the IP addresses with the minimum IP distances to the holes are very low.

Algorithm 3 summarizes the steps taken by the border gateway to deal with the IP hole problem. Since hashing, rehashing and prefix matching processes are done locally by the border gateway, these operations introduce very little delay to the network.

When extending DMap to other network address schemes, such as IPv6, we need to rethink how we deal with the IP hole problem as these network address spaces may have substantially more holes than used address segments. To address such sparse address spaces, we propose to use a two-level indexing method to index each announced address segment: bucket ID and segment ID within that bucket. Suppose we have N buckets, each with a capacity of S segments. We make N large so that S can be kept small. Given a GUID, we run two hash functions, the first one mapping the GUID to bucket ID, and the other one mapping the GUID to the segment ID. Figure 3.3 illustrates the bucketing scheme.

3.3.3 Spatial Locality and Local Replication

The main advantage of DMap lies in its simplicity: hashing a GUID to a random AS. However, this random placement ignores locality, and so may degrade performance. Having multiple replicas partially addresses this problem, but it still has the inherent problem of a direct hashing scheme: the GUID mappings are stored at faraway ASs when the host and requestor are close to each other. Thus, we enhance the baseline DMap for an expected common case of when a requesting node is attached to the same AS as the GUID that it is resolving. To leverage this *spatial locality*, DMAP stores an additional replica of a GUID mapping at its attached AS. When a host registers/updates its GUID, it creates/updates a local copy (at its attached AS) in addition to creating/updating the K "global" copies. When a node needs to lookup a GUID, it sends out a local and a global lookup simultaneously. When the host and the requester are from the same AS, the local request should lead to significantly reduced lookup latency.

3.3.4 Inconsistent GUID→NA Mappings

BGP Churn: Since a change in the prefix announcements directly influences DMap, we analyze the potential effects of BGP churn. A long term study of BGP churn evolution [89] shows that a major reason for churn in the BGP tables is router configuration mistakes or other anomalies. Changes in prefix announcements occur when an AS withdraws a previously announced prefix or announces a new prefix. The actual rate of new prefix announcement and prefix withdrawal is small, with the former dominating the latter.

When an AS withdraws a certain prefix, all the mappings previously hosted by the AS whose GUIDs are hashed to the withdrawn IP addresses will become inaccessible, resulting in what we call *orphan mappings*. To address this problem, we let the withdrawing AS run the IP hole protocol to find a deputy AS for these mappings before withdrawing. It sends a *GUID insert messages* to the deputy AS and deletes its own copy of the mapping. Subsequent queries will then hit an IP hole. Following the same IP hole protocol, they will reach the deputy AS and find the mapping.

Announcing new prefixes can also result in orphan mappings. The GUIDs that were originally hashed to these IP addresses had followed the IP hole procedure to a "deputy" AS, and announcing these addresses now can make the mappings on the deputy AS orphan mappings. As a result, queries that reach the announcing AS will not find the mappings, while the mappings on the deputy AS become inaccessible. To solve this problem, when the announcing AS receives a query and finds the mapping missing, it sends a *GUID migration message* to the deputy AS to relocate the mapping to itself. This operation could cause a negligible one-time overhead, which only occurs for the first query after the announcement.

Mobility: Mobility can also lead to inconsistencies in DMap. Suppose host X, with GUID

G_x, is connected to AS A. As a result, DMap has the mapping $(G_x : A)$. Then suppose X moves to AS A' at time t_0, and its mapping will be updated to $(G_x : A')$ at time t_1. While we expect $t_1 - t_0$ to be small, it is possible for a querying node to get the old mapping right after X has moved. The querying node will then be unable to communicate with X. In this case, the querying node should mark the mapping as obsolete, and keep checking until it receives an updated one.

Router Failure: An AS can lose part or all of its mappings due to router failure. This is a rare event, but we need to address the resulting complication. If a lookup request reaches an AS, but cannot find the mapping due to this problem, the requester will wait for a timeout. Following the timeout, the requester will contact the next mapping replica (remembering we have K replicas in total). We note that the probability for K Internet routes to fail at the same time is extremely low, and thus our replication strategy also improves system resilience and reliability.

3.4 Evaluation

In this section, we present the results from a detailed performance evaluation of the DMap scheme using a mix of qualitative reasoning and event-based simulation.

3.4.1 Storage and Traffic Overhead

To analyze the storage requirements in absence of specifications about the GUID/NA lengths and related headers, we make the following assumptions. We assume flat GUIDs of length 160 bits, each associated with a maximum of 5 NAs (accounting for multi-homed devices) of length 32 bits each. 32 bits of additional overhead per mapping entry is assumed which could include type of service, priority and other meta information. Each mapping entry thus has a size of 160 + 32x5 + 32 = 352 bits. We assume a total of 5 billion GUIDs, roughly equal to the present number of mobile devices, and a replication factor of $K = 5$. Based on the average prefix announcement by individual ASs as determined from a current snapshot of the BGP table [88], the storage requirements per AS, assuming proportional distribution, is only 173 Mbits. This storage requirement is quite modest, even if it is multiplied several times to include non-mobile devices as well as future growth.

The update traffic overhead is also a key parameter of interest in ensuring scalability. The DMap technique reduces the traffic overhead in comparison to other mapping schemes by: (a) Ensuring a single overlay-hop path to a storage location, (b) Not adding any table maintenance traffic as required in DHT schemes. Using a broad estimate of the 5 Billion GUIDs being those of mobile hosts which update their GUID$\rightarrow$NA mapping at an average rate of 100 updates/day, the world-wide combined update traffic would be $\sim$10 Gb/s, a minute fraction of the overall Internet traffic of ~ 50x10^6 Gb/s as of 2010 [83].

3.4.2 Query Response Time and Load

The round trip response time of a query is composed of: (i) K longest prefix matchings at the local gateway router, (ii) network latency between the query source and the chosen destination AS, (iii) the queuing and processing delay of the mapping server at the destination AS and (iv) the return network latency between the destination AS and the query source. Since routers use fast longest prefix match algorithms, requiring on order of 100 instructions per lookup, i.e. $\sim$30 nanoseconds on a 3 GHz processor [90], we ignore this component in our evaluation. Also sufficient resources are assumed at the mapping server to make the queueing and processing delay very small compared to the round trip latency. Note that this process does not add any delays to the normal data packets as the steps described above are only applied to GUID query packets and are assumed to be handled at a separate compute layer at the gateway router.

3.4.2.1 Simulation Setup and Input Workloads

We develop a discrete-event simulator consisting of $\sim$26000 nodes, each emulating an AS. The connectivity graph of the network, inter-AS and intra-AS connectivity latencies, and the announced IP prefix list are derived from measurement driven data as described below. We consider three types of events: GUID inserts, GUID updates and GUID lookups.[1]

We use the AS-level topology of the current Internet as our network model by extracting the following real-measurement data from the DIMES database [92]: (i) Connectivity graph

[1]The source code for our simulator is available at [91]

containing 26,424 ASs and 90,267 direct links between them, (ii) Average end-to-end latencies between each pair of AS and within each AS. The DIMES database provides end-to-end median latency for about 9 million pairs of hosts which are either within the same AS or in different ASs. From this dataset, we extract the average inter-AS and intra-AS latency since we only work with an AS-level network topology in our simulation. Due to the inherent incompleteness of real-trace data, intra-AS latency numbers are not available for about 6% of the ASs that are involved in the storage or transit of the mapping data. For these ASs, we use the median value (3.5 ms) of the set of available intra-AS latencies as a working solution.

Since our scheme allocates GUIDs to ASs according to the prefix announcements, we use a complete list of IP prefixes advertised in the Internet default free zone (DFZ), as seen by APNIC's router at DIX-IE in Tokyo, Japan [88]. This dataset consists of roughly 330,000 prefixes spanning close to 52% of the 32 bit IP address space which is consistent with recent estimates [89] about the size of the prefix tables in DFZ routers. We confirm our results with two other prefix tables taken from BGP routers in the continental USA and Europe respectively and observe similar trends.

To discard any location bias and to incorporate the global scale of operation, we use another dataset from DIMES that contains the number of end-nodes connected to ASs to characterize the distribution of the source of GUID insert and query. Each GUID in our simulation originates from a randomly picked source AS, where the probability of choosing a certain AS is weighted in proportion to the number of end-nodes found in that AS.

The number of queries for any GUID depends on its popularity amongst Internet hosts. In order to capture the effects of the wide variations in host popularity, we use a Mandelbrot-Zipf distribution [93, 94] to model the varying host popularity. The Mandelbrot-Zipf distribution defines the probability of accessing an object at rank k out of N available objects as:

$$p(k) = \frac{H}{(k+q)^{\alpha}}, \tag{3.1}$$

where $H = 1/\sum_{k=1}^{N} 1/(k+q)^{\alpha}$, with α determining the skewness and q affecting the "flatness" of the peak. We use a value of $\alpha = 1.02, q = 100$ following the arguments in [94].

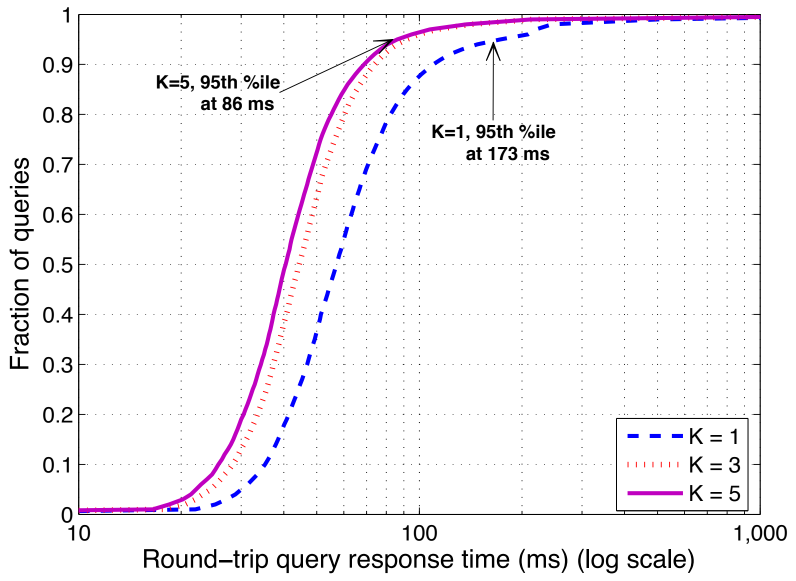


Figure 3.4: Round trip query response times

3.4.2.2 Evaluation Results

We present two sets of results that characterize the query response time and the load distribution of our scheme respectively.

Query Response Time: We evaluate the query response time for DMap by inserting 10^5 GUIDs and generating 10^6 queries according to the popularity model. By repeated trials with increasing number of GUIDs/queries, we verified that the response times converged after reaching the above configuration and larger numbers are not necessary. When we store a mapping at multiple locations, i.e., $K > 1$, in the results below, we assume that the querying node has sufficient information to choose the location with the lowest response time. We note that in today's Internet, this information is only partially available, but at the least, each AS has hop count information for reaching all other ASs through the routing protocol. Using least hop count instead of lowest response time leads to similar results albeit with marginally increased latencies. We would also emphasize that many techniques are being proposed to better estimate the response times [95].

Figure 3.4 plots the cumulative distribution function (CDF) of the round trip query response times with varying K values. We make two observations. First, with $K = 5$, 95% of the queries complete within 86ms. This is well within the range needed for voice call handoffs [13]. Indeed, given that many WiFi and IP handoff protocols are often on the order of 0.5-1 second [96, 97], DMap updates would not introduce an undue additional burden. Second, storing each GUID mapping in multiple locations can significantly reduce query response times, as it allows a querying node to choose the replica that is "closest" to itself, thus addressing the locality of the requests. The effect of increasing K can be clearly seen with the leftward shift of the CDF curve as we increase the value of K. In particular, the mean, median and 95^{th} percentile query latencies of $K = 1$ and $K = 5$ cases are tabulated in Table 3.1, which shows a marked decrease in the tail of the response time distribution.

However, even the curve for $K = 5$ has a relatively long tail. This long tail arises from a few queries originating from those ASs with unusually long intra-AS response times, according to the DIMES dataset. For example, the 18 queries with the longest response times all originated from AS 23951, a small AS registered in Indonesia with a one-way latency of more than 2.3 seconds on each of its outgoing links.

Impact of BGP Churn: The above study assumes that the BGP table at the query origin exactly reflects the current state of the Internet. However, BGP tables at different places in the Internet can be inconsistent because of new prefix announcements or prefix withdrawals. This inconsistency may have an adverse impact on the overall query response times as the query may reach an AS which does not host the requested mapping. In this situation, the AS will then reply with a "GUID missing" message, and the querying node will have to contact another replica. Thus, a query may require multiple round-trips to different ASs for each resolution. We note that the probability of two churns occurring at the same time for the same GUID is

K	**Round Trip Query Response Time (ms)**		
	Mean	**Median**	**95th percentile**
1	74.5	57.1	172.8
5	49.1	40.5	86.1

Table 3.1: Query Response Time Statistics for $K = 1, 5$

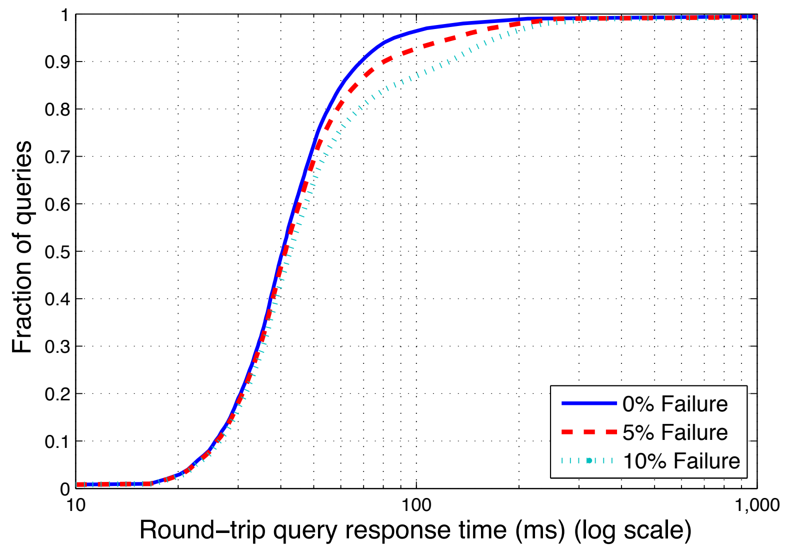


Figure 3.5: Effect of BGP Churn on query response times.

negligible. Here, we conduct a set of experiments to quantify the impact of this inconsistency caused by BGP churn. In the experiments, we vary the percentage of prefixes that are newly announced or withdrawn from 0 to 10%. Figure 3.5 plots the CDF of the query response times for $K = 5$ and 0% to 10% lookup failures. A 5% failure rate, which already seems pessimistic according to [98, 99], shifts the median and 95th percentile from 40.5ms and 86.1ms to 41.3ms and 129.1ms, respectively.

Storage Distribution: We next study the distribution of GUID$\rightarrow$NA mappings amongst ASs to evaluate DMap's ability to spread the storage load proportional to the size of the ASs. We measure storage load using the *Normalized Load Ratio* (NLR) at each AS, which is defined as the ratio of the percentage of GUIDs assigned to an AS divided by the percentage of IP addresses advertised by that AS. For example if an AS announces a $/8$ prefix, corresponding to 0.39% of the 32 bit IP space and is assigned 20,000 out of a total of 1 Million GUIDs, i.e., 2% of GUIDs, then its normalized load would be 2/0.39 $\simeq$ 5. Ideally, each AS's NLR would be 1.

Figure 3.6 plots the CDF of the NLR when we inserted from 10^5 to 10^7 GUIDs, with $K = 5$. We observe that for 10^7 GUIDs, 93% of the ASes had NLRs between 0.4 and 1.6. Further, we observe that as the number of GUIDs increases from 10^5 to 10^7, the CDF becomes much

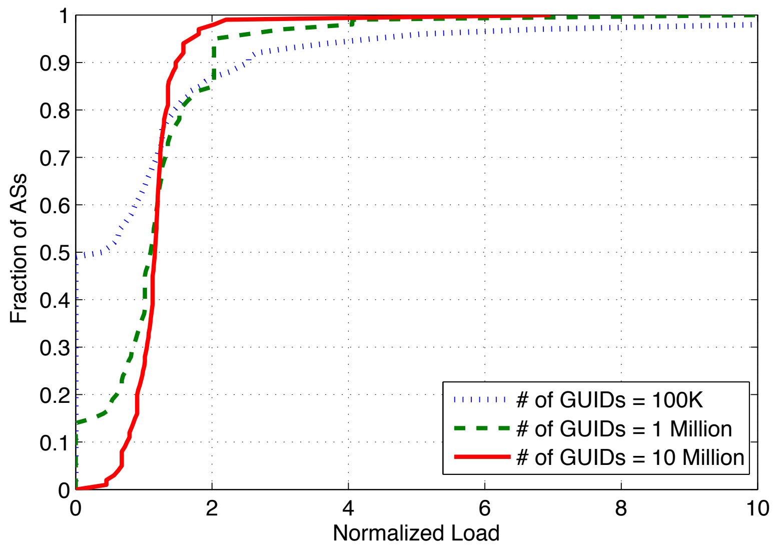


Figure 3.6: Normalized Load Ratio per AS

sharper around NLR equal to 1 (with a shorter tail). This suggests that DMap can distribute the storage load better when the system scales. These results show that DMap does a very good job of spreading out the storage load proportional to the percentage of the IP space that an AS claims.

Interestingly, the median NLR value is 1.16. The fact that the median NLR value is greater than 1 is expected since in addition to its fair share of GUIDs, many ASs are also allocated a portion of the GUIDs that has hashed values after M retries falling in the IP holes as described in Section 3.3.2.

3.5 Analytical Model for Query Response Time

In this section we present an analytical model for an upper bound on the query response time and parametrically study its dependence on the Internet topology and replication factor K. While the simulation framework of Section 4.5 shows the response time performance of DMap based on the current Internet topology, this analytical model allows us to estimate its performance based on a predicted future Internet model.

3.5.1 The Jellyfish Model

An accurate parametric model of the Internet topology is known to be a difficult endeavor due to the inherent complexities in routing policies, detour paths and limited visibility of the intra-AS structure [100]. The Jellyfish model, however, has been found to closely follow the evolution of the Internet topology at a relatively coarse scale [101]. In this work, we build a Jellyfish model based on the PoP-level Internet topology. We first label the node with the highest degree as the root v_0 and the maximal clique[1] containing v_0 as the *core*, denoted as $Shell$-0. Let v be a non-root node and j a non-negative integer. The smallest path length[2] from v to a node in the core is its *distance to the core*. We use $Shell$-j to denote the set of nodes whose degree is more than 1 (intermediate nodes) and whose distance to the core is j. We use $Hang$-j to denote the set of nodes whose degree is 1 (leaf nodes) and whose distance to the core is $j + 1$. These are standard notations in the Jellyfish model [100]. Then we have

$$Layer(j) = Shell\text{-}j \cup Hang\text{-}(j-1) \text{ for } j \geq 1,$$

and $Layer(0) = Shell$-0. We further denote the total number of layers in the Internet PoP topology as N and the percentage of nodes in layer j by r_j; if n is the total number of nodes in G, then $r_j = |Layer(j)|/n$. The separation of one degree nodes at each layer distinguishes between stub connections and transit connections which makes the model much closer to the Internet topology than a standard tree structure.

3.5.2 Upper Bound for Query Response Times

Following the above model, if we assume no peer links between the nodes inside each layer, then the distance between any two nodes s and t (in layers j_s and j_t respectively), $d(s,t)$, is at most $j_s + j_t + 1$. Note that since the core forms a completely connected graph, all hops in the core are of length one. To drive a simple parametric upper bound for the query response time, we assume that the network address space is uniformly distributed among the PoPs and all addresses within a PoP behave in an identical fashion. Following the algorithm described in Section 3.3, let the source of a GUID query belong to PoP s and let $t_1, t_2, \ldots, t_K$ be the

[1]clique: a completely connected subgraph of G.

[2]path length: the number of edges in the path, $i.e.$, that of the involved PoPs in the path minus 1.

destination PoPs for the query determined by the K hash functions $h_1, h_2, \dots, h_K$ applied on the GUID G. Assuming a linear relationship between PoP path length and response time, the query response time $\tau(s, G)$, is thus given by

$$\tau(s, G) = c_0 \cdot \min_{1 \le i \le K} d(s, t_i) + c_1, \tag{3.2}$$

where c_0 and c_1 are constants. In order to average over all possible source and destination PoPs, we treat $d(s, t_i)$ as a random variable and find its probability distribution based on the r_j values defined in the previous subsection. In the analysis, we use the standard notations $Pr(\cdot)$ and $Pr(\cdot \mid \cdot)$ for the probability and conditional probability of argument events, and $\mathbb{E}(\cdot)$ for the expected value of a random variable.

Given a uniformly selected source PoP, we have $Pr(s \in Layer(j)) = r_j$. Note that since DMap actually chooses a network address uniformly over all possible addresses, the j^{th} layer could include a different number of addresses than the ratio r_j. Our analysis assumes the uniform distribution of addresses among PoPs but the model can be easily extended to non-uniform distributions by considering weights w_s proportional to the number of addresses in PoP s. Since accurate estimates of such a distribution is not directly available through any of the Internet measurement frameworks, we assume $w_s = 1$ for all s. Based on the same assumptions, we have $Pr(t_i \in Layer(j_i)) = r_{j_i}$ for each $i = 1, 2, \dots, K$ and $j_i = 0, 1, 2, 3, 4, \dots, N-1$. Thus the conditional probability of $d(s, t_i) \ge l + 1$ given that $s \in Layer(j)$ is at most the percentage of nodes in $Layer(l-j) \cup Layer(l+1-j) \cdots \cup Layer(N-1)$. (Since s is in $Layer(j)$, $d(s, t_i) = l + 1$ when t_i is in $Layer(l-j)$, $d(s, t_i) = l + 2$ when t_i is in $Layer(l+1-j)$ and so on in the worst case). In other words,

$$Pr\left(d(s, t_i) > l \;\middle|\; s \in Layer(j)\right) \le p_{j,l},$$

$$\text{where } p_{j,l} \overset{def}{=} r_{l-j} + r_{l+1-j} + r_{l+2-j} \cdots$$

$$\Rightarrow Pr\left(\min_{1 \le i \le K} d(s, t_i) > l \;\middle|\; s \in Layer(j)\right) \le p_{j,l}^K,$$

$$\Rightarrow Pr\left(\min_{1 \le i \le K} d(s, t_i) \le l \;\middle|\; s \in Layer(j)\right) > 1 - p_{j,l}^K,$$

$$\Rightarrow Pr\left(\min_{1 \le i \le K} d(s, t_i) \le l\right) > \sum_{j=0}^{N-1} r_j (1 - p_{j,l}^K).$$

This provides an upper bound for the CDF of average distance from the source to the closest

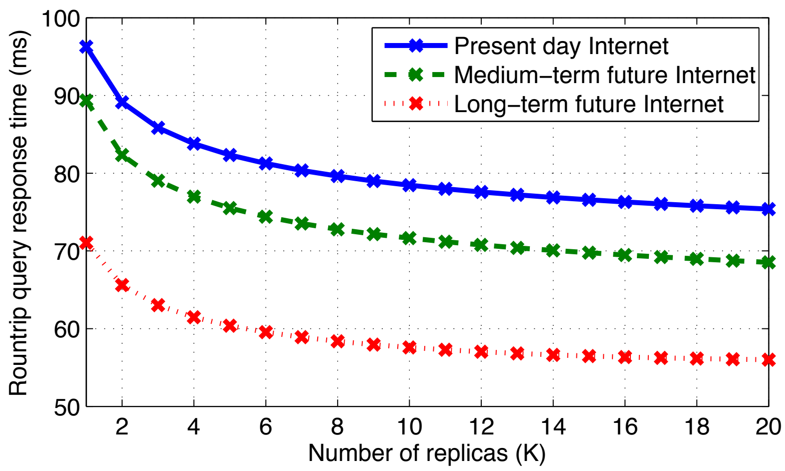


Figure 3.7: Analytical upper bound of query response times with the Internet topology evolution.

destination. Thus the probability that $\min_{1\le i\le K} d(s,t_i) > l$ is at most $1-q_l$, where we define q_l as:

$$q_l \overset{def}{=} \sum_{j=0}^{N-1} r_j \left(1-p_{j,l}^K\right),$$

Finally, noting that the diameter of our PoP graph is $(N-1)+(N-1)+1 = 2N-1$ or less,

$$\begin{aligned} \mathbb{E}\left(\min_{1\le i\le K} d(s,t_i)\right) &< \sum_{l=1}^{2N-1}(1-q_l) \\ \Rightarrow \mathbb{E}\left(\tau(s,G)\right) &< c_0\left(\sum_{l=1}^{2N-1}(1-q_l)\right)+c_1. \end{aligned} \tag{3.3}$$

3.5.3 Analytical Results

We use the formulation derived above to study the response time upper bound in three different scenarios with varying number of replicas K. The first scenario reflects the current Internet topology, for which we use measured data from the iPlane project [102] for setting the parameters $r_j, j = 1,2,\ldots,N$. The data set shows a graph of 193,376 nodes within 8 layers and more than 60% of the nodes residing in layers 3 and 4. The next two scenarios model the medium-term and long-term future Internet topologies. In order to come up with models for the future Internet, we leverage the following two distinct trends observed from the widely regarded CAIDA measurement framework [103]: (i) The number of nodes are growing almost linearly with time, (ii) The topology graph is getting flatter with time, i.e. ASs are obtaining more direct paths to the core. Extrapolating these trends, we model the medium-term (5-10 years) future Internet as having 20% more nodes than present contained in 6 layers. Similarly, the long-term (25-30 years) future Internet model contains double the number of nodes contained in 4 layers. Figure 3.7 shows the analytical upper bound of average query response time for the three scenarios using the measured least squared error values for $c_0, c_1 = 10.6, 8.3$. The plot shows that based on the predicted future Internet topology, response time upper bounds for DMap queries become smaller with the evolution. Also, the analysis clearly indicates that increasing the replica number results in diminishing returns beyond a few replicas. We note that actual values for the query response upper bound will typically be smaller than the ones obtained in this analysis, since we did not consider the presence of peering links between nodes in the same layer.

3.6 Related Work

Given the importance of locator/identifier separation schemes in both current and future networks, various architectures for mapping identifiers to locators have been proposed and studied. Most of the early mapping schemes [86, 87, 76, 77] assumed aggregatable identifier spaces and proposed ideas based on that vantage point. However, this assumption is too restrictive making such schemes not applicable to many recent mainstream proposals such as HIP [19], AIP [73] and MobilityFirst [16] which propose flat identifiers. Our approach, in contrast, targets a flexible resolution service by not making any assumptions about identifier hierarchy or locator structure.

There are some recent mapping architecture proposals that incorporate flat identifier space such as DHT-MAP [104], SLIMS [105]. However these approaches either incur high lookup latency, making it not applicable to highly mobile environment, or high management overhead which limits scalability. For example, the DHT based scheme in [104] can entail up to 8 logical hops introducing an average latency of about 900ms as per their assumptions.

In contrast, our scheme aims for much lower latencies by employing the one-hop hashing approach and ensures minimum management overhead for feasible deployment on a global scale. We argue that making use of network entities and the IP reachability information already available through the underlying routing infrastructure provides a practical and scalable approach to realize mapping resolvers. Reference [106] uses a similar in-network hashing scheme to target the different but related problem of name-based routing in enterprise networks.

This work also focuses on a global-scale simulation to validate the design, which has been neglected in most of the prior works referenced above. Reference [76] is a recent exception which presents a trace based simulation using the iPlane dataset [102]. Our simulation approach is more realistic than that of [76] on two counts: (a) We use a larger dataset from DIMES [92] to extract AS level connectivity and latency information. The DIMES dataset is based on measurements from $\sim$1000 vantage points compared to $\sim$200 for iPlane, resulting in information for about twice the number of ASs as compared to iPlane; (b) To generate resolution lookup events, [76] uses DNS lookup traces from two particular source locations which introduces a significant locality bias in their results. In contrast, we globally distribute lookup source locations by weighting the chances of choosing a particular source location (source AS) in proportion to the available data on number of end nodes near that location. The basic intuition here is to mimic realistic deployment where more lookup requests will be generated from more densely populated areas.

3.7 Concluding Remarks

In this chapter, we presented the concept, design and evaluation of DMap, a scheme for low latency, scalable name resolution service in the future Internet. DMap distributes name to address mappings amongst Internet routers using an in-network single-hop hashing technique that derives the address of the storage router directly from a flat, globally unique identifier. In contrast to other DHT-based techniques, DMap does not require any table maintenance overhead since we use network level reachability information already available through existing routing protocols. In addition, DMap supports arbitrary name and address structures making it more widely applicable than prior techniques. Through a large-scale discrete-event simulation, we show that the proposed DMap method achieves low latencies with a mean value of $\sim$50 ms and 95th percentile value of $\sim$100 ms and good storage distribution among participating routers.

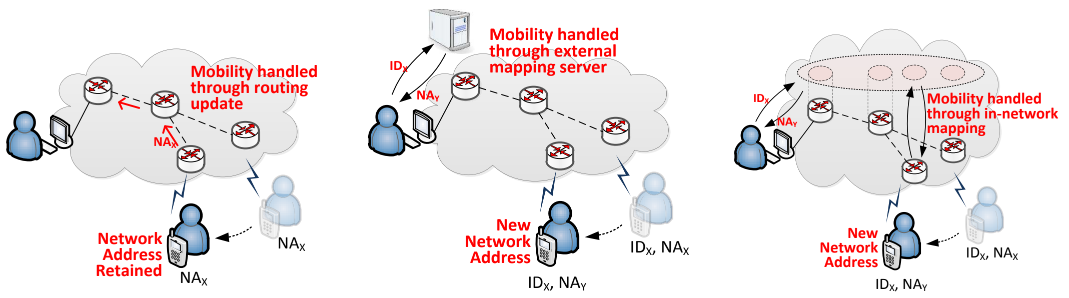
In further work, we plan to consider other variations of the proposed DMap distribution scheme - for example GUIDs can be hashed directly to AS numbers or allocation sizes can be varied to reflect economic incentives at ASs. We also plan to extend the scope of this work by studying a feasible in-network caching methods that builds on top of the basic DMap scheme. Since our scheme interacts with the hosts, the inter-domain routing protocol and the Internet routers, security is a critical requirement at each level. The MobilityFirst project [16], takes a holistic approach towards self certification based security, which tie in well into the relevant aspects of our scheme. Our future work plan also includes incorporating the transient effects of BGP updates, misconfigurations and router failures.

On the validation and evaluation front, there is an ongoing effort to implement a proof-of-concept global scale DMap system using the GENI (global environment for network innovation) framework. A first DMap prototype was demonstrated at the GENI Engineering Conference-12 in Kansas City and efforts are currently under way to fully instrument the latency and overhead measurements necessary to evaluate scalability and performance. If the GENI experiments successfully confirm DMap performance, there are also further plans to use the proposed technique as part of a complete identifier-based protocol stack in the MobilityFirst future Internet architecture project.

Chapter 4

Edge-aware Interdomain Routing Protocol

In recognition of the fact that wireless access and mobility at the edge also have implications for routing between networks and some of the resulting requirements cannot be met by current BGP solutions, we propose a Edge-aware Interdomain Routing protocol. Our objective for this work is to explore and understand Internet scale routing mechanisms in view of emerging mobility services and use the results to influence evolving standards rather than to change BGP in a single step. This component of the MobilityFirst architecture is a clean-slate inter-domain routing protocol designed to meet the needs of the future mobile Internet. In particular, we propose EIR (edge-aware inter-domain routing) as a general solution for a range of mobility-related requirements including device and network migration, cross-domain end-host multi-homing, global roaming agreement setup and wireless edge peering. The EIR protocol provides enhanced information about network topology and edge network properties in order to enable networks across the Internet to make better routing decisions than currently possible with BGP. This is accomplished with a telescopic network state dissemination protocol which makes the entire network graph visible while keeping the routing overhead within limits. EIR enables autonomous systems to optionally expose internal network topology and aggregate properties such as bandwidth, availability and variability thus enabling corresponding networks to select paths which take into account both service requirements (such as dual-homing or multicast/anycast) and edge network constraints (e.g. LTE vs. WiFi). Further, EIR is designed to work in conjunction with late binding of names to addresses and in-network storage in order to provide robust services in environments with dynamic mobility and disconnection. The design of the EIR is given along with sample use cases (i.e. host mobility, multi-homing, multicast and edge peering) to further explain the benefits of the protocol. This is followed by a two-stage evaluation of EIR including an Internet scale simulation model to verify scalability, and a 200


Mobility handled through routing update
NAX
Network Address Retained
NAX
NAX
Mobility handled through external mapping server
IDX
NAY
New Network Address
IDX, NAY
IDX, NAX
IDX
NAY
Mobility handled through in-network mapping
New Network Address
IDX, NAY
IDX, NAX

(a) Dynamic routing approach (b) Mapping through external service (c) Mapping through in-network service

Figure 4.1: Alternative approaches to handling mobility: (a) Moving end-host retains network address, routing updates ensure delivery. (b) End-host obtains new address, identifier-locator mapping through external service. (c) Mapping through in-network service, routers along the path can re-map bindings.

node experimental ORBIT emulation to validate the protocol design and provide experimental results for selected usage scenarios.

4.1 Introduction

This chapter presents the design and evaluation of a new interdomain protocol for the future mobile Internet. The proposed Edge-Aware Interdomain routing (EIR) protocol was developed as a part of the "MobilityFirst" future Internet Architecture project [107] aimed at a clean-slate redesign of the IP protocol architecture. The MobilityFirst project has a particular emphasis in designing both intra- and inter-domain routing protocols to efficiently support mobility requirements at the edge. In earlier work [108, 109], we have proposed and extensively validated the GSTAR (generalized storage aware routing) protocol as a solution for intra-domain routing in wireless/mobile edge networks. EIR proposed here was motivated by a recognition of the fact that wireless access and mobility at the edge also have implications for routing between networks and some of the resulting requirements cannot be met by current BGP solutions. Our objective for this work is to explore and understand Internet scale routing mechanisms in view of emerging mobility services and use the results to influence evolving standards rather than to change BGP in a single step.

As a motivating example, consider the increasingly important “hetnet” mobile service scenario in which a mobile device may be simultaneously connected to a dynamically changing set of cellular and WiFi networks. It is possible to consider a variety of service objectives for this scenario ranging from “most economical” to “best interface” to “all interfaces”. Since the cellular and WiFi networks will in general be in different Internet domains, routers need to have visibility of the network graph and some awareness of edge network properties in order to make informed forwarding and/or multicast copy decisions. Another emerging use case involves network mobility in which an entire network may migrate from one location to another, indicating the need for a robust routing protocol which is tolerant to disconnections and rapidly changing network graphs at the edge. Clearly, these new requirements are not easily addressed via minor modifications to BGP, indicating the need for a fundamentally different routing approach that leads to improved topology visibility and a degree of wireless edge awareness.

The EIR protocol we propose here is based on the following key features: (1) “aNodes” and “vLinks” as abstractions for optionally exposing aggregated internal network topology; (2) telescopic flooding of network state packets in order to limit routing overhead; (3) late binding of object names to network addresses; and (4) support for specification of a broad range of routing policies. The aNode and vLink abstractions are inspired by the Pathlet routing protocol [110] and are intended to provide a mechanism for autonomous systems (ASs) to optionally express some details about their internal graph and wireless edge properties. The use of network state packets with telescopic flooding is meant to provide fast updates to nearby networks while eventually providing all networks with a global view of the network topology. Late binding is a key element of the design which is predicated on the use of names or “globally unique identifiers” (GUIDs) to identify all network attached object, along with a logically centralized global name resolution service (GNRS) for dynamic binding of names to network addresses (see refs [107, 111] for further details on these components of the MobilityFirst architecture). In our proposed architecture, any router in the network can optionally query the GNRS for an updated name to address binding, thus enabling packets to be delivered correctly even when the routing protocol cannot keep up with the pace of dynamic changes at the edge. Increased capabilities such as edge awareness or late binding in network routers imply the need for enhanced policy specification capabilities that apply to services such as mobility, multi-homing and multicast.

The main contributions of this work are summarized as follows. First, we present a clean-slate inter-domain routing framework designed to support the emerging needs of mobile wireless services at the edge while also maintaining capabilities currently associated with BGP. The proposed EIR protocol is based on telescopic network state updates designed to provide network graph visibility and edge awareness without excessive routing overhead. EIR incorporates "aNode" and "vLink" abstractions that enable networks to optionally expose some of their internal structure in an aggregated manner. The protocol also takes advantage of late binding between names and addresses in order to deal with fast changes in network topology. We identify several use cases (mobility with disconnection, multi-homing, multicast, edge peering) to demonstrate the value of the EIR protocol. EIR is validated using both large-scale simulation and ORBIT testbed emulation with hundreds of routing nodes, and results are presented for metrics such as packet delivery rate, delay and routing overhead for representative use cases.

4.2 Key Principles and Techniques

EIR is designed for architectures which separate the identifier and locator roles of IP addresses. This separation principle is used in several recent proposals (such as LISP [79], HIP [112], AIP [113], MobilityFirst [107], XIA [114]), and is being increasingly deployed by autonomous systems as per a recent measurement report [115]. We use the terminology defined in MobilityFirst [107]: self-certified location-independent end-host 'names' or identifiers are called Globally Unique Identifiers (GUIDs), and the locators are called Network Addresses (NAs). While this split architecture itself provides support for end-host mobility (by dynamically mapping the GUID of a mobile node to NAs corresponding to the current point(s) of attachment), in this work we show the need for supporting wireless links and mobile nodes in the inter-domain routing protocol.

4.2.1 Design Principles

Our design decisions are directed towards enabling and using (i) more information about links (e.g. whether wireless or wired link between networks), and (ii) information about more links (e.g. internal structure of the AS). Here we first present the top-level design principles behind EIR.

In-network mapping of names to addresses: A major point of differentiation of the different split architectures mentioned above is their implementation and use of the name to address mappings. As shown in Fig. 4.1, the mapping infrastructure can either be hosted as services external to the network layer and be accessed only by end-nodes, or alternatively be implemented in-network and be accessible at the network layer by both end-hosts and routers. We make use of the in-network mapping approach for ensuring delivery of packets in the case of fast end-host mobility as depicted in Fig. 4.1(c). Several past works have shown the feasibility of Internet-scale, distributed, in-network mapping infrastructure with extremely small query-response times [116, 111].

Propagating link-level information in inter-domain routing: BGP does not differentiate between wired and wireless inter-network links, making it difficult to make routing decisions based on capacity constraints. For example, in an early in-flight WiFi implementation, Boeing associated each flight with an IP address block which was announced into the global routing system from different locations as the plane moved [117]. Other networks receiving such announcements had no idea that the last hop for this path had a ground-to-plane wireless link instead of the usual high-capacity peering-point wired link and thus might have sent excess traffic towards this network without realizing the capacity constraint. In EIR, coarse-grained link-level information about each inter-network link is propagated through the routing protocol to enable networks to make forwarding decisions based on aggregate edge network properties.

Increased visibility of alternative paths: More often than not, there are multiple routes available between any two networks in the Internet and those routes can entail vastly different properties [118]. In BGP, a network might learn about many routes to a destination but can only select and propagate *one* best route to other networks, which leads to a myopic view of the network graph. In order to support the increasingly common use-cases of multi-path, multicast and multi-network operations, EIR entails network-wide visibility of multiple possible paths between each pair of networks. In addition, EIR incorporates a mechanism that allows networks to realize policy routing beyond common routing policies as seen today in BGP. Section 4.3.3 discusses how EIR handles routing policies in detail.

Flexibility in exposing internal structure: EIR enables flexibility in the amount of internal network structure that a domain announces to the other networks. Traffic engineering, differential network services and source routing can be done more easily and efficiently when networks have *more fine-grained* view of multiple possible paths; i.e. not only AS-level paths but also intra-AS level paths. The flexibility of network boundaries enables efficient support for network mobility and facilitates virtual network formation.

4.2.2 Enabling Techniques

The EIR protocol combines the following four techniques in order to support the requirements of mobile nodes and networks.

1. Network-wide visibility of aggregated nodes: In EIR, the granularity of implementation of the inter-domain routing protocol is changed from AS-level to *aNode*-level, as shown in Fig. 4.2. We define aNode or aggregated node as a set of routers belonging to the same AS which share certain common properties. Each AS can group routers into one or more aNodes based on internally relevant criteria (e.g. routers in geographic proximity or a group of wireless routers deployed in a campus can constitute an aNode). ASs can dynamically change the number and composition of aNodes that it exposes to other networks. Connectivity between aNodes is represented in terms of virtual links or *vLinks*. Aggregated information about aNodes and vLinks in a nework are disseminated to every other AS in order to provide a complete view of the network graph. As elaborated in Sec. 4.3, the aNode and vLink abstractions allow each AS to flexibly control the amount of internal structure exposed to other networks.

2. Telescopic state dissemination for scalability: While having richer network states helps routers to make better routing decisions, it comes at the cost of scalability. In particular, if each routing update is flooded to every other AS in the network, the resulting traffic overhead would clearly make the system unscalable beyond 1000's of nodes. Hence, the design challenge is how to control the amount of state being disseminated to minimize the overhead. That brings the second key technique: *telescopic route dissemination*. The basic idea of this dissemination technique is to dampen the flood emanating from a certain node as it progresses through the network by algorithmically varying the amount of time a node holds an update message before forwarding it to its next-hop neighbors. This hold time at a given node receiving the update is set as a function of the the hop count between that node and the source of the update - larger the hop count, greater the hold time. This results in all nodes eventually getting updates from all other nodes, but with distant nodes having less up-to-date information about each other. The term ‘telescopic’ comes from the analogy of distant nodes seeing each other through the reverse-end of a telescope, i.e. they are visible but less clearly so. Details about the type of telescopic functions used in EIR and further intuitions behind this approach is provided in Sec. 4.3.2.

3. Late-binding for mobility support As a side effect of telescopic route update dissemination, network states that a network observed from far away could be obsoleted during transit of the packet and hence result in routing failure. To address this issue, EIR incorporates the additional design feature of *spatially late name-to-address binding*. Late name-to-address binding serves as a fail-safe mechanism that allows routers actively react to link variations and mobility of end nodes as well as networks. In particular, EIR makes use of a fast in-network global naming resolution service, GNRS [111], to retrieve the current network location of the destination.

4. Explicit policy expression through link and network attribute

EIR handles routing policies by separating route classification from route selection on the control plane, sharing the same idea with Morpheus [118]. Networks express their policies for individual links by announcing their set of routing criteria which is then disseminated through out the whole Internet. Each policy objective is represented as a separate attribute. For example, business relationship could be one attribute with three possible values as “customer”, “peer”, and “backup”. When computing the routing and forwarding table, each network uses a weighted-sum decision process to balance trade-offs among different objectives (described in Section 4.3.3)

4.3 EIR Protocol Design

While BGP is sufficient for basic inter-domain routing with static ASs, it trades flexibility in route selections and router-level link state information for a high level of abstraction and scalability. In contrast, we argue for a more balanced architecture that reveals enough internal state

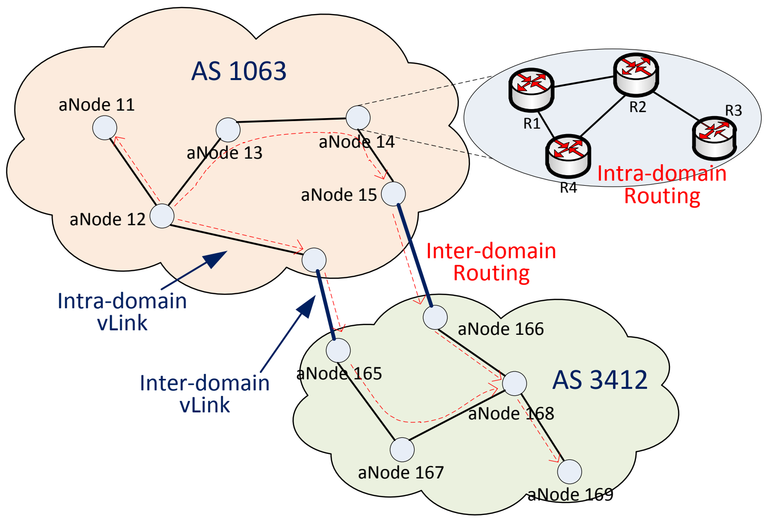


Figure 4.2: Overall architecture of *aNodes* and *vLinks*

of the network so that network entities can make a smarter decision in message delivery, satisfying different requirement of today's services, but also have flexible aggregation capability to make the architecture scalable. We contend that a network entity that wants to deliver a packet to a far away destination does not need to know the most up-to-date states around that destination node until the packet gets closer to the destination. Merely knowing the existence of possible paths and the approximate condition of paths connecting the two endpoints is useful to make smarter routing decision.

The overall architecture is illustrated in Figure 4.2. Each AS has an option of dividing its routers or other networked entities (such as access points and base-stations) into one or more than one group (called aggregated nodes or *aNodes*). Entities belonging to the same aNode typically share some operational or physical attributes. aNodes are connected via *vLinks*, which are a single-hop or multiple-hop connection. Both aNodes and the vLinks between the aNodes are characterized by a set of property values. These property values constitute the link state packet that is propagated from one aNode in the system to all other aNodes. For example, in Figure 4.2, the announcements from aNode 12 in AS 1063 is propagated to all aNodes in both the networks; however the rate of propagation of such updates decreases as the updates move further out from that aNode (this method of routing dissemination is implemented through the use of telescopic functions as described in Section 4.3.2). Thus aNode 13 receives updates from aNode 12 far more often than aNode 169, which is several hops away. In the following subsection, we describe the key design features of EIR including its building blocks of aNodes and vLinks(4.3.1), route dissemination protocol with telescopic technique (4.3.2), routing policy handling(4.3.3) and route selection (4.3.4).

4.3.1 Building Blocks

4.3.1.1 Aggregated node (aNode)

An aggregated node (or aNode) is defined by an AS to represent the internal structure of the AS or to group a set of networked entities with some common properties for management purposes. As examples, possible compositions of aNodes include: the entire AS; group of routers in a geographical area; all routers that support flow-based routing (for example through OpenFlow); wireless routers on bus/train/plane networks; and cell-site routers in flat LTE networks. Note that a single physical router could belong to different aNodes simultaneously. There are two main motivations behind defining new aNodes in an AS: (i) to achieve traffic engineering goals by exposing the internal structure of the AS, and (ii) to express the edge network information so that traffic can be delivered to the hosts connected to the aNode more efficiently. An AS creates an aNode by assigning a globally unique aNode number to the router(s) belonging to the aNode. Similar to the way AS numbers are currently assigned, in EIR each AS is assigned a range of aNode numbers with the size of the allocated range being proportional to the size of the AS. Internal to the aNode, any addressing and routing scheme can be used to suit the needs of the network. For example, one network might choose to have a flat addressing scheme for seamless intra-network mobility support while another might have a hierarchical name space for higher scalability.

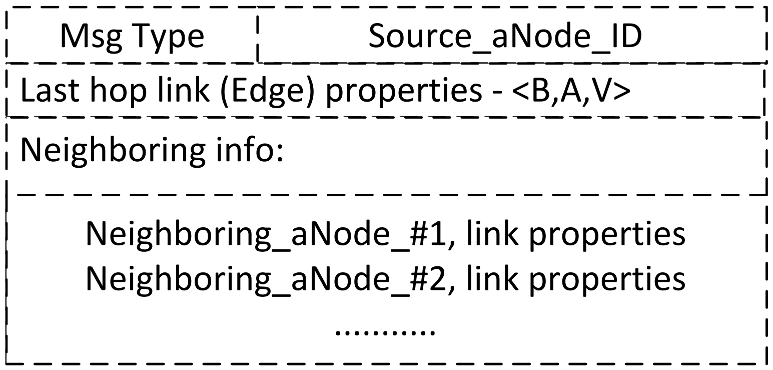
4.3.1.2 Virtual link (vLink)

The link connecting two aNodes or a sequence of aNodes $a_1 \rightarrow a_2 \ldots \rightarrow a_n$ is termed as vLink. Similar to aNode, the notion of vLink provides an abstraction for physical connectivity between aNodes. That allows networks to partially expose their internal connectivity structure while keeping it at the level of detail that fit the networks' need. For example, an AS X could announce different cut-through paths with different quality of service by announcing many multi-hop vLinks; it can advertise its internal structure by announcing all the aNodes that it has along with single-hop vLinks connecting those aNodes; and letting outside ASes to decide by which path inside X outside ASes want to route through. Along with aNodes and its properties, vLinks information are announced by every AS in the network using telescopic link-state dissemination, which is described next.

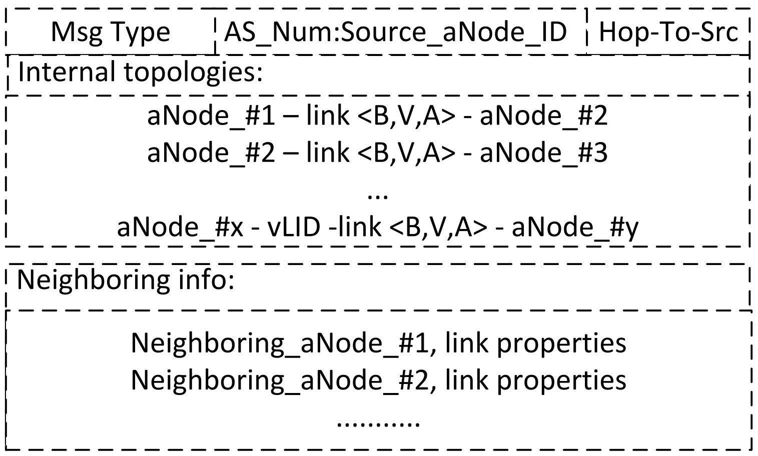
4.3.2 Route Dissemination

The internal structure of a domain is expressed through a graph of aNodes. Periodically, each aNode broadcasts to its neighboring aNode the current state of its network, which includes three major parameters: aNode absolute bandwidth (B), bandwidth variation range (V), and availability rate (A). Absolute bandwidth of an aNode could be the amount of traffic it can carry through if it is a transit aNode; or it could be the aggregated bandwidth of the links through which the aNode connect to the end-host if it is a stub aNode. For example, an aNode that is composed of a single airplane might be an aNode that has absolute bandwidth equal to the bandwidth that it could deliver to all passengers. By varying the three parameters on different aNodes, a domain can control traffic patterns that traverses into and inside its network. In addition, with its fine-grain internal structure exposed to outside networks, a domain can also offer its clients with flexible route section as a value-added service. A domain uses the same set of properties, $<$B,V,A$>$, to describe its vLinks which similarly represent vLinks' absolute bandwidth, variation range and availability.

EIR uses link-state routing throughout the whole Internet in conjunction with telescopic route dissemination mechanism. Specifically, route update messages consisting of both internal and external properties of a network are *periodically* disseminated by ASs. These messages fall into one of two categories: (i) inter-AS messages which describe the network state, called *nSP* and (ii) intra-AS messages which describe the aNode states and its associated vLinks, called *aSP*. An aNode broadcasts aSPs to its directly-connected neighboring aNodes to advertise its internal properties and conditions of links. ASs can use gossiping or any other mechanism to


Msg Type
Source_aNode_ID
Last hop link (Edge) properties - <B,A,V>
Neighboring info:
Neighboring_aNode_#1, link properties
Neighboring_aNode_#2, link properties
....

(a) Intra-AS route update structure


Msg Type
AS_Num:Source_aNode_ID
Hop-To-Src
Internal topologies:
aNode_#1 – link <B,V,A> - aNode_#2
aNode_#2 – link <B,V,A> - aNode_#3
...
aNode_#x - vLID -link <B,V,A> - aNode_#y
Neighboring info:
Neighboring_aNode_#1, link properties
Neighboring_aNode_#2, link properties
....

(b) Inter-AS route update structure

Figure 4.3: (a) An aNode state packet contains an aNode properties and properties of links from the aNode to its neighbouring aNodes, (b) A network state packet has internal network topology, and its neighboring information.

further propagate the aSPs internal to the AS. Border aNodes, upon receiving the aSPs from all the aNodes inside the AS, construct nSPs by combining the complete view of internal network with export policies of the AS. The nSPs are then announced to neighboring ASs. Fig. 4.3 shows the structure of the aNode state packet and the network state packet.

The border aNodes also relay nSPs originated from other ASs in a *telescopic* manner, which means that the relaying rate of a particular border aNode is determined by the distance, i.e. hop count, between the originator and the relaying border aNode. For example, if an aNode makes 2 updates in a minute, its first-hop neighbors might relay all the updates but the 2-hop neighbors curb half the updates and relay only the most recent update every minute. As a result, a router would get more frequent (hence up-to-date) routing updates from routers that are closer to it. Different telescopic functions can be defined by changing the relation between the hold-delay (time for which an aNode holds a received nSP before relaying it to other neighbors) and the hop-count. The steeper the increase in hold-delay per additional hop, greater is the reduction in traffic overhead, but it also leads to a corresponding increase in the time taken by far-away nodes to receive the updates. To explore this tradeoff, we test six different heuristically defined functions. The following equations characterize the telescopic functions in terms of the relation between hold-delay (denoted by y) and the hop count (denoted by x), and Fig 4.4 illustrate their shapes.

$$\text{Constant:} \quad y_1 = A$$

$$\text{Linear:} \quad y_2 = Ax$$

$$\text{Exponential:} \quad y_3 = A\exp^{(x-1)}$$

$$\text{Constant-Linear:} \quad y_4 = \begin{cases} A, \text{ if } x < \alpha \\ A(x-\alpha+1), \text{ if } x \geq \alpha \end{cases}$$

$$\text{Constant-Exp:} \quad y_5 = \begin{cases} A, \text{ if } x < \alpha \\ A\exp^{(x-\alpha)}, \text{ if } x \geq \alpha \end{cases}$$

$$\text{Constant-Exp-Constant:} \quad y_6 = \begin{cases} A, \text{ if } x < \alpha \\ A\exp^{x-\alpha}, \text{ if } \alpha \leq x < \beta \\ A\exp^{(\beta-\alpha)}, \text{ if } x \geq \beta \end{cases}$$

4.3.3 Policy Handling

Through telescopic link-state and network-state dissemination, all ASes have complete visibility of the whole network. Hence, it requires a mechanism to allows domains control and enforce their routing policies, achieving their policy objectives. In response, EIR adopts a mechanism to support broad range of routing policies, ranging from common types of inter-domain business relationship arising from the inherent hierarchy of legacy Internet to many other flexible routing policies that is beneficial to both ISPs and their customers.

Similar to the idea of representing different policies objectives by different attributes from

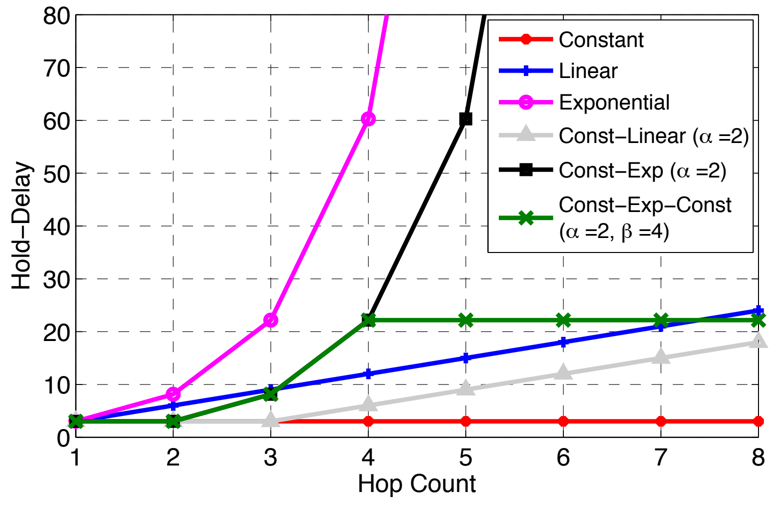


Figure 4.4: Shape of different telescopic functions used for reducing routing traffic overhead

Morpheus [118], EIR allows domain to tag vLinks and aNodes to describe their policy objective. For example, to express business relationship, a domain may tag a vLink with “customer”, “peer”, or “provider”; to express the truth worthy level of an aNode, a domain may tag a route as “suspicious”, “neutral”, or “truth-worthy”. The idea of tagging vLinks and aNode to express policies is analogous to using <B,V,A> to express the quality of those entities. The only different is that these three parameters are more commonly used among all vLinks and aNodes while policies objectives are widely different and domain-dependent. These tagging information is then disseminated through out the Internet in a way that is similar to how network state packets (nSPs) are propagated. When compute forwarding tables, networks use a weighted-sum decision process to select the route that balances trade-offs among different objectives as described in [118].

It is worth noting that EIR naturally supports domains to *setup global roaming agreement*, which is an emerging need area mobile era. In particular, a domain (hosting domain) that is willing to provide network connectivity for client of other domain (remote domain) tags their incoming links and its aNodes with “roaming” tag to indicate that it is willing to accept roaming agreement. Assuming it agrees with the hosting domain on the terms and conditions of the agreement, the remote domain could register its name to the roaming partner entry of the hosting domain on the GNRS, finishing the setup procedure. When a remote domain’s client migrates to and associates with the hosting domain, the domain will first verifies that the client belongs to the remote domain using the self-certifying GUID of that client. Note that EIR is based on the identifier/locator separation in which a network entity has a self-certified global identifier. Once the verification is completed, the hosting domain will allow up stream traffic from the client and update GNRS with a GUID-to-address mapping for that client so that others network entity can reach the remote domain's client.

4.3.4 Route selection

After each aNode learns about all available vLinks, other aNodes and the routing policies, it can build a graph in which each aNode is a vertex and each vLink is a single weighted edge (where which the weight can be derived from the link properties). The routers comprising the aNode can then run different weighted-sum algorithms based on different criteria on this graph to produce a sequence of edges to each destination. For example, Dijkstra's algorithm can be applied to find paths with the smallest latency or hop-count, while Max-Flow Min-Cut algorithm can be applied to find paths with highest stability or availability.

It is also important to note that EIR supports multi-cast inherently by utilizing two key components - telescopic link-state routing dissemination and the GNRS. By using the GNRS, multi-cast groups are represented by GUIDs or a set of GUIDs, thus eliminating the need for any multi-cast group management protocol. The complete network graph allows nodes to decide branching points more efficiently.

4.4 Case Studies

We now present the following three case studies to illustrate how EIR can by used to support dynamism in the edge networks:

1. Dynamic mobility support - handled through late binding and storage aware routing.
2. Cross-domain multi-homing - enabled by providing the complete view of the network graph.
3. Wireless edge peering - supported through the use of link level properties and nSP flooding.

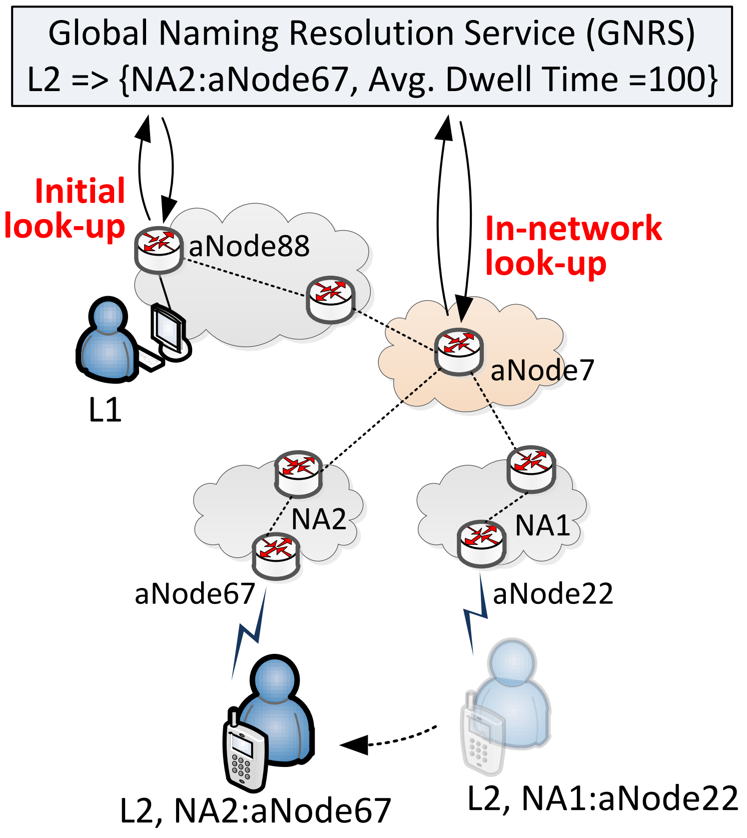


Figure 4.5: Example of mobility support through late-binding of name-to-address

4.4.1 Dynamic Mobility Support

In this section we show the basic technique through which EIR supports delivery of in-transit packets during end-host or network mobility. In this example, a fixed-host L1 wants to send packets to another (fast-moving) end-host L2 which is initially connected to aNode22. In order to do so, L1 simply sends the packet to its default router which does a GNRS look-up to ascertain the current network address (NA1:aNode22) to which the packet should be sent to. The GNRS, in addition to storing the up-to-date network address(es) corresponding to the permanent identifier of each host, also stores the average dwell time of the moving hosts in a network. This is calculated by the GNRS by averaging the time difference between successive network address updates from the host.

In this example, the default router of L1 receives the average dwell time for L2 and decides (based on its small value) that packets sent to the currently connected aNode, i.e. aNode22, might not get delivered since the host is likely to move out of that aNode. In this situation, it utilizes the complete view of the network graph available at each aNode through EIR, to find a transit aNode (say aNode7) whose latency to the destination aNode is roughly equal to the average dwell time of L2 obtained from the GNRS. Packets are then forwarded to aNode7, at which point another look-up is made to the GNRS. While the packet is in-transit, it is likely that the GNRS has been updated to reflect the current network address of L2 as it joins NA2:aNode67. Thus packets are received by the end-host through aNode67 instead of being sent to the old network address.

The main intuition behind in-network look-ups is that for fast-moving hosts/networks, it is better to send packets towards the destination up to an aNode which is close enough to the destination such that the name-to-address binding done at that aNode is likely to be still valid. While this simplistic example might suggest that late-binding is only useful if the dwell time of a host in each network is equal to or less than the packet transit time, the same technique can be used in situations which comprise of disconnections, toggling between networks (4G and WiFi), and network mobility.

4.4.2 Cross-domain multi-homing

While devices with multiple interfaces are increasingly common (for example almost all smartphones have both Wi-Fi and 2G/3G/4G radio front-ends), making use of the parallel connectivity opportunities through existing schemes is still extremely challenging. EIR enables a ‘network-based’ multi-homing scheme where the bulk of sophistication lies inside the network rather than at the end-host stacks, as in the case of most prior proposals on multi-homing [119, 120].

EIR accommodates end-host multi-homing through three key design components: (i) the global naming resolution service that provides fast in-network name to address mapping (ii) differential service identifiers (SID) that allows end-host to express its multi-path requirements, and (iii) fine-grained network graph, which allows the source router to select the path depending on the end-host requirements. Figure 4.6 shows an example of how EIR can be utilized in the end-host multi-homing scenario. Consider the case where an application on end-host L1 wants to send packets to a multi-homed host L2, which is associated to aNode89 (a basestation) in

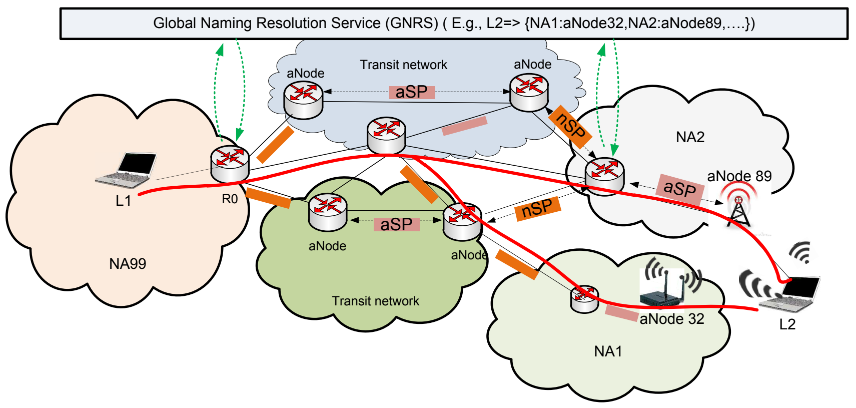


Figure 4.6: An example of EIR routing in end-host multi-homing scenario

network NA2 through a 4G interface and to aNode32 (a WiFi AP) in network NA1 through a WiFi interface. Prior to the transfer of packets, L2 sequentially updates the GNRS with the network addresses that it obtains through the two networks. Thus the GNRS stores an entry of the form: $< L2 => \{NA1 : aNode32, NA2 : aNode89\} >$. In order to send packets to L2, the host L1 does not need to keep track of the multiple interfaces, nor implement a data-striping algorithm. Instead, it simply hands out packets labeled with L2 as the destination, to the transport layer. Based on the application's service requirements, the transport layer sets an appropriate value for the service ID. For example, if it is a time-sensitive application, a low-latency service will be selected; whereas, if it is a bandwidth-intensive application, a high-bandwidth service will be specified (in which case all interfaces might be utilized for data transmission). Let's assume in this case the service L1 selects is "fastest-service". Once the service ID is specified, the packet is handed to L1's default router R0. The router R0 does a GNRS look-up to ascertain the aNodes to which L2 is currently connected, i.e. $< aNode32, aNode89 >$. Through EIR, R0 has a complete and fine-grained view of the entire network graph, thus it can compare multiple possible paths to reach L2, from which the best route and the associated aNode is selected. The router then routes the packet to the designated aNode (say aNode89).

If there are no physical changes in the link quality along the path, the packets would reach aNode89 and get delivered to L2. However, in the event of link quality fluctuation, routers along the selected path can react in response to the fluctuation when using EIR. Let's consider the scenario where the link between aNode78 and aNode89 is broken while the packet is in transit from R0 to aNode89. As the packet reaches the border aNode78 of NA2, due to the telescopic nature of EIR, routers in aNode78 realize the broken link, making L2 unreachable through aNode89, since the routers have fresher information about quality of links closer to aNode78. As a result, instead of forwarding the packet to the next hop, the router executes another naming look-up to find L2's latest address to which the packet is then delivered accordingly through the neighboring transit network to NA1, to aNode32 and then L2.

4.4.3 Wireless edge peering

Peering between autonomous domains in the Internet is one of the most important, yet least understood technique used in the Internet. Different ASs employ different types of peering agreements with different number of neighboring ASs and a recent report showing the presence of 75% more peering links than previously known [121] shows the lack of a clear structure to specify, infer, and instantiate peering relations between networks.

Through the use of more granular primitives at the interdomain level (aNodes instead of ASs) and the inclusion of link-level information in the routing updates, EIR enables a new form of wireless peering relations between edge networks. Consider the case of two small enterprise networks N1 and N2 which operate in geographically close locations (e.g. on different floors of a building) and have different Internet service providers N3 and N4. Due to the geographical proximity, some wireless routers in both networks can connect to each other, for example using the bridging-mode available in many enterprise WiFi APs [122]. Using the aNode abstraction, N1 and N2 can then establish a wireless edge peering link by exchanging the aNode specific policy specification with each other. This wireless peering link would keep the two networks connected even if networks N3 and N4 both are undergoing failures, and can help one network to use the connectivity of the other network in case either one of N3 and N4 has a link failure.

We believe that wireless peering would be increasingly important for the mobile-dominant future Internet, especially for supporting disaster-recovery (when wired connections to ISPs might fail) and congestion (to maintain partial edge-connectivity when the main links become too congested).

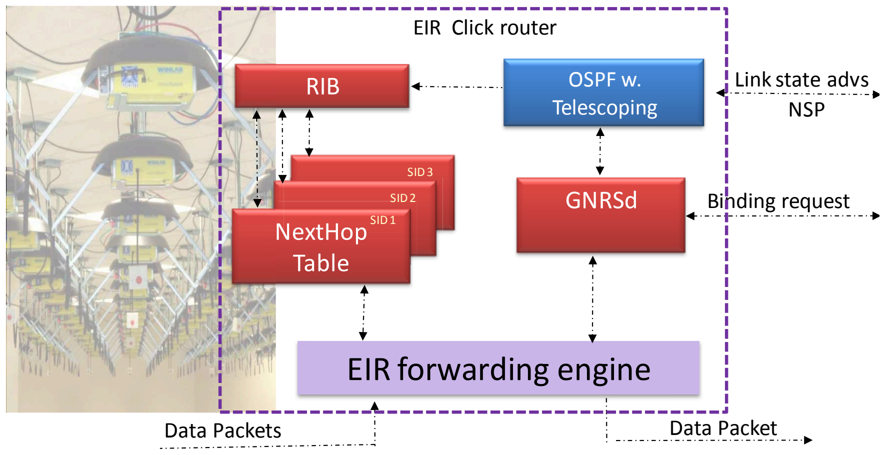


Figure 4.7: Overall architecture of the Click router prototype based on ORBIT nodes

4.5 Evaluation

In this section, we evaluate the EIR protocol in terms of scalability and mobility support capability through a large-scale prototype evaluation and an Internet-scale simulation study. Sec. 4.5.1 describes the implementation details and the results from testbed-based experiments, and Sec. 4.5.2 describes the setup and insights from a ∼26,000 node simulation effort.

4.5.1 Implementation

To measure the performance and implementation feasibility of EIR, we built a prototype router (based on the Click modular router design [123]) and deployed a network of 200 physical machines on the ORBIT testbed [124].

4.5.1.1 Prototype Design

Fig. 4.7 shows the overall structure of our Click implementation. Our router consists of two components: control plane and data plane. The control plane sends aSP and nSP packets as per the specifications outlined in Sec. 4.3 and also receives and forwards aSP and nSP packets sent by other routers. Upon receiving a routing update packet (with a higher sequence number than it currently has for that source), the router takes the following steps:

First, it calculates the duration for which it needs to hold the received update before relaying it to other neighbors (i.e. the hold-delay) based on the telescopic function being used (see

Fig. 4.4). The router then uses the information contained in the update packet to re-compute its routing table using Dijkstra's shortest path algorithm. Changes in the computed paths are reflected in the forwarding table which is used in the data plane. Lastly, to help the late-binding operation in the data plane, the control plane builds another table called the late-binding table. Each entry of this table contains a full path and the corresponding costs to a destination and is used only for late-binding operations.

The data plane sends and receives data packets to and from other routers. The GNRS server, which is implemented as a daemon in one of the nodes of the testbed, is used extensively by the data plane. As explained in the dynamic mobility and cross-domain multi-homing use-cases in Sec. 4.4, the GNRS server is queried by the data plane whenever the aNode address corresponding to the destination of a packet needs to be ascertained.

4.5.1.2 Evaluation Methodology

Topology Generation: Since it is infeasible to emulate the complete AS-level graph of the current Internet, we use 200 physical nodes of the ORBIT testbed with a scaled-down topology which mimics the AS-level structure of the Internet. In order to do so, we first extract the degree distribution and the latency distribution of the measured AS-level graph from the DIMES database [125]. Next, we build a Jellyfish topology [101] consisting of 200 nodes by matching the distribution of ASs in each layer and the proportion of links between layers, to the values ascertained from the DIMES dataset. Finally, the real-world measured latency values are used to assign link delays in our topology in a manner that preserves the latency-distribution. Fig. 4.8 compares the CDF of the latency values used in our topology with that of the complete AS-level graph obtained from the DIMES database.

Performance Metrics In evaluating our system performance, we consider several metrics. We compute the overhead caused by aSP and nSP advertisements when using different telescopic functions (as described in Sec. 4.3.2) and for different values of update interval. To show the effectiveness of using late-binding and rebinding techniques, we compute the percentage of lost packets when a sender tries to send data packets to a mobile receiver. 'Rebinding' here refers to the case when packets sent from a source are not binded to the destination aNode while the packet is in transit, but forwarded to the destination through a GNRS re-lookup only

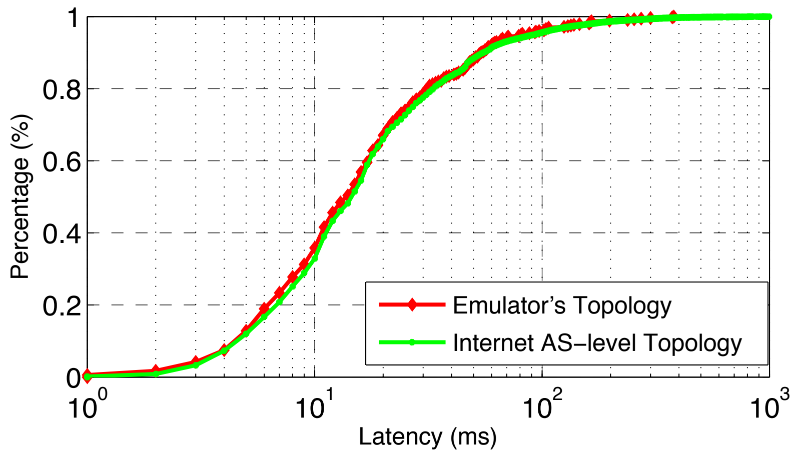


Figure 4.8: CDF of inter-AS latency as measured in the Internet's AS-level graph and the inter-node latency used in our 200 node topology.

if the host has moved out of the original destination.

We use the term 'mobility interval' to indicate the duration an end-host stays connected to an aNode. For the mobility experiments, we assume fixed senders and mobile receivers. For collecting each data-point, we randomly select the attachment aNode for the sender and two attachment aNodes for the mobile receiver. The sender sends 50,000 data packets each of size 64 bytes to the receiver at a rate of 1000 packets per second. We repeat this experiment while varying the mobility interval and present the average results over 100 runs.

4.5.1.3 Results

Fig. 4.9 shows the overhead of aSP and nSP advertisement using three telescopic functions. At all update intervals, the overhead incurred by function 1 is greatest and the overhead incurred by function 3 is the smallest. The reason is that the duration we hold the update packets is longest for the 1st function. Fig. 4.10 shows the CDF of the percent of the aNode population that receives the update packets as a function of time.

Fig. 4.11 shows the percent of lost packets when a sender sends data packets to a mobile receiver with and without using the late-binding and re-binding techniques. In the early-binding case, the percent of lost packets will increase as the receiver moves often. This is because we only resolve the attachment aNode at the sender. Whenever the receiver moves to another node

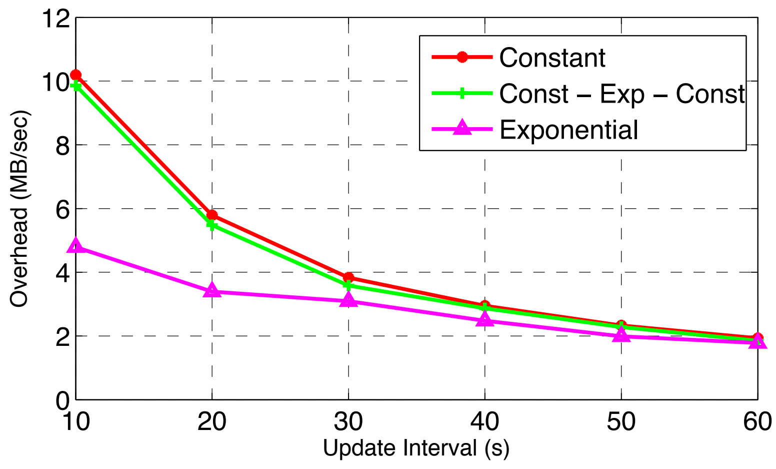


Figure 4.9: Overhead of update packets under different telescopic functions

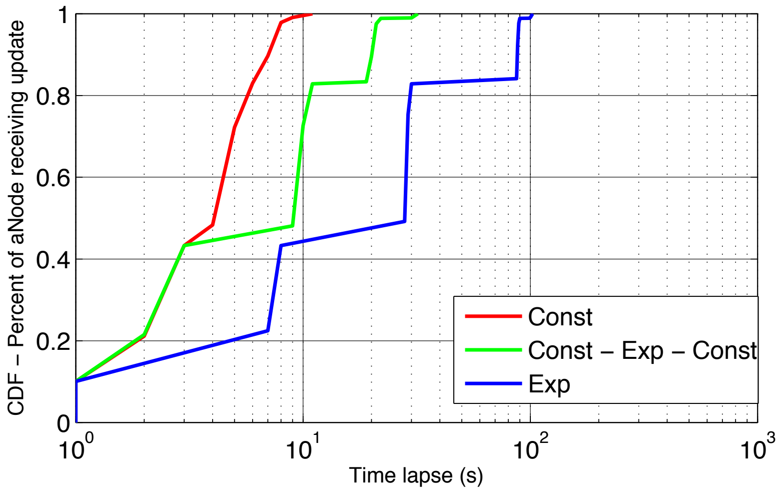


Figure 4.10: CDF of percent of aNodes receive update packet from one node

it will not receive the packets sent from the sender. Our results also confirm that late-binding and rebinding ensure packet delivery in mobility.

In Fig. 4.12 and Fig. 4.13, the performance of both late-binding and rebinding are shown for end-to-end latency and the stretch. Both late-binding and rebinding maintain low stretch at all mobility interval, and re-binding results in increased latency as expected.

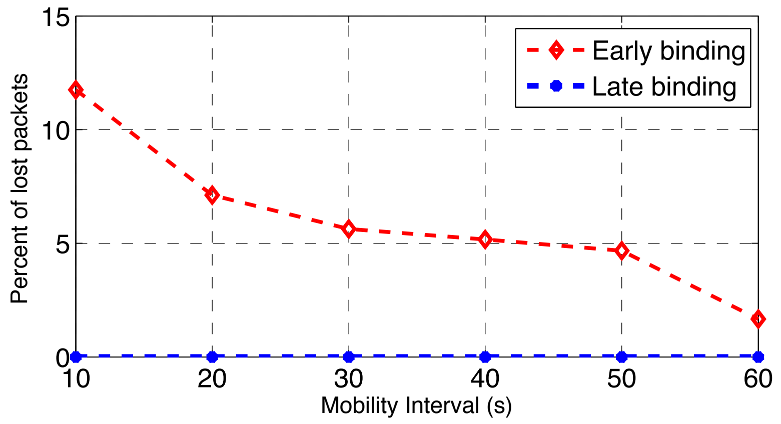


Figure 4.11: CDF of percent of aNodes receive update packet from one node (Will change)

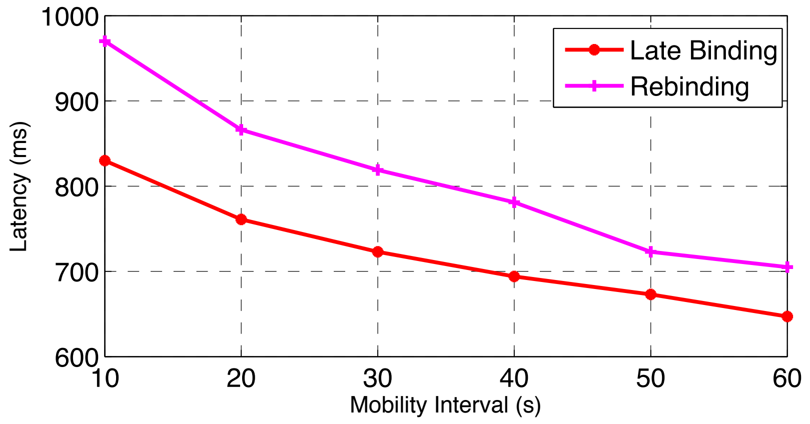


Figure 4.12: End-to-end latency when using late-binding and rebinding technique

4.5.2 Internet-scale Simulation

To evaluate the protocol at a global scale, we designed and implemented a custom-built discrete event-driven simulation to reflect the actual Internet topology according provided by DIMES [125]. Through the simulation, we evaluate routing event dissemination overhead and dissemination latency.

4.5.2.1 Methodology

The simulator takes AS-level topology of the current Internet as the network model by extracting the following real-measurements from the DIMES database: (i) Connectivity graphs containing 26235 ASs and ∼100K links between them, (ii) Link latency between each pair of

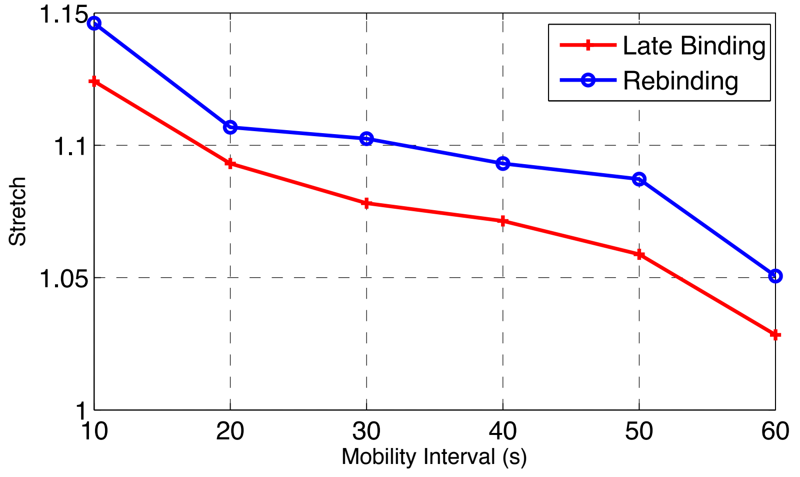


Figure 4.13: Stretch when using late-binding and rebinding technique

AS. Because the number of dissemination packages increases exponentially with the number of nodes in the topology, our simulator cannot emulate the event propagation of all 26K ASs for a long duration of time. Therefore, we limit the duration of the simulation so that our server, a Linux machine with 32 cores and 75 GB of memory, could handle.

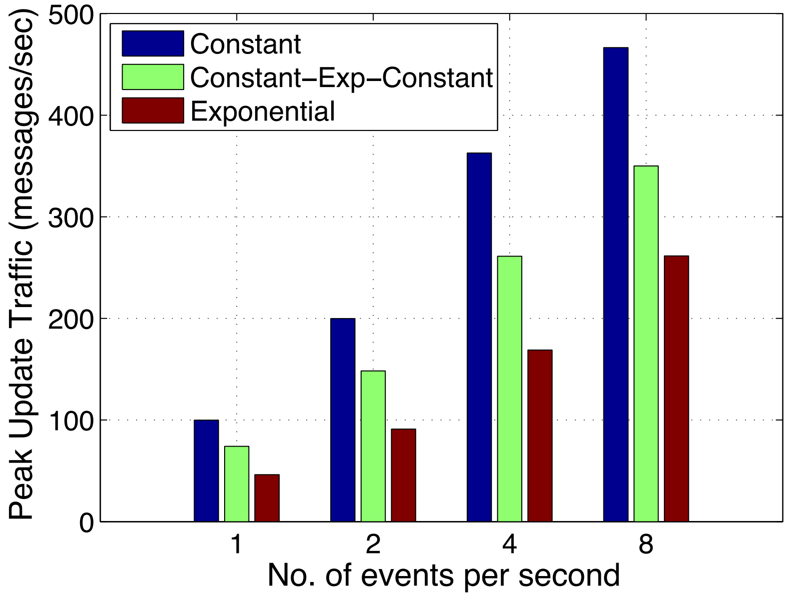


Figure 4.16: Routing event update overhead with different telescopic functions

In particular, we generate 60 second worth of routing event with different routing event rate, ranging from 1 to 8 events per second, which also covers the measured BGP routing event rate

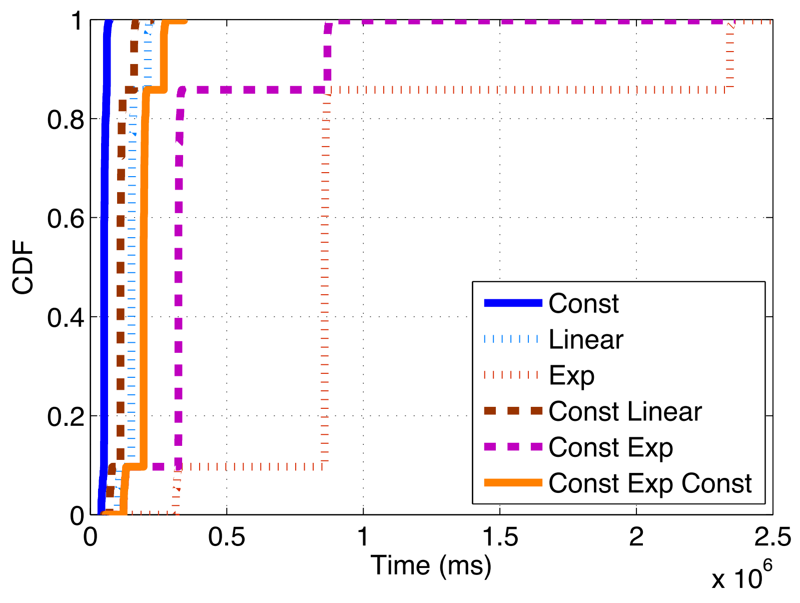


Figure 4.14: Longest routing event update time

of around 1.8 event per second. In the simulation, for each routing update event, we simulate the propagation process from the original aNode to all other aNode in the topology. We then count the number of packets it generates and the amount of time it takes to disseminate throughout the network.

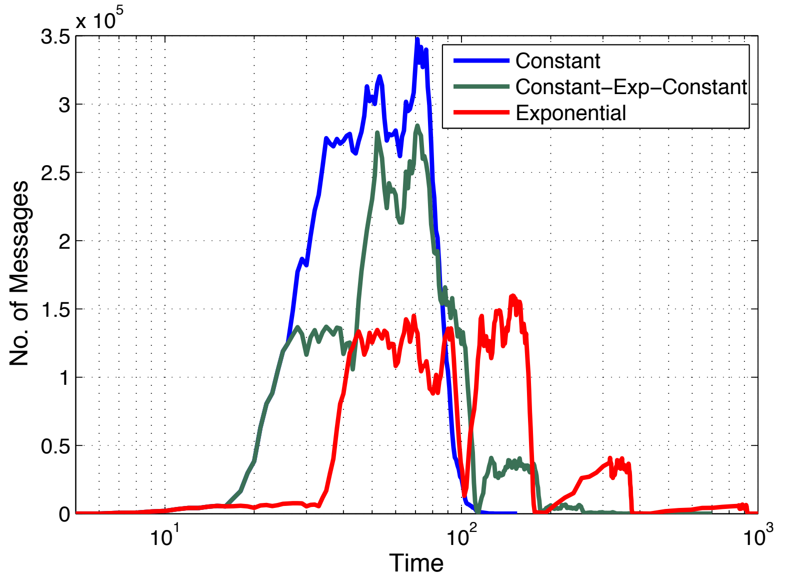


Figure 4.17: Routing update message dissemination with different telescopic functions

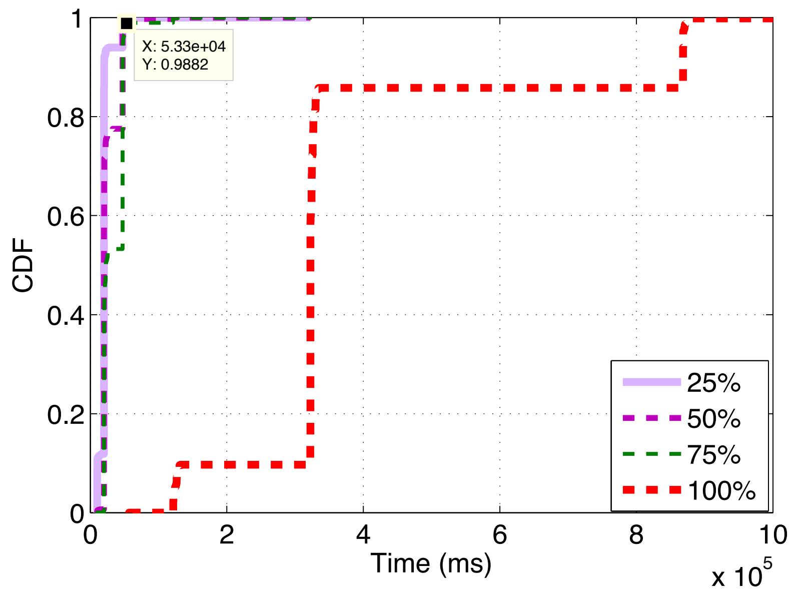


Figure 4.15: Routing event update time with different percentile of recipients

4.5.2.2 Routing update latency and overhead

Figure 4.14 shows the CDF of the time required for every AS in the graph to get the update generated. This captures the worst-case scenario since some ASs are linked through very high-latency links. This plot shows that when using the constant functions, all ASs receive the update in a short time but even with other telescopic functions except the exponential and the constant exponential, the update times are not very long. Figure 4.15 shows the time when different percentages of ASs get the update made from an AS for the constant-exponential-constant telescopic function. This shows that the previous graph was really the worst case scenario as most ASs get the update much ahead of the last AS.

Figure 4.16 shows the peak traffic overhead when using different telescopic functions for different values of update generation rate. The gain when using the constant-exponential-constant function increases with the increase in the generation rate. Figure 4.17 shows the number of update messages being sent as a function of time for three different telescopic function. As expected, the constant function leads to the highest peak but a narrower shape whereas the constant-exponential-constant function shows a good balance between the peak rate and the reduction offered in overhead.

4.6 Concluding Remarks

In this work, we have presented the design and evaluation of a novel edge-aware interdomain routing protocol intended as a clean-slate solution for the future mobile Internet. The EIR protocol is based on telescopic flooding of a ”network state packet (NSP)” along with late binding of end-point names to locators at intermediate routers in the network. In EIR, internal network topology and state can optionally be exposed by autonomous systems using the ”aNode” and ”vLink” abstractions described here. Dissemination of NSPs across the Internet provides improved visibility of the network topology as a whole along with aggregated information about internal structure for those networks which choose it make it available. These mechanisms enable routers to make informed routing decisions for mobility-oriented services such as roaming with intermittent disconnection, multi-homing and edge peering. The EIR protocol has also been extensively validated using both large-scale simulation and ORBIT testbed emulation using 200 network nodes running a real-time Click software implementation of the routers. The results confirm that the telescopic NSP flooding approach effectively limits routing overhead to acceptable values, and that the late-binding EIR protocol provides good performance for a mobility service scenario with frequent migration of clients across network domains. Future work on EIR includes design of policy mechanisms qualitatively similar to those in BGP, as well as more realistic performance evaluation via integration into the MobilityFirst deployment on the meso-scale experimental GENI network.

Chapter 5

Future Directions and Conclusion

This dissertation describes new applications and services that composable mobile systems could bring. Taking concrete steps towards realizing composable mobile systems, we proposed three new components to enrich communication methods. We proposed a new communication method for capacitive touch enabled devices, which enable a wide range of applications that are suitable for composable computing. Furthermore, under the umbrella of a mobile-centric network architecture, MobilityFirst, we presented a new naming resolution service and an inter-domain routing protocol that provide better support for highly mobile and dynamic environments as found in composable mobile systems. However, there are still many challenges, not limited to communication issues, preventing the proliferation of fully composable mobile architecture. The following sections suggest directions for future research.

5.1 Extracting Information from Shifting Contexts

In order to provide the best computing experience, the system itself must be able to observe, model, and predict the user's environment and user's states. Devising techniques that help mobile devices to gain a comprehensive view of the surrounding context, hence, is a critical mission. One possible extension of our work along this vein is distilling context information through mobile crowd-sourcing, location sensing, and combining, in nontraditional ways, other sensing capabilities of mobile devices (e.g. capacitive sensors, light sensors, etc). We explore a technique that integrates wireless wearable devices with hardware adjuncts to provide spatial context information with a high level of accuracy from objects and people for indoor environment. For example, accurately estimating relative angles and range information between nearby devices is a hard problem, yet of importance to many context-aware applications such as augmented reality, autonomous automotive systems, smart manufacturing systems, etc. Existing indoor localization techniques could not meet the applications' requirements given the resource constraints. RF-based techniques can easily provide a link to a database but suffer from multipath effects and the need for careful calibration if used to determine spatial orientation. Camera-based techniques are computing intensive and battery-hungry. Optical links, on the other hand, can provide accurate spatial orientation but are less reliable and more energy intensive than radio for transmitting data. To get the best of both technologies, we attempt to utilize radio-assisted free-space optical links to estimate the azimuth and range to objects in the environment while providing detailed information about them. Another aspect deserving research attention is inferring human factors in mobile systems. Mobile computing today has been personalized to meet users' needs and enhance their experience more than ever before. For example, video content is encoded to different bit rates before being streamed to an end-user, adapting to the computational capability and screen size of the client's device. However, this personalization is merely done to the device, not to the individual actively using that device. This limitation is due to the lack of techniques to help the device understand the associated human factors. We are exploring new methods to help mobile devices appreciate more about its active users. In particular, we will continue to work on improving the performance of capacitive touch communication by exploring other modulation and demodulation techniques, revising hardware design, and getting access to the lower level and finer-grain capacitive sensing information.

5.2 Minimizing User Attention and Effort in Human-System Interactions

The system should automatically adapt to changes in the environment as well as user's intentions. In many cases, the interaction between the user and the system is secondary to the user's primary task. Hence, the interaction should take the least amount of the user's attention and avoid distracting him from the primary task. For example, Bill, in our initial scenario in Chapter 1, ideally should be able to focus on his presentation without worrying about interacting with his matchbox computer to connect to the projector in the meeting room. For that to happen, one should explore new communication methods and new human-computer interfaces for everyday and continuous use of the system, allowing unobtrusive and quick access to the system with very little effort from the users.

5.3 Support Mobile Communication at Wireless Edge Networks

Deep inside the network, DMap shows the immense benefit of leveraging computing power of the distributed routers across the global. We could further explore how to take advantage of other network resources to improve in-network services. For instance, one could leverage the ever-large storage capability of network entities to enable storage-aware routing that would enhance disruption tolerant communications, where packets make forwarding progress despite the absence of a contemporaneous end-to-end route to the destination (due to mobility, for example). It is also promissing to study how we can dynamically place caches at the edge networks that are very close to users, and to exercise caching utilizing the knowledge of the current network's conditions and content servers made available by the edge-aware routing paradigm. The flexibility that software-defined networking (SDN) provides could change the way wireless edge networks are designed, built and operated to allow the dynamics happening at the edge to be gracefully handled. To deliver agility for ad-hoc mobile networks, one could greatly benefit from SDN by tailoring the networks to fit their needs without any redundant functionality, to minimize the overhead and satisfy the strict resource constrains of mobile devices. We want to study how SDN could be used to enforce network management tasks such as device discovery, access control, and dynamic spectrum access for coexistence between heterogeneous radio systems.

5.4 Protecting User Privacy and Securing Context Information

While making context information available to mobile devices is critical to the success of the provisioned mobile platform, in the wrong hands that information could not only reveal subjects' private information but also seriously damage personal, social and professional life of the individual. For example, leakage of a user's frequently visited locations could expose his or her ostracized political view, alternative lifestyles or medical problems. One important aspect of research is exploring techniques to both secure context information and provide context information privacy protection.

Above possible research areas are those directly related to realizing composable mobile systems. Many other practical issues, for example energy harvesting for mobile components, reliable information delivery between devices, or even easing infrastructure implementation, require more research attention. Like any exploration of an immature research field, this dissertation poses more questions than it answers. Hopefully, the trend toward composable mobile system will continue to emerge in the near future, obviating the need for advanced research in the topics suggested in this chapter. Maybe then, we will see composable mobile systems at every corner in everyday life.

References

[1] J. Yang, S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser, and R. P. Martin, "Detecting driver phone use leveraging car speakers," in *Proceedings of ACM MobiCom*, 2011.

[2] G. Chandrasekaran, T. Vu, A. Varshavsky, M. Gruteser, R. P. Martin, J. Yang, and Y. Chen, "Tracking vehicular speed variations by warping mobile phone signal strengths," in *Proceedings of IEEE PerCom*, 2011.

[3] J. Yang, S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser, and R. P. Martin, "Sensing Driver Phone Use with Acoustic Ranging through Car Speakers," in *IEEE Transactions on Mobile Computing*, 2012.

[4] G. Chandrasekaran, T. Vu, A. Varshavsky, M. Gruteser, R. P. Martin, J. Yang, and Y. Chen, "Vehicular speed estimation using received signal strength from mobile phones," in *Proceedings of ACM Ubicomp*, 2010.

[5] M. Weiser, "The computer for the 21st century," *Scientific american*, vol. 265, no. 3, pp. 94–104, 1991.

[6] A. Fox, B. Johanson, P. Hanrahan, and T. Winograd, "Integrating information appliances into an interactive workspace," *Computer Graphics and Applications, IEEE*, vol. 20, no. 3, pp. 54–65, 2000.

[7] B. Brumitt, B. Meyers, J. Krumm, A. Kern, and S. Shafer, "Easyliving: Technologies for intelligent environments," in *Handheld and ubiquitous computing*. Springer, 2000, pp. 12–29.

[8] R. Want, T. Pering, S. Sud, and B. Rosario, "Dynamic composable computing," in *Proceedings of the 9th workshop on Mobile computing systems and applications*, ser. HotMobile '08. New York, NY, USA: ACM, 2008, pp. 17–21. [Online]. Available: http://doi.acm.org/10.1145/1411759.1411765

[9] M. Satyanarayanan, M. A. Kozuch, C. J. Helfrich, and D. R. O'Hallaron, "Towards seamless mobility on pervasive hardware," *Pervasive Mob. Comput.*, vol. 1, no. 2, pp. 157–189, Jul. 2005. [Online]. Available: http://dx.doi.org/10.1016/j.pmcj.2005.03.005

[10] B. A. Myers, "Using handhelds and pcs together," *Commun. ACM*, vol. 44, no. 11, pp. 34–41, Nov. 2001. [Online]. Available: http://doi.acm.org/10.1145/384150.384159

[11] G. Shen, Y. Li, and Y. Zhang, "Mobius: enable together-viewing video experience across two mobile devices," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, ser. MobiSys '07. New York, NY, USA: ACM, 2007, pp. 30–42. [Online]. Available: http://doi.acm.org/10.1145/1247660.1247667

[12] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar, and J. Light, "The personal server: Changing the way we think about ubiquitous computing," in *In Proceedings of Ubicomp 2002: 4th International Conference on Ubiquitous Computing, Goteborg*. Springer-Verlag, 2002, pp. 194–209.

[13] 3GPP: Mobile Broadband Standard, http://www.3gpp.org/.

[14] S. Gundavelli, K. Leung, V. Devarapalli, C. K., and P. B., *Proxy Mobile IPv6*, IETF Internet Standard, RFC 5213, Aug. 2008.

[15] NSF Future Internet Architecture Project, http://www.nets-fia.net/.

[16] MobilityFirst Future Internet Architecture Project, http://mobilityfirst.winlab.rutgers.edu/.

[17] J. Saltzer, *On the Naming and Binding of Network Destinations*, IETF Internet Standard, RFC 1498, Aug. 1993.

[18] C. J. Bennett, S. W. Edge, and A. J. Hinchley, "Issues in the interconnection of datagram networks," Jul. 1977.

[19] R. Moskowitz and P. Nikander, *Host Identity Protocol (HIP) Architecture*, IETF Internet Standard, RFC 4423, May 2006.

[20] J. Pan, R. Jain, S. Paul, and C. So-in, "MILSA: A New Evolutionary Architecture for Scalability, Mobility, and Multihoming in the Future Internet," *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 8, pp. 1344–1362, Oct. 2010.

[21] T. Vu, A. Baid, S. Gao, M. Gruteser, R. Howard, J. Lindqvist, P. Spasojevic, and J. Walling, "Distinguishing users with capacitive touch communication," in *Proceeding of ACM MobiCom*, 2012.

[22] T. Vu, A. Baid, Y. Zhang, T. D. Nguyen, J. Fukuyama, R. P. Martin, and D. Raychaudhuri, "DMap: A Shared Hosting Scheme for Dynamic Identifier to Locator Mappings in the Global Internet," in *Proceeding of IEEE ICDCS*, 2012.

[23] T. Vu, A. Baid, H. Nguyen, and D. Raychaudhuri, "Eir: Edge-aware interdomain routing protocol for the future mobile internet," in *submission*, 2013.

[24] E. Sofge, "Can a smartphone make your car smarter?" *MSN Autos*, 2010.

[25] Craftsman, "Assurelink garage doors openers."

[26] Belkin, "Wemo home automation system."

[27] A. K. Karlson, A. B. Brush, and S. Schechter, "Can i borrow your phone?: understanding concerns when sharing mobile phones," in *Proc. of CHI*, Apr. 2009.

[28] C. Foresman, "Apple facing class-action lawsuit over kids' in-app purchases," *Ars Technica*, Apr. 2011.

[29] K. Hawkey and K. M. Inkpen, "Examining the content and privacy of web browsing incidental information," in *Proc. of WWW*, May 2006.

[30] N. Ben-Asher, N. Kirschnick, H. Sieger, J. Meyer, A. Ben-Oved, and S. Möller, "On the need for different security methods on mobile phones," in *Proc. of MobileHCI*, Aug. 2011.

[31] E. A. Johnson, "Touch Displays: A Programmed Man-Machine Interface," *Ergonomics*, vol. 10, no. 2, pp. 271–277, 1967.

[32] W. Westerman and J. G. Elias, "US 0232567 Patent - Capacitive Sensing Arrangment," Patent, 10, number = US 0232567, type = Patent,, 2006.

[33] "Fundamentals of electrostatic discharge," http://www.esda.org/fundamentalsp5.html, 2010.

[34] M. Sato, I. Poupyrev, and C. Harrison, "Touche: enhancing touch interaction on humans, screens, liquids, and everyday objects," in *Proc. of CHI*, New York, NY, USA, 2012, pp. 483–492. [Online]. Available: http://doi.acm.org/10.1145/2207676.2207743

[35] P. Dietz and D. Leigh, "Diamondtouch: a multi-user touch technology," in *Proc. of UIST*, Nov. 2001.

[36] "Technology review article: Pushing the limits of the touch screen," 2011.

[37] V. Roth, P. Schmidt, and B. Güldenring, "The IR ring: authenticating users' touches on a multi-touch display," in *Proc. of UIST*, Oct. 2010.

[38] H. Bojinov and D. Boneh, "Mobile token-based authentication on a budget," in *Proc. of HotMobile*, Feb. 2011.

[39] A. Braun and P. Hamisu, "Using the human body field as a medium for natural interaction," in *Proc. of PETRAE*. ACM, 2009, p. 50.

[40] P. Dunphy, A. P. Heiner, and N. Asokan, "A closer look at recognition-based graphical passwords on mobile devices," in *Proc. of SOUPS*, Jul. 2010.

[41] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proc. of CCS*, Oct. 2010.

[42] A. Nicholson, M. D. Corner, and B. D. Noble, "Mobile Device Security using Transient Authentication," *IEEE TMC*, vol. 5, no. 11, pp. 1489–1502, November 2006. [Online]. Available: http://prisms.cs.umass.edu/mcorner/papers/tmc_2005.pdf

[43] "Ringbow," http://ringbow.com/ringbow/.

[44] "Motorola atrix: Technical specifications," http://bit.ly/f7nfEX, 2011.

[45] "Sony - mofiria finger-vien pattern recognition," 2009. [Online]. Available: http://tinyurl.com/c7l9e5

[46] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino, "Impact of Artificial "Gummy" Fingers on Fingerprint Systems," in *Proc. of SPIE Vol. 4677*, 2002.

[47] J. Galbally-Herrero, J. Fierrez-Aguilar, J. D. Rodriguez-Gonzalez, F. Alonso-Fernandez, J. Ortega-Garcia, and M. Tapiador, "On the vulnerability of fingerprint verification systems to fake fingerprints attacks," in *Proc. of Carnahan Conferences on Security Technology*, Oct. 2006.

[48] S. Kurkovsky, T. Carpenter, and C. MacDonald, “Experiments with simple iris recognition for mobile phones,” in *Proc. of Information Technology: New Generations*, april 2010, pp. 1293 –1294.

[49] “Mobile biometry: European funded project (fp7-2007-ict-1),” http://www.mobioproject.org/, 2011.

[50] M. Faundez-Zanuy, “On the vulnerability of biometric security systems,” *Aerospace and Electronic Systems Magazine, IEEE*, vol. 19, no. 6, pp. 3 – 8, June 2004.

[51] A. Adler, “Vulnerabilities in biometric encryption systems,” in *Proc. of AVBPA*, 2005.

[52] L. Thalheim and J. Krissler, “Body check: Biometric access protection devices and their programs put to the test,” *ct Magazine*, Nov. 2002.

[53] “Face recognition on galaxy nexus is very neat, but easily compromised with the picture?” http://tiny.cc/gj50gw, 2011.

[54] A. K. Jain, K. Nandakumar, and A. Nagar, “Biometric template security,” *EURASIP J. Adv. Signal Process*, vol. 2008, pp. 113:1–113:17, Jan. 2008.

[55] G. Bailador, C. Sanchez-Avila, J. Guerra-Casanova, and A. de Santos Sierra, “Analysis of pattern recognition techniques for in-air signature biometrics,” *Pattern Recognition*, vol. 44, no. 10-11, pp. 2468 – 2478, 2011.

[56] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, “User evaluation of lightweight user authentication with a single tri-axis accelerometer,” in *Proc. of MobileHCI*, Sep. 2009.

[57] M. L. Mazurek, J. P. Arsenault, J. Bresee, N. Gupta, I. Ion, C. Johns, D. Lee, Y. Liang, J. Olsen, B. Salmon, R. Shay, K. Vaniea, L. Bauer, L. F. Cranor, G. R. Ganger, and M. K. Reiter, “Access control for home data sharing: Attitudes, needs and practices,” in *Proc. of CHI*, May 2010.

[58] Y. Shaked and A. Wool, “Cracking the bluetooth pin,” in *Proc. of MobiSys*, 2005, pp. 39–50.

[59] “Nearfield communications forum,” http://www.nfc-forum.org/, 2011.

[60] E. Haselsteiner and K. Breitfuss, “Security in Near Field Communication (NFC),” in *Proc. of the Workshop on RFID Security*, 2006.

[61] R. Mayrhofer and H. Gellersen, “Shake well before use: Intuitive and secure pairing of mobile devices,” *IEEE TMC*, vol. 8, no. 6, pp. 792 –806, Jun. 2009.

[62] S. Mathur, R. Miller, A. Varshavsky, W. Trappe, and N. Mandayam, “Proximate: proximity-based secure pairing using ambient wireless signals,” in *Proc. of MobiSys*, Jun. 2011.

[63] F. Stajano and R. J. Anderson, “The resurrecting duckling: Security issues for ad-hoc wireless networks,” in *Proc. of Security Protocols*, Apr. 2000.

[64] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong, “Talking to strangers: Authentication in ad-hoc wireless networks,” in *Proc. of NDSS*, Feb. 2002.

[65] C. Gehrmann, C. J. Mitchell, and K. Nyberg, "Manual authentication for wireless devices," *CryptoBytes*, Spring 2004.

[66] F. Xie, T. Furon, and C. Fontaine, "On-off keying modulation and tardos fingerprinting," in *Proc. of ACM workshop on MM & Sec.* ACM, 2008, pp. 101–106.

[67] "Atmel maxtouch mxt1386 specifications," http://www.atmel.com/devices/MXT1386.aspx.

[68] "Afg3000 series arbitrary function generators," http://www.tek.com/AFG3000.

[69] "Msp430 ultra-low power 16-bit microcontrollers," http://msp430.com.

[70] "Amplifier transistors bc548b," http://onsemi.com.

[71] "An mcu-based buck-boost converter for battery chargers," http://www.st.com/.

[72] C. Cornelius, J. Sorber, R. Peterson, J. Skinner, R. Halter, and D. Kotz, "Who Wears Me? Bioimpedance as a Passive Biometric," *Proc. of HealthSec Workshop*, 2012.

[73] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker, "Accountable Internet Protocol (AIP)," in *Proc. ACM SIGCOMM*, Aug. 2008.

[74] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker, "ROFL: Routing on Flat Labels," in *In SIGCOMM*, 2006, pp. 363–374.

[75] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, ser. ICN '11, 2011, pp. 1–6.

[76] L. Jakab, A. Cabellos-Aparicio, F. Coras, D. Saucez, and O. Bonaventure, "LISP-TREE: a DNS hierarchy to support the lisp mapping system," *IEEE J.Sel. A. Commun.*, vol. 28, pp. 1332–1343, October 2010.

[77] L. Mathy and L. Iannone, "LISP-DHT: towards a DHT to map identifiers onto locators," in *Proceedings of the 2008 ACM CoNEXT Conference*, ser. CoNEXT '08, 2008, pp. 61:1–61:6.

[78] E. T. Li, *Recommendation for a Routing Architecture*, IETF Internet Standard, RFC 6115, Feb. 2011.

[79] D. Farinacci, D. Fuller, D. Oran, and D. Meyer, *Locator/ID separation protocol (LISP)*, IETF Internet Draft, draft-farinacci-lisp-12.txt, Sep. 2009.

[80] C. Vogt, "Six/one router: a scalable and backwards compatible solution for provider-independent addressing," in *Proceedings of MobiArch '08*, 2008, pp. 13–18.

[81] D. Jen, M. Meisel, D. Massey, L. Wang, B. Zhang, and L. Zhang, "Apt: a practical tunneling architecture for routing scalability," *UCLA Computer*, pp. 1–9, 2008.

[82] D. Farinacci, V. Fuller, D. Lewis, and D. Meyer, *LISP Mobile Node*, IETF Internet Draft, draft-meyer-lisp-mn-05.txt, May 2011.

[83] "Cisco visual networking index: Global mobile data traffic forecast update, 2010-2015," Cisco White Paper, Feb. 2011.

[84] L. Monnerat and C. Amorim, “D1HT: a distributed one hop hash table,” in *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, 2006, p. 10 pp.

[85] A. Gupta, B. Liskov, and R. Rodrigues, “One hop lookups for peer-to-peer overlays,” in *Proceedings of the 9th conference on Hot Topics in Operating Systems - Volume 9*. Berkeley, CA, USA: USENIX Association, 2003, pp. 2–2. [Online]. Available: http://portal.acm.org/citation.cfm?id=1251054.1251056

[86] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, *LISP Alternative Topology (LISP+ALT)*, IETF Internet Draft, draft-ietf-lisp-alt-06.txt, March 2011.

[87] D. Jen, M. Meisel, D. Massey, L. Wang, Z. B., and Z. L., *APT: A Practical Transit Mapping Service*, IETF Internet Draft, draft-jen-apt-01.txt, May 2008.

[88] BGP Routing Table Analysis - DIX-IE Data, http://thyme.apnic.net/current/.

[89] A. Elmokashfi, A. Kvalbein, and C. Dovrolis, “BGP Churn Evolution: a Perspective from the Core,” *2010 Proceedings IEEE INFOCOM*, pp. 1–9, 2010.

[90] M. Degermark, A. Brodnik, S. Carlsson, and S. Pink, “Small forwarding tables for fast routing lookups,” *SIGCOMM Comput. Commun. Rev.*, vol. 27, pp. 3–14, October 1997.

[91] “DIHT Simulation source code - MobilityFirst Project,” http://www.winlab.rutgers.edu/ tamvu/NRS/sourceCode/, 2011.

[92] Y. Shavitt and E. Shir, DIMES - Letting the Internet Measure Itself, http://www.netdimes.org/.

[93] S. A. Krashakov, A. B. Teslyuk, and L. N. Shchur, “On the universality of rank distributions of website popularity,” *Computer Networks*, vol. 50, no. 11, pp. 1769 – 1780, 2006.

[94] O. Saleh and M. Hefeeda, “Modeling and caching of peer-to-peer traffic,” in *Proceedings of the Proceedings of the 2006 IEEE International Conference on Network Protocols*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 249–258.

[95] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, “Pathlet routing,” in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, ser. SIGCOMM ’09. New York, NY, USA: ACM, 2009, pp. 111–122.

[96] V. Vassiliou and Z. Zinonos, “An analysis of the handover latency components in mobile ipv6,” *Journal of Internet Engineering*, vol. 3, no. 1, pp. 230–240, Dec 2009.

[97] A. Mishra, M. Shin, and W. Arbaugh, “An empirical analysis of the IEEE 802.11 MAC layer handoff process,” *SIGCOMM Comput. Commun. Rev.*, vol. 33, pp. 93–102, April 2003.

[98] S. Y. Qiu, P. D. McDaniel, F. Monrose, and A. D. Rubin, “Characterizing address use structure and stability of origin advertisement in inter-domain routing,” in *Proceedings of the 11th IEEE Symposium on Computers and Communications*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 489–496.

[99] R. Mahajan, D. Wetherall, and T. Anderson, "A study of BGP origin as changes and partial connectivity," Oct. 2001. [Online]. Available: http://www.routeviews.org/papers/nanog_origin.pdf

[100] Y. He, G. Siganos, and M. Faloutsos, "Internet topology." in *Encyclopedia of Complexity and Systems Science*, 2009, pp. 4930–4947.

[101] S. L. Tauro, C. Palmer, G. Siganos, and M. Faloutsos, "A simple conceptual model for the Internet topology," in *IEEE GLOBECOM '01*, vol. 3, 2001, pp. 1667–1671 vol.3.

[102] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: an information plane for distributed services," in *Proceedings of the 7th symposium on Operating systems design and implementation*, ser. OSDI '06, 2006, pp. 367–380.

[103] "CAIDA : The cooperative association for internet data analysis," 2011. [Online]. Available: http://www.caida.org/

[104] H. Luo, Y. Qin, and H. Zhang, "A DHT-Based Identifier-to-Locator Mapping Approach for a Scalable Internet," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, pp. 1790–1802, December 2009.

[105] J. Hou, Y. Liu, and Z. Gong, "Silms: A scalable and secure identifier-to-locator mapping service system design for future internet," *Computer Science and Engineering, International Workshop on*, vol. 2, pp. 54–58, 2009.

[106] C. Kim, M. Caesar, and J. Rexford, "Floodless in SEATTLE: a scalable ethernet architecture for large enterprises," in *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, ser. SIGCOMM '08. New York, NY, USA: ACM, 2008, pp. 3–14. [Online]. Available: http://doi.acm.org/10.1145/1402958.1402961

[107] D. Raychaudhuri, K. Nagaraja, and A. Venkataramani, "MobilityFirst: A Robust and Trustworthy Mobility-centric Architecture for the Future Internet," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 16, no. 3, pp. 2–13, Dec. 2012.

[108] S. C. Nelson, G. Bhanage, and D. Raychaudhuri, "GSTAR: Generalized storage-aware routing for MobilityFirst in the future mobile Internet," in *Proceedings of MobiArch'11*, 2011, pp. 19–24.

[109] N. Somani, A. Chanda, S. C. Nelson, and D. Raychaudhuri, "Storage-Aware Routing for Robust and Efficient Services in the Future Mobile Internet," in *Proceedings of ICC FutureNet V, 2012*.

[110] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet Routing," in *Proceedings of SIGCOMM '09*, 2009, pp. 111–122.

[111] T. Vu et al., "DMap: A Shared Hosting Scheme for Dynamic Identifier to Locator Mappings in the Global Internet," in *Proceedings of ICDCS '12*, 2012, pp. 698–707.

[112] R. Moskowitz and P. Nikander, *Host Identity Protocol (HIP) Architecture*, IETF Internet Standard, RFC 4423, May 2006.

[113] D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker, "Accountable Internet Protocol (AIP)," in *Proc. ACM SIGCOMM*, Aug. 2008.

[114] D. Han et al., "XIA: Efficient Support for Evolvable Internetworking," in *Proceedings of NSDI'12*, 2012, pp. 23–23.

[115] D. Saucez, L. Iannone, and B. Donnet, "A First Measurement Look at the Deployment and Evolution of the Locator/ID Separation Protocol," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 2, pp. 37–43, Apr. 2013.

[116] C. Kim, M. Caesar, and J. Rexford, "Floodless in SEATTLE: A Scalable Ethernet Architecture for Large Enterprises," in *Proceedings of SIGCOMM'08*, 2008, pp. 3–14.

[117] B. Abarbanel, "Implementing global network mobility using BGP," NANOG Presentation, http://www.nanog.org/meeting-archives/nanog31/presentations/abarbanel.pdf, May 2004.

[118] Y. Wang, I. Avramopoulos, and J. Rexford, "Design for Configurability: Rethinking Interdomain Routing Policies from the Ground Up," *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 3, pp. 336 –348, April 2009.

[119] M. Py, *Multi Homing Translation Protocol (MHTP)*, IETF Internet Draft, draft-py-multi6-mhtp-01.txt, May 2002.

[120] J. Iyengar, P. Amer, and R. Stewart, "Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths," *Networking, IEEE/ACM Transactions on*, vol. 14, no. 5, pp. 951–964, 2006.

[121] B. Augustin, B. Krishnamurthy, and W. Willinger, "IXPs: mapped?" in *Proceedings of IMC '09*, 2009, pp. 336–349.

[122] Ruckus Wireless, "Long-Range 802.11n (5GHz) Wi-Fi Backhaul," http://www.ruckuswireless.com/products/zoneflex-outdoor/7731.

[123] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[124] D. Raychaudhuri et al., "Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols," in *Proc. of IEEE WCNC '05*, vol. 3, March 2005, pp. 1664–1669.

[125] Y. Shavitt and E. Shir, "Dimes: let the internet measure itself," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 71–74, Oct. 2005.