Performance Evaluation and Management of the Internet

インターネットの性能評価と管理に関する研究

February, 2014

Masaki Fukushima

福嶋　正機

Performance Evaluation and Management of the Internet

インターネットの性能評価と管理に関する研究

Waseda University

Graduate School of Fundamental Science and Engineering

February, 2014

Masaki Fukushima

福嶋　正機

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Since the early developments and deployments of TCP/IP in the 1970s, the Internet has evolved to the indispensable infrastructure for many areas of our society, ranging from communication, commerce, health care, public administration, politics to education. Since this success of the Internet goes far beyond the extent that the original designer of the Internet architecture supposed, the current Internet are facing various technological challenges that were not predicted in its early days. Furthermore, as a result of this success, the numerous investments in the current Internet ecosystem create a barrier to introduce drastic changes to its current architecture.

For example, the Internet traffic has been increasing far beyond the prediction in the past. In particular, mobile network traffic is recently expected to continue increasing exponentially. Due to this increasing traffic demand, it is becoming more and more challenging for service providers to provision sufficient network resources such as link bandwidth and node capacity in a timely manner to keep the quality of experience (QoE) perceived by users of their services at a certain level. Besides, the networking research community has begun to recognize that the fundamental solution of these problems is difficult with the current Internet architecture based on Internet Protocol (IP).

In order to overcome these challenging issues of the current Internet, many efforts to develop *future network* technologies are being carried out. These efforts have achieved a certain level of success, establishing individual technologies to address each issue of the current Internet. In a large-scale system like the Internet, however, a collection of individual technologies is not sufficient as a solution of practical issues. In addition to the individual technologies, operational aspect of the Internet is essential to address these issues.

1.2 Motivation

In this section, we describe the research areas investigated in this study and the motivations that drive this study in more detail.

First, mobile devices are increasingly becoming inevitable tools to access the Internet. “Systems beyond IMT-2000” [1] require allocation of spectrum resources to realize sufficient capacity, e.g. up to 100 Mbit/s under high mobility environments, to accommodate the demand of upcoming services and applications. However, spectrum resources suitable for these purposes are limited. Furthermore, compared to current homogeneous cellular systems, the design of systems beyond IMT-2000 includes the allocation of different high bandwidth wireless systems and user mobility between the heterogeneous systems. For the successful launch of systems beyond IMT-2000, it is essential to address the problem of frequency resource allocation among different systems Although it is critical what fraction of traffic is accommodated by each system for appropriate frequency assignment, the capability of “seamless system interworking” in future mobile networks makes it difficult to be estimated. Routing in systems beyond IMT-2000 supports seamless roaming among the systems by maintaining active connections, and then, users can seamlessly select either system. In these networks, macrocell cellular systems such as the third generation (3G) systems including CDMA2000 1xEV-DO (or HDR [2]) and high bandwidth microcell wireless systems such as Wireless LAN (WLAN) will complement each other. When high bandwidth and smaller cell size systems such as WLAN is integrated in seamless interworking environment, user’s mobility is not a negligible factor. Traffic distribution among different bit rate systems depends on the user’s mobility. However, it is not known how this distribution impacts on the total network throughput of the systems.

Second, assessing QoE is essential for network operators to retain customer satisfaction. in order to appropriately design and size the network capacity, it is required to measure QoE perceived by users to deduce the network status from users’ point point of view. However, compared to traditional quality of service QoS criteria from network perspective such as link utilization and packet loss ratio, there are some barriers to measure actual QoE. For example, measurement of QoE in the network (not in the users’ device) is inefficient because it requires some dedicated device capable of deep packet inspection (DPI). It is not realistic to deploy such costly devices to hundreds of PoPs in a large operator’s network.

Third, privisioning network resources in response to QoE degradation is also inevitable for network operators. As multiple access technologies to the Internet become available to users, such as ADSL, FTTH and 3G/4G wireless, Internet Service Providers (ISPs) are facing a mixture of challenges that may seem harder than ever to fulfill concurrently, such as (1) extending the *footprint* to cover a large user base and multiple access means per user, (2) reducing operational *costs*, (3) improving network *availability*, and

(4) maintaining *operational confidentiality.* The emerging concept of network virtualization (NV) recently proposed in various projects [3, 4, 5] is expected to help ISPs achieve some of the goals, especially (1) to (3), simultaneously. For example, virtual collocation [6, 7] separates *Infrastructure Providers* (*InPs*) that provide multiple isolated slices of physical resources and *Service Providers* (*SPs*) that utilize slices to operate virtual networks, in order for the SPs to *cost-effectively* extend their network *footprints* on top of multiple InPs without investing in physical infrastructure and to improve network *availability* by splicing multiple paths [8]. However, the last bullet (4) mentioned above, *operational confidentiality*, may remain unresolved despite such a new concept of NV to the rescue for achieving the diverse mixture of the goals. We observe that an ISP often strives to keep its competitors away from the operational practices developed to survive business tussles. It is common for followers of market leaders in various businesses to analyze their strategies/operations and to employ them to catch up [9]. Free-riding on other ISPs' operational expertise turns out to be quite effective, since ISPs often cultivate the market in similar geographical regions [10, 11]. For instance, in the German wireless telecommunication market, followers have adopted such a "herding strategy" to bring severe price-cutting competition, and profits have plunged 50% in five years [12]. In the light of these observations, one must note that NV, in fact, may have a negative impact on the *confidentiality of SPs' operational information*, since virtual collocation [6, 7] implies that an InP runs its own SP service on its top as well as on the other (competing) InPs that have access to the operational details of the SP on top of them. For example, virtual collocation may allow each of two competing InPs, such as AT&T and Verizon, to extend the footprint of their own SP services over the resources of the other, although this endangers the operational confidentiality of both SPs. Therefore, if NV is to be employed to satisfy all the goals (1) to (4) of ISPs mentioned previously, the challenge is to achieve secure network operations of SPs without disclosing too much information to the underlying InPs.

Lastly, efficiency of content distribution has emerged as a pressing issue that needs to be addressed by future networks. Recently, Information-Centric Networking (ICN) [13] is getting more and more attentions in the future internet research community, as a means for enabling scalable and cost-efficient content distribution, intrinsic mobility, and multihoming. Content-Centric Networking (CCN) [14] pursued in Named-Data Networking (NDN) project is a promising ICN architecture that employs a hierarchical content naming scheme. In CCN, *content names* are location independent and there is no notion of *locator* like IP address. Instead of name resolution like DNS, the bindings between a content name and its content source locations are gradually resolved by routers in *a hop-by-hop basis.* Each CCN router has a Forwarding Information Base (FIB), which binds every content name prefix to the next hops (i.e., the outgoing faces) toward its content sources. When a router needs to forward an Interest packet for a content name, it looks up the name in its FIB by longest prefix match, retrieves the next hop information, and forwards the packet to the next hop routers. Such hop-by-hop content locating naturally supports efficient content distribution, mobility, and multihoming. A popular content can be hosted by many content sources without managing many locators like in current CDNs. Moreover, each content source can change/add its attachment points to the network without globally advertising the new locators of these points to its content consumers or routing packets via an indirection point like in Mobile IP. On the other hand, such location independent names raise scalability issues on FIB [15, 16]. Since location independent names are assigned to content sources regardless of their topological locations, name prefixes are hard to aggregate and thus the FIBs of CCN routers will be far larger than those of current IP routers. Thus, it is crucial to efficiently store and lookup such large FIBs. Fortunately, the downside trend of DRAM cost will enable CCN routers to store large FIBs on memory. FIB lookup latency issue is, however, more challenging due to the large latency of DRAM access and the complexity of longest prefix matching on variable- and unbounded-length names. Software-based FIB mechanisms employ a hash table [15] or trie [17]. Regardless of the underlying data structures, they need to seek the longest matching prefix through all candidate prefix lengths (in descending order with a hash table, or ascending order with a trie), and thus the number of random accesses to the DRAM per lookup is proportional to the name length, which makes the FIB lookup latency proportional to the name length and FIB throughput inversely proportional to the name length. In order to eliminate this limitation, hardware-based FIB mechanisms are proposed [16, 18], which store a Bloom filter in a low-latency on-chip (SRAM) memory and populate it with prefixes. For each prefix length, the fast Bloom filter is checked first, and only if it gives a positive result, a slow hash table in an off-chip DRAM is probed to retrieve the next hop information. If given a sufficiently large on-chip memory, a name lookup involves only a single DRAM access regardless of the name length. However, this hardware-based FIB requires an expensive on-chip memory of the size proportional to the number of prefixes, which makes its Internet-scale deployment in backbone routers infeasible.

1.3 Research Proposal

In this study, we pose some key challenging issues of the operational aspect of the future networks and propose solutions to address these operational issues.

First, we need to design mobile networks that can accommodate the growing traffic demand. Recently, there have been many proposals to increase the capacity of mobile networks by combining various wireless access technologies and multiple radio frequency bands. However, it is not clear, if such heterogeneous wireless access systems (e.g., cellar systems and WLAN systems) are interworking, how the traffic is distributed to each systems, how mobility of users affects the traffic distribution, and how such interwroking affects QoE perceived by users. This study shows that such mobility of users has considerable impacts on the total network throughput of the systems beyond IMT-2000. We also propose an analysis method to evaluate these problems based on the theory of queueing networks. In the proposed model the user's data transfer requests are modeled at the flow level, not the packet level. Such a modeling approach is widely accepted for modeling both wired and wireless systems [19, 20]. The proposed approach consists of three steps. First, the target system is characterized according to the *system model* described in Sect. 2.3. Secondly, conventional and proposed *queueing network models* are described in Sect. 2.4. Next, the system model parameters are mapped onto the queueing network model parameters as shown in Sect. 2.5. Finally, the queueing network model is analyzed as described in Sect. 2.6.

Second, in order to address the issue of QoE measurement, we propose an efficient QoE measurement method that only requires to count the number of control messages in the transport layer. We also verify the effectiveness of the proposed method by using actual traffic trace. The proposed measurement method is lightweight. It extracts 6-bit control flags of TCP (Transmission Control Protocol) packets. The idea is based on the unique feature of flag ratios which is discovered by our exhaustive search for the new indexes of network traffic. By the use of flag ratios, one can tell if the network is really congested. It is much simpler than the conventional network monitoring by a network analyzer. The well-known monitoring method is based on the utilization parameter of a communication circuit which ranges from 0% to 100%. One cannot tell the line is congested even if the factor is 100%. 100% means full utilization and does not give any further information. To calculate the real performance of the network, one should estimate the throughput or effective speed of each user. The estimation needs much calculation. Our new method tries to correlate ratios of TCP control flags and network congestion. The result shows the usefulness of this new method. This study analyzes the reason why the flag ratios show the unique feature.

Third, if it is detected that QoE of a network service is degraded due to increasing traffic demand, it is required to assign more network resources such as link or router to the service. In order to flexibly provision such resources in a timely manner, network virtualization technologies have been proposed. By using network virtualization, a service provider can create a wide-area virtual network by purchasing virtual links and routers from infrastructure providers. In such a virtual network environment, however, the service provider has a concern on the confidentiality of its operational information. Namely, the underlying infrastructure providers can, at least technically, investigate any information stored in the service provider's virtual routers. In order to address this confidentiality issue, we propose a new routing protocol, which applies a cryptographic protocol called secure multiparty computation and enables virtual routers to perform routing computation without disclosing any routing information to the underlying infrastructure providers. We show that the proposed protocol has practically reasonable performance by experiments on a testbed. More conceretely, we propose a solution for *Minimum Disclosure Routing (MDR)*, the problem for an SP overlaid on top of multiple InPs to minimize the disclosure of its virtual network routing information to the underlying InPs. We posit that MDR can lift the SPs' barrier to entry in employing network virtualization, since the disclosure of confidential routing information prevents SPs from utilizing virtualized network resources from InPs; mainly for the following two reasons. First, SPs are generally sensitive to the disclosure of *any* of their routing information. Second, the disclosure of specific routing information such as path/link cost leads to the disclosure of more sensitive information such as quality, bandwidth, and topology. For example, the quality of services can be inferred from path cost information [21]. Bandwidth is easily estimated from link cost because there is a well-known practice that link cost should be configured to the inverse of link bandwidth [22]. Topology and link costs can be inferred from path cost information even if the link costs of individual links are not directly disclosed by the routing protocol (as in RIP or OSPF inter-area routing) [23]. The above mentioned information gives the competitors of an SP significant advantage, which can be exploited to develop services that are more efficient and less costly than the SP's one. Such cunning competitors can easily defeat the SP in the highly competitive market. Our contributions in this study are two-fold. First, as the continuation of our preliminary report[24] with an informal description of MDR and sketch of the proposed solution, this study formally defines MDR and proposes a detailed design of the solution based on our extension to *Secure Multiparty Computation* (*SMC*) [25], where multiple parties cooperatively compute a function from each party's confidential input. The generic SMC protocol [26] is not simply applicable to MDR, since it requires full-mesh connectivity to share the intermediate computation results among participants, while routing is a problem to establish such logical full-mesh connectivity. Thus, we design a new primitive operation called TRANSFER to securely transfer intermediate results between neighboring virtual routers and allow these intermediate results to be shared among them. Second, we implement the proposed solution in our testbed and evaluate its performance by experiments as well as by the analytical model for comparison. We reveal the proposed solution is supposed to achieve secure routing without degrading the convergence time despite the overhead of the SMC protocol. Accordingly, we conclude that our proposed solution to MDR viably complements the concept of network virtualization to help fulfill all the contradicting goals of ISPs concurrently.

After all, the above mentioned problems of increasing traffic demand have a common source; the current Internet architecture does not necessarily fit with the current use of the Internet. Originally, the Internet was designed as a network for hosts to communicate with each other. In contrast, the users of today use the Internet as a network to distribute or access contents. Recently, it is pointed out that this mismatch is a critical limitation of the current Internet architecture based on IP. The concept of content-centric networking (CCN) is a promising approach to overcome this limitation. Although the CCN architecture has been an active research area, the operational aspect of CCN is not yet attracted sufficient interest. For example, if CCN is operated in the Internet scale, the forwarding information base (FIB) of CCN routers must store hundreds of millions of name prefixes, because the content names in CCN is assigned regardless of their topological locations. This results in the problem of looking up such a huge FIB with longest prefix matching. In this study, we propose a new efficient FIB lookup scheme and verify the effectiveness of the proposed scheme by using actual Internet topology data. More concretely, we propose a new scheme to improve the efficiency of FIB lookup, which can be applied to the software-based FIBs for faster lookup and to the hardware-based FIBs to reduce on-chip memory. The proposed scheme is motivated by the observation that Interest packets matching a non-aggregatable prefix are forwarded by the same prefix length at every hop. Therefore, by exploiting the information on the longest matching prefix length in the previous hop, each CCN router could find the longest matching prefix without prefix seeking. Although, in the current CCN protocol, a router cannot learn the prefix length matched in the previous hop, it is easy to add some link-local header that carries the prefix length information along with each Interest packet. This does not violate the hop-by-hop principle of CCN, and can be incrementally deployed in the network.

Chapter 2

Modeling Interworking Mobile Systems

2.1 Introduction

“Systems beyond IMT-2000” [1] require allocation of spectrum resources to realize sufficient capacity, e.g. up to 100 Mbit/s under high mobility environments, to accommodate the demand of upcoming services and applications. However, spectrum resources suitable for these purposes are limited. Furthermore, compared to current homogeneous cellular systems, the design of systems beyond IMT-2000 includes the allocation of different high bandwidth wireless systems and user mobility between the heterogeneous systems. For the successful launch of systems beyond IMT-2000, it is essential to address the problem of frequency resource allocation among different systems

Although it is critical what fraction of traffic is accommodated by each system for appropriate frequency assignment, the capability of “seamless system interworking” in future mobile networks makes it difficult to be estimated. Routing in systems beyond IMT-2000 supports seamless roaming among the systems by maintaining active connections, and then, users can seamlessly select either system. In these networks, macrocell cellular systems such as the third generation (3G) systems including CDMA2000 1xEV-DO (or HDR [2]) and high bandwidth microcell wireless systems such as Wireless LAN (WLAN) will complement each other. When high bandwidth and smaller cell size systems such as WLAN is integrated in seamless interworking environment, user’s mobility is not a negligible factor. As we describe in Sect. 2.2, traffic distribution among different bit rate systems depends on the user’s mobility. This paper shows that such mobility of users has considerable impacts on the total network throughput of the systems beyond IMT-2000. We also propose an analysis method to evaluate these problems based on the theory of queueing networks.

In the proposed model the user’s data transfer requests are modeled at the flow level, not the packet level. Such a modeling approach is widely accepted for modeling both wired and wireless systems [19, 20]. The proposed approach consists of three steps. First, the target system is characterized according to the *system model* described in Sect. 2.3. Secondly, conventional and proposed *queueing network models* are described in Sect. 2.4. Next, the system model parameters are mapped onto the queueing network model parameters as shown in Sect. 2.5. Finally, the queueing network model is analyzed as described in Sect. 2.6.

2.2 Mobility Impacts on Traffic Distribution

In this section, we describe the relationship between mobility and bandwidth-sharing in seamless interworking environments. The point is that mobility has impacts on the traffic distribution among different bit rate systems. In this study, the term “traffic” means the amount of data transfered over the wireless links.

In the existing mobile systems, a user’s mobility events such as hand-off are typically related to the time scale of minutes or larger order. Contrary, data transfer events such as web page requests are related to time scale of seconds order. Because these two time scales are sufficiently separated, it is quite reasonable for the existing studies modeling mobile system to focus on either time scale. Some studies analyze discrete channel systems such as 2G cellular systems with mobility (e.g. [27, 28]) and others analyze packet-based bandwidth-sharing systems such as WLAN or 1xEV-DO without mobility (e.g. [20]). There has been no intense demand for comprehensive evaluation or modeling framework involving both of the two time scales simultaneously.

The relevant time scales of systems are, however, unclear in future seamless interworking environments. For example, consider a vehicle equipped with a car navigation system connected to the network. It may employ WLAN for downloading high-resolution map data in an urban area. The car, however, must switch to 3G cellular systems when it accelerates or leaves the WLAN service area. Some future systems will employ smaller cell size due to the use of higher frequency bands. Additionally, user-perceived performance of the systems will be more sensitive to the user’s mobility due to adaptive modulation technology. Under such environments, the time scale of mobility approximates that of bandwidth-sharing behavior.

If the time scale of mobility and bandwidth-sharing behavior is not sufficiently separated, traffic distribution among different bit rate systems depends on the user’s mobility. The following intuitive example describes how mobility affects traffic distribution among different bit rate systems. Let h_1 and h_2 be the data rate of cellular and WLAN respectively. Consider a system with only one user moving about the cellular cell at a constant velocity. Given the trajectory of the user, the position of the user is a function of Vt, where V is the velocity of the user and t is the time. Because the bit rate is also a function of the position, the rate available to a mobile can be written as $H(Vt)$. The value of H depends on which area the mobile resides and its steady state probability $\Pr(H = h_i) = p_i$. We assume that user's data transfer requests are independent of the position of the user. If $V \to 0$, the traffic is accommodated by a single system depending on the initial state and the overall traffic distribution simply follows the distribution of H, i.e. $p_1 : p_2$. Meanwhile, if $V \to \infty$, the fraction of traffic accommodated by a system is proportional to the product of its data rate and steady state probability, i.e. the traffic distribution is $h_1 p_1 : h_2 p_2$. The bit rate of WLAN is much higher than cellular system ($h_2 > h_1$). The ratio of these data rates is nearly ten times because the typical data rate of 3G cellular systems is a few megabits per second and that of WLAN systems is a few tens of megabits per second. Consequently, the fraction of traffic accommodated by the higher bandwidth system increases and the load of lower bandwidth systems is reduced. Actual systems fall on the somewhere between the above two extreme cases.

This paper examines how interaction between mobility events and bandwidth-sharing events is a major factor in defining the traffic distribution among different systems. Because none of the existing models targets such a mixture of two time scales, a novel methodology is required to consider the two types of events in a unified model. This paper presents a modeling framework and method to analyze such unified models based on the theory of queueing networks. Joint process of "independent" behavior such as mobility and "interactive" behavior such as bandwidth sharing cannot be analyzed by conventional queueing network models. The proposed analysis method yields request arrival rates as a set of nonlinear traffic equations with state-dependent routing. Assuming independence among nodes in the queueing network, the proposed method reduces the computational cost compared to the brute force Markov model approach. The accuracy of the proposed method is verified numerically by comparison with event-driven simulations. The method provides a quantitative analysis for understanding the impact of mobility on the bandwidth sharing and traffic distribution in the seamless interworking environments.

2.3 System Model

The mobile network in this study consists of one or more base stations. Users arrive at the network, request data transfer, move about the service area of the network during transfer, are occasionally handed over between base stations and finally complete the request and depart from the network. Various systems can be modeled with the proposed model. The notations used in the model are summarized in Table 2.1.

Table 2.1: Notation Summary

notation	system model	queueing network model
λ	total arrival rate of users	arrival rate of customers from outside
$\mathcal{R}$	request state space	–
$\boldsymbol{a}^{\mathrm{R}}$	request state initial distribution	–
$\boldsymbol{P}^{\mathrm{R}}$	request state transition probability matrix	–
$\boldsymbol{\nu}^{\mathrm{R}}$	request size parameter	–
$\mathcal{M}$	mobility state space	–
$\boldsymbol{a}^{\mathrm{M}}$	mobility state initial distribution	–
$\boldsymbol{P}^{\mathrm{M}}$	mobility state transition probability matrix	–
$\boldsymbol{\nu}^{\mathrm{M}}$	mobility state duration parameter	–
$\mathcal{J}$	set of resources (typically base stations)	set of nodes
$\mathcal{U} = \mathcal{R} \times \mathcal{M}$	user state space	set of customer classes
$\mathcal{U}_j$	set of user states that resource j serves	set of customer classes at node j
$h_j(u)$	bandwidth provided to user state u by resource j	–
$\Phi_j(n_j) = \phi_{j1}(n_j)$	multi-user diversity gain	request service rate of node j with n_j customers
$\mathcal{D} = \{1, 2\}$	1=request, 2=mobility	set of service types
λ_{ju}	–	arrival rate of class u customer at node j from outside
μ_{jud}	–	type d service requirement of class u customer at node j
$r_{jud,kv}$	–	service-type-dependent routing probability
α_{ju}	–	total arrival rate of class u customer at node j

2.3.1 User Model

Users arrive at the network according to a Poisson process with rate λ. The behavior of a user is characterized by *request parameters* and *mobility parameters*.

Request Parameters

A user requests data transfer according to a Markov chain on a discrete state space $\mathcal{R}$. The initial state distribution is $\boldsymbol{a}^{\mathrm{R}} = (a_i^{\mathrm{R}};\ i \in \mathcal{R})$ and the state transition probability is $\boldsymbol{P}^{\mathrm{R}} = (p_{ij}^{\mathrm{R}};\ i,j \in \mathcal{R})$. The superscript "R" means these parameters are related to the request of users. $\boldsymbol{P}^{\mathrm{R}}$ is a substochastic matrix, i.e. $\sum_{j\in\mathcal{R}} p_{ij}^{\mathrm{R}} \leq 1$. After request completion at state i, the user changes its state to j with probability p_{ij}^{R} or leaves the network with probability $1 - \sum_{j\in\mathcal{R}} p_{ij}^{\mathrm{R}}$. Departure from the network by request state transition means that the user terminates the connection with the mobile network. At state i, the user's request size follows exponential distribution with rate ν_i^{R}. Let vector $\boldsymbol{\nu}^{\mathrm{R}} = (\nu_i^{\mathrm{R}};\ i \in \mathcal{R})$. Request size typically represents the transferred data size.

Mobility Parameters

Mobility is modeled as a Markov chain on the discrete state space $\mathcal{M}$. The initial state distribution is $\boldsymbol{a}^{\mathrm{M}} = (a_i^{\mathrm{M}};\ i \in \mathcal{M})$ and the state transition probability is $\boldsymbol{P}^{\mathrm{M}} = (p_{ij}^{\mathrm{M}};\ i,j \in \mathcal{M})$. The superscript "M" means these parameters are related to the mobility of users. Again $\boldsymbol{P}^{\mathrm{M}}$ is a substochastic matrix. State duration time at state i follows exponential distribution with rate ν_i^{M}. Let $\boldsymbol{\nu}^{\mathrm{M}} = (\nu_i^{\mathrm{M}};\ i \in \mathcal{M})$. Mobility states typically represent a mobile's discretized location, velocity, direction or a combination of them. After state duration expiration at state i, the user changes its state to j with probability p_{ij}^{M} or leaves the network with probability $1 - \sum_{j\in\mathcal{M}} p_{ij}^{\mathrm{M}}$. Departure from the network by mobility state transition means that the user moves outside the service areas of the network. For example, if mobility of a user is represented as the base station that the user is served, mobility state space is a set of base stations, mobility state change represents hand-over between base stations and state duration represents dwell time to the area of the base station. Mobility state space might be more fine-grained such as a set of small square grids dividing whole are of a mobile system. In this case, mobility state is a pair of grid and traveling direction (north, south, west or east) of the mobile. Mobility state changes when the mobile changes its direction as well as the mobile moves from a grid to a neighboring gird.

Consequently, the state of a user is a discrete-state stochastic process on state space $\mathcal{U} = \mathcal{R} \times \mathcal{M}$. A user state is denoted by $u = (u_1, u_2),\ u_1 \in \mathcal{R}, u_2 \in \mathcal{M}$. Because request and mobility state durations are described as exponential distributions, a user's state is completely characterized over state space $\mathcal{U}$. Although we use exponential distribution, general distributions can be approximated by phase-type distribution at the expense of additional computational cost.

2.3.2 Base Station Model

The mobile systems in the network consist of *resources*. A resource is typically a base station in omni-cell systems or a sector of a base station in sector-cell systems. A resource is shared by and assigned to users according to some kind of multiplexing mechanism, e.g. the time slot in the TDM system or spreading code in the CDM system. The set of all resources in the network is denoted by $\mathcal{J}$. The set of user states that resource $j \in \mathcal{J}$ provides the service is $\mathcal{U}_j \subseteq \mathcal{U}$. All resources are exclusive to the user state space they serve, i.e. $|\mathcal{U}_j \cap \mathcal{U}_k| = 0$ for all $j \neq k$.

Some systems transfer data at rates depending on the signal strength, e.g. 1xEV-DO adaptive modulation [29] and IEEE802.11 WLAN data rate fallback. The data rate provided for a user is therefore a function of user state $u \in \mathcal{U}_j$ and denoted by $h_j(u)$. The total throughput of a resource may depend on the number of simultaneous users, e.g. multi-user diversity in 1xEV-DO [30, 29]. The service rate of resource j with n_j users is $\Phi_j(n_j)$. Note that $\Phi_j(n_j)$ is normalized to satisfy $\Phi_j(1) = 1$. Consequently, the request of a user with user state u sharing resource j with n_j users including itself is served at rate $\Phi_j(n_j)h_j(u)/n_j$.

There might be virtual resources corresponding to the user's thinking time or browsing time. The service rate of these resources is $\Phi_j(n_j) = n_j$, i.e. unit service rate per user regardless of n_j. The mean request size at such a resource is therefore the mean duration of idle time.

2.4 Queueing Network Model

We use a queueing network model to analyze the system described in Sect. 2.3. Note that we use the word *customer* to denote the entity in the queueing model and *user* to denote the entity in the system model.

2.4.1 Conventional Queueing Network

In this sub-section, the conventional queueing network model described in the literature [31, 32] is summarized. A network consists of one or more nodes and the set of all nodes in the network is denoted by $\mathcal{J}$. There are multiple classes of customers in the network. Customer classes are defined node by node. Let $\mathcal{U}_j$ be the set of all customer classes at node j. The arrival rate from the outside of the network and mean service requirement of customer class $u \in \mathcal{U}_j$ at node j is λ_{ju} and μ_{ju}^{-1} respectively. The total service rate provided by node j with n_j customers is $\phi_j(n_j)$. Service is provided according to the *symmetric service discipline* [31] with parameter $\delta_j(\ell, n_j)$. Proportion $\delta_j(\ell, n_j)$ of the total service rate is directed to the customer in position ℓ, $\ell = 1, 2, \ldots, n_j$. A typical example of symmetric service is a server-sharing queue with parameter $\delta_j(\ell, n_j) = 1/n_j$, which divides the service rate evenly among n_j customers. Another application of symmetric service is infinite server with parameters $\phi_j(n_j) = n_j$ and $\delta_j(\ell, n_j) = 1/n_j$. Customers are always served at unit rate in infinite servers.

When a class u customer departs from node j, the customer arrives at node k as class v with routing probability $r_{ju,kv}$. The customer also leaves the network with probability $r_{ju,0}$. Because the routing probability is a probability distribution,

$$\sum_{k \in \mathcal{J}} \sum_{v \in \mathcal{U}_k} r_{ju,kv} + r_{ju,0} = 1, \quad j \in \mathcal{J}, u \in \mathcal{U}_j. \tag{2.1}$$

Although the network of symmetric queues can represent a very general class of networks and is proved to have tractable product form solutions, this model cannot model the system model described in Sect. 2.3. In our system model, the routing probability depends on the state of node because request state transition rate is divided among users while mobility state transition is independent of other users. Thus nodes must have the property of server-sharing queue and infinite server simultaneously.

2.4.2 Queueing Network with Multiple Service Types

We extend the queueing network model described in the previous sub-section by introducing *multiple service types* and *service-type-dependent routing.* A node in the queueing network with multiple service types has multiple types of service facility and the routing probability depends on the type of service the customer has completed.

This model also has parameters $\mathcal{J}$, $\mathcal{U}_j$, λ_{ju} and $\delta_j(\ell, n_j)$. Let $\mathcal{D}$ be the set of all service types. Each type of service is provided by the symmetric service discipline. Type $d \in \mathcal{D}$ service is provided at rate $\phi_{jd}(n_j)$. Thus a customer at position ℓ of node j with n_j customers is provided type d service at rate $\phi_{jd}(n_j)\delta_j(\ell, n_j)$. The mean service requirement of class u customer for type d service at node j is μ_{jud}^{-1}.

Customers are provided all types of services simultaneously. When any type of service is completed, the other types of services are immediately aborted and the customer departs from the node. When a class u customer departs from node j by type d service completion, the customer arrives at node k as class v with routing probability $r_{jud,kv}$ or leaves the network with probability $r_{jud,0}$, therefore

$$\begin{aligned} \sum_{k \in \mathcal{J}} \sum_{v \in \mathcal{U}_k} r_{jud,kv} + r_{jud,0} = 1, & \\ j \in \mathcal{J},\ u \in \mathcal{U}_j,\ d \in \mathcal{D}. & \end{aligned} \tag{2.2}$$

2.5 Mapping to Queueing Network Model

In this section, the system model introduced in Sect. 2.3 is described in the framework of the queueing network with multiple service types. The set of system resources corresponds to the set of nodes in queueing network $\mathcal{J}$. There are two types of service, i.e. $\mathcal{D} = \{1, 2\}$. Service type 1 and 2 represent *request service* and *mobility service* respectively. User state and customer class are one-to-one correspondence. Thus the set of customer classes is the user state space $\mathcal{U} = \mathcal{R} \times \mathcal{M}$.

Let u, v be user states and $u = (u_1, u_2),\ v = (v_1, v_2),\ u_1, v_1 \in \mathcal{R},\ u_2, v_2 \in \mathcal{M}$. The total arrival rate λ is distributed among nodes and customer classes according to the initial distribution of the underlying Markov chains and thus $\lambda_{ju} = \lambda a^{\mathrm{R}}_{u_1} a^{\mathrm{M}}_{u_2}$. The request service requirement is related to the mean request size and data transfer rate of the user and $\mu_{ju1} = \nu^{\mathrm{R}}_{u_1} h_j(u)$. The request service rate is equal to the service rate of the resource. Thus $\phi_{j1}(n_j) = \Phi_j(n_j)$. Mobility service is essentially described as an infinite server in the queueing network, i.e. $\phi_{j2}(n_j) = n_j$. The mobility service requirement is the same as the mobility state duration and $\mu_{ju2} = \nu^{\mathrm{M}}_{u_2}$. Because both the request and mobility service rate is shared evenly among users, $\delta_j(\ell, n_j) = 1/n_j$.

Type 1 service completion represents the request state transition and type 2 service completion represents the mobility state transition. Therefore, the service-type-dependent routing probability is described with the state transition probability of the underlying Markov chains as

$$r_{jud,kv} = \begin{cases} p^{\mathrm{R}}_{u_1 v_1}, & d = 1, u_2 = v_2, \\ p^{\mathrm{M}}_{u_2 v_2}, & d = 2, u_1 = v_1, \\ 0, & \text{otherwise}. \end{cases} \tag{2.3}$$

2.6 Analysis of Queueing Network Models

2.6.1 Analysis of Conventional Queueing Network

An analysis of a conventional queueing network without the service type described in Sect. 2.4.1 is presented in the literature [31, 32]. Let α_{ju} and β_{ju} be the overall average arrival and departure rates of customer class u at node j. They are related by *traffic equations*

$$\alpha_{ju} = \lambda_{ju} + \sum_{k \in \mathcal{J}} \sum_{v \in \mathcal{U}_k} \beta_{kv} r_{kv,ju},\ j \in \mathcal{J},\ u \in \mathcal{U}_j. \tag{2.4}$$

Because $\beta_{kv} = \alpha_{kv}$ in the conventional queueing network model, Eq. (2.4) is a set of linear equations related to $\boldsymbol{\alpha} = (\alpha_{ju}; j \in \mathcal{J}, u \in \mathcal{U}_j)$. The network has a product form solution and the steady state probability of each node is the same as that of a node with the Poisson arrival rate $\boldsymbol{\alpha}_j = (\alpha_{ju}; u \in \mathcal{U}_j)$. Let $\rho_{ju} = \alpha_{ju}/\mu_{ju}$ and $\rho_j = \sum_{u \in \mathcal{U}_j} \rho_{ju}$. The steady state probability of n_j

customers at node j is

$$\pi_j(n_j) = b_j^{-1} \frac{\rho_j^{n_j}}{\prod_{\ell=1}^{n_j} \phi_j(\ell)} \tag{2.5}$$

where b_j is the normalization constant given by

$$b_j = \sum_{n=0}^{\infty} \frac{\rho_j^n}{\prod_{\ell=1}^{n} \phi_j(\ell)}. \tag{2.6}$$

2.6.2 Analysis of Network with Service Types

The analysis of queueing network with service types is summarized in this section. Details of the analysis are shown in 2.8.

Because it is intractable to directly analyze the network with the type-dependent service requirement μ_{jud}, we consider a network with the type-independent service requirement μ_{ju} whose transition rates are approximately equivalent to that of the original network. The equivalent network is characterized by μ_{ju}, $r'_{jud,kv}$, $\phi'_{jd}(n_j)$. These parameters are determined by minimizing the residual of the approximation.

Let α_{ju} be the overall average arrival rate of class u customer at node j of the equivalent network and $\phi_j(n_j) = \sum_{d\in\mathcal{D}} \phi'_{jd}(n_j)$. The average departure rate of the equivalent network β_{jud} is

$$\beta_{jud} = {b_j}^{-1} \alpha_{ju} \sum_{n_j=1}^{\infty} \frac{\phi'_{jd}(n_j) {\rho_j}^{n_j-1}}{\prod_{\ell=1}^{n_j} \phi_j(\ell)}. \tag{2.7}$$

Departure rate β_{jud} satisfies $\sum_{d\in\mathcal{D}} \beta_{jud} = \alpha_{ju}$. The steady state probability of a node can be given by applying Eq. (2.5) to the equivalent network. Let $\boldsymbol{\alpha} = (\alpha_{ju};\ j \in \mathcal{J}, u \in \mathcal{U}_j)$. The departure rate β_{jud} is a nonlinear function of $\boldsymbol{\alpha}$ because ρ_j is a function of $\boldsymbol{\alpha}$. Let departure rate β_{jud} be denoted by $\beta_{jud}(\boldsymbol{\alpha})$ to emphasize that they are functions of $\boldsymbol{\alpha}$. The traffic equations are

$$\alpha_{ju} = \lambda_{ju} + \sum_{k\in\mathcal{J}} \sum_{v\in\mathcal{U}_k} \sum_{d\in\mathcal{D}} \beta_{kvd}(\boldsymbol{\alpha}) r'_{kvd,ju}, \qquad j \in \mathcal{J},\ u \in \mathcal{U}_j. \tag{2.8}$$

The above traffic equation is a set of nonlinear equations related to $\boldsymbol{\alpha}$. Solving the traffic equation is a kind of *fixed point problem* and an iterative method can be applied [32]. (A vector $\boldsymbol{x}$ is called a fixed point of a function f if $f(\boldsymbol{x}) = \boldsymbol{x}$.) The larger the state space of the underlying Markov chain, the more computational cost required to solve a network with multiple service types. By virtue of the independence assumption, the computational cost of the proposed analysis method is restricted to the order of $O(|\mathcal{U}|)$. Note that this is much smaller than that of the brute force Markov model such as $O(|\mathcal{U}|^M)$ where M is the maximum number of users in the network.

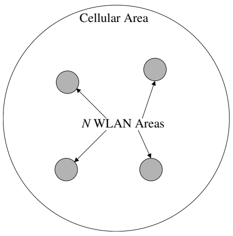


Figure 2.1: A Cellular Cell and N WLAN APs

2.7 Numerical Results

In this section we show some numerical examples of heterogeneous interworking environments. The results of the proposed analysis give us useful insights into the traffic distribution among different bit rate systems.

2.7.1 Cellular-WLAN Overlay

Consider a cellular-WLAN overlay environment like Fig. 2.1. There are two types of systems: a 3G cellular cell and N WLAN access points (APs) surrounded by the cellular cell. Users always use WLAN when it is available. Users switch between the systems when they cross the boundary of the systems. It is assumed that the hand-off between the systems involves no overhead. The above system is represented as a $(N+1)$-node queueing network model. Thus $\mathcal{J} = \{1, \ldots, N+1\}$. The node 1 corresponds to the cellular base station and $2, \ldots, N+1$ to the WLAN APs.

In the existing WLAN systems, it might be difficult to have the communications for the terminal with high speed mobility or moving the edge of the WLAN area due to hand-off overhead. There are some research activities in this area [33, 34]. Some of these systems employ dual wireless interfaces approach to minimize hand-off overhead and preserve communication during hand-off even at the edge of the WLAN area. For example, [33] concludes that handover time of WLAN can be improved to 61 milliseconds. This corresponds to hand-off region of 1.7 meters under velocity of 100 km/h. In this study, we consider that WLAN or similar microcell systems can provide communications and hand-off users under high-speed mobile environments.

In this study, we assume that the bottleneck of the system is data rate, not the number of connected users. Therefore, we do not limit the number of simultaneously connected users in both cellular and WLAN systems. This is a reasonable assumption with packet-based data-oriented wireless systems. For example, in CDMA2000 1xEV-DO[2], 59 users can simultaneously connect to a sector and share the bandwidth in TDM manner. Because the limitation of 59 connections is sufficently large, our evaluations are based on the parameters that this limitation do not practically affect the performance.

Users arrive according to a Poisson process with rate λ , download data once and then depart from the system. The request size follows exponential distribution with mean G. The request state space $\mathcal{R} = \{1\}$. The request state 1 shows that the user is downloading data. The request parameters are determined as follows

$$\boldsymbol{a}^{\mathrm{R}} = (1),\ \boldsymbol{P}^{\mathrm{R}} = (0),\ \boldsymbol{\nu}^{\mathrm{R}} = (G^{-1}). \tag{2.9}$$

The mobility state space $\mathcal{M} = \{1, \ldots, N+1\}$. The mobility state j corresponds to the area of node j. Let S_j, $j = 1, \ldots, N+1$, be the surface area of the area j and $S = \sum_{j=1}^{N+1} S_j$. Assuming uniform distribution of users, the initial probability of a mobility state is proportional to the corresponding surface area:

$$\boldsymbol{a}^{\mathrm{M}} = \left(\frac{S_1}{S}, \frac{S_2}{S}, \ldots, \frac{S_{N+1}}{S}\right). \tag{2.10}$$

We determine the dwell time of each area according to the fluid flow model shown in [35, 36]. Let V be the mean velocity of mobiles, L_{jk}, $j \neq k$, be the boundary length between two areas j and k, and $L_j = \sum_{k \neq j} L_{jk}$. Assuming WLAN areas do not overlap or border on each other, a user leaving a WLAN area always moves to the cellular area and vice versa,

$$\boldsymbol{P}^{\mathrm{M}} = \begin{pmatrix} 0 & \frac{L_{1,2}}{L_1} & \cdots & \frac{L_{1,N+1}}{L_1} \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 0 \end{pmatrix}, \tag{2.11}$$

$$\boldsymbol{\nu}^{\mathrm{M}} = \left(\frac{VL_1}{\pi S_1}, \frac{VL_2}{\pi S_2}, \ldots, \frac{VL_{N+1}}{\pi S_{N+1}}\right). \tag{2.12}$$

In this mobility model, each area is represented by its surface area and the boundary length. The matrix $\boldsymbol{P}^{\mathrm{M}}$ indicates that transition probability is proportional to the length of the boundary between the source and destination areas. The vector $\boldsymbol{\nu}^{\mathrm{M}}$ indicates that average duration time is proportional to the surface area of the area and inversely proportional to its boundary length. Duration time parameter represents the average duration time because this mobility model is based on Markov chain. In this numerical evaluation, we assume all areas have circular shape. [1]

Both types of systems are modeled as server-sharing queues with the unit service rate, thus $\Phi_j(n_j) = 1, \delta_j(\ell, n_j) = 1/n_j$ for $j = 1, 2$. All cells are circular and the radius of cell j is R_j. Because there is no overlapping

[1] Generally, the shape is not necessarily circular. In the above mobility model, the shape of area is partially represented by boundary length and surface area. For example, consider two areas with the same surface area: one has a smooth circular shape and the other has a jagged shape. The latter has longer boundary length. Therefore, the jagged area has larger ingress transition probability and smaller duration time. This means that a mobile travelling around the edge of the jagged area frequently steps in and out the the area.

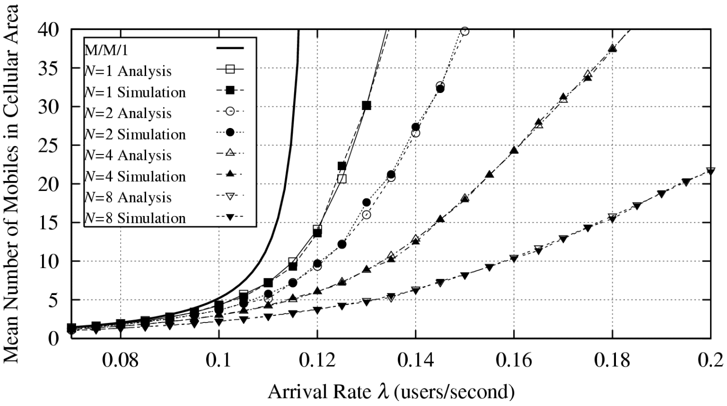


Figure 2.2: Mean number of active cellular users. The number of WLAN APs N is varied. The mobile velocity $V = 20$ km/h.

between the WLAN cell, $S_j = \pi {R_j}^2$, $L_j = L_{j0} = 2\pi R_j$ for $j = 2, \ldots, N+1$, and $S_1 = \pi {R_1}^2 - \sum_{j=2}^{N+1} S_j$, $L_1 = \sum_{j=2}^{N+1} L_j$.

We use parameter set $R_1 = 1$ km, $h_1(u) = 1$ Mbit/s and $R_j = 100$ m, $h_j(u) = 11$ Mbit/s for $j = 2, \ldots, N+1$. The mean request size $G = 1$ MB. The above queueing network model is evaluated with event-level simulations and the proposed analysis method. The numerical results are shown in Fig. 2.2 for mobile velocity $V = 20$ km/h and $N = 1$ to 8. This result shows the number of active cellular users as a function of user arrival rate. Active cellular user is a user who is actively occupying bandwidth of the cellular base station. In our request model Eq. (2.9), users are always using system bandwidth. Therefore, the number of active users is equal to the number of users in cellular area. It is one the essential evaluation factors of the system performance because the cellular system is the bottleneck of the interworking system. The M/M/1 model corresponding to the model without WLAN is also plotted for comparison. Although the multiple WLAN APs reduce load of the cellular system, the amount of improvement per access point is monotonically decreased. The analysis presents sufficiently accurate estimates compared to the simulation results.

Figure 2.3 shows mean response time. Mean response time is the time required to complete download request. This is one of the significant performance measures from user's perspective. Naturally, more WLAN APs contribute to improvement of response time. As shown in Fig. 2.2 and 2.3, mean response time is strongly correlated to the number of active cellular users. Figure 2.4 shows fractions of traffic carried by WLAN by simulations. In this figure, traffic means the amount of data transferred from the base stations to the mobiles. If the arrival rate is low, the fractions approximate the surface area ratio of WLAN system. As the arrival rate increases, the fractions also increase because users tend to be handed-off to the WLAN area before completing download due to congestion in the cellular area.

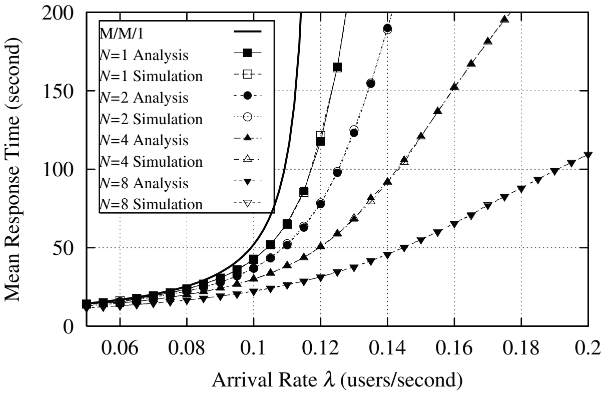


Figure 2.3: Mean response time. The number of WLAN APs N is varied. The mobile velocity $V = 20$ km/h.

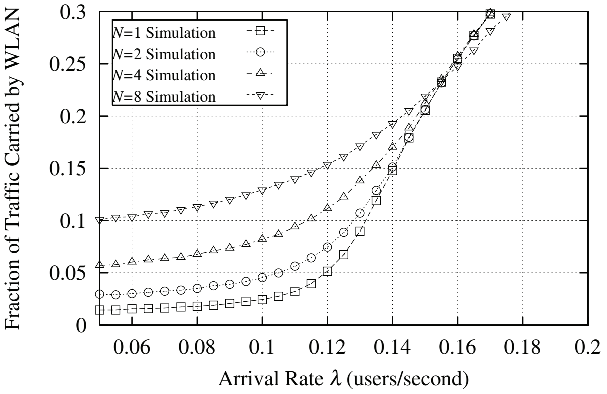


Figure 2.4: Fraction of traffic carried by WLAN APs. The number of WLAN APs N is varied. The mobile velocity $V = 20$ km/h.

2.7.2 Various Degrees of Mobility

In this subsection, we show how the degree of mobility, i.e. the velocity of mobile, impacts on the performance of the interworking systems.

We evaluate the same system as the previous subsection except for the number of WLAN APs and the mobile velocity. The number of WLAN APs N is fixed to 1 and the velocity of mobile V is varied from 3 km/h to 100km/h. This corresponds to the typical range of mobile velocity from pedestrian to high-speed vehicle. Figure 2.5 shows the result for various degrees of mobile velocity V. The point of this result is that the number of active cellular users depends on the mobile velocity. The higher mobility results in the decrease in the number of active users in the cellular area. Figure 2.6 shows mean response time. Mean response time also depends on the mobile velocity. Figure 2.7 shows the fraction of traffic carried by WLAN system. The traffic

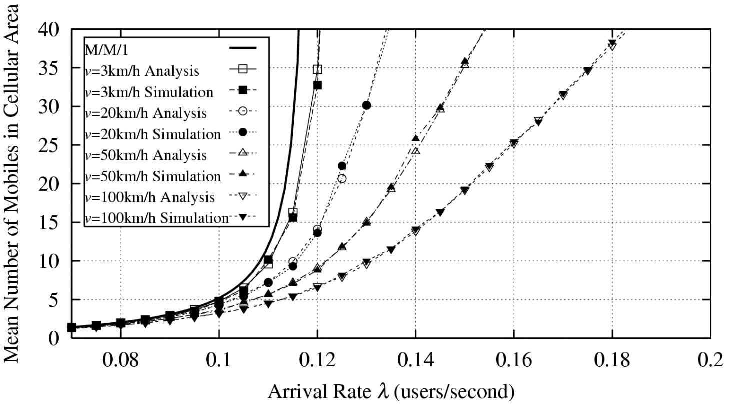


Figure 2.5: Mean number of active cellular users. The mobile velocity V is varied. The number of WLAN APs $N = 1$.

distribution between cellular and WLAN systems obviously depends on the mobile velocity. The amount of traffic accommodated by cellular system is reduced if the mobility is increased as described in Sect. 2.2. In seamless interworking environments, mobile transmits or receives larger amount of data when it happens to reside in the area of higher bandwidth systems.

In Fig. 2.5, the sensitivity to the mobility variation is well observed from the proposed analysis. A comparison between Fig. 2.2 and Fig. 2.5 reveals that mobility has an impact comparable to the number of WLAN APs. This implies that the mobility parameters are essential for the system design of mobile networks with seamless interworking. Although the volume of traffic offered to a system is the most fundamental factor for capacity planning of networks, even this parameter cannot be simply determined without taking into account the impact of mobility. Results of the numerical studies indicate that increases in user mobility and the number of WLAN APs are comparable factors from a network capacity perspective. The analysis method provides objective and quantitative insights for this problem.

2.7.3 Granularity of Mobility Model

The mobility model [35] we adopt is first order approximation. Although this contributes to the tractability of the analysis, higher order statistics of the mobility pattern may affect these results. To verify this, we study the case with a more fine-grained mobility model. The whole area is divided into square grids as shown in Fig. 2.8. The size of grids is determined to equalize the area of a grid to that of a circular WLAN cell in our coarse mobility model. By comparison between the analysis results of the coarse mobility model and the more fine-grained grid simulation results, we can verify the accuracy of our coarse mobility model. As mentioned in Sect. 2.7.1, we assume that the overhead of hand-off is not critical factor of system performance in future

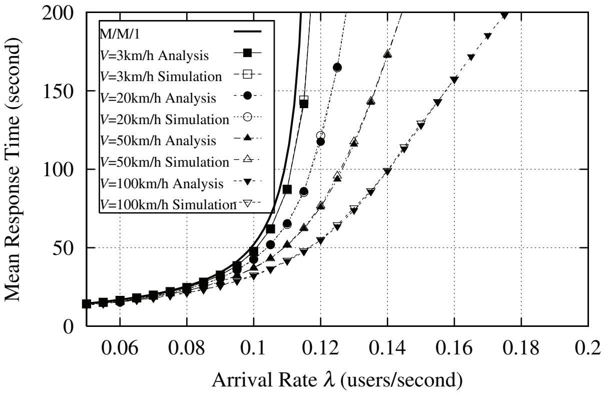


Figure 2.6: Mean response time. The mobile velocity V is varied. The number of WLAN APs $N = 1$.

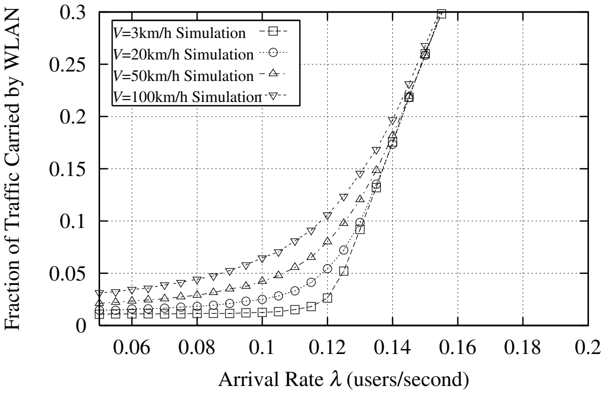


Figure 2.7: Fraction of traffic carried by WLAN APs. The mobile velocity V is varied. The number of WLAN APs $N = 1$.

interworking environment. Therefore, our grid model does not include hand-off regions. If the overhead of hand-off were not negligible, the grid mobility model would not be applicable to evaluate the performance of mobile systems with hand-off.

Users move across grids according to a Markov mobility model. The mobility state space is a set of grid and direction pairs. Direction is chosen from north, west, south and east. Users move straight and turn in a randomly chosen direction and so forth. The number of grids in which users move straight before changing direction follows geometric distribution and its mean K is a parameter. WLAN APs are uniformly distributed without overlapping. The results of multiple simulation runs are averaged. The result of this model is compared with our prior analysis in Fig. 2.9.

Because the hand-off user arrival process in grid model is correlated with user position, the memory of the arrival process is longer than that of first-

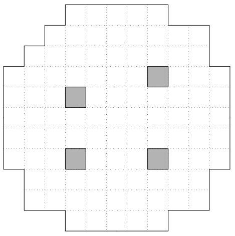


Figure 2.8: Grid Mobility Model

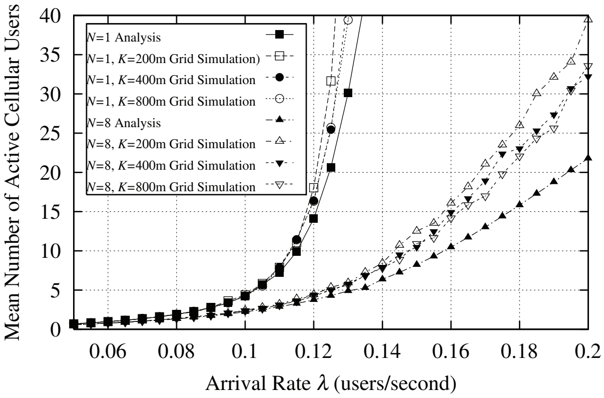


Figure 2.9: Mean number of active cellular users with different granularities of mobility model. The mean distance that mobiles move in a direction K is varied. The number of WLAN APs $N = 1$. The mobile velocity $V = 20$ km/h.

order approximation model, i.e. the arrival process is more bursty. Therefore, the system performance with grid model can be worse than that with first-order approximation model.

Although the grid model results in Fig. 2.9 follow this tendency, the overall characteristics of mobility impact can be observed from our analysis. These results suggest that the granularity of mobility model is not critical for evaluating first order performance measures like the mean number of active users or the average transfer time. This is not true for conventional mobile network systems where the primary performance measure is call blocking probability that is related to higher order statistics of the distribution of the number of users.

2.7.4 Network with Multiple Cells

The previous examples include only a single cellular cell. In this section, we show results of network with multiple cellular cell shown in Fig. 2.10. A hexagonal center cell is surrounded by other six hexagonal cells. The radius

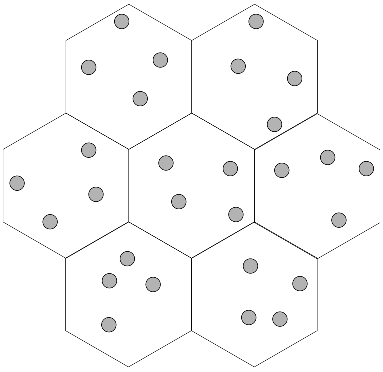


Figure 2.10: Multiple Cellular Cells and WLAN APs

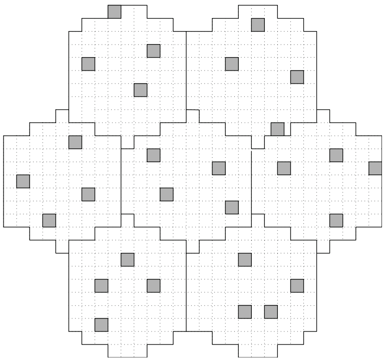


Figure 2.11: Grid Mobility Model for Multiple Cellular Cells

of each hexagonal cell is 1 km. Each cell includes N WLAN APs. We evaluate this network by three methods: analysis of queueing network model with coarse mobility model, simulation of the same queueing network model and simulation of fine-grained grid mobility model shown in Fig. 2.11 with $K = 500$ m.

Figures 2.12, 2.13 include the results for $V = 3$ km and 60 km. Each figure shows the mean number of active cellular users in the central cell for $N = 1$ or 8. The analysis results agree well with the coarse mobility model simulation. These results confirm that the proposed analysis method is applicable to multiple cell networks. Furthermore, these results approximate the characteristics of fine-grained mobility model.

2.8 Analysis of Network with Service Types

2.8.1 Characterization of Queueing Network

Prior to analysis of the queueing network with multiple service types we summarize what characterizes the stochastic dynamics of the queueing net-

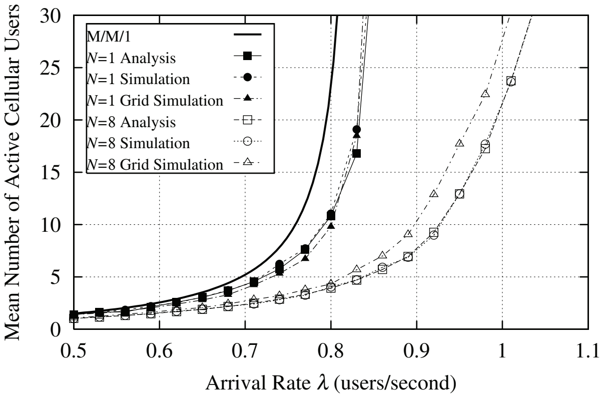


Figure 2.12: Mean number of active cellular users with multiple-cell model. The number of WLAN APs N is varied. Mobile velocity $V = 3$ km/h.

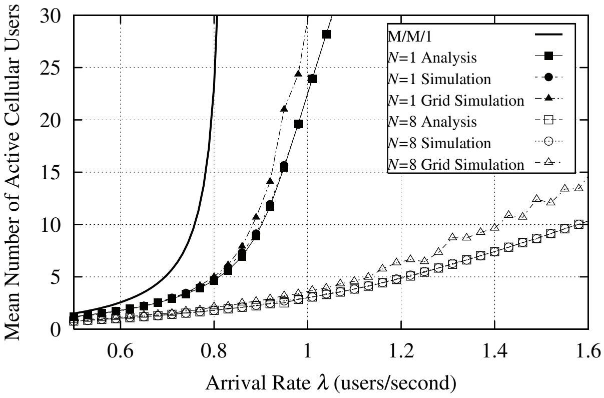


Figure 2.13: Mean number of active cellular users with multiple-cell model. The number of WLAN APs N is varied. Mobile velocity $V = 60$ km/h.

work. Because we assume an exponentially distributed service requirement, the state of a node in the network can be described as a continuous time stochastic process on a discrete state space. Let $\mathcal{S}_j$ be the state space of node j and $x_j, x'_j \in \mathcal{S}_j$ be states of node j. Then a node in the network is characterized by the following information [32]:

- $p^{\mathrm{A}}_{ju}(x_j, x'_j)$ = *arrival effects*: the probability that a class u arrival at node j changes the state from x_j to x'_j.

- $q^{\mathrm{D}}_{ju}(x_j, x'_j)$ = *departure transition rates*: the rate at which class u departures change the state of node j from x_j to x'_j.

- $q^{\mathrm{I}}_{ju}(x_j, x'_j)$ = *internal transition rates*: the rate at which internal transitions change the state of node j from x_j to x'_j.

Because a node never changes its state without arrival or departure in our model, we suppress the internal transition rate $q^{\mathrm{I}}_{ju}(x_j, x'_j)$ in the following discussion.

The state of the network is a continuous time Markov chain. Let $\mathcal{S} = \prod_{j\in\mathcal{J}} \mathcal{S}_j$ be the state space of the network. Then a state of the network is a vector of states of nodes such as $\boldsymbol{x} = (x_j;\ j \in \mathcal{J}) \in \mathcal{S}$. The transition rates of the network are [32]

$$q(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j,k\in\mathcal{J}} q_{jk}(\boldsymbol{x}, \boldsymbol{y}), \quad \boldsymbol{x}, \boldsymbol{y} \in \mathcal{S} \tag{2.13}$$

where $q_{jk}(\boldsymbol{x}, \boldsymbol{y})$ are the transition rates by movements of the customer from node j to k and can be written as

$$\begin{aligned} &q_{jk}(\boldsymbol{x}, \boldsymbol{y}) \\ &= \begin{cases} \displaystyle\sum_{x'_j\in\mathcal{S}_j} \sum_{u,v\in\mathcal{U}_j} q^{\mathrm{D}}_{ju}(x_j, x'_j) r_{ju,jv} p^{\mathrm{A}}_{jv}(x'_j, y_j) \\ \qquad\qquad \cdot 1_{\{y_\ell = x_\ell, \ell\neq j\}}, \qquad j = k, \\ \displaystyle\sum_{u\in\mathcal{U}_j} \sum_{v\in\mathcal{U}_k} q^{\mathrm{D}}_{ju}(x_j, y_j) r_{ju,kv} p^{\mathrm{A}}_{kv}(x_k, y_k) \\ \qquad\qquad \cdot 1_{\{y_\ell = x_\ell, \ell\neq j,k\}}, \qquad j \neq k, \end{cases} \end{aligned} \tag{2.14}$$

where $1_{\{A\}}$ is an indicator function that is 1 if event A occurs and 0 otherwise.

From Eq. (2.13) and (2.14), having the same $p^{\mathrm{A}}_{ju}(x_j, y_j)$ and $q^{\mathrm{D}}_{ju}(x_j, y_j) r_{ju,kv}$ for all $j, k \in \mathcal{J}, u \in \mathcal{U}_j, v \in \mathcal{U}_k$ is a sufficient condition for the two networks to have the same transition rates $q(\boldsymbol{x}, \boldsymbol{y})$ and thus the same steady state probability.

2.8.2 Analysis

Let n_j be the number of customers at node j and $c_j(\ell)$ be the class of customer in position ℓ of node j and $\boldsymbol{c}_j = (c_j(1), \ldots, c_j(n_j))$. In our model, the state of a node can be described as a continuous time stochastic process in a state space described by the form of vector $\boldsymbol{c}_j$ because we assume class-dependent exponentially distributed service requirements and a position-dependent service discipline. The arrival effect of the network is

$$p^{\mathrm{A}}_{ju}(\boldsymbol{c}_j, \boldsymbol{c}'_j) = \sum_{\ell=1}^{n_j+1} \delta_j(\ell, n_j + 1) 1_{\{\boldsymbol{c}'_j = \boldsymbol{c}_j \oplus \mathbf{e}_\ell(u)\}} \tag{2.15}$$

where the notation $\boldsymbol{c}_j \oplus \mathbf{e}_\ell(u)$ represents the state after a class u arrival in position ℓ.

We can extend the discussion in Sect. 2.8.1 to the network with multiple service types while we use the notation $\mathcal{U}_j, \mathcal{D}, q^{\mathrm{D}}_{jud}, r_{jud,kv}$. In the network with multiple service types, the rate whereby a class u customer in position ℓ of node j with n_j customers departs by type d service completion is

$$\phi_{jd}(n_j)\delta_j(\ell, n_j)\mu_{jud}. \tag{2.16}$$

The departure transition rates of the node are therefore

$$q^{\mathrm{D}}_{jud}(\boldsymbol{c}_j, \boldsymbol{c}'_j) = \sum_{\ell=1}^{n_j} \phi_{jd}(n_j)\delta_j(\ell, n_j)\mu_{jud} \cdot 1_{\{\boldsymbol{c}'_j = \boldsymbol{c}_j \ominus \mathbf{e}_\ell,\ c_j(\ell)=u\}} \tag{2.17}$$

where the notation $\boldsymbol{c}_j \ominus \mathbf{e}_\ell$ represents the state after the customer in position ℓ leaves the node.

Because it is intractable to directly analyze the network with type-dependent service requirement μ_{jud}, we consider an *equivalent transition rate network* with the type-independent service requirement μ_{ju}. The equivalent transition rate network is a network whose transition rates defined as Eq. (2.13) are equal to that of the original network. The state space $\mathcal{S}$ is common to the two networks. Because the service type is a property related to departure rates of the nodes, the arrival effects p^{A}_{ju} shown in Eq. (2.15) are also common to the two networks. Let $r'_{jud,kv}$ be the routing probability of the equivalent network, $\phi'_{jd}(n_j)$ and $q'^{\mathrm{D}}_{jud}(\boldsymbol{c}_j, \boldsymbol{c}'_j)$ be the service rate and the departure transition rate of node j in the equivalent network and

$$q'^{\mathrm{D}}_{jud}(\boldsymbol{c}_j, \boldsymbol{c}'_j) = \sum_{\ell=1}^{n_j} \phi'_{jd}(n_j)\delta_j(\ell, n_j)\mu_{ju} \cdot 1_{\{\boldsymbol{c}'_j = \boldsymbol{c}_j \ominus \mathbf{e}_\ell,\ c_j(\ell)=u\}}. \tag{2.18}$$

From the discussion in the previous section, the following equation gives a sufficient condition for the two networks to have the same transition rates and thus the same steady state distribution:

$$q^{\mathrm{D}}_{jud}(x_j, y_j) r_{jud,kv} = q'^{\mathrm{D}}_{jud}(x_j, y_j) r'_{jud,kv}. \tag{2.19}$$

From Eq. (2.17),(2.18) and (2.19)

$$\phi_{jd}(n_j)\mu_{jud} r_{jud,kv} = \phi'_{jd}(n_j)\mu_{ju} r'_{jud,kv}. \tag{2.20}$$

Let $w_{jd} = \phi'_{jd}(n_j)/\phi_{jd}(n_j)$ and Eq. (2.20) can be written as

$$\mu_{jud} r_{jud,kv} = w_{jd}\mu_{ju} r'_{jud,kv}. \tag{2.21}$$

The equivalent network has the parameters w_{jd}, μ_{ju} and $r'_{jud,kv}$ that satisfy Eq. (2.21). Summing up both sides of Eq. (2.21) for kv yields

$$\mu_{jud} = w_{jd}\mu_{ju}. \tag{2.22}$$

Given j, the left side of Eq. (2.22) has $|\mathcal{U}_j| \times |\mathcal{D}|$ parameters and the right side has $|\mathcal{U}_j|+|\mathcal{D}|$ parameters. Therefore, an exactly equivalent network does not generally exist. Because feedback loop, $r_{jud,ju}$, has less impact on the state of the network, an approximation to satisfy Eq. (2.21) for all $kv \neq ju$ can be applied and

$$\begin{aligned}\sum_{kv\neq ju} \mu_{jud} r_{jud,kv} &= \sum_{kv\neq ju} w_{jd}\mu_{ju} r'_{jud,kv},\\ \mu_{jud}(1-r_{jud,ju}) &= w_{jd}\mu_{ju}(1-r'_{jud,ju}).\end{aligned}\tag{2.23}$$

Because $r'_{jud,ju}$ is probability,

$$\mu_{jud}(1-r_{jud,ju}) \leq w_{jd}\mu_{ju}.\tag{2.24}$$

The inaccuracy factor of the approximation

$$\mu_{jud} r_{jud,ju} - w_{jd}\mu_{ju} r'_{jud,ju} = \mu_{jud} - w_{jd}\mu_{ju}\tag{2.25}$$

should be minimized. Consequently w_{jd} and μ_{ju} are determined by solving the following optimization problem for each node j:

$$\begin{aligned}\text{Find} \quad & w_{jd}, \mu_{ju},\\ \text{to minimize} \quad & \sum_{u\in\mathcal{U}_j, d\in\mathcal{D}} \left|\frac{\mu_{jud} - w_{jd}\mu_{ju}}{\mu_{jud}}\right|,\\ \text{subject to} \quad & \mu_{jud}(1-r_{jud,ju}) \leq w_{jd}\mu_{ju},\\ & w_{jd} \geq 0, \mu_{ju} \geq 0.\end{aligned}\tag{2.26}$$

The routing probability of the equivalent network is determined as follows:

$$r'_{jud,ju} = 1 - \frac{\mu_{jud}}{w_{jd}\mu_{ju}}(1-r_{jud,ju}),\tag{2.27}$$

$$r'_{jud,kv} = \frac{\mu_{jud}}{w_{jd}\mu_{ju}} r_{jud,kv}, \quad \text{for } kv \neq ju.\tag{2.28}$$

We assume that the steady state probability of a node in the network with multiple service types can be approximated by the steady state probability of a node in isolation under Poisson arrival with the same rate. Let α_{ju} be the overall average arrival rate of class u customer at node j of the equivalent network and $\phi_j(n_j) = \sum_{d\in\mathcal{D}} \phi'_{jd}(n_j)$. The steady state probability about $\boldsymbol{c}_j$ is

$$\pi_j(\boldsymbol{c}_j) = b_j^{-1} \prod_{\ell=1}^{n_j} \frac{\rho_{jc_j(\ell)}}{\phi_j(\ell)}.\tag{2.29}$$

where $\rho_j = \sum_{u\in\mathcal{U}_j} \rho_{ju}$, $\rho_{ju} = \frac{\alpha_{ju}}{\mu_{ju}}$ and

$$b_j = \sum_{n=0}^{\infty} \frac{{\rho_j}^n}{\prod_{\ell=1}^{n} \phi_j(\ell)}.\tag{2.30}$$

The average departure rate of the equivalent network β_{jud} is

$$\begin{aligned}
\beta_{jud} &= \sum_{\boldsymbol{c}_j \in \mathcal{S}_j} \pi_j(\boldsymbol{c}_j) \sum_{\boldsymbol{c}'_j \in \mathcal{S}_j} q'^{\mathrm{D}}_{jud}(\boldsymbol{c}_j, \boldsymbol{c}'_j) \\
&\approx \sum_{\boldsymbol{c}_j \in \mathcal{S}_j} \left({b_j}^{-1} \prod_{\ell=1}^{n_j} \frac{\rho_{jc_j(\ell)}}{\phi_j(\ell)} \right) \\
&\qquad\qquad \left(\sum_{\ell=1}^{n_j} \phi'_{jd}(n_j) \delta_j(\ell, n_j) \mu_{ju} 1_{\{c_j(\ell)=u\}} \right) \\
&= {b_j}^{-1} \alpha_{ju} \sum_{n_j=1}^{\infty} \frac{\phi'_{jd}(n_j) {\rho_j}^{n_j-1}}{\prod_{\ell=1}^{n_j} \phi_j(\ell)}. \qquad (2.31)
\end{aligned}$$

2.9 Conclusion

It was discussed that user's mobility impacts on the traffic distribution in the systems beyond IMT-2000. Under these environments, the time scales of bandwidth sharing and mobility cannot be simply separated. Numerical results for cellular-WLAN overlay environments were examined to demonstrate that the mobility of users has a significant impact on the traffic distribution between the different systems and its impact is possibly comparable to the number of WLAN APs. A framework for the performance evaluation of such systems was proposed. A queueing network model with nonlinear traffic equations was applied taking into account the independence among nodes in the network. The applicability of the proposed analysis method was verified through numerical results. The proposed model and analysis provide insights for those problems involved in frequency allocation, capacity planning and deployment of future seamless system-interworking environments.

To realize spectrally efficient networks, efficient operation and deployment are essential as well as physical layer efficiency of the individual system. The convergence of different systems with diverse characteristics in systems beyond IMT-2000 makes the deliberate deployment scenario even more important. This includes optimal deployment of WLAN access points and optimization of operational parameters. Furthermore, systems in the next decade might be developed based on interworking environments from the beginning of their design. Such systems cannot operate if deployment and operation are not aware of interworking.

We conclude by acknowledging that for many years the QoS of cellular network has been captive to the spectral efficiency of physical layer technologies and the assigned frequency bandwidth. However, as the paper demonstrates, user mobility is a key factor in defining capacities of systems beyond IMT-2000 as well as WLAN provisioning. It allows mobile operators to meet rising user expectations for future services effectively and efficiently.

Chapter 3

Analysis of TCP Flags in Congested Network

3.1 Introduction

There have been many tools and utilities for network monitoring. However, they give little information about the real performance of data transfer.

For example, Fig. 3.1 illustrates utilization of the communication link which connects Waseda University and IMnet (Inter-Ministry Research Information Network[37]). The vertical axis shows average utilization of the link every five minutes. The bandwidth of this link is 1.5 Mbps at the time of this measurement (it is 100 Mbps at present.) The link is utilized 100% in the daytime in the direction from the IMnet to Waseda University. The graph is simple. However, there is no further information.

According to our earlier survey, HTTP (Hyper Text Transfer Protocol) traffic occupies more than 65% of the traffic from IMnet to Waseda University. It is worth while investigating the HTTP traffic on this link.

Fig. 3.2 shows the average throughput of HTTP on the same day. The throughput is calculated from the log file of HTTP proxy servers at Waseda University. Throughput is indicated as time (in seconds) required to transfer a file of 7KB. The size of 7KB is selected because it is the average size of files transfered by HTTP through the same proxy servers. The average size

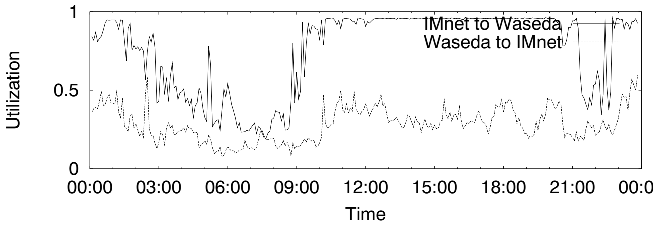


Figure 3.1: Traffic between Waseda University and IMnet on a day

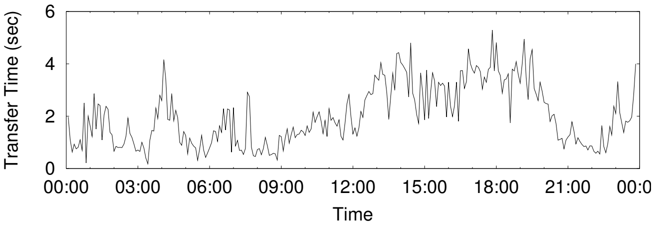


Figure 3.2: Average transfer time of HTTP on a day

is calculated from the same log file at proxy servers. It is easily observed that the performance of HTTP is fluctuating while the utilization of the link is constantly 100%. We would like to measure the degree of congestion which reflects the real performance of HTTP without analyzing the huge log files. This paper proposes a new method of performance measurement which collects a piece of information from the data packets, and estimates the performance by a simple calculation.

This study was performed through Special Coordination Funds for promoting science and technology of the Science and Technology Agency of the Japanese Government.

3.2 TCP Control Flags

TCP/IP is the protocol on the Internet[38][39]. TCP (Transmission Control Protocol) provides reliable connections over IP (Internet Protocol). IP is not a reliable protocol. IP packets may be discarded. Moreover, the order of the IP packets may be changed from the original order. On such unreliable IP protocol, TCP ensures reliable communications by performing the following functions.

Connection management: TCP is a connection-oriented protocol. It establishes a virtual circuit prior to a communication. A virtual circuit provides a dedicated communication link to be used by application programs. An application program only has to send or receive data through a virtual circuit. TCP manages the virtual circuit.

Acknowledgment: In TCP communications, the receiver sends an acknowledgment packet to the sender if a data packet is delivered to the receiver safely. When no acknowledgment packet is returned to the sender in a certain amount of time, the sender assumes that the packet is lost, and retransmits the packet. By the retransmission method, TCP ensures reliable communications over IP protocol.

Sequence number: IP may deliver packets in a different order than the order of transmission. There is also a chance of having duplicated packets because the loss of an acknowledgment packet invokes retransmission of the same data. To cope with these problems, TCP puts numbers to each TCP packet. These are called the sequence number. The receiver of TCP packets can reconstruct the original data properly when duplication of packets or disordered packets occurs.

TCP controls transmission of packets to ensure reliability. TCP has control flags in the TCP header. The control flags occupy a 6-bit field in a TCP packet header. Each bit of control flag has the following meaning.

FIN (Fin Flag): The sender transmits a FIN flag when it has no more data to transmit. Both ends of a virtual circuit normally exchange a pair of packets with FIN flags at the end of a connection.

SYN (Synchronize Flag): SYN flags are used to synchronize the sequence number. Both ends normally exchange a pair of packets with SYN flags at the establishment of a new connection.

RST (Reset Flag): One end sends a packet with a RST flag when it wants to abort the connection.

PSH (Push Flag): When the sender requests the receiver to deliver the data to the application program immediately, it puts a PSH flag. The sender normally sets this flag on when the transmission of a packet empties the buffer area of the sender.

ACK (Acknowledgment Flag): ACK means acknowledgment. There is a related field in a TCP header which contains the acknowledged sequence number. Normally all packets except for the first packet in a connection have ACK flags.

URG (Urgent Flag): This flag means the packet contains some urgent data. This flag is rarely used.

TCP has an adaptive capability. TCP controls transmission of packets in accordance with the condition of a network. For instance, TCP would restrict the number of packets to be transmitted if it encounters network congestion. TCP control flags are used to manage the transmission of packets.

This paper develops a new method of estimating network congestion by measuring TCP behavior. We use TCP control flags. The result shows that the some ratios of the TCP flags are closely related to network congestion.

3.3 Measurement

3.3.1 Network

Fig. 3.3 shows the topology of the network under investigation. Waseda University is connected to the Internet through IMnet[37]. The bandwidth

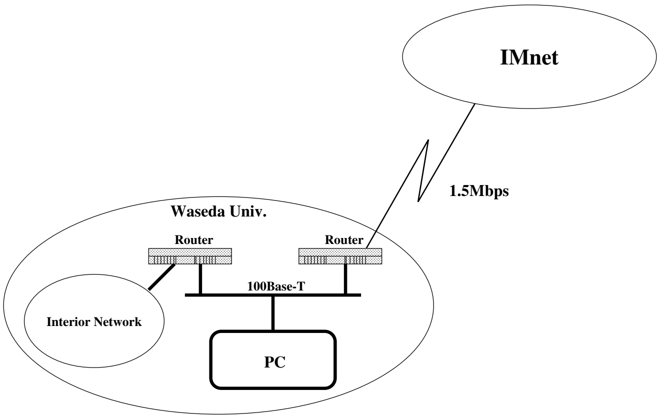


Figure 3.3: Topology of network

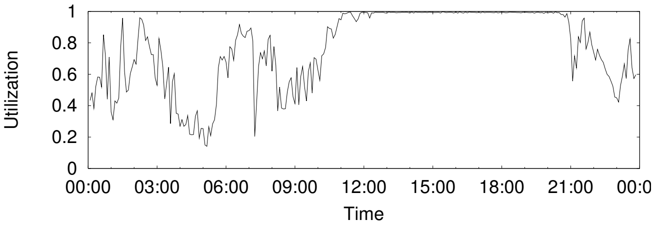


Figure 3.4: Utilization Parameter on December 16

of the link is 1.5 Mbps. We attach a computer to the Ethernet segment to which the border router is connected. The computer collects packets on the Ethernet segment. Thus, we can investigate the traffic between Waseda University and the IMnet.

We investigate HTTP traffic, i.e. Web applications. The computer collects all packets whose source port number or destination port number is 80. We select HTTP because it occupies the largest portion of the traffic.

The computer is equipped with PentiumII 300 MHz CPU and 128MB of memory. The operating system is BSD/OS. We wrote a program for collecting data. It is build with Packet Capture Library. This program counts the number of TCP flags, and the results are stored in disk files every five minutes.

The results shown in this paper were collected from December 12 to December 23, 1997. This period includes the beginning of the winter vacation at Waseda university. Thus, a wide variety of traffic patterns can be observed.

3.3.2 Ratios and the Correlation

We calculate the correlation coefficients between the two parameters and various flag ratios which will be described later.

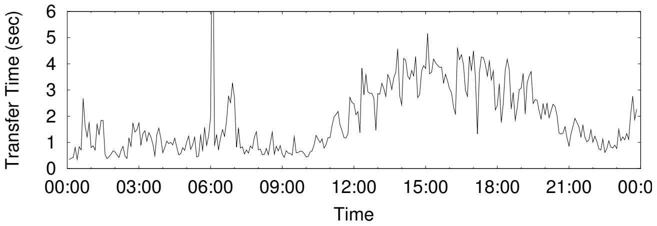


Figure 3.5: HTTP transfer time Parameter on December 16

Table 3.1: Correlation with Utilization

a	b	Correlation Coefficient bet. a/b and Utilization
Syn	Fin	-0.7298
Fin	Syn	0.7196
FRA	Fin	0.6610
Psh	Fin	-0.6464
FRA	ALL	0.6237
FPA	ALL	0.6178
Fin	ALL	0.6149
FA	ALL	0.5923
Psh	Syn	-0.5614
FA	Fin	-0.5151

Utilization Parameter: Utilization of the link in the direction from IMnet to Waseda University. The utilization parameter is independently got from the router using SNMP (Simple Network Management Protocol).

HTTP transfer time Parameter: Performance of HTTP data transfer over the link. It is calculated from log files of proxy servers. The value is expressed in the time (in seconds) which is required to transfer a 7KB file by HTTP.

Both parameters and the measured flag counts are time series data. The unit of time is five minutes. This paper uses the data collected on December 16 in order to investigate the correlation coefficient. Fig. 3.4 and Fig. 3.5 show two parameters: utilization and HTTP transfer time, respectively. In the following sections, data on December 16 are used as a main sample.

Table 3.1 and Table 3.2 show flag ratios with high correlation coefficients. The names of flags used in the tables are the following.

ALL: number of all packets.

Flag name (e.g. Fin): number of packets which have the specific flag.

Table 3.2: Correlation with HTTP transfer time

a	b	Correlation Coefficient bet. a/b and Transfer Time
FRA	Fin	0.7440
Fin	Syn	0.7362
FRA	ALL	0.7264
Syn	Fin	−0.7092
Fin	ALL	0.6543
FPA	ALL	0.6481
FA	ALL	0.6238
Psh	Fin	−0.5760
S	ALL	0.5330
FA	Fin	−0.5294

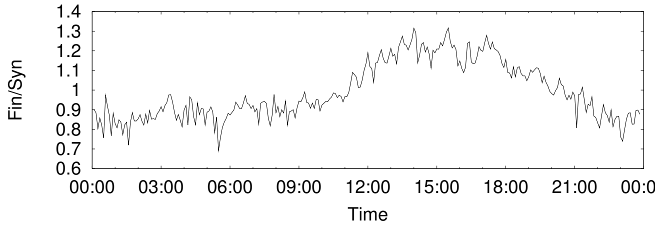


Figure 3.6: ratio of all FIN flags to all SYN flags

One or more first letters of flag name (e.g. FRA): number of packets which have all the flags represented by their first letters and have no other flags.

Three flag ratios are selected because they have a high correlation coefficient with both Utilization and HTTP transfer time. Graphs of these three flag ratios are shown in Fig. 3.6, 3.7 and 3.8.

Fin/Syn: ratio of all FIN flags to all SYN flags.

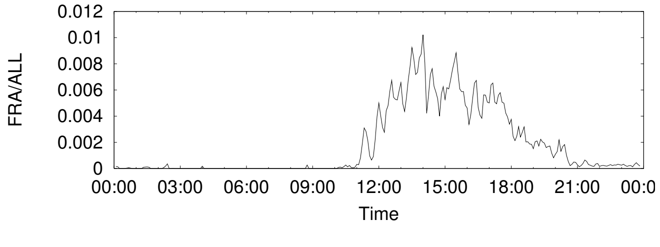


Figure 3.7: ratio of FIN-RST-ACK flags to all packets

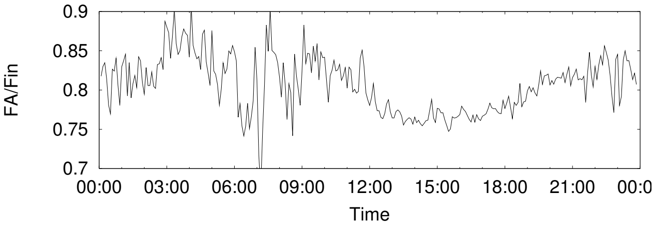


Figure 3.8: ratio of FIN-ACK flags to all FIN flags

FRA/ALL:
ratio of FIN-RST-ACK flags to all packets.

FA/Fin: ratio of FIN-ACK flags to all FIN flags.

3.4 Application

3.4.1 Congestion

All of the three flag ratios are fluctuating prominently when the utilization is nearly 100% and the link is congested. We try to estimate the congestion of the link based on those three flag ratios. In this section, the word "congestion" is used when utilization is over 98%.

We apply *discriminant analysis.* Discriminant analysis is a statistical method of differentiating sample data. The sample data is represented by variables, $x_1, x_2, \ldots, x_p$, which are also called *explanatory variables.* A linear discriminant function with p explanatory variables $x_1, x_2, \ldots, x_p$ is formulated as follows,

$$Z = a_0 + a_1x_1 + a_2x_2 + \cdots + a_px_p$$

If the value of Z is positive, the sample data is classified in group A. If the value of Z is negative, the sample data is classified in group B. There is a known method to determine the coefficients, $a_0, a_1, a_2, \ldots, a_p$, which minimizes the error in discrimination with given samples.

We construct a linear discriminant function based on the data collected on December 16. Three flag ratios mentioned in Sect 3.3.2 are used as explanatory variables. The discriminant function takes the form of (3.1). Group A stands for "congestion", and group B means "not congested".

$$\begin{aligned} DS &= -26.435 + 38.957 \cdot (Fin/Syn) \\ &\quad +462.13 \cdot (FRA/ALL) - 17.821 \cdot (FA/Fin) \end{aligned} \tag{3.1}$$

The communication link is estimated to be congested when the value of DS (*Discriminant Score*) is positive. We applied (3.1) to the data of

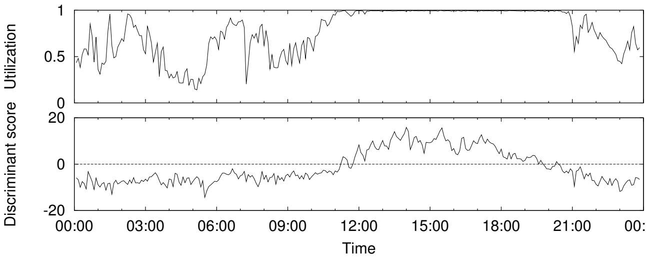


Figure 3.9: Utilization and congestion discriminant score

Table 3.3: Evaluation of congestion discriminant

Date	utilization average	hitting score(%)
12	0.785	90.9
13	0.676	85.3
14	0.402	97.6
15	0.724	98.6
16	0.769	94.4
17	0.684	92.3
18	0.664	87.1
19	0.684	88.5
20	0.475	95.8
21	0.421	98.3
22	0.678	85.3
23	0.398	99.7

December 16. Fig. 3.9 shows the result. The estimation of congestion hits at 94.4% of the points on December 16.

Secondly, we apply (3.1) to estimate congestion on the other days. Table 3.3 shows the summary of the result. This table also includes the utilization average of the day.

The hitting scores are at least 85%. Formula (3.1) is constructed from the data of December 16, the same formula can be applied to the data of the other days. There are relatively low hit scores among days when the utilization average ranges from 0.6 to 0.7. These days require more subtle discriminants than other days. The heavily congested or little congested days have high hit scores because the utilization patterns are simple on these days.

3.4.2 HTTP Performance

We aim to estimate HTTP performance from the three flag ratios. We use the data collected in the daytime when the link was congested. The reason for utilizing daytime data is the following.

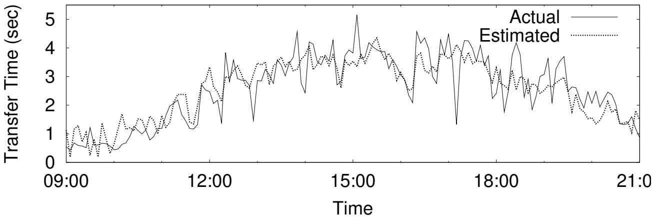


Figure 3.10: Estimation of HTTP transfer time

- At night, there are not many accesses to the proxy server. HTTP transfer time calculated from log files of the proxy server shows sporadic values.
- At night, the link is not congested. HTTP transfer time is mainly determined by the other part of the Internet. Thus, these data are not adequate to estimate the traffic load of the link under investigation.

We adopt *multiple regression analysis*. Multiple regression analysis is a popular statistical method of investigating the relationship between sample data and a dependent variable. The method can predict the value of a dependent variable from sample data, which are also called *explanatory variables*. A multiple regression equation is formulated as

$$y = a_0 + a_1x_1 + a_2x_2 + \cdots + a_px_p$$

where y is the dependent variable and $x_1, x_2, \ldots, x_p$ are the explanatory variables. There is a standard procedure called *least square method* to determine the coefficients, $a_0, a_1, a_2, \ldots, a_p$. The least square method minimizes the error between the predicted y and the actual value.

We apply the multiple regression analysis using the least square method to the data of December 16, then construct a multiple regression equation which calculates ETT ($Estimated\ Transfer\ Time$).

$$\begin{aligned} ETT = {} & 3.4789 + 10.849 \cdot (Fin/Syn) \\ & -225.67 \cdot (FRA/ALL) - 15.281 \cdot (FA/Fin) \end{aligned} \tag{3.2}$$

We apply (3.2) to the data from which (3.2) was constructed. Fig. 3.10 shows the result. This estimation seems correct, and the correlation coefficient between the actual HTTP performance and the estimated HTTP performance was 0.845.

To investigate the usefulness of this method, we apply (3.2) to the data of the other days. As a result, correlation coefficients are at least 0.7 from data collected in the daytime of weekdays, when the link is highly utilized. This correlation coefficient is lower than the result from the data from which

Table 3.4: Evaluation of HTTP transfer time estimation

Date	Utilization average	correlation coefficient
12	0.785	0.808
13	0.676	0.817
14	0.402	-0.073
15	0.724	0.836
16	0.769	0.845
17	0.684	0.813
18	0.664	0.764
19	0.684	0.709
20	0.475	-0.034
21	0.421	-0.100
22	0.678	0.666
23	0.398	0.119

(3.2) was constructed but still correct to a certain degree. From the data of holidays, however, the correlation coefficient was nearly 0, and on some holidays it gives negative values. The latter example illustrates the limitation of our method.

3.5 Analysis

In (3.1) and (3.2), the ratio of FIN to SYN is the most significant factor to estimate the load of the link. To investigate this phenomenon, we analyze the number of SYN flags and FIN flags in each TCP connection with port number 80 in the same communication link.

We count packets which are set with only SYN flags on as the beginning of a connection. Connections are categorized in two groups depending on the behavior of connection termination.

Terminated Connections: Connections which have explicit termination of the connection. A connection closed gracefully or aborted by a RST flag is categorized in this group.

Timeout Connections: Connections which have no explicit termination. When a connection exchanges no packet for 240 seconds, then the tracing of this connection is stopped and this connection is counted as timeout connection.

We investigate TCP connections during the daytime when the link is congested. We observed 130,244 connections in one hour. 99% of these connections are terminated connections. The average FIN/SYN ratio during the same period is 1.038. (The number of FIN flags is more than the number of SYN flags.)

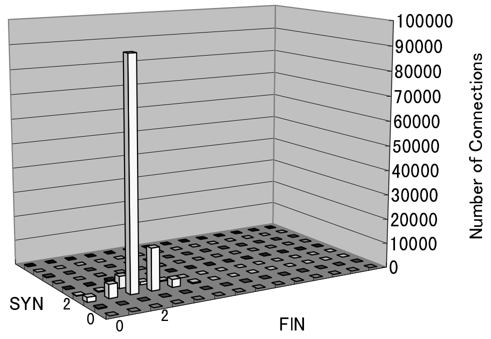


Figure 3.11: Number of Terminated Connections (WASEDA)

3.5.1 Terminated Connections

Fig. 3.11 shows the number of connections according to the SYN count and the FIN count in all terminated connections.

Fig. 3.11 displays that most parts of connections have two SYN packets. Compared to this stable number of SYN, the number of FIN shows more variance.

The most frequent combination is two SYN packets and two FIN packets. This is the most graceful behavior. There are other combinations besides the two SYN – two FIN combination. However, the numbers are small relative to the former combination. The second most frequent combination is two SYN packets and three FIN packets. This combination is the major factor of the unbalance in the number of FIN flags and SYN flags.

3.5.2 Timeout Connections

Fig. 3.12 shows the number for all timeout connections.

The most common number of SYN flags is again two. This is the same as for terminated connections. However, the number of SYN flags shows more variance in timeout connections than in terminated connections.

With respect to the number of FIN flags, timeout connections show quite different patterns from terminated connections. In the timeout connections, the number of FIN flags shows much more variance than in the terminated connections. The combinations in the timeout connections are divided into two clusters. In Fig. 3.12, one cluster covers from zero to three FIN flags. The most frequent combination in this cluster is two SYN flags and one FIN flag. This combination is also the most frequent one in the whole timeout connections.

The other cluster covers from seven to ten FIN flags. The most frequent combination in this cluster is two SYN flags and nine FIN flags. This combination is the second most frequent one in the whole timeout connections. It

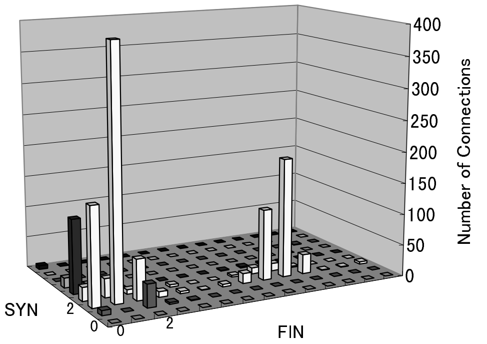


Figure 3.12: Number of Timeout Connections (WASEDA)

is not normal to have nine FIN flags to two SYN flags. The FIN flags are repeatedly re-sent by TCP routine because of absence of the acknowledgment.

3.5.3 Another Network

To verify these results, we investigate packet traces from another network. We use OC3mon/Coral [40] and capture packets between APAN (Asia-Pacific Advanced Network)[41] Tokyo Exchange Point and STAR TAP (U.S.).

We collect packets for 6 days in August 1999. Packet capturing is performed intermittently, i.e. 10 minutes per hour. We traced all TCP packets and observed 1,547,517 connections. Routing asymmetricity in backbone network occasionally makes TCP connections observable only one-way. In fact half of the captured connections constitutes such one-way traffic. Because this traffic complicates connection tracing and analysis, we simply ignored this traffic. Finally we count 831,901 bidirectional terminated connections.

Fig. 3.13 shows the number of SYN and FIN packets in these connections. This figure is similar to Fig. 3.11 except for the existence of one SYN and zero FIN connections. These connections are aborted by RST segment. Our previous terminated connections has a small number of aborted connections. At Waseda University most HTTP traffic is generated by proxy servers. The proxy servers are more graceful in connection termination than human users. Thus the number of aborted connection in Fig. 3.11 is much smaller than Fig. 3.13.

3.6 Conclusion

This paper clarifies the following.

- Correlation was observed between network congestion and some ratios of TCP flags.
- One can tell if the link is congested based on the value of flag ratios.

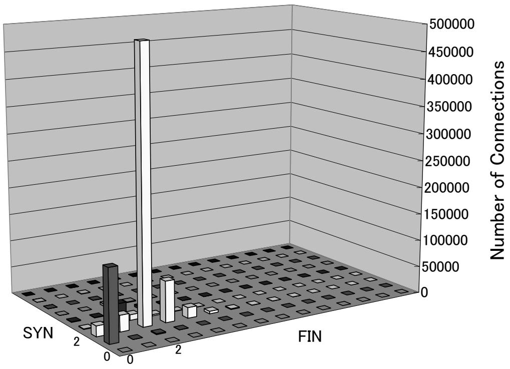


Figure 3.13: Number of Terminated Connections (APAN)

- In a heavily congested network, it was possible to estimate the performance of HTTP from the ratios of flags.
- The ratio of FIN to SYN in HTTP traffic is significantly larger than 1.0 when the network is congested. The most significant factor of this imbalance is the existence of TCP connections with two SYN packets and three FIN packets.

The formulas in this paper are based on the measurement of a particular network. It is possible to cover more networks if we can put a computer to collect TCP packets. We have a plan to cover more networks. The statistical method described in this paper might be applied to other application protocols, if the coefficients in the formulas are determined based on the measured data for the application. We have a plan to investigate other application protocols.

The method proposed in this paper uses a 6-bit field of TCP packets. The amount of data is small enough to be collected through existing network probes, e.g. RMON (Remote Network Monitoring). Moreover, the calculation of ratios is simple.

Up to now, many monitoring tools and utilities have been proposed. For example, a network operator can observe the utilization of a communication link by monitoring routers through Simple Network Management Protocol (SNMP). A good example is the Multi-Router Traffic Grapher (MRTG)[42] which periodically generates graphs of utilization. Measurement of utilization is a simple and effective method. However, utilization figures provide little information about actual performance of data transfer. When the link is utilized 100%, it only tells us the full utilization. One cannot tell whether the link is over-loaded or not. Our method can complement the existing tools, and it gives more information on the network status.

Chapter 4

Minimum Disclosure Routing

4.1 Introduction

As multiple access technologies to the Internet become available to users, such as ADSL, FTTH and 3G/4G wireless, Internet Service Providers (ISPs) are facing a mixture of challenges that may seem harder than ever to fulfill concurrently, such as (1) extending the *footprint* to cover a large user base and multiple access means per user, (2) reducing operational *costs*, (3) improving network *availability*, and (4) maintaining *operational confidentiality.*

The emerging concept of network virtualization (NV) recently proposed in various projects [3, 4, 5] is expected to help ISPs achieve some of the goals, especially (1) to (3), simultaneously. For example, virtual collocation [6, 7] separates *Infrastructure Providers* (*InPs*) that provide multiple isolated slices of physical resources and *Service Providers* (*SPs*) that utilize slices to operate virtual networks, in order for the SPs to *cost-effectively* extend their network *footprints* on top of multiple InPs without investing in physical infrastructure and to improve network *availability* by splicing multiple paths [8].

However, the last bullet (4) mentioned above, *operational confidentiality*, may remain unresolved despite such a new concept of NV to the rescue for achieving the diverse mixture of the goals. We observe that an ISP often strives to keep its competitors away from the operational practices developed to survive business tussles. It is common for followers of market leaders in various businesses to analyze their strategies/operations and to employ them to catch up [9]. Free-riding on other ISPs' operational expertise turns out to be quite effective, since ISPs often cultivate the market in similar geographical regions [10, 11]. For instance, in the German wireless telecommunication market, followers have adopted such a "herding strategy" to bring severe price-cutting competition, and profits have plunged 50% in five years [12]. In the light of these observations, one must note that NV, in fact, may have a negative impact on the *confidentiality of SPs' operational information*, since virtual collocation [6, 7] implies that an InP runs its own SP service on its top as well as on the other (competing) InPs that have access to the operational details of the SP on top of them. For example, virtual collocation may allow each of two competing InPs, such as AT&T and Verizon, to extend the footprint of their own SP services over the resources of the other, although this endangers the operational confidentiality of both SPs. Therefore, if NV is to be employed to satisfy all the goals (1) to (4) of ISPs mentioned previously, the challenge is to achieve secure network operations of SPs without disclosing too much information to the underlying InPs.

In this paper, we propose a solution for *Minimum Disclosure Routing (MDR)*, the problem for an SP overlaid on top of multiple InPs to minimize the disclosure of its virtual network routing information to the underlying InPs. We posit that MDR can lift the SPs' barrier to entry in employing network virtualization, since the disclosure of confidential routing information prevents SPs from utilizing virtualized network resources from InPs; mainly for the following two reasons. First, SPs are generally sensitive to the disclosure of *any* of their routing information. Second, the disclosure of specific routing information such as path/link cost leads to the disclosure of more sensitive information such as quality, bandwidth, and topology. For example, the quality of services can be inferred from path cost information [21]. Bandwidth is easily estimated from link cost because there is a well-known practice that link cost should be configured to the inverse of link bandwidth [22]. Topology and link costs can be inferred from path cost information even if the link costs of individual links are not directly disclosed by the routing protocol (as in RIP or OSPF inter-area routing) [23]. The above mentioned information gives the competitors of an SP significant advantage, which can be exploited to develop services that are more efficient and less costly than the SP's one. Such cunning competitors can easily defeat the SP in the highly competitive market.

Our contributions in this paper are two-fold. First, this paper formally defines MDR and proposes a detailed design of the solution based on our extension to *Secure Multiparty Computation* (*SMC*) [25], where multiple parties cooperatively compute a function from each party's confidential input. The generic SMC protocol [26] is not simply applicable to MDR, since it requires full-mesh connectivity to share the intermediate computation results among participants, while routing is a problem to establish such logical full-mesh connectivity. Thus, we design a new primitive operation called TRANSFER to securely transfer intermediate results between neighboring virtual routers and allow these intermediate results to be shared among them. Second, we implement the proposed solution in our testbed and evaluate its performance by experiments as well as by the analytical model for comparison. We reveal the proposed solution is supposed to achieve secure routing without degrading the convergence time despite the overhead of the SMC protocol. Accordingly, we conclude that our proposed solution to MDR viably complements the concept of network virtualization to help fulfill all the contradicting goals (1) to (4) of ISPs concurrently.

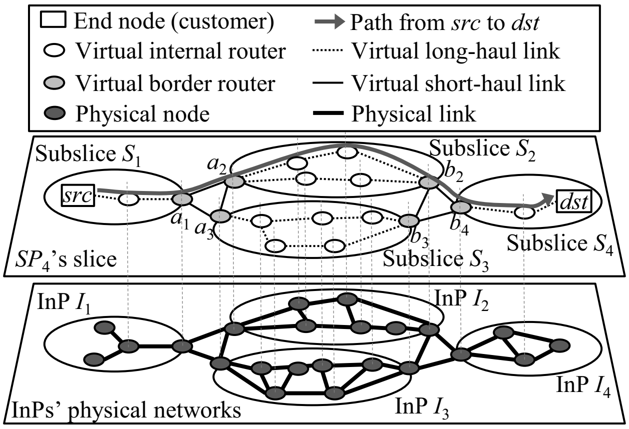


Figure 4.1: A network virtualization example (an SP operating over four InPs).

The rest of the paper is organized as follows. Section 4.2 defines the problem and Section 4.3 overviews the solution. Sections 4.4 and 4.5 elaborate and evaluate the solution. Section 4.6 discusses the important issues and Section 4.7 introduces related work. Finally Section 4.10 briefly concludes.

4.2 Problem

4.2.1 A Walk-through Scenario

We consider the network virtualization scenario shown in Fig. 4.1 where four infrastructure providers (InPs) I_1, I_2, I_3, and I_4 are operating. Suppose in this example that InP I_4 is also operating its own service provider (SP) denoted by SP_4 on top of the four InPs. Namely, SP_4 purchases virtual routers and virtual links from its three business competitors, I_1, I_2, and I_3, builds a *slice* (i.e., a virtual network) running an arbitrary networking protocol, and provides end-to-end services to its customers *src* and *dst*. For the sake of discussion, we introduce the term *subslice* to denote a subgraph of a slice on top of each InP, in other words, a set of virtual routers and virtual links each InP is hosting. For example, in Fig. 4.1, the subslice S_1 is a part of SP_4's slice on top of InP I_1. Virtual routers on the borders of the subslices are interconnected at two peering locations, $\{a_1, a_2, a_3\}$ and $\{b_2, b_3, b_4\}$, as is often the case with the Internet of today [10, 43].

Suppose SP_4 is willing to implement shortest-path routing, that is, to route a packet from *src* in S_1 to *dst* in S_4 through the minimum-cost path. The links within a peering location such as the link $\{a_1, a_2\}$ are supposed to have no cost since they are short-haul links, typically with large bandwidth and low delay. For the sake of simplicity, suppose all the other links have cost 1 since they are long-haul links. Consequently, the cost of the shortest

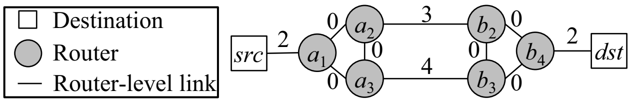


Figure 4.2: The router-level topology of the slice shown in Fig. 4.1. Each number attached to a link is its cost.

path from src to dst (denoted by a thick line traversing S_2 in Fig. 4.1) is seven because it contains seven long-haul links.

From a security perspective, each InP has access to all the information stored on all the virtual routers in the subslice it is hosting. Now, SP_4 faces the problem that its virtual routers must calculate the shortest path while disclosing only the encrypted or fragmented topology information, which cannot be reconstructed into something meaningful to the other InPs, e.g., a_1 must find that packets for dst should exit to a_2 rather than to a_3 without learning that the costs of the paths a_2-dst through S_2 and a_3-dst through S_3 are respectively 5 and 6. If SP_4 could not ensure the confidentiality of its operational information, it would not use the virtualized network resources from its competitors.

4.2.2 Network Model

We consider the following general model of an SP's virtual network. In this paper, we focus on inter-subslice routing since confidentiality issues arise when information is exchanged between subslices. In contrast, intra-subslice routing between virtual internal routers within a subslice does not involve any confidentiality issue. This is because intra-subslice routing requires exchanging messages only within the subslice. (For example, intra-subslice routing within the subslice S_1 requires no message exchange with the neighboring subslices S_2 or S_3, and thus there is no concern that routing information of S_1 is disclosed to InP 2 or 3.) Inter-subslice routing is performed by a set R of virtual border routers (hereafter referred to as "routers") for a set U of destinations (e.g., IP prefixes). As shown in Fig. 4.2, the *router-level topology* is an undirected graph $G_R = (V_R, E_R)$ where the nodes $V_R = R \cup U$ are connected by a set E_R of virtual links (hereafter referred to as "router-level links" or simply "links"). Note that, because we omit virtual internal routers in our model, a link may be an intra-subslice path between two nodes (e.g., the path between a_2 and b_2 in Fig. 4.1 is considered a link in Fig. 4.2.) Each link $e \in E_R$ between two nodes $v, v' \in V_R$ is described by an unordered pair (i.e., a set of two elements) $e = \{v, v'\}$, and has its own link cost $w(e)$. If there is no such link e between v, v', we suppose the cost $w(e) = \infty$. Each short-haul link between two routers in a single peering location (like $\{a_1, a_2\}$) has a cost 0. In Sections 4.2, 4.3 and 4.4, we describe routing to a single destination $u \in U$, and in Section 4.5, we evaluate the impact of the routing table size $|U|$ on the routing performance.

4.2.3 Minimum Disclosure Routing (MDR)

As shown in Section 4.2.1, an SP may encounter an MDR problem, e.g., the router a_1 needs to obtain its next-hop information without disclosing any *confidential routing information* (such as the costs of the paths a_2-dst and a_3-dst) to its underlying InPs. MDR is formulated as a problem for routers R to collectively compute a distributed function f that

- takes *local topology information* as an input from each router $r \in R$, which is the set of all the links directly connected to the router r, denoted by $links_r \subset E_R$,
- exchanges only encrypted or fragmented *routing information* among routers as intermediate computation results, and
- gives *next-hop information* as an output to each router r, which is denoted by the link $nexthop_r \in links_r$. The router r forwards each packet for the destination u towards $nexthop_r$.

From a bird's eye view of the SP, MDR is a problem of computing a function $T_R = f(G_R)$ without disclosing any routing information to underlying InPs, where the input is topology information G_R and the output is a *next-hop tree* $T_R = \{nexthop_r \mid r \in R\}$ rooted at the destination u. (If the SP is willing to implement shortest path routing, T_R should be a shortest path tree.) Consequently, an InP hosting a subslice S_i obtains only the partial topology/next-hop information on the subslice it is hosting, i.e., subgraphs $G_R \cap S_i$ and $T_R \cap S_i$. It cannot obtain topology/next-hop information on the other subslices and any routing information.

4.2.4 Threat Model

Any solution to a security problem needs a clear definition of the threat model. Thus, for MDR, we assume an InP to be an *honest-but-curious* adversary in security jargon, i.e., an adversary that may passively collect information but will not actively attack the system.

To be more precise, we assume that an InP can observe any information (including program code, data and protocol messages) stored on any (virtual) router it is hosting. This is because, technically, an InP has access to the information stored in the registers, memory, and storage on its top [44]. However, we assume that the InP will not maliciously add, modify or remove any such information since such active attacks can be traced by the SP. Besides, we assume that two or more InPs will not collude to expose the SP's confidential routing information.

The threat model defined above implies that none of the existing routing algorithms, such as RIP and OSPF, can achieve MDR because they require the routers to send and receive confidential routing information (e.g., path/link costs and topology) between different subslices and thus disclose it to the underlying InPs. BGP also cannot achieve MDR even if it is applied to inter-subslice routing by assigning a private AS number to each subslice. Although an SP can operate BGP without disclosing the internal topology of a subslice to other subslices, it cannot protect subslice-level topology information. Furthermore, BGP is not necessarily optimal when applied to an SP's intra-domain routing. (We discuss these issues further in Section 4.6.2.) Most importantly, even if an SP naïvely encrypts confidential routing information, underlying InPs still have access to this information. This is because the SP must store the decryption key on top of the InPs' memory and storage in order for the SP's routers to decrypt the encrypted information.

4.3 Overview of the Proposed Solution

Our approach to achieving MDR is to extend the generic SMC by defining a new primitive to transfer secrets in addition to the standard primitives so that our extended SMC may be applied to the distributed routing problem of the MDR.

4.3.1 Secure Multiparty Computation (SMC)

SMC is defined as the cooperative computation of a function F where a set of parties, say parties $1, \ldots, n$, provide inputs $x_1, \ldots, x_n$ and cooperatively calculate outputs $(y_1, \ldots, y_n) = F(x_1, \ldots, x_n)$ of the function, while keeping each party's input x_i and output y_i invisible from other parties. SMC requires that no intermediate computation results be disclosed to any party, since they might contain information that may infer the other parties' inputs and outputs. Although the requirement of SMC appears infeasible, surprisingly, there are several generic SMC protocols for an arbitrary function F *if every pair of parties has a communication channel between them (i.e., if the parties have full-mesh connectivity)* [25, 45, 26].

The generic SMC protocol [26] consists of three primitives. First, each party i invokes a primitive called SHARE to encode its secret input x_i by using the *secret sharing scheme* (SSS) [46] into an n-tuple of *shares* denoted by $[\![x_i]\!] = ([x_i]_1, ..., [x_i]_n)$, where each share $[x_i]_j$ is a piece of encoded information distributed to a party j. No single party can decode the secret unless a certain subset of the other parties discloses its shares to this single party for decoding. Then, they invoke another primitive called COMPUTE to compute the function F from the shares of the inputs $x_1, \ldots, x_n$, while all the intermediate computation results are also shared among them, until they obtain shares of the final outputs $y_1, \ldots, y_n$. Finally, each party i invokes a

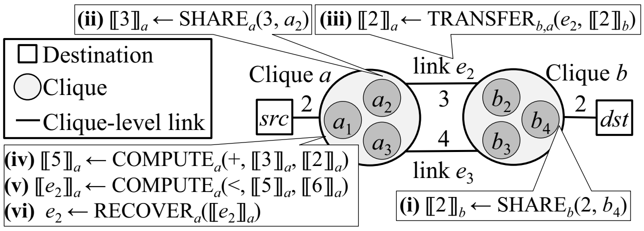

(ii) $[\![3]\!]_a \leftarrow \text{SHARE}_a(3, a_2)$
(iii) $[\![2]\!]_a \leftarrow \text{TRANSFER}_{b,a}(e_2, [\![2]\!]_b)$
Destination
Clique
Clique-level link
Clique a
link e_2
Clique b
src
2
a_1
a_2
a_3
3
4
b_2
b_3
b_4
2
dst
link e_3
(iv) $[\![5]\!]_a \leftarrow \text{COMPUTE}_a(+, [\![3]\!]_a, [\![2]\!]_a)$
(v) $[\![e_2]\!]_a \leftarrow \text{COMPUTE}_a(<, [\![5]\!]_a, [\![6]\!]_a)$
(vi) $e_2 \leftarrow \text{RECOVER}_a([\![e_2]\!]_a)$
(i) $[\![2]\!]_b \leftarrow \text{SHARE}_b(2, b_4)$

Figure 4.3: The clique-level topology of the slice shown in Fig. 4.1. The primitive invocations (i) to (vi) are described in Section 4.3.3.

primitive called RECOVER to collect shares $[\![y_i]\!] = ([y_i]_1, ..., [y_i]_n)$ from the other parties and decode its secret output y_i.

4.3.2 Our Extension to SMC

From an SMC perspective, MDR is also an SMC problem: how SP's routers R can compute a function f that takes local topology information $links_r$ as an input from each router $r \in R$ and gives next-hop information $nexthop_r$ as an output to each router r. However, the generic SMC protocol cannot be applied to MDR due to the chicken-or-egg dilemma—all the three primitives, SHARE, COMPUTE, and RECOVER, require full-mesh connectivity between the routers to share the intermediate results among them, while MDR is a problem of logically establishing such full-mesh connectivity on a partially connected topology G_R.

In order to address this dilemma, exercising the insight that a distributed routing problem can often be addressed by divide-and-conquer approaches, we consider decomposing MDR into sub-problems, each of which can be solved by the COMPUTE primitive among a locally connected subset of routers. In particular, we focus on cliques in the router-level topology; a *clique* is defined as a subset of routers fully connected by short-haul links within a single peering location. For example, there are two cliques in Fig. 4.2, $a = \{a_1, a_2, a_3\}$ and $b = \{b_2, b_3, b_4\}$, which are connected by two long-haul links, $e_2 = \{a_2, b_2\}$ and $e_3 = \{a_3, b_3\}$, and compose the clique-level topology shown in Fig. 4.3.

In a nutshell, we solve MDR by running a distributed routing algorithm in the clique-level topology. In this algorithm, we need to transfer the intermediate computation results (i.e., sub-problems' inputs/outputs generated by SHARE and COMPUTE) between neighboring cliques in the clique-level topology. To this end, we design a new primitive called TRANSFER to securely transfer the secret shared in the clique to its neighboring clique. This primitive plays an essential role in extending SMC to a distributed routing algorithm. In the specific example of Fig. 4.3, it seems that all the six routers in the two cliques can establish full-mesh connectivity if the two routers,

a_2 and b_2, at both ends of the link e_2 forward messages between a_1 and b_4. However, such local forwarding cannot solve the connectivity problem in general topologies where more than two cliques exist. For example, if a subslice hosted by another InP, say subslice S_5, is connected to the subslice S_4, the router a_1 needs to send messages to the routers in S_5 via S_2 and S_4. Such multi-hop connectivity problem cannot be solved by local forwarding between two neighboring cliques, and requires multi-hop routing between cliques, that is, MDR.

By definition of the clique, its routers can invoke the three primitives of the generic SMC protocol, which require full-mesh connectivity. Furthermore, since each router in a clique (i.e., a peering location) is from a different InP, none of the underlying InPs may recover the secret protected by the secret sharing scheme. See Section 4.4 for details of the four primitives.

4.3.3 Walk-through Scenario Revisited

By using Fig. 4.3, we revisit the same example in Section 4.2.1 to sketch our idea to solve MDR with the four primitives (i.e., how the router a_1 obtains the next-hop information for *dst*.) In Fig. 4.3, each invocation of the primitives is denoted with a subscript identifying the clique in which it takes place. For example, SHARE$_b$ denotes an invocation of SHARE in the clique b. Also, $[\![x]\!]_b$ denotes a 3-tuple of shares of a secret x (e.g., a link/path cost), each of which is held by one of the three routers in the clique b.

First, as shown in Fig. 4.3 **(i)**, the router b_4 shares the secret cost 2 of the link to *dst* by invoking SHARE$_b$ among the clique b. This cost should be invisible from the other subslices S_1, S_2, S_3 because it includes the costs of links in the subslice S_4. Also, in Fig. 4.3 **(ii)**, a_2 shares the secret cost 3 of the link e_2 among the clique a.

Subsequently, these secret costs are flooded in the clique-level topology by advertising them between neighboring cliques. For instance, in Fig. 4.3 **(iii)**, the clique b advertises the secret cost 2 originated from b_4 to the neighboring clique a via the link e_2 by invoking TRANSFER$_{b,a}$. During this flooding process, the secret costs of the links along the flooding path are accumulated to obtain the path cost. For instance, in Fig. 4.3 **(iv)**, the clique a adds the secret cost 3 of e_2 to the secret cost 2 advertised via e_2 by invoking COMPUTE$_a$ and obtains the accumulated secret path cost 5 from the clique a to *dst* via e_2. Likewise, omitted in Fig. 4.3, the clique a also obtains the secret path cost 6 from the clique a to *dst* via e_3.

Finally, in Fig. 4.3 **(v)**, the clique a is ready to compare two secret path costs, 5 and 6, advertised via e_2 and e_3 respectively by invoking COMPUTE$_a$, and in Fig. 4.3 **(vi)**, obtain secret next-hop information e_2. By invoking RECOVER$_a$, the routers in the clique a (including a_1) find that e_2 is the link to their next hop for *dst*. Throughout the entire process, none of the secret cost values are disclosed to the underlying InPs because all of these secrets are protected by the secret sharing scheme.

4.4 Details of the Proposed Solution

We describe the proposed solution for MDR in detail.

4.4.1 Topology and Protocol Notation

We consider the following topology model by generalizing the example shown in Fig. 4.2 and 4.3. The routers R on a router-level topology G_R defined in Section 4.2.2 are partitioned into a set C of cliques by peering locations. The router-level links E_R are also divided into two disjoint subsets; the short-haul links E_I between routers within each clique and the long-haul links E_C between routers in different cliques. For any short-haul link $e \in E_I$, its cost $w(e)$ equals zero. The *clique-level topology* is an undirected multigraph $G_C = (V_C, E_C)$, where nodes $V_C = C \cup U$ are connected by the long-haul links E_C. Thus, these links E_C are also referred to as *clique-level links.*

A protocol in this network is denoted by capital letters like "PROTOCOL$_V(X)$" where V is the set of participating routers and X is the collection of inputs from them. Also, each protocol procedure is denoted by small caps like "PROCEDURE$_r(x)$" where $r \in V$ is the router running this procedure and x is the input from this router.

4.4.2 Secret Sharing among InPs

In our MDR protocol, the SP protects every piece of confidential information from n underlying InPs (InP $1, \dots,$ InP n) by using a secret sharing scheme (SSS) [46]. The SSS consists of two algorithms; an encoding algorithm that divides a secret x into n shares denoted by $[\![x]\!] = ([x]_1, \dots, [x]_n)$, and a decoding algorithm that reconstructs the original secret x from any two or more shares out of these n shares.

Note that each InP is assigned to a fixed index in the vector of the shares, namely, InP i can access only the i-th share $[x]_i$ of the secret. Moreover, this index is common in all secrets protected by the SSS (e.g., for two secrets x_1 and x_2, InP i is always given $[x_1]_i$ and $[x_2]_i$.) This is because, when a clique performs the COMPUTE primitive on a function with multiple secret inputs (e.g., $y = F(x_1, x_2)$), all the input/output shares provided by/to an InP must have an identical index.

Actually, each clique consists of routers from a subset of the InPs. Let a clique c consist of routers from a subset of InPs denoted by their index set $I \subseteq \{1, \dots, n\}$. By letting c_i be the router hosted by InP i, this clique can be represented by $c = \{c_i \mid i \in I\}$. Also, $[\![x]\!]_c$ denotes the subset of the shares $[\![x]\!]$ actually held by the routers in this clique, namely, $[\![x]\!]_c = ([x]_i \mid i \in I)$.

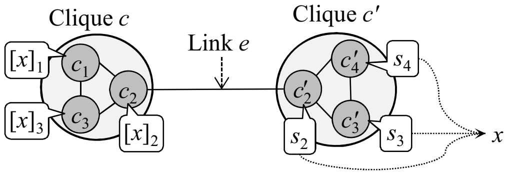


Figure 4.4: TRANSFER from a clique c to its neighboring clique c'. Inputs are shares $([x]_1, [x]_2, [x]_3)$ of a secret x. Expected outputs are (s_2, s_3, s_4) that can be decoded into the secret x.

4.4.3 Primitive Subprotocols within a Clique

We briefly describe the three primitives introduced in Section 4.3, SHARE, RECOVER, and COMPUTE, which are parts of a generic SMC protocol called the BGW protocol [25, 26].

SHARE is a protocol for sharing a secret x among the members of a clique c. Its invocation is denoted by $[\![x]\!]_c \leftarrow \text{SHARE}_c(x, c_j)$, which is implemented as follows. First, the router c_j who owns the secret x invokes a procedure called $\textsc{Share-Send}_{c_j}(x)$. This procedure encodes the secret x into a set of shares $[\![x]\!]_c$ by the SSS encoding algorithm, sends each share $[x]_i\ (i \neq j)$ to a router c_i, and returns $[x]_j$ as its own share. Then, each of the other routers, $c_i\ (i \neq j)$, invokes another procedure called $\textsc{Share-Recv}_{c_i}(c_j)$. This procedure receives a share $[x]_i$ sent by the sender c_j and stores it in this router's local storage.

RECOVER is a protocol for recovering a secret x shared among the members of the clique c. Its invocation is denoted by $x \leftarrow \text{RECOVER}_c([\![x]\!]_c)$. In this protocol, each router c_i invokes a procedure called $\textsc{Recover}_{c_i}([x]_i)$. This procedure sends its local share $[x]_i$ to every other router in c, then in turn receives the share from every other router, and finally decodes the secret x by the SSS decoding algorithm.

COMPUTE is a protocol for computing a function $y = F(x)$ in the clique c, where both of the input x and output y are shared among the members of the clique. Its invocation is denoted by $[\![y]\!]_c \leftarrow \text{COMPUTE}_c(F, [\![x]\!]_c)$. In this protocol, each router c_i invokes a procedure called $\textsc{Compute}_{c_i}(F, [x]_i)$. This procedure provides a share $[x]_i$ of the secret input x, and obtains a share $[y]_i$ of the secret output y. Each router learns nothing regarding the secrets x, y and intermediate computation results. Note that the input can be a vector of multiple inputs (e.g., $y = F(x_1, x_2)$) provided that each router has all the shares with an identical index (e.g., c_i has $[x_1]_i$ and $[x_2]_i$.) See [25, 26] for the details of the procedure, which are beyond the scope of this paper.

4.4.4 TRANSFER Primitive between Two Cliques

We describe TRANSFER, which is a new primitive we design for extending SMC to a distributed routing protocol. As shown in Fig. 4.4, TRANSFER is a protocol for transferring a secret x shared in the clique c to its neighboring clique c' via a clique-level link e between them. Let I and J denote the index sets of InPs that host routers in the clique c and c', respectively (e.g., in Fig. 4.4, $I = \{1, 2, 3\}$ and $J = \{2, 3, 4\}$). Its invocation is denoted by $[\![x]\!]_{c'} \leftarrow \text{TRANSFER}_{c,c'}(e, [\![x]\!]_c)$. In this protocol, each sender router $c_i \in c$ invokes a procedure called $\textsc{Transfer-Send}_{c_i}(e, [x]_i)$, and each receiver router $c'_j \in c'$ invokes a procedure called $\textsc{Transfer-Recv}_{c'_j}(e)$.

There could be a naïve solution where each router c_i in the sender clique sends its share $[x]_i$ to the corresponding router c'_i in the receiver clique, both of which are hosted by InP i. Such a naïve solution, however, does not work if there is some InP that appears in c' but does not in c (i.e., $J - I \neq \phi$). For example, in Fig. 4.4, clique $c = \{c_1, c_2, c_3\}$ has no share that can be simply sent to c'_4.

Therefore, TRANSFER must somehow generate the missing shares for all such InP $j \in J - I$. If one of the routers in the sender clique c had access to the secret x, we could simply use the SSS encoding algorithm to generate the missing shares. However, this is not the case with TRANSFER; for example, no party has access to the secret x if it is an output from COMPUTE on a function of multiple secret inputs originated from different secret owners.

In order to achieve the goal of TRANSFER described above, TRANSFER must provide every receiver router c'_j with an output denoted by s_j that satisfies the following two requirements:

(i) The collection of outputs $(s_j \mid j \in J)$ must constitute correct shares that can be decoded into the original secret x.

(ii) No InP other than InP j obtains the output s_j.

Prior to explaining our TRANSFER design, we need to look further into the SSS decoding algorithm because our TRANSFER implementation depends on its mathematical property. As described in Section 4.8, decoding a secret is computing a weighted sum of its shares;

$$\sum_{i \in I} \alpha_i^I [x]_i = x, \tag{4.1}$$

where α_i^I is a constant independent from either the secret or its shares. (See Section 4.8 for the details of the constant.)

First, we show that s_j satisfying the above requirement (i) can be obtained as a function of the input shares $([x]_i \mid i \in I)$. Equation (4.1) implies that the requirement (i) can be rephrased by the equation

$$\sum_{j \in J} \alpha_j^J s_j = x. \tag{4.2}$$

To obtain such s_j, we apply the SSS in two-fold as illustrated in Fig. 4.5. Initially, each router c_i in the sender clique has a share $[x]_i$. Each router c_i can

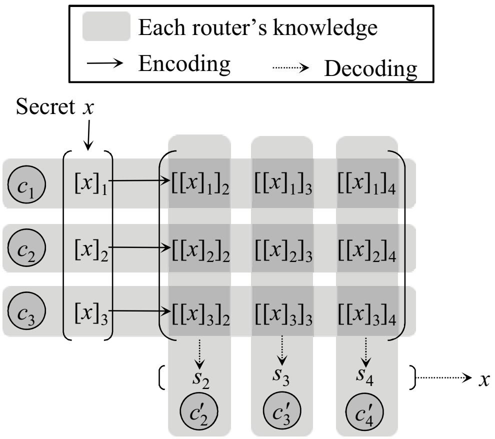


Figure 4.5: How TRANSFER generates new shares (s_2, s_3, s_4) that can be decoded into the original secret x (in order to fulfill the requirement illustrated in Fig. 4.4).

further apply the SSS encoding algorithm to obtain a row of two-fold shares $([[x]_i]_j \mid j \in J)$. As a whole, a matrix of two-fold shares $([[x]_i]_j \mid i \in I,\, j \in J)$ is obtained. By definition, this matrix of shares can be decoded into the secret x by applying two-fold decoding in the reverse order, i.e.,

$$\sum_{i\in I} \alpha_i^I \left(\sum_{j\in J} \alpha_j^J [[x]_i]_j \right) = x. \tag{4.3}$$

By noting that decoding is a weighted sum of shares, swapping the order of decoding (i.e., the order of summation) does not affect the final output x, namely,

$$\sum_{j\in J} \alpha_j^J \left(\sum_{i\in I} \alpha_i^I [[x]_i]_j \right) = x. \tag{4.4}$$

Since (4.4) can be regarded as an instance of (4.2), we can define s_j as

$$s_j = \sum_{i\in I} \alpha_i^I [[x]_i]_j. \tag{4.5}$$

Equation (4.5) shows that the share s_j satisfying the requirement (i) is derived as the weighted sum of the column j of the share matrix.

Second, we show how each receiver router c'_j obtains s_j defined by (4.5), while satisfying the requirement (ii). Equation (4.5) implies that each receiver router c'_j can locally compute s_j if c'_j has the column j of the share matrix, in other words, if every sender router c_i can send its share $[[x]_i]_j$ to c'_j. This is easily achievable by making the two routers at both ends of the link e (e.g., c_2 and c'_2 in Fig. 4.4) relay messages between the two cliques. Besides, in order to satisfy the requirement (ii), the sender router and receiver router encrypt messages between them. This prevents the relaying routers from eavesdropping on the contents of the messages. The encryption key can be pre-shared or exchanged on demand by some existing key exchange protocol like Diffie-Hellman.

4.4.5 MDR Protocol

We describe our solution for the MDR problem defined in Section 4.2.3. The protocol $\text{MDRP}_R(G_R, u)$ described in Algorithm 1 is the shortest path MDR protocol running on routers R in a router-level topology G_R for a destination $u \in U$. This protocol runs the Bellman-Ford routing algorithm [47] on the clique-level topology, while secret costs are invisible from underlying InPs by our primitives defined in Section 4.4.3 and 4.4.4.

Suppose a clique c is hosted by InPs I. Algorithm 1 shows the procedure $\textsc{Mdrp}_{c_i}(\mathit{links}_{c_i}, u)$ that runs on a router c_i hosted by InP i in the clique c and computes its $\mathit{nexthop}_{c_i}$ for the destination u. In this procedure, clique-level topology is described as follows. The clique-level next-hop of c takes a value on $\{e_0, \ldots, e_K\}$ where e_0 is the direct clique-level link to the destination u and $e_1, \ldots, e_K$ are the K clique-level links to its neighboring cliques.

First, as the **(1) initial input** to the protocol (in lines 2-17 of Algorithm 1), this router c_i shares this clique's local topology information described by the costs $w_0, \ldots, w_K$ of the links $e_0, ..., e_K$, respectively. For each link $e_k \in \{e_1, \ldots, e_K\}$ to a neighboring clique, if this router is directly connected to e_k, this router shares its cost w_k by invoking $\textsc{Share-Send}_{ci}$. Otherwise, this router receives its share $[w_k]_r$ by invoking $\textsc{Share-Recv}_{c_i}$. Because we consider shortest path routing, the direct clique-level link e_0 to the destination should be the shortest of the direct router-level links from all the routers in this clique to the destination. Hence, its cost $w_0 = min(d_j \mid j \in I)$ where $d_j = w(\{c_j, u\})$ is the cost of the direct router-level link from a router c_j. (From the definition of the cost function $w(\cdot)$, this cost $d_j = \infty$ if the router c_j has no direct router-level link to the destination.) To obtain its share $[w_0]_i$, the router c_i shares the costs $(d_j \mid j \in I)$ of these router-level links by invoking $\textsc{Share-Send}_{c_i}/\textsc{Share-Recv}_{c_i}$ and $\textsc{Compute}_{c_i}(min, (d_j \mid j \in I))$. This share $[w_0]_i$ is also the initial value of the share $[D]_i$ of this clique's current distance D, namely the shortest path cost, to the destination.

Then, at **(2) each iteration** *step* (in lines 18-25), the router transfers $[D]_i$ via each link $e_k \in \{e_1, \ldots, e_K\}$ to a neighboring clique by invoking $\textsc{Transfer-Send}_{c_i}$. In turn, via each link e_k, the router receives a share $[D_k]_i$ of the neighbor's current distance D_k to the destination by invoking $\textsc{Transfer-Recv}_{c_i}$. Using these shares $[D_1]_i, \ldots, [D_K]_i$ of distances, this router updates its share $[D]_i$ of this clique's distance by computing the function

$$\begin{aligned} D &= \mathit{UpdateDistance}(w_0, \ldots, w_K, D_1, \ldots, D_K) \\ &= \min(w_0, w_1 + D_1, \ldots, w_K + D_K) \end{aligned} \tag{4.6}$$

Algorithm 1 $T_R \leftarrow \mathrm{MDRP}_R(G_R, u)$

```
procedure MDRP$_{c_i}$($links_{c_i}$, $u$)
    for $k \leftarrow 1$ to $K$ do                                    ▷ (1) Initial input
        if $e_k \in links_{c_i}$ then
            $[w_k]_i \leftarrow$ SHARE-SEND$_{c_i}(w(e_k))$
        else
            $[w_k]_i \leftarrow$ SHARE-RECV$_{c_i}(c \cap e_k)$
        end if
    end for
    for $j \in I$ do
        if $i = j$ then
            $[d_j]_i \leftarrow$ SHARE-SEND$_{c_i}(w(\{c_j, u\}))$
        else
            $[d_j]_i \leftarrow$ SHARE-RECV$_{c_i}(c_j)$
        end if
    end for
    $[w_0]_i \leftarrow$ COMPUTE$_{c_i}(min, ([d_j]_i \mid j \in I))$
    $[D]_i \leftarrow [w_0]_i$
    for $step \leftarrow 1$ to $step_{max}$ do                        ▷ (2) Each iteration
        for $k \leftarrow 1$ to $K$ do
            TRANSFER-SEND$_{c_i}(e_k, [D]_i)$
            $[D_k]_i \leftarrow$ TRANSFER-RECV$_{c_i}(e_k)$
        end for
        if $step < step_{max}$ then
            $[D]_i \leftarrow$ COMPUTE$_{c_i}$($UpdateDistance$,
                      $[w_0]_i, \ldots, [w_K]_i, [D_1]_i, \ldots, [D_K]_i$)
        else                                                          ▷ (3) Final Output
            $[k']_i \leftarrow$ COMPUTE$_{c_i}$($NextHopLinkID$,
                      $[w_0]_i, \ldots, [w_K]_i, [D_1]_i, \ldots, [D_K]_i$)
            $k' \leftarrow$ RECOVER$_{c_i}([k']_i)$
            if $e_{k'} \in links_{c_i}$ then
                $nexthop_{c_i} \leftarrow e_{k'}$
            else
                $nexthop_{c_i} \leftarrow \{c_i, c \cap e_{k'}\}$
            end if
            return $nexthop_{c_i}$
        end if
    end for
end procedure
```

Table 4.1: Three parameter settings for different scales of networks

Parameter	Small	Medium	Large
Degree of a clique (K)	4	8	12
Distance bit-length ($\lvert D\rvert$)	4 bits	8 bits	16 bits
Link bandwidth (B)	1 Gbps	1 Gbps	10 Gbps

through $\textsc{Compute}_{c_i}$ on *UpdateDistance*.

Finally, this protocol converges and produces the **(3) final output** (in lines 26-35). See Section 4.5 for the impact of the number of iterations, $step_{max}$, required for convergence on the performance. The router obtains the link ID $k' \in \{0, ..., K\}$ for the clique-level next hop by computing the function

$$\begin{aligned} k' &= \mathit{NextHopLinkID}(w_0, \ldots, w_K, D_1, \ldots, D_K) \\ &= \operatorname{argmin}(w_0, w_1 + D_1, \ldots, w_K + D_K) \end{aligned} \tag{4.7}$$

through $\textsc{Compute}_{c_i}$ on *NextHopLinkID* and $\textsc{Recover}_{c_i}$. If the router c_i itself has the link $e_{k'}$, the router-level $\mathit{nexthop}_{c_i}$ is this link. Otherwise, $\mathit{nexthop}_{c_i}$ is the link to the router that is connected to $e_{k'}$, namely the router $c \cap e_{k'}$. Our solution for the MDR problem completes since every router's nexthop is calculated securely.

4.5 Performance Evaluation

Our proposed solution is expected to have overhead in route calculations for conducting SMC compared to the existing routing algorithms. From a usability perspective, we must ensure that the proposed solution has comparable route convergence time compared to that of the existing non-secure solutions. To this end, this section examines the extra latency incurred by the SMC protocol in various networks, namely, the small-, medium- and large-scale settings shown in Table 4.1.

According to the measurement report [48], we assume the degree K of a clique (i.e., the number of clique-level links connected to a clique) at a typical peering point ranges from 4 (the average degree of POPs) to 12 (the average degree of backbone routers). The bit-length of distance varies from 4 bits (the maximum distance is 15 as in RIP) to 16 bits (as in OSPF).

We do not consider the impact of statistical traffic variation in the evaluation below. This is because resource isolation is one of the fundamental functionalities provided by network virtualization infrastructure, and thus the control plane resources of the SP's slice are easily isolated from its data plane resources. This enables the routing protocol running in the control plane to be unaffected by the data plane traffic variation. The evaluation

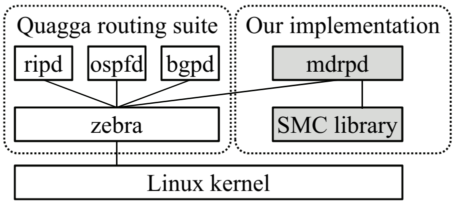


Figure 4.6: Software structure of our MDRP implementation. Shaded boxes are the components we developed.

under statistical traffic variation is one of the important advanced topics we should investigate in the future work.

4.5.1 Experiments on the Testbed

For the performance evaluation with the small-scale setting, we implement the proposed solution as a routing module of the Quagga routing software suite [49] on Linux and carry out experiments on the testbed in our lab. As shown in Fig. 4.6, our implementation consists of two software modules. One is a routing protocol daemon called mdrpd that runs as an extension module of Quagga-0.99.16. The other is a library implementing the generic SMC protocol. Since both modules run at the user level, they require neither a specific kernel version nor any kernel modification. (For reference, we use vanilla linux-2.6.26 kernel with SMP option enabled.) Fig. 4.7 shows the topology of our testbed, which consists of 12 routers and 4 cliques hosted by 6 InPs. This topology represents two competing nationwide backbone InPs peering with four local access InPs in four cities, like LA, Seattle, Houston, and NY, as is often the case with the Internet of today [10, 43]. Every router runs an instance of mdrpd, and every pair of neighboring mdrpds is connected by a TCP connection. This TCP connection carries all of the messages generated by the SMC protocol implemented in the SMC library.

In every step of our method, each clique of routers needs to compute *UpdateDistance* defined by Equation (4.6) for every destination $u \in U$. Thus, we implement a single function *UpdateDistanceVector* that updates a vector of M distances in parallel, where $M = |U|$ is the number of destinations (i.e., the number of IP prefixes). Each of the four access InPs injects $M/4$ IP prefixes to the proposed solution.

In our experiments for the small-scale setting, we use GbE at every link in the network and incur no artificial delay between neighboring routers in a clique (resulting in 61 usec delay, which is usual in GbE environments). For the other parameters, see the small-scale setting in Table 4.1 and additional parameters in Table 4.2. Since *UpdateDistanceVector* can be decomposed for each destination, we can leverage multi-threaded parallel computation on multi-core CPUs. In all our experiments, the number of threads on the

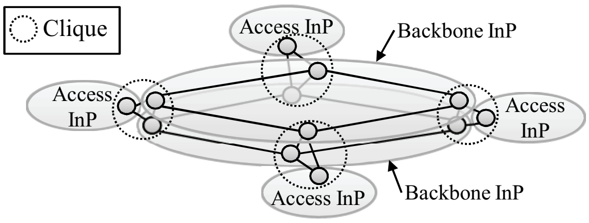


Figure 4.7: Topology of our experiment testbed.

Table 4.2: Parameters in experiments

Router CPU	Opteron 2.2 GHz
Transport of the routing protocol	TCP
Size of clique ($\|c\|$)	3
Size of share of secret bit (s)	3 bits
# of parallel threads ($n_{parallel}$)	1, 2, 4, 8, 16
# of destinations (M)	$2000, \ldots, 10000$

routers under evaluation is set to less than that of the CPU cores so that each thread may exclusively occupy a CPU core. The size s of each share of a secret bit is determined as shown in Section 4.8. The number of destinations, M, is evaluated up to 10000 in all the settings, which is considered sufficiently large for the number of IP prefixes in the IGP routing inside an SP's slice.

4.5.2 Analytical Model

Additionally, we develop an analytical performance evaluation model in order to evaluate the proposed solution with the medium- and large-scale settings that cannot be examined by the experiments.

The SMC protocol requires each router to exchange messages with every other router in its clique and to perform computation on the received messages, thus, it must incur extra latency for both computation and communication compared to the non-secure version without SMC. In this analytical model, we focus on the latency of an invocation of COMPUTE on *UpdateDistanceVector*, denoted by T_{update}, which is expressed as

$$T_{update} = T_{comm} + T_{comp}, \tag{4.8}$$

where T_{comm} and T_{comp} are latencies for communication and computation, respectively.

Prior to describing T_{comm} and T_{comp} further, we need to show how the generic SMC protocol [26] is applied to a generic function such as *UpdateDistanceVector*, which is usually expressed as a complicated mathematical equation like (4.6). The generic SMC protocol consists of two sub-protocols; one is for computing the addition of two secret bits and the other is for computing the

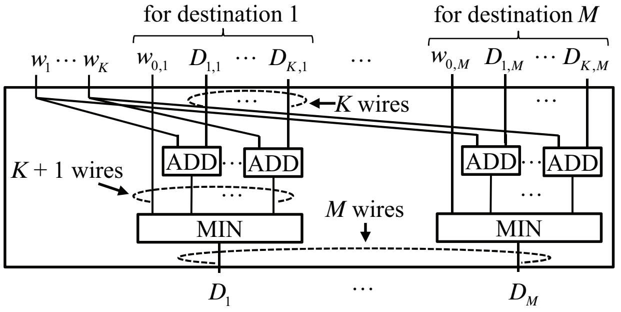


Figure 4.8: Block-level (not gate-level) circuit representation of *Update-DistanceVector*.

Table 4.3: Parameters in the analytical model

# of addition gates (*asize*)	$MK(18\lvert D\rvert + 5)$
# of multiplication gates (*msize*)	$MK(12\lvert D\rvert + 2)$
Multiplication depth (*mdepth*)	$\lvert D\rvert+\log(K+1)\{$ $\log(\lvert D\rvert+1)+2\}$
Delay of links within a clique (P)	61 usec
Computing an addition gate (T_{add})	14.1 nsec
Computing a multiplication gate (T_{mul})	81.7 nsec

multiplication of two secret bits. Therefore, in order to compute a generic function by these sub-protocols, the function first needs to be decomposed to a circuit of bitwise addition/multiplication gates. See Section 4.9 for how *UpdateDistanceVector* can be decomposed to such a circuit. Fig. 4.8 shows the high-level circuit representation of *UpdateDistanceVector*.

The performance of the SMC protocol depends on the complexity of the circuit. Table 4.3 summarizes the complexity parameters. The size parameters, *asize* and *msize*, denote the numbers of addition gates and multiplication gates, respectively. The parameter *mdepth* denotes the number of multiplication gates that need to be computed in sequence due to their dependency in the circuit.

In the SMC protocol, these addition and multiplication gates are computed in a gate-by-gate manner. From the mathematical property of the secret sharing scheme [46], addition gates can be locally computed by each router using only its own share (i.e., the equation $[x]_i + [y]_i = [x + y]_i$ holds for two secret bits x and y.) In contrast, multiplication gates require communication between routers in a clique besides the local computations within each router.

Therefore, the communication latency T_{comm} is attributed to the multiplication sub-protocol. It is invoked *msize* times, and each time, a router needs to transmit a s-bit share to the other routers in the clique via a link of bandwidth B. Transmissions regarding multiple gates can be performed in bulk only if these gates have no mutual dependency. In order to resolve the dependency between multiplication gates, *mdepth* communication rounds are required, each of which takes time of one-way propagation delay denoted by P. Thus, the communication latency is formulated as

$$T_{comm} = msize \cdot s/B + mdepth \cdot P. \tag{4.9}$$

The delay P shown in Table 4.3 is measured in our testbed (half the RTT measured by ping on a GbE link).

The computation latency T_{comp} is the sum of computation time required for the addition gates and the multiplication gates. Since, as shown in Fig. 4.8, the circuit in question can be decomposed per each destination, we can leverage parallel computation, and thus T_{comp} can be expressed as

$$T_{comp} = (asize \cdot T_{add} + msize \cdot T_{mul})/n_{parallel}, \tag{4.10}$$

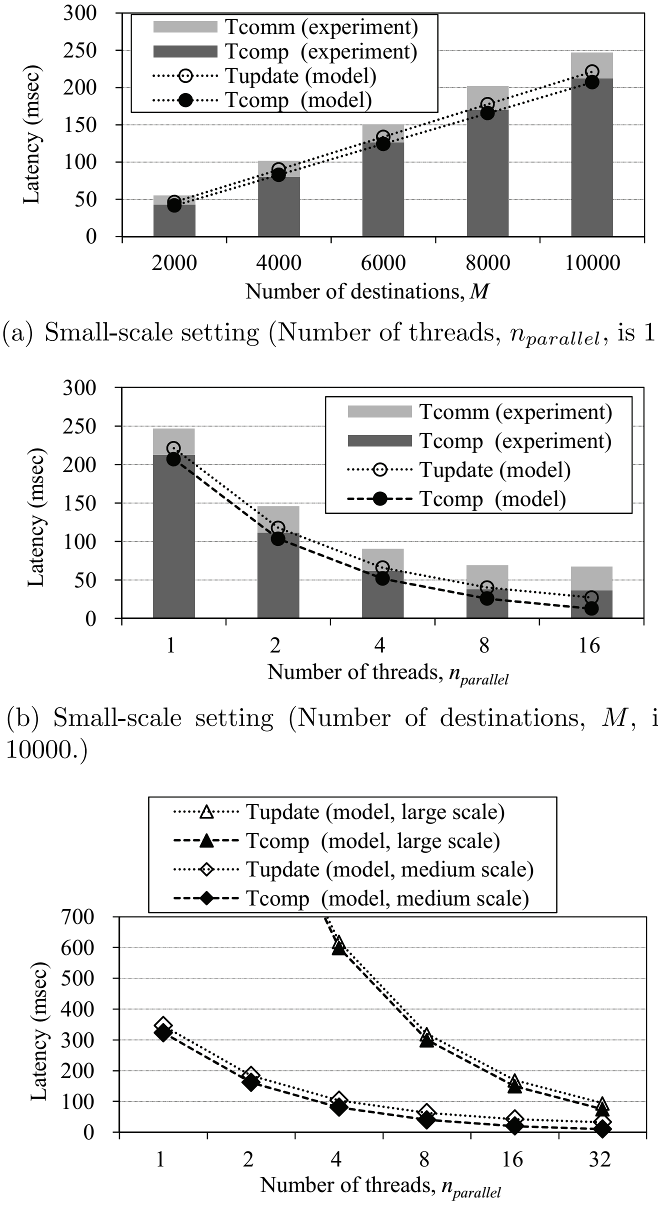
where T_{add} and T_{mul} are the latencies required for local computation per addition gate and multiplication gate, respectively, and $n_{parallel}$ is the number of CPU cores. The values of T_{add} and T_{mul} shown in Table 4.3 are measured by micro-benchmarking in our testbed. The computation latency for each gate is rather small (tens of nanoseconds) because our SMC protocol is an information-theoretic scheme, rather than a cryptographic scheme.

4.5.3 Evaluation Results

In the small-scale setting, we evaluate the performance both experimentally and analytically. Fig. 4.9(a) shows T_{update} as the number of destinations increases. (Note that the results of experiments are shown without confidence intervals because our experiments involve no statistical factors such as traffic fluctuation.) On the whole, the analytical model effectively predicts the linear increase of the latency. In particular, the absolute value of computation latency T_{comp} is accurately predicted. On the other hand, the prediction of the total latency, $T_{update} = T_{comm} + T_{comp}$, is somehow optimistic because the model does not consider the communication overhead incurred by TCP used in the experiment.

We evaluate the performance improvement by parallel computation for a large number of destinations (M = 10k entries). Fig. 4.9(b) shows the latency reduction as the number of threads used increases. We observe that the latency T_{update} is reduced down to sub-hundred milliseconds when $n_{parallel}$ is set to 4 or more; both experimentally and analytically.

These results indicate that our analytical model is accurate enough to predict the performance of the proposed solution under different parameters. Thus, we project the performance for the medium- and large-scale settings using the analytical model. From Fig. 4.9(c), we observe that the latency T_{update} decreases to sub-hundred milliseconds when $n_{parallel}$ is 32, even in the



(a) Small-scale setting (Number of threads, $n_{parallel}$, is 1.)

(b) Small-scale setting (Number of destinations, M, is 10000.)

(c) Medium- and large-scale settings (Number of destinations, M, is 10000.)

Figure 4.9: Latency of COMPUTE(*UpdateDistanceVector*)

large-scale setting. Note that state-of-the-art multi-core CPUs and GPGPU can achieve $n_{parallel} = 32$ easily.

Consequently, the proposed solution converges within a second, even in large-scale networks, because it requires as many number (denoted by $step_{max}$ in Algorithm 1) of invocations of *UpdateDistanceVector* as the diameter of the network, which is estimated as at most 10 in a Tier-1 network [50]. Since the convergence time of the well-engineered OSPF is in the order of sub-seconds [51], we conclude that the convergence time in the proposed solution is comparable to that in typical routing algorithms. Thus, the proposed solution is viable, since it is secure, yet no worse than the typical routing algorithms in terms of convergence time.

4.6 Discussion

4.6.1 Security of the MDR Protocol

Any solution to a security problem needs a thorough analysis of the security level it provides. The security of the existing primitives, SHARE, RECOVER, and COMPUTE, is already established in literature [25, 26]. Thus, in this subsection, we discuss the security of the proposed TRANSFER primitive and MDRP. Since we suppose that InPs are honest-but-curious adversaries, they would try to reverse engineer the SP's routing as much as possible by passively investigating the two types of information discussed below. We show that our solution is secure even though InPs fully exploit both types of information. [1]

Explicit Information

One type of information is that explicitly disclosed to InPs in protocol messages. TRANSFER and MDRP do not have such explicit disclosure of secret information. In TRANSFER, explicit information disclosed to InP $j \in J$ is the messages $([[x]_i]_j \mid i \in I)$ sent to the receiver router c'_j. Each message $[[x]_i]_j$ is a uniformly distributed random variable generated by the SSS encoding algorithm. To run such a probabilistic algorithm, each sender router c_i has a random number generator. Because these random number generators are mutually independent, the generated messages are also independent random variables and thus have no correlation. Such a set of independent random variables carries no information useful for InP j to infer the original secret x.

In MDRP, information explicitly disclosed to InPs is the collection of all the outputs and messages received in the four primitives. As discussed above,

[1] We informally outline the proof of the security because strict formal proof of protocol security (e.g., according to the mathematical framework proposed by [52]) is far beyond the scope of a networking paper.

these primitives are ensured to explicitly disclose no confidential information to InPs. All the information disclosed to InPs is protected by the SSS, which is information-theoretically secure based on our assumption of no collusion between two or more InPs.

Implicit Information (Side Channel Attack)

The other type of information is that implicitly disclosed to InPs through so-called *side channel*. TRANSFER and MDRP have no such side channel that discloses secret information to the underlying InPs. In general, typical side channels are the time required for computation, the pattern of memory access, the timing of message transmission/reception, and the sizes of messages. The sources of these side-channel information are the conditional branches in software programs. If a program branches based on a condition determined by any secret input, the resulting program execution behavior differs depending on the secret input, and these differences can be observed by adversaries through the side channel mentioned above. If a program has no such sensitive branch, the program is called *oblivious* [53]. An oblivious program behaves identically regardless of its secret input, except for the contents of both the memory it writes in and the messages it sends.

As shown in Algorithm 1, our MDRP is oblivious because it has no conditional branch that depends on the secret routing information (i.e., D, D_k, and w_k). Instead, all of its conditional branches depend on only the non-secret information, e.g., the parameters regarding the network size ($step_{\max}$, K, and I) and the InP's local information (i and $links_{c_i}$). Consequently, the proposed MDRP has no side channel that leaks confidential routing information to the underlying InP. TRANSFER also has no side channel in the same way, although we omit the detailed pseudo code representation of TRANSFER.

4.6.2 Using BGP without SMC Protocol

One might consider BGP to be a practical solution to MDR problems even without a costly security mechanism like SMC. This is because BGP is designed to work between administrative domains, namely autonomous systems (ASes), and subslices in MDR are also administrative domains. We could run BGP between subslices if we assign a private AS number to each subslice. BGP can provide routing without disclosing internal topology and link costs inside each subslice. In spite of this similarity between inter-domain routing and MDR, BGP cannot solve MDR problems because of the following two reasons.

First, BGP cannot provide optimal routing. Since MDR is a kind of intra-domain routing problem, solutions for MDR must provide routing that is as optimal as achieved by ordinary intra-domain routing algorithms as such as the distance vector or the link state algorithm. Neither of BGP's minimum

AS-hop routing nor policy-based routing has such optimality. Minimum AS-hop routing could lead to non-optimal path because shorter AS-level path does not necessarily mean shorter router-level path. BGP's policy control knobs [54] can be used to implement routing policies other than the minimum AS-hop routing. These knobs are, however, not designed to implement optimal routing. Rather they are designed to express each AS's business relationships with directly neighboring ASes such as customer-provider and peering. On the other hand, globally optimal routing requires some global view of the network. In the case of MDR, they are end-to-end path costs or whole topology information including link costs of the entire SP's slice, which cannot be expressed by AS-path information or the policy control knobs of BGP.

Second, even if the non-optimality of BGP is acceptable for an SP, BGP discloses some topology information to underlying InPs. For example, AS path information advertised in BGP contains partial information regarding AS-level (i.e., subslice-level) topology of the SP's slice. This violates the requirement of MDR described in Section 4.2. We discuss a possible countermeasure for this in Section 4.6.3.

4.6.3 Using the Existing Routing Protocols with SMC Protocol

Although we implement our solution as a completely new protocol, there can be alternative implementation. A promising approach is implementing the solution as an extension to some existing routing protocol, which is familiar to the current network operators.

RIP is obviously the easiest choice, because our MDR solution is based on a distance vector algorithm similar to the algorithm of RIP.

OSPF can be used in two application scenarios; intra-area routing and inter-area routing. Intra-area routing of OSPF is link state routing, which discloses much topology information between the routers in the same area. If the SP's entire network is designed as a single OSPF area, we need to minimize such disclosure by the SMC protocol. This approach is highly expensive in terms of communication and computation cost of the SMC protocol, and thus unrealistic. Alternatively, if each subslice is assigned to a distinct OSPF area, our solution can be applied to the inter-area routing of OSPF because it is a kind of distance vector routing between border routers.

BGP can be applied to routing between subslices if we assign a private AS number to each subslice. This approach costs higher than application to RIP due to the difference in the operations required in the underlying routing algorithm. Distance vector algorithms like RIP require integer operations such as addition and comparison of distance values. In contrast, path vector algorithms like BGP require list operations such as concatenation and comparisons of AS path. These differences result in the differences in the logic circuits computed by the SMC protocol. For example, each wire in Fig. 4.8 that carries a distance value is replaced by a wire that carries a list of AS numbers. Similarly, integer addition (ADD) and comparison (MIN) blocks are replaced by list concatenation and comparison blocks.

Moreover, since BGP is not a simple path vector protocol but a sophisticated mechanism to express policies of ASes, applying SMC protocol to BGP incurs further overhead to protect such highly sensitive policy information of the SP. These policies are expressed by the control knobs provided by BGP, typically LocalPref, MED, community attribute, and AS path filter [54]. The first two, LocalPref and MED, are easier than the other to implement by SMC protocol because they are integer comparison operations. The other two, community attribute and AS path filter, require more complicated operations like string comparison or even regular expression matching. Although some SMC protocols optimized for regular expression matching have been developed [55, 56], such an operation incurs much more communication and computation overhead than a simple string or list comparison.

In summary, application of our solution to routing algorithms other than the distance vector algorithm incurs a certain level of overhead. Thus, before applying our solution, we should carefully investigate whether SPs really need such complex routing policy or not. Although an SP's slice is spanning multiple InP domains, the slice itself is a single domain network fully controlled by the SP. BGP's policy routing capability might be too much for such intra-domain routing. If traffic engineering is required, shortest path routing with carefully configured link costs may suffice [57].

4.7 Related Work

Security issues in network virtualization have just begun attracting attention. In particular, few studies are directly related to confidentiality of routing in network virtualization. Keller et al. [58] identified the problem of *accountability* in hosted virtual networks. In contrast, our work addresses confidentiality and is complementary to accountability and discusses fundamental security requirements in virtualized environments.

A few proposals exist for confidentiality of topology information in conventional inter-domain networking [59, 60], where several operators must cooperatively provide end-to-end paths. These studies have not adopted any sophisticated computation techniques like SMC and simply hide the information not necessary for computation and disclose some of confidential information required for computation. The *path key* mechanism [59] enables an ISP to establish several disjoint paths across several ISPs while *preserving topology confidentiality* from the other ISPs. A path key carried by an RSVP option is a hint for each ISP's border router to retrieve a series of internal routers on the establishing path. However, the IDs of border routers as well as routing information, such as distance to a destination, are visible to other

ISPs. In an IMS/NGN, *topology-hiding* [60] is required for hiding critical server addresses on a SIP session across several ISPs. However, this is not a routing problem addressed in this paper. We believe our proposed solution is also applicable to these scenarios.

High Assurance Internet Protocol Encryptor (HAIPE) [61] is NSA's interoperability specification for encrypted IP networks. The HAIPE architecture consists of an overlay network connecting some isolated IP networks called *Red enclaves* and its underlay network called *Black core*. An HAIPE device resides between each Red enclave and the Black core, and encrypts all information (including routing information) exchanged between Red enclaves. Since each HAIPE device is assumed to be trustworthy, encryption prevents the Black core from learning any routing information of the overlay network. Conversely, there is no such trustworthy device within InPs' networks in our MDR problem setting.

A large body of work in SMC studies distributed computation without disclosing each party's confidential information [25, 45, 26]. Unfortunately, the generic SMC protocols including [45, 26] and specific SMC protocols (e.g., privacy-preserving shortest path [62] and inter-domain routing between ASes [63]) are not applicable to MDR since they assume every pair of parties has a communication channel between them, while MDR is a problem to establish such logical channels. We break down the problem into smaller local SMC problems that require only local communication. To the best of our knowledge, our work is the first to extend and apply the SMC protocol to distributed routing.

4.8 Secret Sharing Scheme

The secret sharing scheme (SSS) [46] used in our primitives consists of a pair of encoding and decoding algorithms. In the following, all arithmetic operations refer to $GF(p)$ (i.e., modulo-p arithmetic on a set of integers $\{0,\ldots,p-1\}$), where p is the smallest prime number greater than the number of InPs, n. This scheme encodes a secret bit into an element on $GF(p)$. Thus, the size s of each share of a secret bit is $\lceil \log p \rceil$ bits. If a secret consists of a number of bits, this scheme can be applied to each bit of the secret.

This scheme has a threshold parameter t, which specifies the minimum number of shares required for decoding. In our MDR protocol, t is set to two because we assume there is no collusion between InPs.

The encoding algorithm divides a secret bit x into n shares $[\![x]\!] = ([x]_1, \ldots, [x]_n)$. First, it randomly chooses a polynomial q of degree $t-1$ with the constant-term x, i.e., $q(0) = x$. Then, each share $[x]_i$ is the value $q(i)$.

The decoding algorithm reconstructs the secret x by interpolating the polynomial value $q(0)$. Given a set of t points $\{(i, [x]_i) \mid i \in I\}$, where I is an arbitrary index set of size t on $GF(p)$, there exists a unique polynomial q of degree $t-1$ going through these points, and its value $q(0)$ can be computed

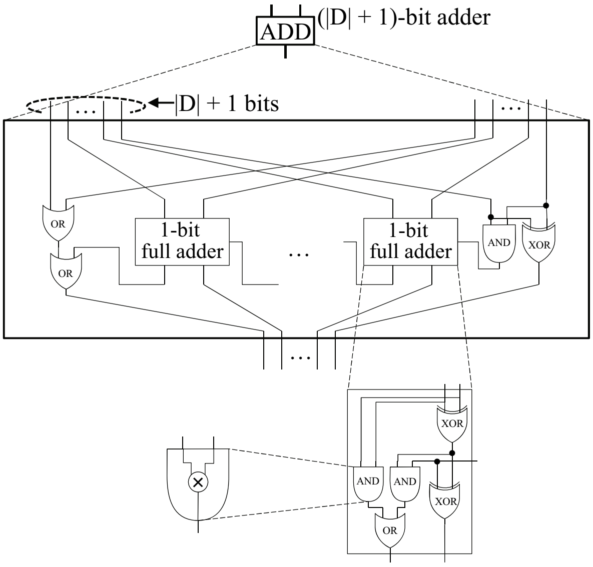


Figure 4.10: Hierarchical decomposition of an ADD block in Fig. 4.8.

by Lagrange's interpolation polynomial

$$q(0) = \sum_{i \in I} \alpha_i^I [x]_i, \quad \text{where} \quad \alpha_i^I = \prod_{k \in I - \{i\}} \frac{(p-1)k}{i-k}. \tag{4.11}$$

4.9 *UpdateDistanceVector* Circuit

We describe how the *UpdateDistanceVector* function can be decomposed to a circuit of bitwise addition and multiplication gates. We also derive its circuit complexity parameter *msize*, the number of multiplication gates in the circuit. (Although we omit the other complexity parameters in Table 4.3, namely *asize* and *mdepth*, due to the space limitation, they can also be derived in a similar fashion.) In Fig. 4.8, each wire in the circuit carries a distance value of length $|D|+1$ bits. ($|D|$ bits represent a finite distance and the additional bit is an infinity flag.)

Each ADD block in Fig. 4.8 is a $(|D|+1)$-bit adder, which can be decomposed to a sequence of $|D|-1$ one-bit full adders as shown in Fig. 4.10. Each full adder is further decomposed to logic gates such as AND, OR, and XOR. Finally, each logic gate is further decomposed into addition and multiplication gates, e.g., "x AND $y = xy$" and "x OR $y = x + y - xy$". Obviously, $msize(AND) = 1$ and $msize(OR) = 1$, where $msize(c)$ denotes the number of multiplication gates in a sub-circuit c. By summing up the number of multiplication gates from bottom up, $msize(ADD) = 4|D| - 2$ is obtained.

Each MIN block is a minimum selector with $K+1$ inputs, which can be decomposed into a binary tree of minimum selectors with two inputs.

Although we omit for brevity, each minimum selectors can be decomposed further into addition and multiplication gates like the ADD block above, and the number of multiplication gates $\mathit{msize}(\mathit{MIN}) = K(8|D| + 4)$ is obtained.

By noting that *UpdateDistanceVector* consists of MK ADD blocks and M MIN blocks, the total number of multiplication gates is obtained as

$$\begin{aligned}\mathit{msize} &= \mathit{msize}(\mathit{UpdateDistanceVector})\\ &= MK \cdot \mathit{msize}(\mathit{ADD}) + M \cdot \mathit{msize}(\mathit{MIN})\\ &= MK(12|D| + 2).\end{aligned}$$

4.10 Conclusion

We posit that operational confidentiality is crucial for enabling the virtual collocation of SPs on top of InPs via network virtualization (NV) in real business scenarios. We focus on Minimum Disclosure Routing (MDR) to enable an SP to route packets without disclosing routing information to InPs and propose that the extension to the generic Secure Multiparty Computation (SMC) securely achieves MDR. We implement the proposed MDR protocol and evaluate its performance, both experimentally and analytically. Our study reveals that the proposal is feasible since the extra latency overhead incurred in the convergence time in our secure routing protocol is within subseconds on large Tier-1 ISP networks and comparable to the convergence time in well-engineered intra-domain routing algorithms. The solution presented in this paper sheds light on the path for network virtualization for use in resolving all the challenges for the ISPs of today, (1) *footprint*, (2) *costs*, (3) *availability*, and especially (4) *operational confidentiality*, concurrently.

Chapter 5

Efficient Lookup Scheme for Name Prefixes

5.1 Introduction

Sustainability of the society has emerged as a pressing issue that needs to be addressed by information technologies. Recently, Information-Centric Networking (ICN) [13] is getting more and more attentions in the future internet research community, as a means for improving the sustainability of the society [64, 65] as well as enabling scalable and cost-efficient content distribution, intrinsic mobility, and multihoming. Content-Centric Networking (CCN) [14] pursued in Named-Data Networking (NDN) project is a promising ICN architecture that employs a hierarchical content naming scheme.

In CCN, *content names* are location independent and there is no notion of *locator* like IP address. Instead of name resolution like DNS, the bindings between a content name and its content source locations are gradually resolved by routers in *a hop-by-hop basis*. Each CCN router has a Forwarding Information Base (FIB), which binds every content name prefix to the next hops (i.e., the outgoing faces) toward its content sources. When a router needs to forward an Interest packet for a content name, it looks up the name in its FIB by longest prefix match, retrieves the next hop information, and forwards the packet to the next hop routers. Such hop-by-hop content locating naturally supports efficient content distribution, mobility, and multihoming. A popular content can be hosted by many content sources without managing many locators like in current CDNs. Moreover, each content source can change/add its attachment points to the network without globally advertising the new locators of these points to its content consumers or routing packets via an indirection point like in Mobile IP.

On the other hand, such location independent names raise scalability issues on FIB [15, 16]. Since location independent names are assigned to content sources regardless of their topological locations, name prefixes are hard to aggregate and thus the FIBs of CCN routers will be far larger than those of current IP routers. Thus, it is crucial to efficiently store and lookup such large FIBs. Fortunately, the downside trend of DRAM cost will enable CCN routers to store large FIBs on memory. FIB lookup latency issue is, however, more challenging due to the large latency of DRAM access and the complexity of longest prefix matching on variable- and unbounded-length names. Software-based FIB mechanisms employ a hash table [15] or trie [17]. Regardless of the underlying data structures, they need to seek the longest matching prefix through all candidate prefix lengths (in descending order with a hash table, or ascending order with a trie), and thus the number of random accesses to the DRAM per lookup is proportional to the name length, which makes the FIB lookup latency proportional to the name length and FIB throughput inversely proportional to the name length. In order to eliminate this limitation, hardware-based FIB mechanisms are proposed [16, 18], which store a Bloom filter in a low-latency on-chip (SRAM) memory and populate it with prefixes. For each prefix length, the fast Bloom filter is checked first, and only if it gives a positive result, a slow hash table in an off-chip DRAM is probed to retrieve the next hop information. If given a sufficiently large on-chip memory, a name lookup involves only a single DRAM access regardless of the name length. However, this hardware-based FIB requires an expensive on-chip memory of the size proportional to the number of prefixes, which makes its Internet-scale deployment in backbone routers infeasible.

We propose a new scheme to improve the efficiency of FIB lookup, which can be applied to the software-based FIBs for faster lookup and to the hardware-based FIBs to reduce on-chip memory. The proposed scheme is motivated by the observation that Interest packets matching a non-aggregatable prefix are forwarded by the same prefix length at every hop. Therefore, by exploiting the information on the longest matching prefix length in the previous hop, each CCN router could find the longest matching prefix without prefix seeking. Although, in the current CCN protocol, a router cannot learn the prefix length matched in the previous hop, it is easy to add some link-local header that carries the prefix length information along with each Interest packet. This does not violate the hop-by-hop principle of CCN, and can be incrementally deployed in the network.

In this study, we propose the new FIB lookup scheme and conduct thorough evaluation based on empirical data from the current Internet. These results suggest that the proposed scheme is a promising approach to improve the efficiency of the CCN-enabled future Internet.

5.2 Background

This section briefly reviews the design of CCN routers' forwarding plane and its bottleneck, and describes typical scenarios where non-aggregatable prefixes pose serious scaling issues.

5.2.1 Packet Forwarding in CCN

In CCN [14], contents are identified by a variable length name (like */foo/bar/-filename*), which consists components (*foo*, *bar*, and *filename*) and "/" as delimiters between components. The length of a name is counted by the number of components in it. CCN defines two packet types. A content consumer sends an *Interest* packet carrying the name of the desired content, and the packet is forwarded to some content sources. Upon its reception, the content sources respond by sending a *Data* packet carrying the requested content, and the packet is forwarded back to the consumer. In order to forward these packets, each router has three tables. When a router receives an Interest packet for a content name, it first looks up the name in the *Content Store (CS)*, which caches the Data packets it forwarded. If the CS has any matching Data, it is forwarded back to the previous hop. Otherwise, the router looks up the name in the *Pending Interest Table (PIT)*, which stores the binding between pending interest names (names of the Interest it forwarded but the response Data packet is not yet received) and their incoming faces. If the PIT has any matching entry, the incoming face is added to the entry. Otherwise, the router looks up the name by longest prefix match in the *Forwarding Information Base (FIB)*, which stores the bindings between name prefixes and one or more outgoing faces. If the FIB has any matching entry, the packet is forwarded according to the FIB entry, and a new PIT entry is created. When a Data packet is received, it is stored in the CS, and if the PIT has any matching entry, the Data packet is forwarded back to the incoming faces of the PIT entry.

As Perino et al. [16] pointed out, FIB lookup is the most critical bottleneck of CCN routers, especially in high-speed backbone routers. This is because it requires longest prefix matching, and to make matters worse, location independent names are generally hard to aggregate, and thus a router's FIB in a large scale network is loaded by hundreds of millions of non-aggregatable prefixes. In contrast to FIB lookup, CS and PIT lookups are not so serious bottleneck, because they do not necessarily require longest prefix matching. Besides, since CS and PIT entries are created on demand and eventually expire, their numbers are limited by the volume of active traffic, while FIB entries are maintained even for all potential, not necessarily active content sources.

5.2.2 Problems of Non-Aggregatable Prefixes

If CCN is deployed at Internet scale, we must face a challenge of designing CCN routers that meet forwarding latency requirements, while accommodating a large number of non-aggregatable prefixes in their FIBs.

One obvious origin of non-aggregatable prefixes is the Internet backbone routing table advertised in CCN-capable BGP [66], because many contents are published under provider-independent prefixes. According to a rough estimate, a full routing table of BGP includes 620 million prefixes, which is the number of web server hostnames as of December 2012 [67]. This is a far more challenging number compared to 0.43 million prefixes of the current BGP [68]. CCN-based backbone network operators need hardware-based high-speed backbone routers that can forward Interest packets at tens or hundreds of Mpps at a moderate cost.

Another potential origin of non-aggregatable prefixes is user-supplied contents. In CCN, users communicate with each other by bi-directional Interest exchange like Voice-over-CCN [69]. Such applications publish contents under user-specific prefixes like */AccessProvider.com/User*. While the BGP routing table stores only the aggregated prefix */AccessProvider.com/*, user-specific prefixes cannot be aggregated within the access provider's network, because the name *User* is flat and location independent. In order to forward Interests to the location where *User* is attached to the access provider, its internal routers need to store millions or tens of millions of non-aggregatable user-specific prefixes.

Similar problems could occur in the cloud infrastructure of Online Social Network (OSN) providers. For example, Interest packets for */OSN.com/User* need to be forwarded to the server hosting *User*'s contents such as blog posts, tweets, movies, and photos. Currently, such contents are located in randomly chosen servers by some distributed key-value store mechanism like distributed hash table. Recently, however, inefficiency of random placement has been pointed out [70, 71], because interests in OSN have strong spatial locality. In future, OSN providers might need to locate contents of strongly related users in a specific server located close to those users. This requires internal routers of large-scale OSN providers to store hundreds of millions of non-aggregatable user-specific prefixes.

5.3 Related Work

Some FIB mechanisms for CCN routers have been proposed, each of which suffers from scaling issues due to memory access latency when they are populated with a large number of non-aggregatable prefixes.

Software-based FIBs for general purpose CPUs are classified by their underlying data structures. *Hash table* provides exact matching with O(1) comparisons. Longest prefix matching is possible by seeking through all candidate prefix lengths in descending order. At the worst case, for a name of length B, this requires $O(B)$ comparisons. CCNx implements a *Name Prefix Hash Table (NPHT)* [15] that combines FIB and PIT functions into a single hash table. Each NPHT entry has a "parent pointer". If an Interest's filename component (excluding its sequence number component) hits the entry added by its preceding Interests for the same file, the parent pointer enables faster prefix seeking by reducing the number of name comparisons required for longest prefix matching. *Trie* is another well-known data structure supporting longest prefix matching, which seeks through the candidate prefix lengths in ascending order. Wang et al. [17] proposed a trie-based FIB for CCN with an efficient name component encoding scheme. (Although they claim that their scheme can support a few million lookups per second on a PC-based software implementation, their evaluation only considers the domain name parts of content names, and does not fully consider variable- and unbounded-length names.)

Regardless of the underlying data structures, these software-based FIBs need to seek through the candidate prefix lengths. Since CCN FIB is generally too large to fit into a small on-chip (SRAM) cache of a general purpose CPU, a random access to a high-latency off-chip (DRAM) memory is required for every prefix length. Consequently, the number of random accesses to the off-chip memory is proportional to the name length, which makes the FIB lookup latency proportional to the name length. Parallel processing does not address this issue, because the bottleneck is the access port of the off-chip memory, not the processor. Also, note that the parent pointer of NPHT does not address this issue, because following a parent pointer is also a random access to the off-chip memory.

In order to eliminate this limitation, *hardware-based FIBs* are proposed, which can be implemented as an ASIC or FPGA chip. Unlike IP forwarding engines, it is infeasible to implement a large CCN FIB by TCAM, due to its high cost and energy consumption. Therefore, these hardware-based FIBs exploit low-latency on-chip memory to minimize access to high-latency off-chip memory. *Bloom filter* based FIB was first proposed by Dharmapurikar et al. [72] for IP routers. Bloom filter [73] is a randomized data structure for storing a set of items. It is more space-efficient than a hash table because, firstly, it stores only keys and no related value, and secondly, it supports only an approximate set membership query function. Namely, it might return a false positive result when the queried item does not belong to the set. The false positive probability depends on the size of the Bloom filter. By exploiting this space efficiency, Dharmapurikar et al. [72] use a Bloom filter on an on-chip memory to minimize access to a hash table on an off-chip memory. The Bloom filter is populated with prefixes. For each prefix length, the fast Bloom filter is checked first, and only if it gives a positive result, the slow hash table is probed to retrieve the next hop information. If given a sufficiently large on-chip memory, each lookup involves only a single access to the off-chip memory regardless of the address length. Perino et al. [16] study the feasibility of the Bloom filter based FIB for CCN routers. They conclude that today's technology is not ready to support an Internet-scale CCN deployment in backbone routers, mainly because a Bloom filter based FIB requires an expensive on-chip memory whose size is proportional to the number of prefixes. Otherwise, the false positive probability of the Bloom filter increases, and thus prefix seeking is required as in the software-based FIBs. Based on the conceptual design of Perino, et al [16], Varvello, et al. [18] proceed one step further into a more concrete CCN router design, which distributes FIB to many line cards. The number of line cards required by this design is proportional to the number of prefixes. Since each line card needs a large on-chip memory, this design also suffers from the similar cost issue.

5.4 Exploiting Neighbors' Prefix Length Information

We propose a new scheme to improve the efficiency of FIB lookup, which is based on the prefix length information from the previous hop router. It can mitigate the scaling issue that the existing FIB mechanisms suffer in the scenarios described in Section II. The proposed scheme is sufficiently generic to be applied to the existing hardware/software FIB mechanisms described in Section III.

The proposed scheme is inspired by two architectural *invariants* [74] observed in the CCN architecture. One is *the latency of off-chip memory*, which is the critical bottleneck of FIB lookup and its drastic improvement will not be expected in near future, because an enormous number of non-aggregatable prefixes prevents routers from effectively caching FIB entries in a low-latency on-chip memory. The other is *the length of a non-aggregatable prefix.* If a router forwards an Interest packet by a non-aggregatable prefix, the next hop router is likely to forward the packet by the same prefix length. The proposed scheme minimizes the impact of the off-chip memory latency by exploiting the length of non-aggregatable prefixes.

5.4.1 Conventional FIB Lookup

The conventional FIB lookup function, denoted by $faces \leftarrow FIBLookup(N)$, takes a name N as an input, seeks the longest matching prefix length L through the candidate prefix lengths, retrieves the set of outgoing *faces* (to be precise, IDs of faces), and returns the *faces*.

Although this interface seems a very natural design choice from a single router viewpoint, it could be inefficient from a collective viewpoint. Suppose routers on a path have an almost same set of non-aggregatable prefixes in their FIBs. As described in Section II, this is rather the norm than the exception. Suppose the name N of an Interest packet matches a prefix of length L in the non-aggregatable prefixes, and the packet happens to be forwarded along the path. Then, every router on the path redundantly looks up the same name N in the almost same set of prefixes, and seeks the same prefix length L. Obviously, this is a waste of resources, especially for large FIBs.

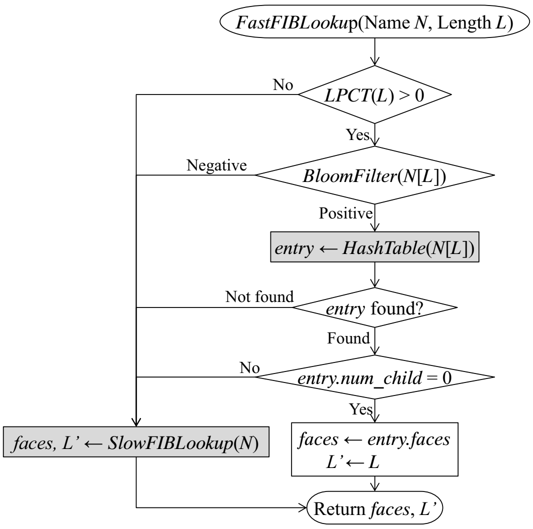


Figure 5.1: Proposed fast FIB lookup function. The shaded boxes need access to high-latency off-chip DRAM.

5.4.2 Proposed Fast FIB Lookup

If a router has a large FIB similar to its neighbor's FIBs, it can reduce prefix seeking by exploiting the longest matching prefix length L in the previous hop. The proposed fast FIB lookup function, denoted by $(\mathit{faces}, L') \leftarrow \mathit{FastFIBLookup}(N, L)$, has additional input L and output L', the longest matching prefix length of the previous hop router and the current router, respectively. The former L is forwarded from the previous hop and the latter L' will be forwarded to the next hop along with the Interest packet.

Fig. 5.1 illustrates the algorithm of *FastFIBLookup*. The two shaded boxes access to the high-latency off-chip DRAM. One shaded box *SlowFIBLookup* is the same as the conventional *FIBLookup*, except for having an additional output L'. This modification is trivial. In the following, we assume *SlowFIBLookup* is implemented by a Bloom filter in an on-chip memory and a hash table in an off-chip memory like Perino et al. [16], because it is the most promising approach to implement high-end CCN routers.

The other shaded box is $\mathit{HashTable}(N[L])$, where $N[L]$ denotes the length-L prefix of the name N, namely, the first L components of the name N (e.g., if N = */foo/bar/filename*, then $N[2]$ = */foo/bar*.) This box probes the FIB hash table in the off-chip memory only for the prefix length L. Note that we can use the result of this probing only if the found prefix of length L is a *leaf prefix*, namely, the prefix has no child prefix in the (hypothetical) prefix tree. This is because, if the prefix has some child prefixes, there is a possibility that one of them or their further descendant gives a match longer than L.

FastFIBLookup has two design objectives. Most importantly, it should avoid calling *SlowFIBLookup* as much as possible. Secondly, it should avoid unnecessarily calling *HashTable*$(N[L])$ if we can predict the hash table probing result is useless. In the best case, this scheme requires only a single access to the off-chip memory. These objectives are achieved as follows.

First, *FastFIBLookup* quickly filters out $N[L]$ if it is obviously a non-leaf prefix. For example, if the previous hop router forwarded a packet by the default prefix ("/"), the prefix of the same length in the current router (namely, the default prefix) is probably a non-leaf. This filtering is done by checking whether the FIB has any leaf prefix of length L, by using a small *Leaf Prefix Count Table (LPCT)*, which holds the number of leaf prefixes for every prefix length. If there is no such prefix, we fall back to *SlowFIBLookup*.

Next, *BloomFilter*$(N[L])$ is called in order to avoid unnecessary hash table probing if we have no such prefix. If its result is negative, we fall back to *SlowFIBLookup*. If the proposed scheme is applied to a software-based FIB without an on-chip Bloom filter, this step is simply omitted.

If the preceding two steps are positive, *HashTable*$(N[L])$ is called. By carefully choosing hash table implementation [75], this step can be done with a single access to the off-chip memory. If a prefix entry is found, and its *num_child* field equals zero, we are sure that it is the longest matching prefix. Otherwise, we fall back to *SlowFIBLookup*.

5.4.3 Additionally Required Information

In the above description, it is assumed that some pieces of information (L, *num_child*, and *LPCT*) are available, which are not required by the conventional FIBs. They can be efficiently obtained as follows.

Firstly, in order to obtain the longest matching prefix length L in the previous hop, we need to modify the protocol between neighboring routers. Since the size of prefix length information is very small, the overhead of adding this information is negligible from the viewpoint of link bandwidth. Another concern than efficiency is the impact of modifying CCN protocol from an architectural perspective. We discuss this issue in Section 5.6.1.

Secondly, the number of child prefixes should be maintained in each prefix entry's *num_child* field, even if the FIB is dynamically updated. This is easily done with a trie-based FIB entry that has explicit child pointers, and can be done with a hash table based FIB entry that has only a parent pointer, by the following procedures. When adding a new leaf prefix to the FIB, its *num_child* is set to zero, and its parent's *num_child* is incremented. If the new prefix has no direct parent (e.g., adding */foo/bar* to a FIB including only the default prefix *"/"*), all of the intermediate prefixes (*/foo*, in the above case) are added as dummy placeholders with *num_child* $= 1$. When deleting an existing prefix from the FIB, its parent's *num_child* is decremented. Any dummy prefix is deleted when its *num_child* becomes zero. Although these update procedures need multiple accesses to the off-chip memory, it is not the bottleneck because FIB update is less frequent than packet forwarding [15].

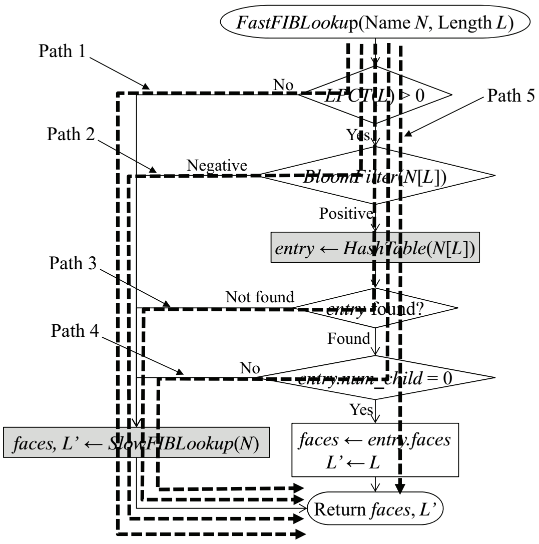


Figure 5.2: Each packet follows one of the five possible execution paths. Path 1 and 2 incur the latency of *SlowFIBLookup*. Path 5 incurs the latency of *HashTable*. Path 3 and 4 incur both of them.

Lastly, the $LPCT$ can be easily maintained by incrementing/decrementing $LPCT(L)$ whenever any non-dummy prefix of length L changes its *num_child* field from one-to-zero/zero-to-one, respectively.

5.5 Evaluation

We evaluate the efficiency of the *FastFIBLookup* scheme compared to that of the conventional *FIBLookup* scheme.

5.5.1 Evaluation Model

The efficiency of FIB lookup is evaluated by its latency, the time required by a router to lookup a name in its FIB. The latency of the conventional FIB lookup is evaluated by the model proposed by Perino et al. [16].

The latency L_{fast} of *FastFIBLookup* is determined by the latencies of *SlowFIBLookup* and *HashTable*, and how often these functions are called. Fig. 5.2 shows five possible execution paths (path 1, ..., 5) in *FastFIBLookup*. Let P_i $(i = 1, \ldots, 5)$ denote the probability that a packet follows execution path i. Then, the latency L_{fast} is defined by

$$\begin{aligned} L_{fast} &= L_{slow}(P_1 + P_2 + P_3 + P_4) \\ &\quad + L_{hash}(P_3 + P_4 + P_5) \end{aligned} \tag{5.1}$$

where L_{slow} and L_{hash} are the latencies of *SlowFIBLookup* and *HashTable*, respectively, and derived as shown in 5.7. The probabilities $P_1, \ldots, P_5$ depend

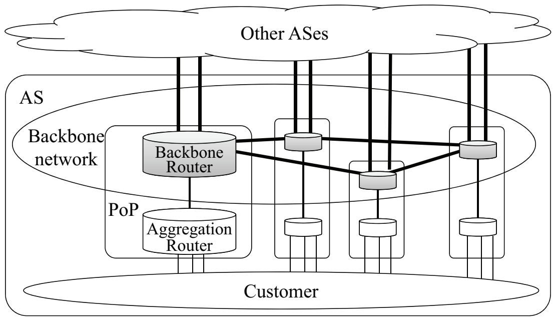


Figure 5.3: Evaluation model for a backbone router.

on the characteristics of the network topology and traffic forwarded by the router.

In order to determine reasonable values of these probabilities experienced by a typical backbone router, we consider a network model of an Autonomous System (AS) shown in Fig. 5.3. This AS has a *backbone network* consisting of backbone routers, each of which is located in a Point-of-Presence (PoP). These backbone routers are connected by backbone links, and running both of BGP and IGP. Each backbone router has a link to an *aggregation router*, which is in charge of forwarding packets between the backbone router and customers accommodated by the PoP. Each backbone router also has links to other ASes.

In this model, a packet from a customer or another AS enters the backbone network at a PoP, then forwarded along a sequence of backbone routers, called *backbone path*, and finally exits to a customer or another AS at a PoP. (Note that backbone routers are neither of sources or sinks of traffic.) When forwarding a packet, the backbone routers use either of the following two types of prefixes. One is *AS-level prefixes* globally advertised by BGP, which is used to forward packets destined for another AS. The other is *PoP-level prefixes* advertised by IGP, which is used to forward packets destined for customers within the AS. Besides, the aggregation router in a PoP forwards packets destined for outside the PoP by the *default prefix* (i.e., a zero-length prefix *"/"*) to its upstream backbone router.

In this evaluation, we assume PoP-level prefixes are aggregatable at AS-level. At first glance, this assumption contradicts with our motivation that name prefixes are non-aggregatable. Actually, this assumption is reasonable by the following reasons. In general, prefix aggregation is used to reduce the number of prefixes. On the other hand, in our evaluation, the number of prefixes is given as a parameter, not the result of aggregation. If the number of prefixes is given, the assumption that prefix length is variable (i.e., prefixes are aggregatable) results in more conservative evaluation (in a sense, the worst-case evaluation), because the proposed scheme can reduce FIB lookup latency only if the prefix length is the same as in the previous

Table 5.1: Each packet forwarded by a backbone router is classified into eight types

Packet type j	The current router's position on the backbone path	Source AS	Destination AS
1	first	same AS	same AS
2	first	same AS	another AS
3	first	another AS	same AS
4	first	another AS	another AS
5	not first	same AS	same AS
6	not first	same AS	another AS
7	not first	another AS	same AS
8	not first	another AS	another AS

Packet type j	Prefix used by the previous hop	Prefix used by the current router	Same prefix?	Path in Fig. 5.2
1	default	PoP-level	no	1
2	default	AS-level	no	1
3	AS-level	PoP-level	no	2 or 3
4	AS-level	AS-level	yes	4 or 5
5	PoP-level	PoP-level	yes	4 or 5
6	AS-level	AS-level	yes	4 or 5
7	PoP-level	PoP-level	yes	4 or 5
8	AS-level	AS-level	yes	4 or 5

hop.

From these observations, each packet forwarded by a backbone router (hereafter called the *current router*) can be classified into eight types shown in Table 5.1. As shown in the leftmost four columns of the table, each packet type is defined by three attributes; the current router's position on the backbone path followed by the packet, the source AS of the packet, and the destination AS of the packet. If the current router is the first router in the backbone path, its previous hop is an aggregation router or a router in a neighboring AS. Otherwise, its previous hop is a neighboring backbone router.

Each packet type uniquely determines the prefix types at the previous hop and the current router. Packets of type 1 and 2 are traffic from customers accommodated by the current router's PoP, and forwarded from the aggregation router by the default prefix, and thus follow the execution path 1 in Fig. 5.2. Namely,

$$P_1 = T_1 + T_2 \tag{5.2}$$

where T_j denote the probability that a packet is type j. Packets of type 3 are traffic from a neighboring AS and destined for customers within the AS.

Table 5.2: Packet type distribution T_j estimated from empirical data

Packet type j	AS 1221	AS 1239	AS 1755	AS 3257	AS 3967	AS 6461	Average
1	0.038	0.030	0.032	0.032	0.032	0.032	0.032
2	0.106	0.083	0.089	0.089	0.089	0.089	0.091
3	0.017	0.013	0.014	0.014	0.014	0.014	0.014
4	0.047	0.037	0.039	0.039	0.039	0.039	0.040
5	0.145	0.154	0.152	0.152	0.152	0.152	0.151
6	0.405	0.428	0.422	0.422	0.422	0.422	0.421
7	0.064	0.068	0.067	0.067	0.067	0.067	0.066
8	0.178	0.188	0.186	0.186	0.186	0.186	0.185

These packets are forwarded by AS-level prefixes in the previous hop, and forwarded by PoP-level prefixes in the current router. Thus, they follow the execution path 2 or 3, depending on whether the Bloom filter results in false positive or not, namely

$$P_2 = T_3(1 - P_{fpos}), \tag{5.3}$$

$$P_3 = T_3 P_{fpos}, \tag{5.4}$$

where P_{fpos} is the false positive probability shown in 5.7. Packets of type 4, 5, 6, 7, and 8 are forwarded by the same prefix in their previous hop router and current router, and thus follow the execution path 4 or 5, depending on whether the prefix is a leaf or not, namely

$$P_4 = (T_4 + T_5 + T_6 + T_7 + T_8)(1 - P_{leaf}), \tag{5.5}$$

$$P_5 = (T_4 + T_5 + T_6 + T_7 + T_8)P_{leaf}, \tag{5.6}$$

where P_{leaf} is the probability that a prefix is a leaf.

The remaining parameters, T_j and P_{leaf}, are determined from empirical data. Unfortunately, there is currently no large-scale commercial CCN network. As a second-best way, we investigated these values of the current Internet.

The packet type distribution T_j ($j = 1, \ldots, 8$) is estimated from empirical data. For each backbone topology of six ASes published by Mahajan et al. [76], we synthesize a traffic matrix by using the gravity model proposed by Roughan [77], and route this traffic by shortest path routing. See 5.8 for the details on how T_j is derived. The results are summarized in Table 5.2. We use the average over the six ASes as the values of T_j.

The probability P_{leaf} is determined from the current IPv4 prefixes advertised in BGP, available from CAIDA [78]. The dataset includes 460441 IPv4 prefixes. We randomly choose an address from the address space covered by the dataset, lookup the longest matching prefix, and investigate whether

Table 5.3: Other evaluation parameters

Parameters	Hardware-based FIB	Software-based FIB
Number of prefixes (n)	620 million [67]	—
On-chip memory size (M)	1 Gbits	—
On-chip memory latency (L_{on})	0.45 ns [16]	—
Off-chip memory latency (L_{off})	15 ns [16]	60 ns [79]
Name length (B)	30 comp. [16]	1, . . ., 30 comp.
Prefix length (L)	—	1, . . ., 15 comp.
Name component length (K)	—	10 bytes [80, 81]
Hash table load factor (α) [82]	—	0.5
CPU clock frequency (F)	—	3 GHz
CPU word length (W)	—	8 bytes

the prefix has any child prefix (i.e., any prefix covered by the prefix) or not. As a result of sampling one million addresses, we obtain $P_{leaf} = 0.685$. For reference, we also evaluate the latency for $P_{leaf} = 0.5$, 0.75, and 1.

5.5.2 Evaluation Results

The proposed scheme is evaluated in two scenarios; the application to a hardware-based FIB like Perino et al. [16], and the application to a software-based FIB implemented by a hash table. Unless otherwise stated, the parameters not discussed in Section 5.5.1 are set as shown in Table 5.3.

Fig. 5.4 shows the latencies of hardware-based FIB as functions of the number of prefixes n for different values of P_{leaf}. The entire on-chip memory is dedicated to the Bloom filter and its size is fixed to 1 Gbits. For small FIBs ($n < 100$ million), both of the conventional and proposed scheme achieve nearly minimum latency of 15 ns, the latency of a single access to off-chip Reduced Latency DRAM (RLDRAM). Namely, each lookup requires only a single hash table probing. As n increases, the latency of the conventional scheme rapidly increases and converges to B times larger than the minimum, where B is the name length, because the false positive probability of the Bloom filter converges to 1 due to the excessive amount of registered prefixes. The latencies of the proposed scheme increase slower. For $n = 620$ million (the number of web server hostnames of today [67]) and $P_{leaf} = 0.685$, the latency of the proposed scheme is less than half of the conventional scheme.

Fig. 5.5 shows the latencies as functions of the on-chip memory size M (i.e., Bloom filter size). For any latency target, the proposed scheme significantly reduces the required on-chip memory size, which is the primary source of the cost increase to support CCN in a backbone router [16].

Although it is currently unrealistic to use a software-based router for forwarding backbone traffic, it would be worthwhile to evaluate the performance

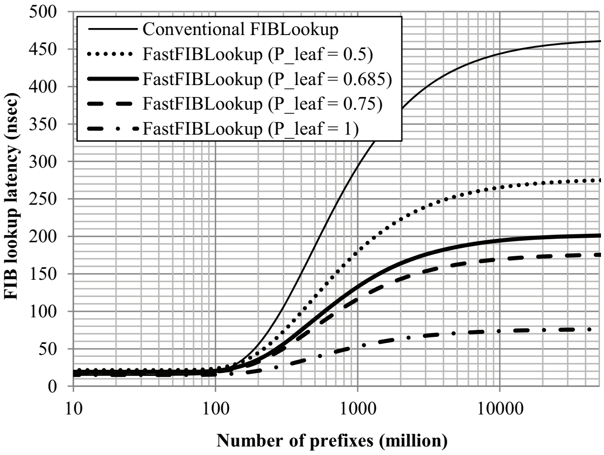


Figure 5.4: The latencies of hardware-based FIBs. The on-chip memory size M is fixed to 1 Gbits, which is a fairly high standard for today's technology.

of the proposed scheme applied to software-based FIBs, due to the following reasons. Firstly, software-based router technologies are rapidly improving in recent years [83, 84]. Secondly, CCN is rather long term research direction and thus the applicability of software-based router in the era of CCN deployment could be broader than that of today. Therefore, based on these assumptions, we evaluate the performance of the proposed scheme applied to software-based FIBs. As a reference, we suppose software-based backbone router's FIB lookup latency must be kept below 230 ns, which is the average interval of packets of 4-KB size (the default packet size of the current CCN implementation [85]) arriving at the rate of 140 Gbps (the line card throughput of state-of-the-art backbone routers [86]).

The performance of the software-based FIB is evaluated, where the FIB is too large to fit into the on-chip cache of a CPU. Fig. 5.6 shows the latencies as functions of the name length B. The minimum latency is 60 ns, the latency of a single access to off-chip DRAM. The latency of the conventional scheme increases linearly to the name length, because every prefix length requires an off-chip DRAM access. The latencies of the proposed scheme increase slower. Even if $B = 30$, the 99.9-percentile of URL length in a traffic trace [16], the latency of the proposed scheme with $P_{leaf} = 0.685$ is less than half of the conventional scheme. In order to keep the lookup latency below 230 ns, software-based FIBs can not support names with arbitrary length. The conventional scheme can support only name with 5 components, whereas the proposed scheme with $P_{leaf} = 0.685$ can increase this limitation to 10 components.

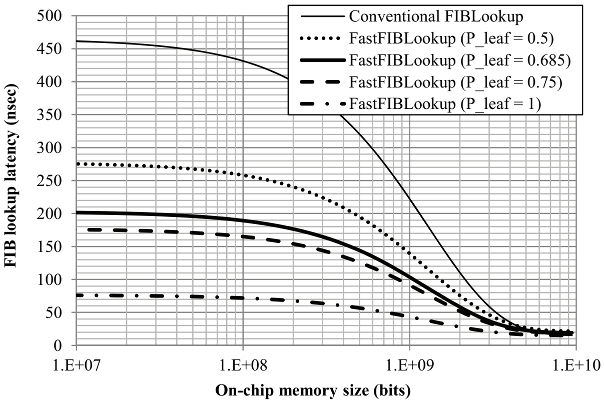


Figure 5.5: The latencies of hardware-based FIBs. The number of prefixes n is fixed to 620 million, which is the same as the number of web server hostnames of today.

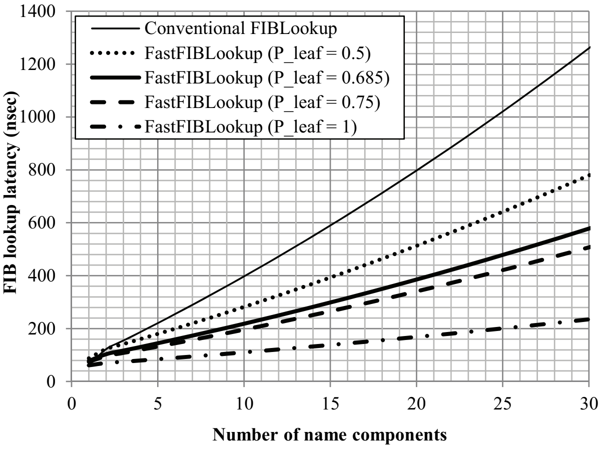


Figure 5.6: The latencies of software-based FIBs.

5.6 Discussion

5.6.1 Impact on CCN architecture

In the standard CCN architecture, there is no protocol to exchange prefix length information with neighboring routers. In order to support this feature, we have two options. One is to modify the networking layer protocol of CCN and add a prefix length field in the Interest packet format. Although this is easy from the viewpoint of implementation cost, it might violate the principle of CCN architecture, because prefix length information is per-hop information and inappropriate to be embedded in the networking layer. Probably, adding some link layer protocol is better solution, which inserts a tiny shim header carrying the prefix length information in front of every Interest packet.

Such a link layer protocol can be easily designed by extending the existing link layer protocol for NDN [87]. Note that such a link layer modification affects neither of the CCN network layer protocol nor its hop-by-hop content locating principle, and thus it is easy to incrementally deploy the proposed scheme.

5.6.2 Security Consideration

We need to discuss whether exchanging prefix length information between neighbors involves any security issue or not, because such information is not exchanged in conventional networking protocols. First of all, since the proposed scheme is a link local protocol, there is no security issue between non-neighboring routers.

As for integrity concerns, there could be a possibility that a router sends an invalid prefix length other than L' to its next hops, intentionally or accidentally. From a receiver's viewpoint, this is rather an efficiency issue and not a security issue, because the proposed scheme strictly checks whether the prefix of that length is a leaf prefix or not. If a name matches with a leaf prefix, it must be the longest matching prefix. Otherwise, the proposed scheme falls back to the conventional scheme. Therefore, no router needs to trust its neighbor to correctly send the longest matching prefix length.

As for confidentiality concerns, some network operator might think prefix length is sensitive information. For example, suppose an access provider has a confidential business contract with a content provider (e.g., */ContentProvider.com/*). This contract requires the access provider to perform some policy-based routing for a specific prefix (e.g., */ContentProvider.com/Movie/PayPerView*). This is implemented by adding the specific prefix to routers' FIBs. If the proposed scheme sends the length of the prefix to another network operator who provides transit between the access provider and the content provider, the transit provider could guess the existence of the confidential business contract. In order to prevent such undesired information disclosure, the access provider can add a "length to export" attribute to the prefix, which is used instead of the actual longest matching prefix length for the packets forwarded to the transit provider's routers. Such a workaround is required for a small fraction of prefixes in the BGP table, and thus it does not affect the overall efficiency of the proposed scheme.

As for availability concerns, it is technically possible for a owner of a router to mount a DoS attack on a neighboring router by intentionally sending invalid prefix length information. From a practical perspective, however, such an attack is not effective in commercial ISP networks, because the attacker must be a neighbor of the victim ISP, and thus the victim ISP can easily identify the attacker.

5.7 Latency Model

For a hardware-based FIB, the latency L_{slow} of *SlowFIBLookup* follows the equation (1) of Perino et al. [16] and the latency L_{hash} of *HashTable* equals the off-chip memory latency L_{off}. The false positive probability P_{fpos} of an M-bit Bloom filter populated by n prefixes is given by

$$P_{\mathit{fpos}} = \left(\frac{1}{2}\right)^{\frac{M \ln 2}{n}}. \tag{5.7}$$

In contrast to the hardware-based FIB, a software-based FIB probes each length one by one in descending order, from the name length B until finding the longest matching prefix length L. Therefore, the latency of *SlowFIBLookup* in a software-based FIB is given by

$$L_{slow} = \sum_{\ell=B}^{L} L_{hash}(\ell). \tag{5.8}$$

The effect of the on-chip cache memory is omitted, because the FIB is too large to fit into the small cache of a CPU. Thus, we have no on-chip Bloom filter, namely, $P_{\mathit{fpos}} = 1$. Also, by carefully implementing *HashTable* with open addressing [75], its latency L_{hash} for a prefix of length ℓ is given by

$$L_{hash}(\ell) = L_{\mathit{off}} + \frac{1}{1-\alpha} \cdot \frac{\ell K}{WF}, \tag{5.9}$$

the sum of the latency of an off-chip memory access and the time required to compare hash keys, where $\frac{1}{1-\alpha}$ is the number of key comparisons required to probe the hash table of load factor α [82] and $\frac{\ell K}{WF}$ is the time required to compare equality of two prefixes of the average component length K bytes, at the rate of one word length W per CPU clock of frequency F.

5.8 Packet Type distribution

Given one of the backbone topologies published by Mahajan et al. [76, 88], the probability T_j that a packet forwarded by a backbone router is type j is derived as the weighted average over all backbone routers in the topology, namely,

$$T_j = \sum_r T_{j,r} \frac{V_r}{\sum_u V_u} \tag{5.10}$$

where $T_{j,r}$ is the probability that a packet forwarded by a specific backbone router r is type j, and V_r is the relative frequency of a packet visiting the router r.

Let C_x^{in} and C_x^{out} be the probability that a packet respectively enters from and exits to a customer accommodated by the PoP of a backbone router x.

Similarly, let A_x^{in} and A_x^{out} be the probability that a packet respectively enters from and exits to other ASes linked with a backbone router x. We follow the gravity model [77], namely, the entering location and exiting location of a packet is assumed to be statistically independent. Then,

$$V_r = \sum_x \sum_y (C_x^{in} + A_x^{in})(C_y^{out} + A_y^{out}) R_{xry} \tag{5.11}$$

where R_{xry} is the probability that a packet entering at a backbone router x and exiting at a backbone router y visits a backbone router r. By assuming equal-cost multipath routing, this equals to the number of shortest path(s) from x via r to y divided by the total number of shortest path(s) from x to y.

For packet type j = 1, 2, 3, and 4 (i.e., the router is the first hop in the backbone path), $T_{j,r}$ are given by

$$T_{1,r} = \sum_y C_r^{in} C_y^{out}, \tag{5.12}$$

$$T_{2,r} = \sum_y C_r^{in} A_y^{out}, \tag{5.13}$$

$$T_{3,r} = \sum_y A_r^{in} C_y^{out}, \tag{5.14}$$

$$T_{4,r} = \sum_y A_r^{in} A_y^{out}. \tag{5.15}$$

For type j =5, 6, 7, and 8 (i.e., the router is not the first hop in the backbone path), $T_{j,r}$ are given by

$$T_{5,r} = \sum_{x \neq r} \sum_y C_x^{in} C_y^{out} R_{xry}, \tag{5.16}$$

$$T_{6,r} = \sum_{x \neq r} \sum_y C_x^{in} A_y^{out} R_{xry}, \tag{5.17}$$

$$T_{7,r} = \sum_{x \neq r} \sum_y A_x^{in} C_y^{out} R_{xry}, \tag{5.18}$$

$$T_{8,r} = \sum_{x \neq r} \sum_y A_x^{in} A_y^{out} R_{xry}. \tag{5.19}$$

We assume that the entering/exiting probabilities, A_x^{in}, A_x^{out}, C_x^{in}, and C_x^{out}, are proportional to the normalized population D_x of the city where the backbone router x is located, namely,

$$\begin{aligned} C_x^{in} &= C^{in} D_x,\ C_x^{out} = C^{out} D_x, \\ A_x^{in} &= A^{in} D_x,\ A_x^{out} = A^{out} D_x. \end{aligned} \tag{5.20}$$

For each backbone router x, its city name is annotated in the topology data [88]. Thus, we determine the value of relative population D_x from the city population statistics by United Nations [89]. All values of D_x are normalized so that $\sum_x D_x = 1$. If multiple backbone routers are located in a city, the population is equally distributed among those routers.

The total probabilities, C^{in}, C^{out}, A^{in}, and A^{out}, are derived by solving the following equations from (5.21) to (5.23). The entering probabilities and exiting probabilities should add up to unity, namely,

$$C^{in} + A^{in} = C^{out} + A^{out} = 1. \tag{5.21}$$

The ratio of upload and download by customers within the AS is given by a parameter α , i.e.,

$$\frac{C^{in}}{C^{out}} = \alpha. \tag{5.22}$$

We set $\alpha = 2.63$ from the traffic statistics of broadband users in Japan [90]. The probability that a packet entering from other ASes is destined for a customer within the AS is given by a parameter β, namely,

$$\frac{C^{out}}{C^{out} + A^{out}} = \beta. \tag{5.23}$$

We set $\beta = 0.264$, which is the inverse of the average AS-hop length 3.7856 [68] as of Nov. 2012. These equations yield a solution

$$\begin{aligned} &C^{in} = \alpha\beta, \qquad C^{out} = \beta, \\ &A^{in} = 1 - \alpha\beta,\ A^{out} = 1 - \beta. \end{aligned} \tag{5.24}$$

5.9 Conclusion

We proposed a new scheme for efficiently looking up non-aggregatable name prefixes in a large FIB. The proposed scheme is based on the observation that the bottleneck of FIB lookup is the random accesses to the high-latency off-chip DRAM for prefix seeking and this can be reduced by exploiting the information on the longest matching prefix length in the previous hop. Our evaluation results show that the proposed scheme significantly improves FIB lookup latency with a reasonable traffic parameters observed in today's Internet.

Chapter 6

Conclusion

The Internet traffic has been increasing far beyond the prediction in the past. In particular, mobile network traffic is recently expected to continue increasing exponentially. Due to this increasing traffic demand, it is becoming more and more challenging for service providers to provision sufficient network resources such as link bandwidth and node capacity in a timely manner to keep the quality of experience (QoE) perceived by users of their services at a certain level. Besides, the networking research community has begun to recognize that the fundamental solution of these problems is difficult with the current Internet architecture based on Internet Protocol (IP).

In order to overcome these challenging issues of the current Internet, many efforts to develop *future network* technologies are being carried out. These efforts have achieved a certain level of success, establishing individual technologies to address each issue of the current Internet. In a large-scale system like the Internet, however, a collection of individual technologies is not sufficient as a solution of practical issues. In addition to the individual technologies, operational aspect of the Internet is essential to address these issues. In this study, we posed several operational issues of the technologies developed for future networks and proposed solutions for them.

In Chapter 2, it was discussed that user's mobility impacts on the traffic distribution in the systems beyond IMT-2000. Under these environments, the time scales of bandwidth sharing and mobility cannot be simply separated. Numerical results for cellular-WLAN overlay environments were examined to demonstrate that the mobility of users has a significant impact on the traffic distribution between the different systems and its impact is possibly comparable to the number of WLAN APs. A framework for the performance evaluation of such systems was proposed. A queueing network model with nonlinear traffic equations was applied taking into account the independence among nodes in the network. The applicability of the proposed analysis method was verified through numerical results. The proposed model and analysis provide insights for those problems involved in frequency allocation, capacity planning and deployment of future seamless system-interworking environments. To realize spectrally efficient networks, efficient operation and deployment are essential as well as physical layer efficiency of the individual system. The convergence of different systems with diverse characteristics in systems beyond IMT-2000 makes the deliberate deployment scenario even more important. This includes optimal deployment of WLAN access points and optimization of operational parameters. Furthermore, systems in the next decade might be developed based on interworking environments from the beginning of their design. Such systems cannot operate if deployment and operation are not aware of interworking.

In Chapter 3, we proposed a new lightweight QoE measurement method. Up to now, many monitoring tools and utilities have been proposed. For example, a network operator can observe the utilization of a communication link by monitoring routers through Simple Network Management Protocol (SNMP). A good example is the Multi-Router Traffic Grapher (MRTG)[42] which periodically generates graphs of utilization. Measurement of utilization is a simple and effective method. However, utilization figures provide little information about actual performance of data transfer. When the link is utilized 100%, it only tells us the full utilization. One cannot tell whether the link is over-loaded or not. Our method can complement the existing tools, and it gives more information on the network status.

In Chapter 4, we posit that operational confidentiality is crucial for enabling the virtual collocation of SPs on top of InPs via network virtualization (NV) in real business scenarios. We focus on Minimum Disclosure Routing (MDR) to enable an SP to route packets without disclosing routing information to InPs and propose that the extension to the generic Secure Multiparty Computation (SMC) securely achieves MDR. We implement the proposed MDR protocol and evaluate its performance, both experimentally and analytically. Our study reveals that the proposal is feasible since the extra latency overhead incurred in the convergence time in our secure routing protocol is within sub-seconds on large Tier-1 ISP networks and comparable to the convergence time in well-engineered intra-domain routing algorithms. The solution presented in this paper sheds light on the path for network virtualization for use in resolving all the challenges for the ISPs of today, footprint, costs, availability, and especially operational confidentiality, concurrently.

In Chapter 5, we proposed a new scheme for efficiently looking up non-aggregatable name prefixes in a large FIB. The proposed scheme is based on the observation that the bottleneck of FIB lookup is the random accesses to the high-latency off-chip DRAM for prefix seeking and this can be reduced by exploiting the information on the longest matching prefix length in the previous hop. Our evaluation results show that the proposed scheme significantly improves FIB lookup latency with a reasonable traffic parameters observed in today's Internet.

We conclude by acknowledging that for many years the user experiences of networks have been captive to the efficiency of individual technologies developed for the current Internet. However, as the paper demonstrates, performance evaluation and measurement are key factors in understanding operational aspect of networks as well as improving user experiences in future networks. By using the proposed solutions, the communication service providers are able to design, operate, and measure their network services more appropriately when they provides services based on heterogeneous mobile networks, network virtualization environments, and content-centric networking. Our contribution is making it easier for service providers to introduce these new technologies into there actual services and thus take the networking industry one step further toward the resolution of the challenging problems the current Internet is facing. It allows network operators to meet rising user expectations for future services effectively and efficiently.

Acknowledgement

It would not have been possible to write this doctoral thesis without the help and support of the kind people around me, to only some of whom it is possible to give particular mention here.

This thesis would not have been possible without the help, support, encouragement and patience of my supervisor, Prof. Shigeki Goto, not to mention his advice and unsurpassed knowledge of networking. Members of Goto Lab. also deserve my sincerest thanks.

I also would like to thank Prof. Jiro Katto and Prof. Tatsuya Mori for carefully reading the draft of the thesis and giving insightful comments and suggestions.

I would like to thank Prof. Akihiro Nakao for his support and valuable discussions on our joint research on network virtualization.

I would like to acknowledge the financial and technical support of KDDI R&D Laboratories, Inc. and its staff. Amongst my colleagues in KDDI R&D Laboratories, Inc., I am particularly grateful to Dr. Yu Watanabe, Dr. Shinichi Nomoto, Dr. Toru Hasegawa, Dr. Hajime Nakamura, Dr. Teruyuki Hasegawa, Dr. Atsushi Tagami, and Dr. Kohei Sugiyama, who helped and contribute great ideas and advices. Without them, this study would not be possible.

Last, but by no means least, I would like to thank my family for their support and great patience at all times.

Bibliography

[1] ITU-R M.1645, "Framework and overall objectives of the future development of IMT-2000 and systems beyond IMT-2000," June 2003.

[2] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushayana, and A. Viterbi, "CDMA/HDR: A bandwidth-efficient high-speed wireless data service for nomadic users," IEEE Communications Magazine, July 2000.

[3] "GENI: Global Environment for Network Innovations." `http://www.geni.net/`.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," ACM SIGCOMM CCR, vol.38, no.2, pp.69–74, 2008.

[5] A. Nakao, R. Ozaki, and Y. Nishida, "CoreLab: An Emerging Network Testbed Employing Hosted Virtual Machine Monitor," ACM ROADS, December 2008.

[6] N. Feamster, L. Gao, and J. Rexford, "How to Lease the Internet in Your Spare Time," ACM SIGCOMM CCR, pp.61–64, January 2007.

[7] N. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," IEEE Communications magazine, vol.47, no.7, pp.20–26, 2009.

[8] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala, "Path splicing," ACM SIGCOMM CCR, vol.38, no.4, pp.27–38, 2008.

[9] D. McLoughlin and D. Aaker, Strategic Market Management: Global Perspectives, John Wiley & Sons, 2010.

[10] W. Norton, "The evolution of the US Internet peering ecosystem," The 31st NANOG meeting, 2004.

[11] "Steel in the Air, AT&T/Cingular Cell Tower Lease Renegotiation." http://www.steelintheair.com/Cingular-and-ATT-Wireless-Cell-Tower-Lease-Negotiations.html.

[12] P. Nattermann, “Best practice does not equal best strategy,” The McKinsey Quarterly, vol.2, no.2000, pp.22–31, 2000.

[13] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, “A survey of information-centric networking,” Communications Magazine, IEEE, vol.50, no.7, pp.26–36, 2012.

[14] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, “Networking Named Content,” ACM CoNEXT, Rome, Italy, Dec 2009.

[15] H. Yuan, T. Song, and P. Crowley, “Scalable ndn forwarding: Concepts, issues and principles,” Computer Communications and Networks (ICCCN), 2012 21st International Conference on, pp.1–9, IEEE, 2012.

[16] D. Perino and M. Varvello, “A reality check for content centric networking,” Proceedings of the ACM SIGCOMM workshop on Information-centric networking, pp.44–49, ACM, 2011.

[17] Y. Wang, K. He, H. Dai, W. Meng, J. Jiang, B. Liu, and Y. Chen, “Scalable name lookup in ndn using effective name component encoding,” Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on, pp.688–697, IEEE, 2012.

[18] M. Varvello, D. Perino, and J. Esteban, “Caesar: a content router for high speed forwarding,” Proceedings of the second edition of the ICN workshop on Information-centric networking, pp.73–78, ACM, 2012.

[19] S.B. Fredj, T. Bonald, A. Proutiere, G. Régnié, and J.W. Roberts, “Statistical bandwidth sharing: A study of congestion at flow level,” SIGCOMM 2001, Aug. 2001.

[20] S. Borst, “User-level performance of channel-aware scheduling algorithms in wireless data networks,” INFOCOM 2003, April 2003.

[21] S. Seetharaman and M. Ammar, “On the interaction between dynamic routing in the native and overlay layers,” IEEE INFOCOM, pp.1–12, 2006.

[22] W. Parkhurst, Cisco OSPF command and configuration handbook, Cisco Press, 2002.

[23] F. Chung, M. Garrett, R. Graham, and D. Shallcross, “Distance realization problems with applications to Internet tomography,” Journal of Computer and System Sciences, vol.63, no.3, pp.432–448, 2001.

[24] M. Fukushima, T. Hasegawa, T. Hasegawa, and A. Nakao, “Minimum disclosure routing for network virtualization,” IEEE INFOCOM Workshop (Global Internet Symposium), pp.858–863, April 2011.

[25] O. Goldreich, Foundations of Cryptography, volume 2, Basic Applications, Cambridge University Press, 2004.

[26] M. Ben-Or, S. Goldwasser, and A. Wigderson, “Completeness theorems for non-cryptographic fault-tolerant distributed computation,” ACM STOC, pp.1–10, 1988.

[27] S.S. Rappaport and L. Hu, “Microcellular communication systems with hierarchical macrocell overlays: Traffic performance models and analysis,” Proc. of IEEE, vol.82, no.9, pp.1383–1397, Sept. 1994.

[28] R.J. Boucherie and N.M. van Dijk, “On a queueing network model for cellular mobile telecommunications networks,” Operations Research, vol.48, no.1, pp.38–49, Jan. 2000.

[29] E. Esteves, P. Black, and M. Gurelli, “Link adaptation techniques for high-speed packet data in third generation cellular systems,” European Wireless 2002, Feb. 2002.

[30] A. Jalali, R. Padovani, and R. Pankaj, “Data throughput of CDMA-HDR a high efficiency high data rate personal communication wireless system,” VTC 2000 Spring, 2000.

[31] F.P. Kelly, Reversibility and Stochastic Networks, John Wiley & Sons, 1979.

[32] X. Chao, M. Miyazawa, and M. Pinedo, Queueing Networks: Customers, Signals and Product Form Solutions, John Wiley & Sons, 1999.

[33] T. Okabe, T. Shizuno, and T. Kitamura, “Wireless LAN access network system for moving vehicles,” Proceedings of IEEE ISCC 2005, June 2005.

[34] A. Salkintzis, G. Dimitriadis, D. Skyrianoglou, N. Passas, and N. Pavlidou, “Seamless continuity of real-time video across UMTS and WLAN networks: Challenges and performance evaluation,” IEEE Wireless Communications, June 2005.

[35] K.L. Yeung and S. Nanda, “Channel management in microcell/macrocell cellular radio systems,” IEEE Transactions on Vehicular Technology, vol.45, no.4, pp.601–612, Nov. 1996.

[36] B. Jabbari, “Teletraffic aspects of evolving and next-generation wireless communication networks,” IEEE Personal Communications, Dec. 1996.

[37] “Imnet (inter-ministry research information network).” http://www.imnet.ad.jp/.

[38] D.E. Comer, Internetworking with TCP/IP vol 1: principles, protocols and architecture, Prenctice Hall, 1995.

[39] W.R. Stevens, TCP/IP Illustrated, Volume 1, The Protocols, Addison-Wesley, 1994.

[40] "Oc3mon/coral." `http://www.caida.org/Tools/CoralReef/`.

[41] "Apan (asia-pacific advanced network)." `http://www.apan.net/`.

[42] "Mrtg (multi router traffic grapher)." `http://ee-staff.ethz.ch/~oeticker/webtools/mrtg/mrtg.html`.

[43] Qwest Business, "Qwest Network Maps." `http://www.qwest-business.com/demos/network-maps.html`.

[44] B. Payne, "XenAccess: An Introspection Library for Xen," ACSAC, 2006.

[45] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," ACM STOC, pp.218–229, 1987.

[46] A. Shamir, "How to share a secret," Communications of the ACM, vol.22, no.11, pp.612–613, 1979.

[47] N. Lynch, Distributed algorithms, Morgan Kaufmann, 1996.

[48] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP topologies with Rocketfuel," IEEE/ACM Transactions on networking, vol.12, no.1, pp.2–16, 2004.

[49] "Quagga routing suite." `http://www.quagga.net/`.

[50] R. Fukumoto, S. Arakawa, T. Takine, and M. Murata, "Analyzing and modeling router–level Internet topology," LNCS, vol.5200, pp.171–182, 2008.

[51] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving sub-second IGP convergence in large IP networks," ACM SIGCOMM CCR, vol.35, no.3, pp.35–44, 2005.

[52] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," IEEE FOCS, pp.136–145, 2001.

[53] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious RAMs," Journal of the ACM, vol.43, no.3, pp.431–473, 1996.

[54] M. Caesar and J. Rexford, "BGP routing policies in ISP networks," IEEE Network, vol.19, no.6, pp.5–11, 2005.

[55] J. Troncoso-Pastoriza, S. Katzenbeisser, and M. Celik, “Privacy preserving error resilient DNA searching through oblivious automata,” ACM CCS, pp.519–528, 2007.

[56] A. Singh, S. Barman, and K. Shukla, “Secure two-party context free language recognition,” Distributed Computing and Internet Technology, pp.117–124, 2005.

[57] B. Fortz, J. Rexford, and M. Thorup, “Traffic engineering with traditional ip routing protocols,” IEEE Communications Magazine, vol.40, no.10, pp.118–124, 2002.

[58] E. Keller, R. Lee, and J. Rexford, “Accountability in hosted virtual networks,” ACM VISA, pp.29–36, 2009.

[59] J. Vasseur and A. Farrel, “Preserving topology confidentiality in inter-domain path computation using a path-key-based mechanism.” RFC 5520, April 2009.

[60] “3GPP TS 23.228 V5.15.0,” June 2006.

[61] United States National Security Agency, “High assurance internet protocol encryptor interoperability specification, v. 3.1.0,” December 2006.

[62] J. Brickell and V. Shmatikov, “Privacy-preserving graph algorithms in the semi-honest model,” LNCS, vol.3788, p.236, 2005.

[63] S. Machiraju and R. Katz, “Verifying global invariants in multi-provider distributed systems,” ACM SIGCOMM HotNets, pp.149–154, 2004.

[64] J. Burke, A. Horn, and A. Marianantoni, “Authenticated lighting control using named data networking,” tech. rep., UCLA, October 2012.

[65] L. Wang, R. Wakikawa, R. Kuntz, R. Vuyyuru, and L. Zhang, “Data naming in vehicle-to-vehicle communications,” Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on, pp.328–333, IEEE, 2012.

[66] S. DiBenedetto, C. Papadopoulos, and D. Massey, “Routing policies in named data networking,” Proceedings of the ACM SIGCOMM workshop on Information-centric networking, pp.38–43, ACM, 2011.

[67] “Netcraft september 2012 web server survey.” `http://news.netcraft.com/archives/2012/09/10/september-2012-web-server-%survey.html`.

[68] “Bgp routing table analysis reports.” `http://bgp.potaroo.net/`.

[69] V. Jacobson, D. Smetters, N. Briggs, M. Plass, P. Stewart, J. Thornton, and R. Braynard, "Voccn: voice-over content-centric networks," Proceedings of the 2009 workshop on Re-architecting the internet, pp.1–6, ACM, 2009.

[70] M. Wittie, V. Pejovic, L. Deek, K. Almeroth, and B. Zhao, "Exploiting locality of interest in online social networks," Proceedings of the 6th International Conference on emerging Networking EXperiments and Technologies (CoNEXT), p.25, ACM, 2010.

[71] S. Scellato, C. Mascolo, M. Musolesi, and V. Latora, "Distance matters: Geo-social metrics for online social networks," Proceedings of the 3rd conference on Online social networks, pp.8–8, USENIX Association, 2010.

[72] S. Dharmapurikar, P. Krishnamurthy, and D. Taylor, "Longest prefix matching using bloom filters," Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, pp.201–212, ACM, 2003.

[73] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," Internet Mathematics, vol.1, no.4, pp.485–509, 2004.

[74] B. Ahlgren, M. Brunner, L. Eggert, R. Hancock, and S. Schmid, "Invariants: a new design methodology for network architectures," Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture, pp.65–70, ACM, 2004.

[75] A. Badam, K. Park, V. Pai, and L. Peterson, "Hashcache: Cache storage for the next billion," Proceedings of the 6th USENIX symposium on Networked systems design and implementation, pp.123–136, USENIX Association, 2009.

[76] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment, pp.231–236, ACM, 2002.

[77] M. Roughan, "Simplifying the synthesis of internet traffic matrices," ACM SIGCOMM CCR, vol.35, no.5, pp.93–96, 2005.

[78] "Routeviews prefix to as mappings dataset (pfx2as)." `http://www.caida.org/data/routing/routeviews-prefix2as.xml`.

[79] D. Levinthal, "Performance analysis guide for intel core i7 processor and intel xeon 5500 processors," Intel Performance Analysis Guide, 2009.

[80] Y. Wang, "A statistical study for file system meta data on high performance computing sites," Master's thesis, Southeast University, 2012.

[81] N. Rowe, “Automatic detection of fake file systems,” International Conference on Intelligence Analysis Methods and Tools, 2005.

[82] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, Introduction to algorithms, pp.241–243, MIT press, 2001.

[83] R. Bolla and R. Bruschi, “Pc-based software routers: high performance and application service support,” Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow, pp.27–32, ACM, 2008.

[84] M. Dobrescu, N. Egi, K. Argyraki, B.G. Chun, K. Fall, G. Iannaccone, A. Knies, M. Manesh, and S. Ratnasamy, “Routebricks: exploiting parallelism to scale software routers,” ACM SOSP, pp.15–28, ACM, 2009.

[85] “Ccnx.” http://www.ccnx.org/.

[86] Cisco Systems, “Cisco crs-3 forwarding processor card.” http://www.cisco.com/en/US/prod/collateral/routers/ps5763/CRS-FP-140_DS%.html.

[87] J. Shi and B. Zhang, “NDNLP: A Link Protocol for NDN,” tech. rep., The University of Arizona, 2012.

[88] “Rocketfuel: An isp topology mapping engine.” http://www.cs.washington.edu/research/networking/rocketfuel/.

[89] United Nations Statistics Division, “Demographic yearbook 2011, table 8,” 2011. http://unstats.un.org/unsd/demographic/products/dyb/dyb2011.htm.

[90] Ministry of Internal Affairs and Communications, “Traffic statistics of the internet in japan (in japanese),” September 2012. http://www.soumu.go.jp/main_content/000177422.pdf.

List of Research Achievement

Journal

- M. Fukushima, A. Tagami, and T. Hasegawa, “ Efficient Lookup Scheme for Non-Aggregatable Name Prefixes and Its Evaluation,” IEICE TRANSACTIONS on Communications, Vol.E96-B, No.12, pp.2953-2963, Dec. 2013.
- M. Fukushima, K. Sugiyama, T. Hasegawa, T. Hasegawa, and A. Nakao, “Minimum Disclosure Routing for Network Virtualization and Its Experimental Evaluation,” IEEE/ACM Transactions on Networking, vol.21, no.6, pp.1839-1851, Dec., 2013.
- M. Fukushima, H. Nakamura, S. Nomoto, and Y. Watanabe, “Modeling of seamless interworking environments for heterogeneous mobile systems,” IEICE TRANSACTIONS on Communications, vol. E89-B, no. 10, pp. 2885-2896, 2006.
- M. Fukushima and S. Goto, “Analysis of TCP flags in congested network,” IEICE TRANSACTIONS on Information and Systems, vol. E83-D, no. 5, pp. 996-1002, 2000.
- H. Khosravi, M. Fukushima, and S. Goto, “An Improved TCP Protocol Machine for Flow Analysis and Network Monitoring,” IEICE TRANSACTIONS on Communications Vol. E86-B, No. 2, pp. 595-603, 2003.
- Hiroki FURUYA, Masaki FUKUSHIMA, Hajime NAKAMURA, Shinichi NOMOTO, “Modeling of Aggregated TCP/IP Traffic on a Bottleneck Link Based on Scaling Behavior”, IEICE TRANSACTIONS on Communications Vol.E85-B No.9 pp.1756-1765, 2002.

Refereed Conference/Workshop

- M. Fukushima, A. Tagami, and T. Hasegawa, “Efficiently Looking Up Non-Aggregatable Name Prefixes by Reducing Prefix Seeking,” in Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on, pp. 3247-3252, 2013.
- M. Fukushima, T. Hasegawa, T. Hasegawa, and A. Nakao, “Minimum Disclosure Routing for Network Virtualization,” in 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2011, pp. 858–863.
- M. Fukushima, H. Nakamura, S. Nomoto, and Y. Watanabe, “Impact of mobility on traffic distribution in seamless interworking environments,” in Vehicular Technology Conference, VTC2004-Fall. IEEE 60th, 2004, vol. 6, pp. 4395 – 4401.
- Fukushima, M.; Nakamura, H.; Nomoto, S., "A playout-buffer-sensitive time slot scheduling for integration of real-time and elastic traffic in wireless networks," Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004. 15th IEEE International

Symposium on , vol.2, no., pp.1188,1192 Vol.2, 5-8 Sept. 2004.

- M. Fukushima and S. Goto, “Analysis of TCP flags in congested network,” in Internet Workshop, 1999. IWS 99, 1999, pp. 151–156.
- Sugiyama, K.; Fukushima, M.; Tagami, A.; Hasegawa, T., "Secure and traceable online image sharing," Global Communications Conference (GLOBECOM), 2012 IEEE , vol., no., pp.2089,2094, 3-7 Dec. 2012.
- Sugiyama, K.; Fukushima, M.; Hasegawa, T., "Anonymous communication for encouraging the reporting of wrongdoing in social groups," Secure Network Protocols (NPSec), 2010 6th IEEE Workshop on , pp.37-42, Oct. 2010.
- Koto, H.; Fukushima, M.; Nomoto, S.; Takahata, F., "Scheduling algorithm based on sender buffer backlog for real-time application in mobile packet networks," Wireless Communications and Networking Conference, 2005 IEEE , vol.1, no., pp.151,157 Vol. 1, 13-17 March 2005.

Others

- 吉田芳明、福嶋正機、北辻佳憲、田上敦士、阿野茂浩、” サービス合成可能なネットワークプラットフォーム”, 信学技報, vol. 113, no. 205, NS2013-77, pp. 27-27, 2013 年 9 月.
- 福嶋正機、長谷川輝之、長谷川亨、中尾彰宏、”ネットワーク仮想化環境における最小開示ルーティング方式の提案”, 電子情報通信学会ネットワーク仮想化研究会、北海道大学、2011 年 7 月.
- 福嶋正機、長谷川輝之、長谷川亨、中尾彰宏、”ネットワーク仮想化環境における最小開示ルーティング方式の提案”, 電子情報通信学会技術研究報告. NS（ネットワークシステム） 111(8), 1-6, 2011-04-14.
- M. Fukushima, T. Hasegawa, and T. Suda, “A new secure computing architecture based on anonymity in a network and its quantitative analysis”, NSF FIND PI meeting, April, 2009.
- 福嶋正機、中村元、”異種移動体システム間におけるシームレスインターワーク環境のモデル化”、 電子情報通信学会 2004 年ソサイエティ大会、2004 年 9 月。
- 福嶋正機、中村元、野本真一、”移動体異種システム間インターワーク環境でのモビリティの影響に関する検討”、電子情報通信学会２００４年総合大会、東京工業大学　大岡山キャンパス、2004 年 3 月.
- 福嶋正機、中村元、野本真一、 “Modeling of mobile environments with heterogeneous system-interworking”, 日本オペレーションズ・リサーチ学会　待ち行列研究部会 2003 年度（第 22 回）シンポジウム、ひこねステーションホテル、2004 年 1 月.
- 福嶋正機、中村元、野本真一、” 異種システム間インターワークを考慮したモバイル環境のモデル化”、 電子情報通信学会技術研究報告. IN, 情報ネットワーク 103(421), 59-64, 2003-11-07.
- 福嶋正機、中村元、野本真一、”移動体異種システム間インターワーキング環境性能評価モデルに関する検討”,電子情報通信学会 2003 年ソサイエティ大会、2003 年 9 月.

- 山本周、福嶋正機、中尾彰宏 、“ネットワーク仮想化基盤サービス設計ツール“, 電子情報通信学会　ネットワーク仮想化研究会、2012 年 11 月.
- 長谷川亨・福嶋正機・北辻佳憲・岡本修一・中尾彰宏、”サービス合成可能なネットワークプラットフォームの提案”、信学技報, vol. 112, no. 230, IN2012-75, pp. 7-12, 2012 年 10 月.
- 杉山浩平、福嶋正機、長谷川輝之、” 安全かつ追跡可能なオンライン画像共有方式の提案”,信学技報 IA, インターネットアーキテクチャ 111(81), 41-46, 2011-06-09.
- 杉山浩平・福嶋正機・長谷川輝之、”集団における問題の通報を促進する匿名通信方式とゲーム理論を用いた評価”、 信学技報, vol. 110, no. 79, ICSS2010-5, pp. 25-30, 2010 年 6 月.
- 杉山浩平、福嶋正機、長谷川輝之、” グループ間の早期警報を実現する匿名通信方式の検討”, 電子情報通信学会総合大会講演論文集 2010 年_通信(2), 34, 2010-03-02.
- 福嶋正機、長谷川輝之、長谷川亨、須田達也、”ネットワークの匿名性を利用して安全に計算を行うアーキテクチャの提案”, 新世代ネットワークワークショップ 2009.
- 岸洋司、福嶋正機、中村元、” 無線アクセス環境における加入者側トラヒック制御法に関する一検討”, 信学技報. RCS, 無線通信システム 106(478), 61-64, 2007-01-17.
- 福嶋正機・岸洋司・中村元、” 適応型レート制御を用いた加入者側トラヒック制御方式の提案”, 信学技報, vol. 106, no. 420, IN2006-124, pp. 61-65, 2006 年 12 月.
- T. Kitahara, M. Fukushima, Y. Kishi, H. Nakamura, “Traffic control system, traffic control method, communication device and computer program”, US Patent 7974203 B2.
- H. Nakamura, M. Fukushima, Y. Kishi, H. Koto, “Apparatus, method and computer program for traffic control”, US Patent 7944838 B2.
- 福嶋正機 、小頭秀行 、岸洋司 、中村元、”トークンバケットによるトラヒック制御装置、方法及びプログラム”, 特許登録番号 4577230.
- 中村元 、福嶋正機 、岸洋司、”トークンバケットによるトラヒック制御装置、方法及びプログラム”、 特許登録番号 4577220.