9-1-2012

Security Issues in Network Virtualization for the Future Internet

Sriram Natarajan
University of Massachusetts - Amherst, natarajan.sriram@gmail.com

SECURITY ISSUES IN NETWORK VIRTUALIZATION FOR THE FUTURE INTERNET

A Dissertation Presented

by

SRIRAM NATARAJAN

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2012

Electrical and Computer Engineering

SECURITY ISSUES IN NETWORK VIRTUALIZATION FOR THE FUTURE INTERNET

A Dissertation Presented

by

SRIRAM NATARAJAN

Approved as to style and content by:

Tilman Wolf, Chair

Weibo Gong, Member

Michael Zink, Member

Prashant Shenoy, Member

Christopher V. Hollot, Department Chair
Electrical and Computer Engineering

To My Parents and My Brother.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to Professor Tilman Wolf, for his guidance, support, constructive feedback, and encouragement. Under his mentorship, I was successful in tackling various problems that I faced during my dissertation and enhance my research skills. My heartfelt thanks to him for accepting me into NSL at a time when I was looking for a transition in my research career. Also, I thank Professor Weibo Gong, Professor Michael Zink, and Professor Prashant Shenoy for being part of my dissertation committee and providing valuable advice and help on my dissertation.

I would like to thank Dr. Shashank Shanbag and Vikram Desai for all the great discussions we have had to tackle some interesting research problems. I thank Dr. Xin Huang at Deutsche Telekom Laboratories, for giving me an opportunity to work with her for four very productive months. The experience that I gained during this summer internship has changed the way I look at research and has given me a great career path ahead. Also, I thank Ali Reza Sharafat at Stanford University for helping me out during this internship.

Finally, I am grateful to my parents and my brother for their encouragement and belief in me. I thank my brother for constantly backing me throughout my graduate studies. Without him this wouldn't have been possible.*Thanks Brother!*.

ABSTRACT

SECURITY ISSUES IN NETWORK VIRTUALIZATION FOR THE FUTURE INTERNET

SEPTEMBER 2012

SRIRAM NATARAJAN

B.E., ANNA UNIVERSITY, CHENNAI, INDIA

M.S., UNIVERSITY OF MASSACHUSETTS, AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Tilman Wolf

Network virtualization promises to play a dominant role in shaping the future Internet by overcoming the Internet ossification problem. Since a single protocol stack cannot accommodate the requirements of diverse application scenarios and network paradigms, it is evident that multiple networks should co-exist on the same network infrastructure. Network virtualization supports this feature by hosting multiple, diverse protocol suites on a shared network infrastructure. Each hosted virtual network instance can dynamically instantiate custom set of protocols and functionalities on the allocated resources (e.g., link bandwidth, CPU, memory) from the network substrate. As this technology matures, it is important to consider the security issues and develop efficient defense mechanisms against potential vulnerabilities in the network architecture.

The architectural separation of network entities (i.e., network infrastructures, hosted virtual networks, and end-users) introduce set of attacks that are to some extent different from what can be observed in the current Internet. Each entity is driven by different objectives and hence it cannot be assumed that they always cooperate to ensure all aspects of the network operate correctly and securely. Instead, the network entities may behave in a non-cooperative or malicious way to gain benefits. This work proposes set of defense mechanisms that addresses the following challenges: 1) How can the network virtualization architecture ensure anonymity and user privacy (i.e., confidential packet forwarding functionality) when virtual networks are hosted on third-party network infrastructures?, and 2) With the introduction of flexibility in customizing the virtual network and the need for intrinsic security guarantees, can there be a virtual network instance that effectively prevents unauthorized network access by curbing the attack traffic close to the source and ensure only authorized traffic is transmitted?.

To address the above challenges, this dissertation proposes multiple defense mechanisms. In a typical virtualized network, the network infrastructure and the virtual network are managed by different administrative entities that may not trust each other, raising the concern that any honest-but-curious network infrastructure provider may snoop on traffic sent by the hosted virtual networks. In such a scenario, the virtual network might hesitate to disclose operational information (e.g., source and destination addresses of network traffic, routing information, etc.) to the infrastructure provider. However, the network infrastructure does need sufficient information to perform packet forwarding. We present Encrypted IP (EncrIP), a protocol for encrypting IP addresses that hides information about the virtual network while still allowing packet forwarding with longest-prefix matching techniques that are implemented in commodity routers. Using probabilistic encryption, EncrIP can avoid that an observer can identify what traffic belongs to the same source-destination pairs.

Our evaluation results show that EncrIP requires only a few MB of memory on the gateways where traffic enters and leaves the network infrastructure. In our prototype implementation of EncrIP on GENI, which uses standard IP header, the success probability of a statistical inference attack to identify packets belonging to the same session is less than 0.001%. Therefore, we believe EncrIP presents a practical solution for protecting privacy in virtualized networks.

While virtualizing the infrastructure components introduces flexibility by reprogramming the protocol stack, it doesn't directly solve the security issues that are encountered in the current Internet. On the contrary, the architecture increases the chances of additive vulnerabilities, thereby increasing the attack space to exploit and launch several attacks. Therefore it is important to consider a virtual network instance that ensures only authorized traffic is transmitted and attack traffic is squelched as close to their source as possible. Network virtualization provides an opportunity to host a network that can guarantee such high-levels of security features thereby protecting both the end systems and the network infrastructure components (i.e., routers, switches, etc.). In this work, we introduce a virtual network instance using capabilities-based network which present a fundamental shift in the security design of network architectures. Instead of permitting the transmission of packets from any source to any destination, routers deny forwarding by default. For a successful transmission, packets need to positively identify themselves and their permissions to each router in the forwarding path. The proposed capabilities-based system uses packet credentials based on Bloom filters. This high-performance design of capabilities makes it feasible that traffic is verified on *every* router in the network and most attack traffic can be contained within a single hop. Our experimental evaluation confirm that less than one percent of attack traffic passes the first hop and the performance overhead can be as low as 6% for large file transfers.

Next, to identify packet forwarding misbehaviors in network virtualization, a controller-based misbehavior detection system is discussed as part of the future work. Overall, this dissertation introduces novel security mechanisms that can be instantiated as inherent security features in the network architecture for the future Internet. The technical challenges in this dissertation involves solving problems from computer networking, network security, principles of protocol design, probability and random processes, and algorithms.

TABLE OF CONTENTS

Page

CHAPTER

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1
INTRODUCTION

The current Internet has been vastly successful in achieving global connectivity between a large number of diverse networks, devices, and users. This success is due to the openness of the architecture and the general philosophy of allowing any system to communicate with any other system on the network. Such open standard has made it possible for everyone to communicate, offer service, make profit thereby creating an indispensable impact on our lives. As a result, the Internet architecture has seen the introduction of diverse end-systems (e.g., mobiles, laptops, smartphones, sensors, etc.) and related protocols to support user requirements. Also, with the introduction of massive influx of applications (e.g., video, financial transaction, shopping, entertainment, business) and communication mediums, it is estimated that the Internet has around 2.26 billion users (growth rate of 528.1% in the 11-year period between 2000 and 2011) connected to various application services around the world.

As of March 2012, Facebook has more than 901 million registered users, where more than 58% of the users log on to the service every day [50]. 72 hours of video are uploaded every minute on Youtube, where 3 billion hours of video are viewed each month. Last year, Youtube had a total of 1 trillion video views [132]. Recently, the amount of time spent accessing the Internet on mobile applications (81 minutes per day) have surpassed the total time spent (74 minutes per day) on desktops and laptops. While such numbers are heartening to see the wide variety of functionalities the Internet has provided, a major shortcoming with the architecture is the difficulty in providing inherent security guarantees.

1.1 Security in Current Internet Architecture

Reports from the US federal agency show a 650% increase in security violations in the last five years from 5503 cases in 2006 to 41776 in 2010 [79]. Network attacks have continued to compromise various business and government organizations ranging from attacking the Japan's Space Agency network [99] to Sony's playstation attack [98]. An illegal intrusion by computer hackers have compromised 20 years of student health information (1982 to 2002) from the University of Massachusetts Amherst's Health Service [112]. Statistics reveal that personal and medical data of around 8 million people were compromised during the last two years [114]. This continuous trend of network security breaches have not just affected personal computers and servers but have targeted smartphones and physical systems as well. Computer attackers eavesdropped on police radio communications using smartphone applications [100], while hackers were able to shutdown the Curran-Gardner public water facility in Springfield, Illinois, USA [101]. Experts believe a co-ordinated distributed denial-of-service (DDoS) attack was launched to bring down popular websites such as "Wikileaks" and "The Pirate Bay" which led to the disruption of their services for three days [133].

In addition to various security issues in the current Internet architecture, user privacy, identity theft, censorship of data, biased monitoring and traffic discrimination are prevalent [89]. While cryptographic protocols can protect the data content, the traffic can still reveal the communicating entities to ISPs in the network path [68]. Reports suggest that ISP's log user browsing data and sell sensitive information to advertisement companies [92], inject RESET packets to bring down P2P connections such as BitTorrent and Gnutella [46] thereby disrupting user activities. Such biased control by ISP's clearly compromises user privacy. While proxy servers and anonymous overlay networks [40] bring in some respite to the privacy issue, several attacks [1], [49] have shown to compromise such systems.

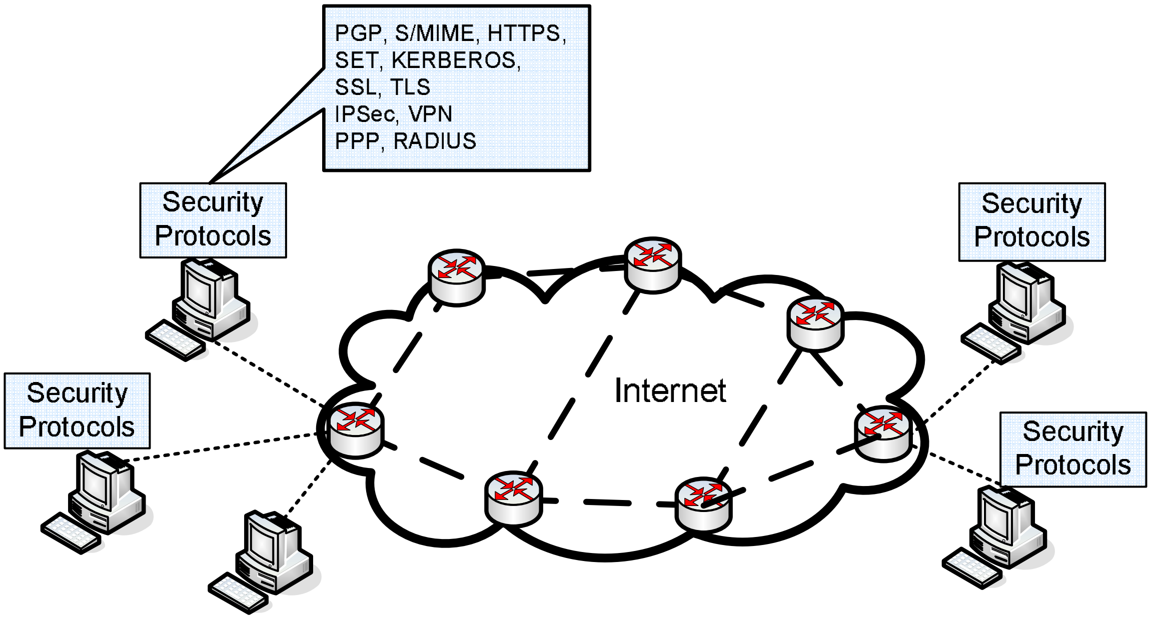

Figure 1.1. Security in Current Internet Architecture.

To address some of the above attacks and vulnerabilities various security features are supported as point solutions in the Internet architecture. Figure 1.1 shows the set of security extensions added to the Internet protocol stack. To provide authentication, confidentiality, integrity, and availability, a number of additions have been developed. These approaches range from cryptographic operations on end-systems and routers (e.g., SSL, VPN) to dedicated traffic monitoring and access control (e.g., firewalls, intrusion detection systems, intrusion prevention systems) to defenses against denial of service (DoS) attacks (e.g., anomaly detection, rate limiting). While such extensions provide point solutions in the defense against specific attacks, it has been difficult to envision how novel security mechanisms can be deployed at an architectural level. Also, there are needs for more security support for new communication paradigms and domain-specific networks such as financial transactions, military communication, and remote medical procedures. For many of these new communication domains, specialized protocol suites have been developed. Due to this specialization, it is not expected that a single protocol stack can satisfy all the needs of a future Internet.

Therefore, to provide inherent security guarantees it is necessary to design an entirely new network architecture that can overcome the shortcomings of the current Internet.

1.2 Network Virtualization Architecture

Next-Generation Internet architectures are currently being explored in the networking research community [52]. Network virtualization is a key technology that is necessary to support diverse protocol suites in the future Internet [52], [8], [51], [36], [111]. The idea of virtualizing the network components has gained significant attention in the context of networking testbeds. This facilitates researchers to evaluate new ideas and test experiments/protocols in realistic scenarios GENI [77], Planetlab [88], Emulab [113], VINI [17]. As a result, virtualization in a shared testbed proved to be successful in overcoming the limitations and complexities of individual physical testbed. To extend the success of virtualized testbeds to realistic network scenarios, [51] proposes a network architecture that separates the role of Internet Service Providers (ISPs) into Infrastructure Providers (managing physical infrastructure) and Service Providers (running customizable network protocols and services).

Network virtualization introduces flexibility to the Internet ossification by separating the functionalities into the following entities:

- Network Infrastructure (NI): provides the physical components required to setup the network (e.g., routers and links). NI efficiently allocates the required network bandwidth and physical resources (device CPU and memory) for each virtual network, ensuring proper resource isolation between them. Related work has explored algorithms for mapping (i.e., resource allocation) [37] and router designs to support virtualization [8, 126].

- Virtual Networks (VN): deploy customizable network protocols by leasing the required infrastructure resources from multiple NIs. Each virtual network is a

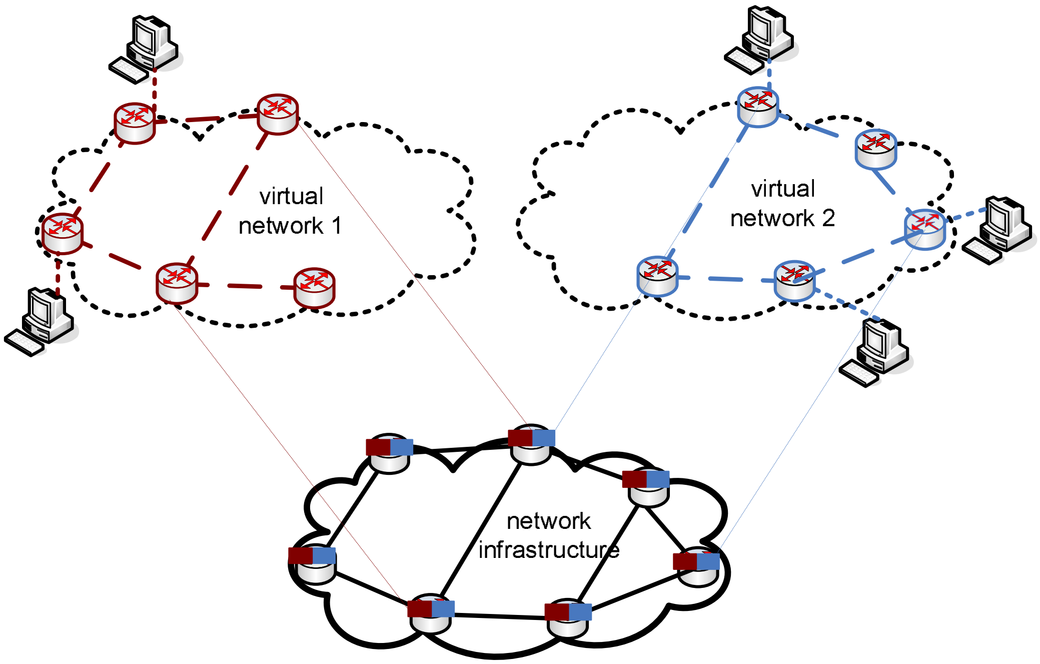

Figure 1.2. Virtualized Network Infrastructure.

combination of multiple virtual routers and links. When initiating a service, the VN confines to Service Level Agreements (SLA) with set of NIs and receives the requested resources. Each VN then instantiates the service (e.g., novel network protocol) on the allocated resources to form a virtual network topology by connecting end-users to the network.

- End-Users: are similar to the current Internet architecture but have the opportunity to choose from multiple virtual network services.

For any virtual network, the above architectural separation reduces the cost involved in setting up the physical resources and reduce maintenance costs.. Each virtual network can provide its users with a custom set of protocols and functionalities (e.g., security features). Figure 1.2 shows two virtual networks sharing the network infrastructure resources. Both VNs deploy their customized network services on the shared infrastructure components and establish end-to-end connectivity between end-users. Once deployed, each VN can then operate the control plane functionalities on the NI resources and direct the NI to perform the required data plane features (e.g., packet forwarding).

Despite the various advantages, hosting multiple virtual networks on a shared network infrastructure introduces new security challenges unlike seen in the current Internet. The VN cannot assume inherent provision of security features by the hosting NI and is oblivious to the malicious activities of the infrastructure. In addition, with the infrastructure resources being shared among multiple virtual networks it presents an opportunity for attackers to co-host malicious network services and attack the legitimate VNs. For the NI, the hosted virtual networks should not launch attacks or access privileged information on the infrastructure. Therefore, to understand the possible security issues in detail this dissertation focuses on identifying the attacks and vulnerabilities that are unique to the virtualized network infrastructure environment. For a successful adaptation of the technology for the future Internet, it is important to address the security issues with effective defense mechanisms.

1.3 Security in Network Virtualization

Some of the security challenges that the proposed network virtualization architecture should address are listed below:

- **Privacy and Confidentiality:** The network infrastructure can introduce biased management practices, snoop confidential information, or launch hidden attacks on the hosted virtual networks. Hence, a secure packet processing methodology (e.g., confidential packet forwarding) should be provided by the NI component with certain level of data transparency (a mechanism to process the packets without exposing the input data) between the hosted VNs and the NIs.

- **Trust:** To setup end-to-end network connectivity, the virtual network service should partner with multiple infrastructure providers with varying levels of agreements and requirements. This requires that the virtual network should trust multiple competing network infrastructures to establish global connectivity. Therefore the architecture should introduce a trustworthy interface or management framework among the network entities.

- **Authorization & Availability:** The architectural separation of virtual networks and network infrastructures provide additive opportunity for attackers to exploit vulnerabilities in both the network services and network components to launch attacks. This can reduce the availability of network resources. Also, with the infrastructure resources being shared among multiple virtual networks it presents an opportunity for attackers to co-host malicious services and attack the legitimate VNs. Therefore, the architecture should ensure only authorized traffic is transmitted and attack traffic is curbed as close to the source as possible.

- **Accountability:** The problem of accountability is to identify if the underlying virtual network service and the infrastructure components are providing the guaranteed/promised level of services. The network entities are subjected to both external and internal attacks and hence it is in the interest of all participating entities to consider the problem of accountability (e.g., violations, anomalies, tampering of resources, deviations from the expected behavior, etc.) in the virtual network provisions.

- **Isolation:** Since multiple virtual networks are co-hosted on a shared infrastructure resource, the architecture should consider effective isolation (e.g., IP address space virtualization, layer-2 addressing, protect network configuration information) between the virtual network resources. Also, for the network infrastructure, the hosted virtual networks should not launch attacks or access privileged information on the infrastructure.

1.4 Challenges and Requirements

This dissertation explores some of the fundamental security issues in the network virtualization architecture by considering a thorough understanding of the attacks and vulnerabilities and develop a secure network virtualization platform for the future Internet. The following technical challenges and open questions are addressed in this dissertation:

- Network Infrastructure:
    - Can the network infrastructure perform packet processing functionality without gaining any information on the operation of virtual networks?
    - How can the network infrastructure ensure that the hosted virtual networks does not introduce any malicious activities on the physical resources?
- Virtual Network:
    - Can the virtual network trust the underlying network infrastructure?
    - Can we have a virtual network instance that guarantees effective security provision for end-users and prevent unauthorized network access to isolate attack traffic?
    - What security features are required for the virtual networks to host their services (e.g., network protocols) on third-party network infrastructures?
- User:
    - What level of security can the user expect from the network virtualization architecture? How different is this from the current Internet?

- Can the user customize/choose the required security functionality for each network connection?
- How can the user identify if the guaranteed level of security is provided by the network services and the infrastructure components?
- How can the user hold the network entities accountable for any security violation?

1.5 Contributions

This dissertation introduces the following security features in the network virtualization architecture:

1. **Design a Confidential Packet Forwarding Technique**: Any honest-but-curious NI component can snoop on traffic sent by the virtual networks. Such activities can identify the communicating entities (source and destination addresses) in each packet, what routes are used inside a virtual network, etc. Revealing such information is undesirable since it compromises the privacy and confidentiality of both users and virtual networks. Therefore, we require a technique that avoids such inferences about traffic while allowing the NI components to perform the data plane functionalities. This introduces the need to forward packets based on the encrypted forwarding address information. The specific contributions of this work are:

   - Identification of the anonymity and user-privacy issue in the context of network virtualization architecture. Virtual networks are hosted on third-party infrastructures and it is in the interest of both users and virtual network providers to protect the operational information (e.g., traffic characteristics, routing information) of the network from snooping infrastructure providers.

- An efficient design of the proposed EncrIP technique to solve the above problem. To provide a practical, confidential packet forwarding functionality, EncrIP uses a combination of prefix-preserving and probabilistic encryption technique. This ensures the effectiveness of the system against statistical inference attack as well as allows the use of conventional forwarding lookup technique.
- A quantitative security analysis of the proposed system with specific details on attack probability, space overhead, and security-space tradeoff introduced in the system
- Prototype implementation and experimental evaluation of the proposed technique. The EncrIP technique was implemented in ProtoGENI experimental testbed to measure the probability of successful attack and measure the performance overhead (i.e., speed of the cryptographic computations per packet, memory cost on edge routers, prefix overhead on core routers) introduced by the technique.

2. **Design a Capabilities-Based Virtual Network Instance**: A secure virtual network instance that introduces a high-performance capabilities-based network to reduce attack traffic (e.g., to prevent unauthorized network access, attack traffic injection, and isolation of attack traffic) and allow only authorized network access. The proposed network architecture requires every router in the network to validate packets to ensure 1-hop containment of malicious traffic and thus avoid the consumption of network resources as this traffic is forwarded to its target. The specific contributions of this work are:

    - A design of a deny-by-default architecture that uses data path credentials to verify packet permissions on every hop.

- An efficient design of data path credentials based on Bloom filters that can be used for high-performance networks. These credentials are of constant size (independent of the path length) and difficult to generate (and thus difficult to fake), but computationally simple to verify for high-speed forwarding.

- A quantitative study of security guarantees that can be provided by such an architecture. A detailed security analysis of how well data path credentials can defend against attacks in unicast, multicast, and network coding settings is also provided.

- Results from a prototype implementation of the proposed protocol on Emulab that demonstrate the effectiveness of the proposed approach in defending against denial-of-service attacks and that show limited overhead and performance degradation from using credentials. The protocol header was implemented between the network layer (i.e., IP) and transport layer. Experimental nodes running Linux Kernel (2.6.21.4) were modified to support the processing of the proposed data path credentials header.

1.6 Organization

The rest of the dissertation is organized as follows: To understand the possible security issues in detail, a detailed description of the security requirements, challenges, attacker capabilities, and possible attack scenarios (attacks and vulnerabilities that are unique to the virtualized networks) in each entity (i.e., network infrastructures, virtual networks, and end-users) in the architecture is discussed in Chapter 2.

Chapter 3 presents a confidential packet forwarding technique that encrypts forwarding address information of the communicating entities and provide a mechanism for infrastructure components to forward packets based on the encrypted address. The proposed encryption function introduces a combination of prefix-preserving and probabilistic encryption technique that ensures the anonymity and data confidentiality of end-users connected to virtual networks (e.g., forwarding addresses, network topology, routing information) from the network infrastructure components. Also, a detailed security analysis and experimental evaluation of the proposed technique is discussed in this chapter.

Chapter 4 presents a capabilities-based virtual network instance that introduces a robust, security-enhanced network architecture. The proposed architecture introduces “data path credentials” which focuses on authorization and availability of legitimate network traffic, by ensuring that only packets that have been positively identified are forwarded in the network. This chapter elaborates the level of security achieved by the capabilities-based network protocol by providing detailed security analysis and experimental evaluation of the proposed architecture.

Chapter 5 summarizes the contribution of this dissertation and discusses the future work. Related work for each of the proposed system is discussed in each chapter.

CHAPTER 2

SECURITY ISSUES IN NETWORK VIRTUALIZATION

In this chapter, we discuss possible security issues in each entity in the architecture by discussing the security model and possible attack scenarios that exhibit some of the security concerns in network virtualization. Figure 2.1 shows the possible combinations in which attacks can compromise different entities in the architecture. For example, (2) indicates an attacker launching attacks (e.g., denial-of-service attack) on the network infrastructure, (3) indicates a scenario when a malicious VN service launches attacks on the end-users, while (7) indicates a network infrastructure entity snooping the virtual network traffic. In the following subsections, we elaborate each combination of attacks shown in Figure 2.1 with possible attack scenarios.

2.1 Users

Various network security issues and related defense mechanisms have been proposed to protect end-systems. However we focus on attacks originating from a malicious virtual network or from a vulnerable network infrastructure that compromises the user.

2.1.1 Security Requirements

The basic security requirement for end-users is to ensure that attacks should not modify the working of the end-system while being able to identify and discard attack traffic.

2.1.2 Attacker Capabilities

The following attacker capabilities define the possible attack scenarios that can be launched on the end-users:

- An attacker can send attack packets to compromise or modify a specific functionality on the end system.
- An attacker can sniff confidential information by introducing statistical inferences on user traffic.
- An attacker can launch a flooding attack to send continuous network traffic and throttle the network bandwidth of the end-user disrupting access to legitimate network service.
- An attacker cannot physically access the end-system but can initiate remote based attacks.

2.1.3 Attack Scenarios

Here we discuss potential vulnerabilities and attacks that can be launched on the end-users by the networking entities.

NI attacks on User: A compromised network infrastructure can selectively drop/modify packets belonging to particular sender or group of senders [12, 39]. The attacker could choose to drop a packet within a particular time window, thereby forcing the sender to reduce their sending rate as they perceive congestion. The attacker could selectively drop queued packets exploiting congestion control protocol at the senders. The VN and the sender are completely unaware of the malicious activity of the NI and hence are subjected to reduced quality of service provision. In chapter 3, we discuss one possible solution to address the data privacy and confidentiality issue (to address unauthorized snooping of traffic) when hosting virtual networks on third-party network infrastructures.

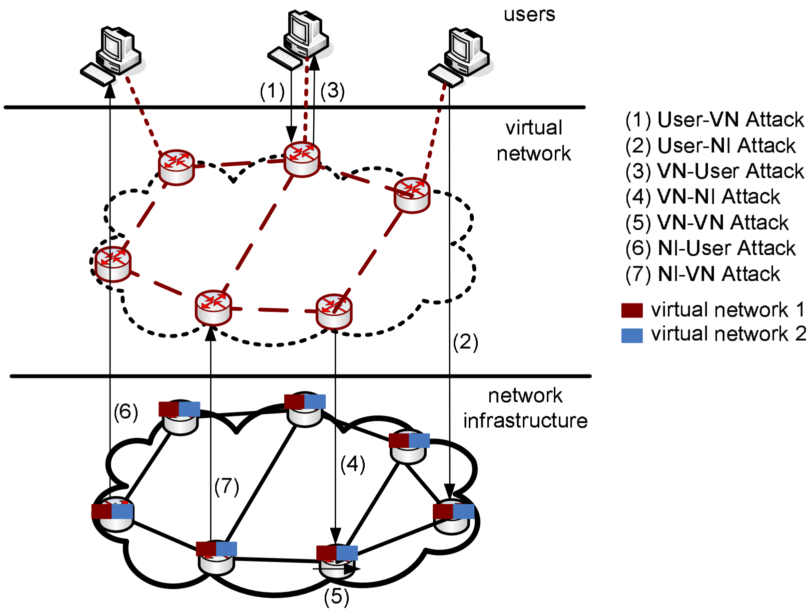

Figure 2.1. Potential Attacks within Virtualized Network.

VN attacks on User: To control network congestion and maintain the promised network access, the VN could introduce protocol specific interference by injecting forged packets to disrupt the legitimate connection. Recent activities by Comcast to inject TCP RESET packets on file sharing protocol connections disrupted user activities bringing down P2P connections such as BitTorrent and Gnutella [46]. Rather than introducing dynamic traffic shaping mechanisms, the company blocked traffic corresponding to P2P protocols by sniffing protocol headers and injecting forged packets to reset the connection, leading to the Net Neutrality debate [54]. Such practices exhibit the level of control the virtual network has on the user traffic, introducing various security issues.

2.2 Virtual Networks

Virtual Networks (VN) can be targeted by attacks generated from the underlying infrastructure (NI), the co-hosted VNs or the users connected to the VN. In this section, we discuss the security model for the hosted VNs explaining the security requirements, attacker capabilities, and attack scenarios.

2.2.1 Security Requirements

To ensure correct protocol processing of the hosted VNs we assume the following security requirements:

- NIs should not snoop or monitor traffic associated to the hosted VN.
- NIs should not modify legitimate traffic or inject malicious traffic that disrupts the working or functionality of the hosted VN.
- A co-hosted VN should not launch side-channel, timing attacks on a vulnerable VN.
- Users should not be able to intrude and modify the functionality of the VN by taking advantage of programmability.
- An inherent access control mechanism should ensure the security of privileged information stored in VN.

2.2.2 Attacker Capabilities

The attacker capabilities that can compromise the hosted VN are:

- An attacker can instantiate a malicious protocol function to modify the normal functionality of the virtual network.
- An attacker can sniff the state of the shared physical resources (e.g., memory cache, CPU) on the network infrastructure to attack the co-hosted VNs.
- An attacker can intentionally modify or selectively manipulate the data traffic associated with a particular VN.

2.2.3 Attack Scenarios

Here we discuss potential vulnerabilities and attacks that can be launched on the VN by illustrating each case with an attack scenario.

NI attacks on VN: Network infrastructures and virtual networks operate under certain service level agreement (SLA). However, a compromised NI can modify the hosted VN information (e.g., network protocols), violate the promised level of services or compromise the operation of the hosted virtual network. This increases the trust and accountability issues when hosting VN on third-party infrastructures, as discussed in [61]. NI can also indulge in biased management practices (unauthorized snooping of traffic) by introducing hidden VN monitoring activities on the network traffic, thus violating user privacy and confidentiality. A solution to this problem is discussed in Chapter 3.

VN attacks on co-hosted VN:

- Network virtualization projects such as [17], [51] propose that the logical isolation between the hosted virtual networks significantly improves the secureness of the system by providing better control and manageability. On the contrary the isolation of resources can lead to entirely new set of network attacks. An attacker could take advantage of the shared infrastructure platform by leasing portion of resources to assess the vulnerabilities and functionalities of the co-hosted VNs. The vulnerable VN could be another competing virtual network provider running a specific service. Once the attacking VN is instantiated, it takes advantage of the placement and launches a cross-VN side channel attack to steal information from the vulnerable VN. An example of such an attack was exhibited in the Amazon EC2 cloud service by [95] by launching cross virtual machine side channel attacks. In our work, we perceive similar attacks can be launched on the co-hosted virtual networks that share the same platform.

- Another example for co-hosted virtual networks causing network configuration integrity problem is discussed in the following example. To accommodate fine-grained control, modern switching substrates (e.g., OpenFlow [70]) maintain forwarding information for each active flow in a flow table. A separate control plane manages flows within a subnetwork by updating this flow information within switches. When using network virtualization, a technique that allows sharing of networking resources among different logical networks, the physical switch and its flow table need to be shared [102]. To maintain configuration integrity, it is essential to implement an effective isolation mechanism for the flow table between virtual network slices [74].

  Current OpenFlow-based switches do not provide flow table isolation in hardware and thus lead to hidden conflicts and misconfiguration of flows. Instead, OpenFlow handles the flow conflict problem by assigning a priority to each flow table entry. When using multiple controllers, effective isolation cannot be achieved with priorities since individual controllers may make modifications to the flow table that lead to hidden flow conflicts (e.g., a flow entry with higher priority shadows another flow entry, flow conflicts [74], [5] or modifying the flow entry that compromises some security-based flow rules [87]). As a result, OpenFlow networks may exhibit network-level routing that is inconsistent with the view of each VN thereby introducing security violations. One practical solution to this problem is discussed in [74].

User attacks on VN: To reduce the complexity of network management of virtual networks, [118] suggests an interesting solution to provide a live router migration technique, transferring the control plane information (network protocol binaries and configuration files) and re-instantiating the data plane state in the new physical router platform. This approach is similar to the live virtual machine migration technique introduced in [38]. During migration of the virtual network state, an attacker sniffing the network traffic can launch a Man-in-the-Middle (migration) attack to eavesdrop the contents of the VN and other confidential information. An example of such an attack in the context of live virtual machine image migration was shown in [80].

2.3 Network Infrastructure

The network infrastructure is vulnerable to attacks originating from the hosted virtual networks or users associated with them. In this section we define our security model explaining the security requirements, attacker capabilities, and attack scenarios with respect to the network infrastructure.

2.3.1 Security Requirements

For a correct functioning of the network infrastructure we assume the following security requirements:

- The hosted VN should not tamper with the allocated NI resources to gain control of the infrastructure.
- NI should ensure complete isolation of physical and network resources between co-hosted virtual networks.
- Legitimate traffic should be processed without any interference, while malicious network traffic should be inferred and discarded.
- NI should support effective access control mechanism to protect from extraction of secret information stored in the infrastructure.

2.3.2 Attacker Capabilities

The following attacker capabilities define the possible attack scenarios that can be launched on the network infrastructure:

- An attacker can send arbitrary data and control packets to flood the network and bring down the NI.
- An attacker can asses the vulnerabilities of the infrastructure from the allocated resources to intrude and take control of the entire infrastructure.
- An attacker cannot physically access the equipments but can initiate remote based attacks.

2.3.3 Attack Scenarios

The following attack scenarios exhibit possible vulnerabilities in the network infrastructures:

User attacks on NI: Virtual network providers require flexibility in customizing their service. Modern routers use general-purpose programmable packet processors to reprogram the router functionality [44]. This feature however introduces new vulnerabilities thereby compromising the network infrastructure. With the introduction of programmability in packet processors, code exploits such as buffer overflows, integer vulnerabilities can introduce security issues. An attacker could inject a data packet that takes advantage of the code vulnerability of the hosted virtual network and modify the operation of the packet processor leading to a denial-of-service attack [32]. This scenario is specific to the customization functionality introduced by VN that can compromise NI components. Hence a secure programming paradigm is required when instantiating the virtual network service by the network infrastructure. A solution to curb attack traffic originating from unauthorized users is discussed in Chapter 4.

VN attacks on NI: A malicious VN can be motivated to attack the infrastructure to disrupt the services hosted by a competing VN. The hosted platform gives extra opportunity to asses the vulnerabilities of the infrastructure, identify router bugs [62] to launch a flooding attack on the network and physical resources of NI. This can bring down the entire network infrastructure, eventually breaking the co-hosted VN. Another scenario is when the attacker wishes to reproduce some features associated to a co-hosted VN service, can manipulate the configurations of NI by extracting secret information and eavesdrop on the hosted VN traffic. An example could be a live video streaming service that can be eavesdropped, reproduced and redirected to a set of unauthorized users.

2.4 Related Work

Network virtualization addresses the Internet ossification issue by providing a shared, customizable platform to host diverse protocol suites, as shown in [8, 51, 52]. Several widely used Internet testbeds (e.g., GENI [77], Emulab [113], Planetlab [88]) use virtualization to separate different experiments and thus demonstrate that such technology is feasible and practical. However, unlike the current Internet architecture, the technology introduces new class of security issues [75]. Modern router designs that support network virtualization require an embedded packet processing platform that can perform custom packet processing for virtual networks that are deployed at runtime [110], [131]. Packet processors in these systems are often implemented using embedded multi-core network processors [120].

The problem of hosting network protocols and services on third-party infrastructures raises serious questions on the trustworthiness of the participating entities. Reference [115] shows the list of ISPs that introduce hidden traffic shaping techniques on peer-to-peer protocols. Such activities indicate the requirement to examine security issues, when hosting virtual networks on the network infrastructures. Information leakage in virtualized network infrastructures are analogous to the cloud computing paradigm. A side channel attack that extracts secret information by targeting co-hosted virtual machines in Amazon EC2 service was shown in [95]. Reference [35] suggests a denial-of-service attack can be launched on the physical network which can bring down all hosted virtual networks.

To address accountability in hosting virtual networks, violation detection using processor extensions and network measurements to monitor service-level agreements are introduced in [61]. Minimum disclosure routing [55] addresses the lack of operational confidentiality when hosting virtual networks on third-party network infrastructure. The technique extends Secure Multiparty Computation [29] to allow NI providers to compute the routing decisions using only local routing information. To identify trustworthy infrastructure providers based on their past experiences and feedback, [71] proposes a trust management framework between network entities. A secure packet processing system that monitors the instruction level operations of the packet processor (to detect malicious attacks) in the NI component was proposed in [124]. Allowing virtual networks to customize the allocated resources by introducing programmability can lead to the introduction of malicious code on the router [33]. Solutions to this problem have been proposed using techniques from embedded system security [32].

2.5 Summary

In this chapter, we discuss some of the security issues that can arise in the network virtualization architecture. To address some of the above challenges, we discuss possible defense mechanisms that guarantee the following security principles: Confidentiality and Privacy, Authorization and Availability, and Accountability of virtual network services. In the subsequent chapters, we discuss two security mechanisms that are required to be included in the network virtualization architecture to ensure that security is not an after thought process, but an integral part of the network virtualization architecture.

CHAPTER 3
CONFIDENTIAL PACKET FORWARDING

In this chapter, a confidential packet forwarding functionality is proposed that protects user privacy when network traffic is forwarded by third-party network infrastructures. In a typical virtualized network, the infrastructure and the virtual network instance may be managed by different administrative entities that may not trust each other. In such a scenario, the virtual network operator might hesitate to disclose network configuration or control information (e.g., source and destination addresses of network traffic, routing information, etc.) to the infrastructure provider. However, the network infrastructure provider does need sufficient information to implement the packet forwarding functionality within the virtual network. Therefore, it is important to develop mechanisms that protect a virtualized network's operational information, while allowing an efficient implementation on the network infrastructure. Some initial ideas on this topic have been published in prior work, which includes an introduction to the proposed encryption technique [75], [76].

The ability to share physical resources among multiple virtual networks presents advantages, including lower operational costs by avoiding the need to maintain separate physical resources. However, the use of virtualization also raises an interesting security problem. Any honest-but-curious network infrastructure provider may snoop on traffic sent by the virtual networks. Such snooping can reveal information about which end-systems communicate with each other, what routes are used inside a virtual network, etc. Note that this information is visible to the NI provider even if cryptographic protocols are used by the end-systems since in existing networks *end-*

system addresses and forwarding tables have to be visible in order to perform packet forwarding. Revealing such operational information may be undesirable for a virtual network, in particular in large-scale networks that span multiple infrastructure providers with whom establishing mutual trust relationships is difficult or impractical. Therefore, this work aims to provide a solution to provide privacy for virtual networks in network virtualization.

The main idea of this work is to encrypt network addresses before they enter the network infrastructure and decrypt them when they leave. Conventional cryptographic techniques cannot be employed directly since the encrypted addresses could not be used by a router's forwarding engine. Therefore, we need to address three fundamental questions in our system:

- How can we encrypt IP addresses such that inferences about traffic (e.g., the same sources-destination pair) can be avoided?

- How can we encrypt IP addresses such that packet forwarding with conventional routers (i.e., using longest prefix match) in the network infrastructure is possible?

- How can we encrypt IP addresses such that the existing Internet Protocol (IP) version 4 header can be used? (In our work, we focus on IPv4 so that we can demonstrate our technique in a real network environment.)

As an answer to these questions, we present Encrypted IP (EncrIP), a protocol that uses probabilistic encryption in a prefix-preserving manner to hide source and destination information while still permitting packet forwarding using longest prefix match. Using EncrIP, network infrastructure providers can forward packets without gaining insights into the internal operation of virtual networks. The specific contributions of our work are the following:

- Design of encryption method that utilizes probabilistic encryption and prefix-preservation to hide address information while allowing efficient packet forwarding.
- Prototype implementation of EncrIP on GENI using an unmodified IP header.
- Evaluation of resource requirements and security characteristics of EncrIP.

Our results show that EncrIP can be implemented using only a few MB of data on gateways at the edge of the virtual network. Forwarding in the virtual network itself can be performed without overhead. Our results show that the success probability of a statistical inference attack, trying to identify which packets belong to the same source-destination pair, is less than 0.001%. We therefore believe that EncrIP presents an effective solution to providing privacy in virtualized networks. We begin our discussion by first discussing the existing privacy schemes and compare them with the proposed EncrIP technique.

3.1 Background

The problem of hosting network protocols and services on third-party infrastructures raises questions on the trustworthiness of the participating entities. Considering the “honest-but-curious” model between the network entities, the VN does not want to expose the data packet (header and payload) when processed by the NI. To put EncrIP into context, we review other existing techniques for privacy and anonymization in networks and compare them in Table 3.1:

- IPsec [63]: IPSec establishes a secure tunnel between gateways using the Encapsulating Security Payload (ESP) protocol suite. The entire packet (including IP header) is encapsulated into another packet with unprotected source and destination IP addresses. VPNs provide the required security and privacy to end-users by establishing a virtual private connection over the public Internet. A secure VPN based on IPSec, SSL/TLS uses a combination of cryptographic functions and tunneling mechanism to provide data confidentiality, data integrity and authentication of end-users. While the technique provides effective user privacy and payload encryption it incurs high processing overhead and requires session maintenance using Internet Key Exchange (IKE). In addition, VPNs are vulnerable to location privacy and are subjected to government crackdown in some countries [78] wherein DNS poisoning was introduced to bring down some popular VPN services. Also, when using a third-party VPN provider's service, the user should trust the VPN provider, who might log user behavior thereby compromising user privacy and confidentiality.

- Tor [40]: Tor is a distributed overlay network that provides strong anonymity guarantees against a global eavesdropper by constructing a network of virtual tunnels that establish a private network path. The system establishes a random pathway using intermediate, trusted Tor relays to provide the required anonymity to the end-user. The connection setup requires a series of cryptographic exchanges between the user and each intermediate Tor relay node in the path thereby ensuring the reduction of traffic analysis performed by adversaries. Unlike other anonymity systems, Tor can hide the originating source from the participating relays as well, since each relay can see no more than one hop in the path.

  While Tor promises to be an effective solution in solving some of the anonymity requirements (e.g., traffic analysis performed by ISPs, government class) in the current Internet, there are various performance issues associated with this system (cryptographic computations involved at each relay node and increased latency overhead [82]), requires end-system software support, and is subjected to government crackdown in some countries [105,106]. In addition, [66] exhibits how Tor users can be traced and profiled when used in conjunction with Bittorrent, peer-to-peer application.

- Address Hiding Protocol (AHP) [90]: AHP hides the source address of traffic in cooperation with the ISPs, similar to "caller ID blocking" in telephone systems. AHP uses a tweakable block cipher based encryption function to encrypt the IP addresses at trusted gateway routers. While the technique has low overhead, it can only guarantee source anonymity (no receiver anonymity and no location privacy) and can cause collisions in long-lived flows. The technique introduces the notion of two IP addresses *hidden* and *sticky* address to enable the feature to work with certain peer-to-peer applications. AHP has resemblance to [107] which provides location privacy in IPv6 addressing.

  As explained in Chapter 2, the role of ISPs are envisioned to be separated into VN and NI providers. In doing so, it is important to identify the role of each network component in the architecture. The EncrIP technique that we introduce in our work has similar goals with the AHP protocol, however, we introduce a system that is suitable for the network virtualization environment by providing the required anonymity to end-users from third-party network infrastructure providers.

- Lightweight Anonymity and Privacy (LAP) [58]: LAP allows end-systems to establish an encrypted path and forward packets on that path with low latency overhead. With such a setting, the technique addresses the tradeoff between the privacy levels achieved and the performance overhead introduced in the system. LAP extensively discusses the technical requirements in achieving the desired privacy and elaborates deployment considerations to work with different next-generation network architectures. To provide backward compatibility with the current Internet, LAP introduces the notion of heterogenous network that

Table 3.1. Comparison of Existing Privacy Techniques with EncrIP.

	VPN [63]	Tor [40]	AHP [90]	LAP [58]	AnonyFlow [72]	EncrIP
Adversary	eavesdropper	eavesdropper	end-system	end-system	end-system	eavesdropper
Source anonymity	yes	yes	yes	yes	yes	yes
Receiver anonymity	yes	yes	no	yes	no	yes
Location privacy	no	yes	no	yes	yes	yes
Session unlinkability	yes	no	no	yes	no	yes
Payload encryption	yes	yes	no	no	no	no
Optimal routing	yes	no	yes	no	yes	yes
Commodity routers	yes	no	yes	no	no	yes
Processing overhead	high	high	low	low	low	low
Implementation	gateways	overlay	gateways	network	gateways	gateways

combines LAP enabled and legacy autonomous domains and tries to achieve the desired security and performance. However, the technique considers a receiver attacker model and supports session unlinkability by requesting new encrypted paths for each session.

- AnonyFlow [72]: AnonyFlow proposes an in-network anonymization service (in cooperation with a trusted network provider) that hides the source identifier from the receiver. The technique addresses deployment challenges (e.g., support long-lived flows, multiple ingress/egress points) by maintaining a centralized global service. AnonyFlow does not provide receiver anonymity or session unlinkability and requires specialized switches [70].

Comparing EncrIP to the other approaches for privacy, it is the only system that can be implemented with commodity routers and still achieve location privacy. While IPsec is most similar to EncrIP, the processing overhead for EncrIP is considerably lower as we show in our results. Therefore, EncrIP presents a unique high-performance privacy solution that provides session unlinkability.

3.2 Confidential Packet Forwarding

The problem of confidential packet forwarding in virtualized networks is illustrated in Figure 3.1. When a virtual network is used to connect multiple subnetworks (e.g., corporate campuses, etc.), the traffic sent via the network infrastructure can be

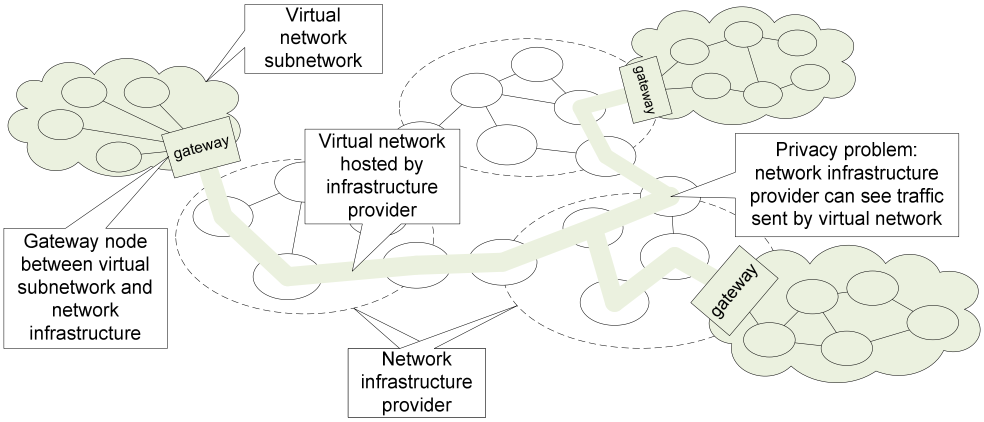
Virtual network subnetwork
gateway
gateway
gateway
Virtual network hosted by infrastructure provider
Privacy problem: network infrastructure provider can see traffic sent by virtual network
Gateway node between virtual subnetwork and network infrastructure
Network infrastructure provider

Figure 3.1. Privacy problem in virtualized networks.

seen by the network infrastructure provider. In our work, we introduce a “gateway” that encrypts network addresses so that the infrastructure provider no longer can determine which end-system is communicating with which other end-system. The presented approach can achieve this privacy more efficiently than IPsec and other approaches and does not require any additional headers.

3.2.1 Security Requirements

To explicitly state the goals of our system and our assumptions about the attacker, we briefly review our security model. The requirements for a confidential packet forwarding system are:

- Address privacy: The identity of the communicating entities in the virtual network should be concealed in the network infrastructure. In our case, this means that the IP addresses of the source and the destination used by the virtual network should not be visible in the network infrastructure.

- Session unlinkability: Packets with the same source-destination address pair should not be linkable. That is, an attacker should not be able to determine if any two packets have the same pair of source and destination addresses.

- Routing privacy: Routing information should not reveal information about address encryption.

Note that in this work we focus on network addresses and how to hide the information inside them. Clearly, higher layer protocols also carry information that can be used to link packets to the same session (e.g., port numbers). However, we assume that appropriate security protocols are used to protect such information. Since higher layers also often encrypt packet payloads (e.g., Transport Layer Security (TLS)), our work only focuses on protecting network layer information.

3.2.2 Attacker Capabilities

In our system, we consider the network infrastructure provider to be the attacker who aims to observe operational information from the hosted virtual networks. The capabilities of the attacker are assumed to be:

- Full access to data plane: The attacker can see all data traffic in network infrastructure.

- Full access to control plane: The attacker can see all forwarding tables and routing information exchanges in the network infrastructure.

To allow for a practical solution to this problem, we need to limit the attacker's capabilities as follows:

- No access outside network infrastructure: The attacker cannot see or modify traffic outside the network infrastructure, especially not on the gateways where traffic transitions into and out of the network infrastructure.

- Honest-but-curious mode of operation: The network infrastructure is assumed to implement forwarding correctly. Incorrect forwarding behavior (e.g., black hole attack) can be detected by other techniques [12,39] and is outside the scope of our work.

3.2.3 Implementation and Performance Considerations

Based on the above security model, there are multiple possible solutions to provide privacy, including those mentioned in related work. However, we also need to consider how such solution can be implemented in a practical network. Therefore, we impose the following additional requirements on our solution:

- Use with commodity routers: The system should allow the use of off-the-shelf routers in the network core. Since routers use longest prefix match to determine forwarding decisions, this requirement implies that any solution needs to maintain the prefix characteristics of the IP address space. Of course, gateway nodes that translate between the virtual network and the network infrastructure do not need to meet this requirement.

- Low processing overhead: The system should not require complex cryptographic computations for every packet that is transmitted. Requiring such computational overhead would lower the network performance that can be achieved.

- Packet Forwarding Rate: The proposed encryption technique should not significantly impact the performance of the lookup operation (packet forwarding rate). Since packet forwarding is based on longest prefix match, the encryption function should be prefix-preserving.

3.2.4 Encryption and Forwarding Process

Based on the above requirements and assumptions, we design a system that can protect IP address information while still allowing efficient packet forwarding using commodity hardware.

Conceptually, the operation of EncrIP is illustrated in Figure 3.2. In normal IP packet forwarding, routers forward packet using a regular forwarding information base (FIB). In EncrIP, packet addresses are first translated into the encrypted domain

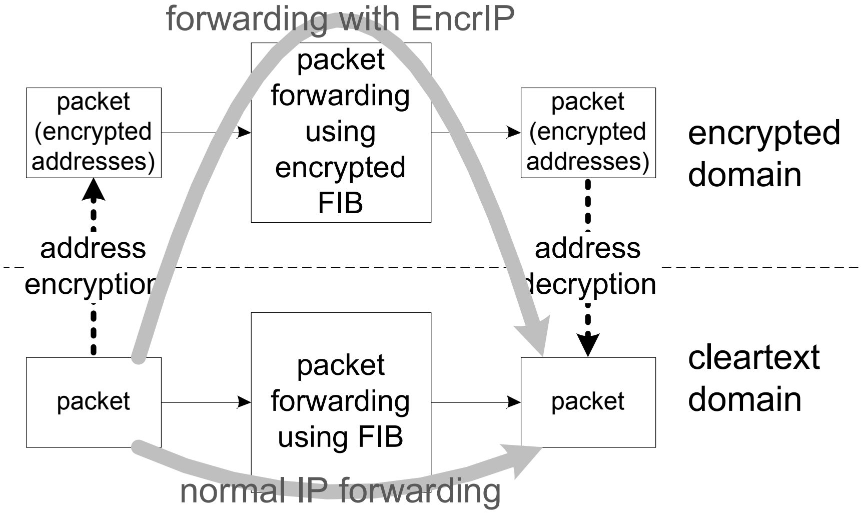

Figure 3.2. Address encryption and decryption processing in EncrIP.

using address encryption, which results in packet that contain encrypted addresses. These packets are then forwarded using a different forwarding information base where prefixes are adjusted to match their encrypted representation. At the exit gateway, the packet addresses are decrypted, which results in unencrypted packets having reached the destination.

3.2.5 Design Choices for Encryption

There are many different ways of how the encryption and decryption process could be implemented. One way to encrypt the IP addresses would involve mapping each original address to a randomly chosen IP address, using a one-to-one mapping technique. For example, a format-preserving encryption [19] is based on such an approach. However, with such a method an adversary can still identify all packets that belong to the same flow and hence this method does not meet our requirements. To avoid a straightforward identification of packets from the same session, we need to consider an encryption function that introduces some form of randomization.

Probabilistic encryption functions introduce such randomness in the encryption process [57]. When the same input information is encrypted multiple times, different encrypted outputs are generated. Since the encryption needs to be reversible, the output of probabilistic encryption is necessarily larger than the input. Thus, every 32-bit IP address is mapped to set of encrypted addresses of length $32 + x$ bits. (We discuss in Section 3.3.5 how the x extra bit can be accommodated in the IP header.)

Fully Homomorphic Encryption (supports both addition and multiplication) has theoretically proved to process the data in the encrypted domain without decrypting the input data [56]. All processing functions are performed in the encrypted domain and hence the infrastructure is completely oblivious to the data being processed. However the practical feasibility of the FHE technique to satisfy our protocol processing requirements and challenges are unclear.

While probabilistic encryption does provide randomization to avoid session linkability, it does not exhibit the prefix-preserving nature that is necessary for using longest prefix match during forwarding. In the next section, we describe our EncrIP encryption technique that uses randomization *and* maintains prefix-relationships, which is the main contribution of our work.

3.2.6 Assumptions

Before discussing the design of our EncrIP function, we enumerate the following assumptions in our solution:

- In our work, we assume an honest-but-curious adversary model (e.g., statistical/passive inferences). For example, the NI providers are assumed to correctly host the VN services but are curious in identifying the communicating entities of a particular connection.

- Edge routers (gateways) in the network are assumed to be secure (operated by the same administrative entity) and hence perform the required cryptographic computations in the system.

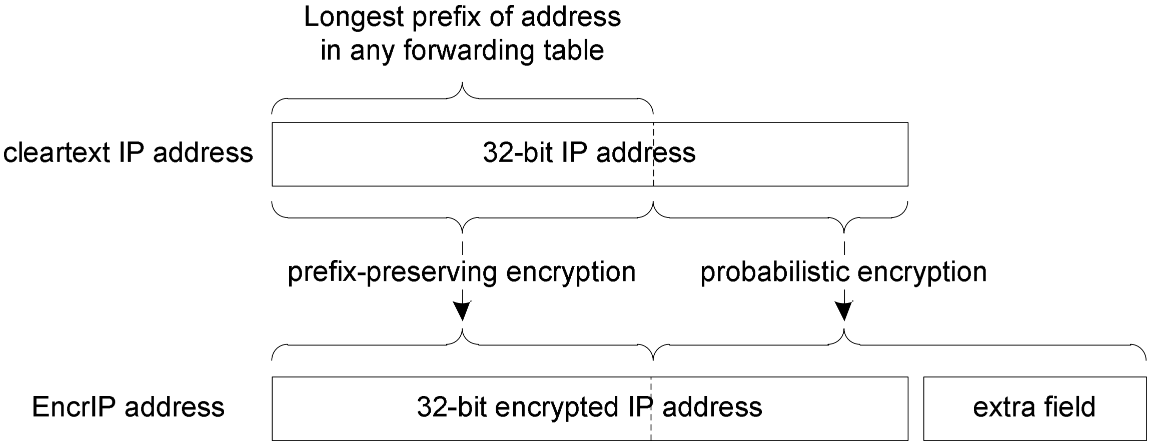
Longest prefix of address in any forwarding table
cleartext IP address
32-bit IP address
prefix-preserving encryption
probabilistic encryption
EncrIP address
32-bit encrypted IP address
extra field

Figure 3.3. Address Transformation in EncrIP.

- The VN provider securely stores the required prefix length information of all IP addresses in the edge routers using a prefix tree data structure.
- Also, the VN provider securely installs the required encrypted addresses in the forwarding table of the NI components.
- We consider the packet payloads and higher protocol header information to be encrypted using conventional cryptographic protocols to achieve the required end-to-end privacy.
- The EncrIP function design is proposed to work with IPv4 addressing, however, the technique can be easily extended to work with IPv6 forwarding address format by varying the tweakable parameter values proposed in the Section 3.3.

3.3 EncrIP: Probabilistic Prefix-Preserving Address Encryption

The overall process by which addresses are encrypted in EncrIP is illustrated in Figure 3.3. Prefix-preserving encryption is used on the portion of the address that needs to maintain prefix properties for forwarding. The number of bits affected by this operation depends on the prefixes that are announced within the virtual network; the length of the longest prefix announced for a given address determines the number of bits that are encrypted with prefix-preservation. The remaining bits in the address are encrypted using probabilistic encryption. As mentioned above, probabilistic encryption maps one cleartext to many ciphertexts. Thus, the output of probabilistic encryption is longer than the input. As a result, the EncrIP address requires additional bits. Note that the bits in the extra field are not necessary for forwarding since they do not contain any information that needs to be matched for longest prefix match.

Let the original address length be represented as n (IPv4 = 32 and IPv6 =128) and the encrypted address length be m $(m = n + x)$, where $x > 0$ represent the length of extra bits in the encrypted address. Let p be the common prefix length and t be the top-hashing length (discussed in section 3.3.2). Therefore, the EncrIP function can be defined as: $EncrIP : \{0,1\}^n \rightarrow \{0,1\}^m$, where, an original address $'a'$ is encrypted to output multiple encrypted addresses $(b_1, b_2, ..., b_i, ..., b_j)$ as follows:

$$EncrIP(a) = [b_1, b_2, ..., b_i, ..., b_j] \tag{3.1}$$

such that the following properties are satisfied,

$a^{1,...,t} \rightarrow b_i^{1,...,t}$	top hashing
$b_i^{t+1} = b_j^{t+1}, ..., b_i^p = b_j^p$	prefix-preserving encryption
$a^{p+1,...,n} \rightarrow b_i^{p+1,...,m}$	probabilistic encryption
$1 \leq i, j \leq 2^x$	number of encrypted addresses

In the next section, we discuss the functionalities of the probabilistic encryption function and the prefix-preserving encryption tree used in our technique.

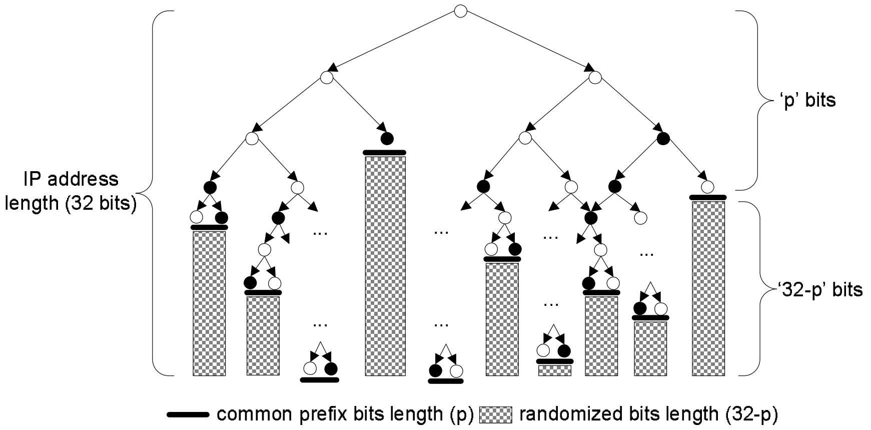
IP address length (32 bits)
'p' bits
'32-p' bits
...
common prefix bits length (p)
randomized bits length (32-p)

Figure 3.4. EncrIP Function.

3.3.1 Probabilistic Encryption

Probabilistic encryption matches well with our security requirements since it introduces randomness, which makes it more difficult for an attacker to link packets that belong to the same session, thereby reducing the impact of statistical inferences introduced by the NI components. Existing probabilistic encryption functions either introduce high latency cryptographic computations [57] or are ineffective with respect to the required compactness of the encrypted address length [45], and hence are not suitable to be used in encrypting the forwarding addresses. To achieve the required randomization, we propose a function with the following properties:

- An encryption function which introduces some form of randomness by converting a deterministic encryption function into a probabilistic encryption [21]. However, the output should match a pseudo random number generator that generates set of encrypted addresses with almost uniform distribution, such as [69].

- A function that avoids collision, such that no two original addresses are mapped to the same encrypted address.

Since the first p bits in the original address are required to maintain the prefix-preserving property, the remaining $n - p$ bits can be randomized to output multiple encrypted addresses. However, the encrypted address length ($m = n + x$) determines (constraints) the level of randomization achieved by the encryption function. The EncrIP function randomizes the address bits as follows: Given an IP address with prefix length p, the probabilistic encryption function considers the last $n - p$ bits (i.e., p+1,...,n) in the original IP address and maps it to $m - p$ bits (i.e., p+1,...,m) in the encrypted address ($a^{p+1,...,n} \rightarrow b_i^{p+1,...,m}$). For example, given an IP address with prefix length /24, the last 8 bits in the original address are randomized to output the last $8 + x$ address bits in the encrypted address. For each incoming packet, the original address bits are mapped to one of the 2^x different encrypted addresses. Assuming an uniform distribution, the identical encrypted address is generated after every $2^x - 1$ packets.

An Optimal Asymmetric Encryption Padding (OAEP) [21] algorithm is used to generate the encryption output. This is done by first padding the input address bits with random bits (using a pseudo random number generator [69]) and then transforming them using a combination of asymmetric encryption (e.g., RSA) and set of random oracles [20] (e.g., cryptographic hash functions) to generate the desired encrypted address bits. The number of bits stored in the extra fields determines how well an encrypted address holds up against statistical inference attack in the network infrastructure. In section 3.5, we quantify this level of security. Next, we discuss the prefix preserving encryption function which is required to encrypt the common prefix bits in the original forwarding address.

3.3.2 Prefix-Preserving Encryption

A prefix-preserving transformation on an address (or part thereof) is defined as follows:

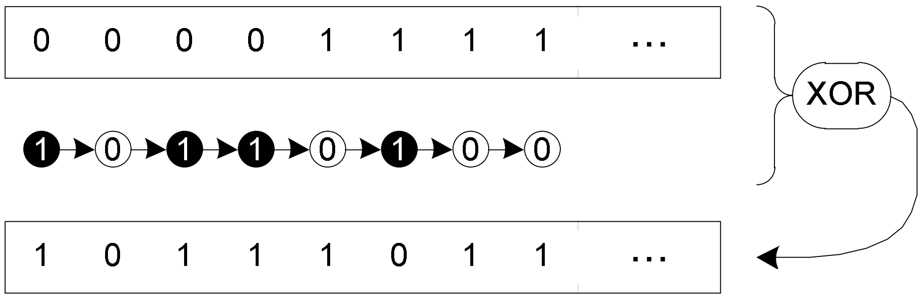

Figure 3.5. Prefix-preserving encryption of addresses.

Definition 1 *A function f is prefix-preserving if any two IP addresses a_1 and a_2 that share a common p-bit prefix also share a p-bit prefix after transformation to $f(a_1)$ and $f(a_2)$.*

As discussed in [127] and [91], the only way to achieve prefix-preserving encryption is to consider a binary prefix tree, T, where nodes indicate which bits need to be flipped during transformation. A cryptographic function is used to determine the state of each node in this "encryption tree" (i.e., white nodes indicating '0' or no flip and black nodes indicating flip or '1'). To encrypt an address, the tree is traversed using the original address bit sequence and node values encountered are recorded. The node values are then used in an exclusive-OR (XOR) operation to generate the encrypted address (or address prefix) as illustrated in Figure 3.5. Since the colors of the nodes in the tree are determined by a cryptographic function and cannot be guessed, the output of the XOR function is encrypted.

In EncrIP, we do not use a full binary tree as has been done in [127] and [91]. Instead, we only use a tree that, for any given address, has a depth equal to the longest matching prefix in any forwarding information base in the network. As illustrated in Figure 3.6, the tree contains only p bits for each address, corresponding to the announced prefix length in the network. The remaining $32-p$ bits (plus the extra field) are randomized using probabilistic encryption.

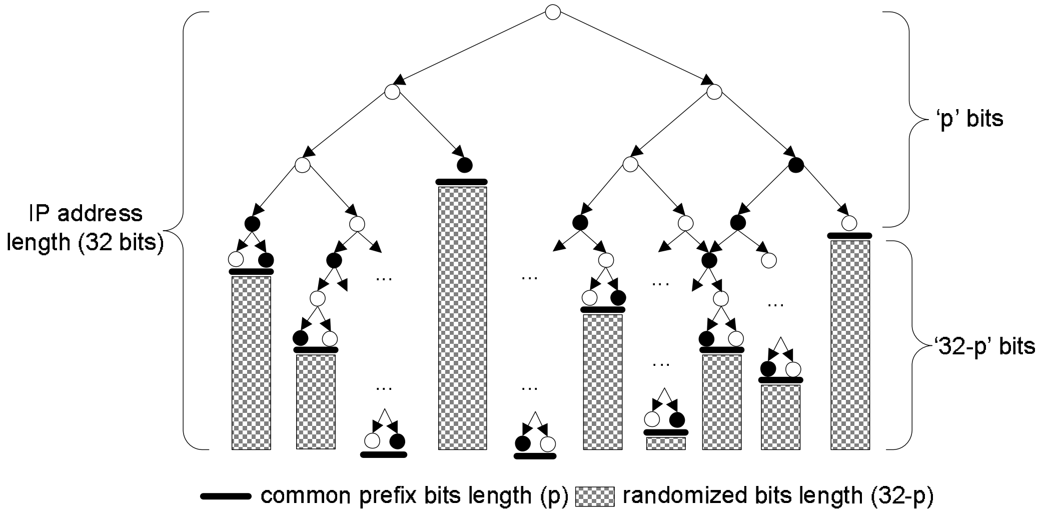
IP address length (32 bits)
'p' bits
'32-p' bits
common prefix bits length (p)
randomized bits length (32-p)

Figure 3.6. Encryption Tree in EncrIP.

3.3.3 Prefix Length Distribution

One key question is how deep the EncrIP encryption tree is for every possible address. The values labeled "EncrIP" in Figure 3.7 shows a sample prefix length distribution of a routing table obtained from the RouteViews project [2] (Routing Table Report, Japan View, RIB date 11/05/2011). One way to create the encryption tree is to keep the full data structure in memory. This approach, however, may be expensive to implement. Therefore, we propose two approaches to reduce implementation cost:

- Prefix expansion: To avoid the need to store many different prefix lengths, prefix length can be expanded to certain predetermined levels. As a result, the information about prefix lengths can be compressed more easily. We consider three prefix expansion schemes: EncrIP-Coarse represents a four-level prefix expansion scheme using prefix length /8, /16, /24 and /32; EncrIP-Mid represents a five-level expansion scheme using prefix lengths /8, /16, /21, /24 and /32; EncrIP-Fine represents a six level expansion scheme using prefix lengths /8, /16, /21, /24, /30 and /32. The prefix expansion schemes with varying number of expansion levels as shown in Table 3.2

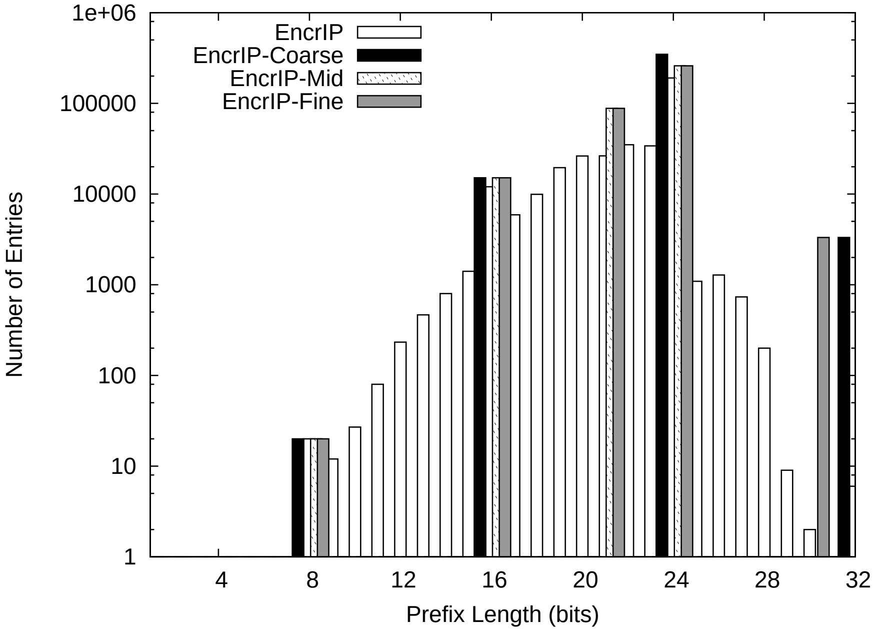

Figure 3.7. IPv4 Prefix Length Distribution.

- Dynamic computation of node colors: Instead of storing the encryption tree in memory, it is also possible to dynamically calculate the node colors based on the cryptographic function used to color the nodes. This approach reduces memory requirements to nearly zero, but generates computational overhead. For some gateway nodes with low data rates and limited memory, this tradeoff may be suitable.

In the remainder of this chapter, we only consider prefix expansion as one possible improvement. While prefix expansion decreases the amount of memory required for the encryption tree in the gateway router, the IP addresses are matched with longer prefixes and hence the number of prefixes that need to be maintained in the network infrastructure increases. Also, the session unlinkability property suffers since the length of the probabilistically encrypted address portion decreases. We evaluate this tradeoff in more detail in Section 3.5.

Table 3.2. Prefix Expansion Levels

Feature Name	Prefix Expansion Level
EncrIP	Original prefix length
EncrIP-Coarse	/8,/16,/24,/32
EncrIP-Mid	/8,/16,/21,/24,/32
EncrIP-Fine	/8,/16,/21,/24,/30,/32

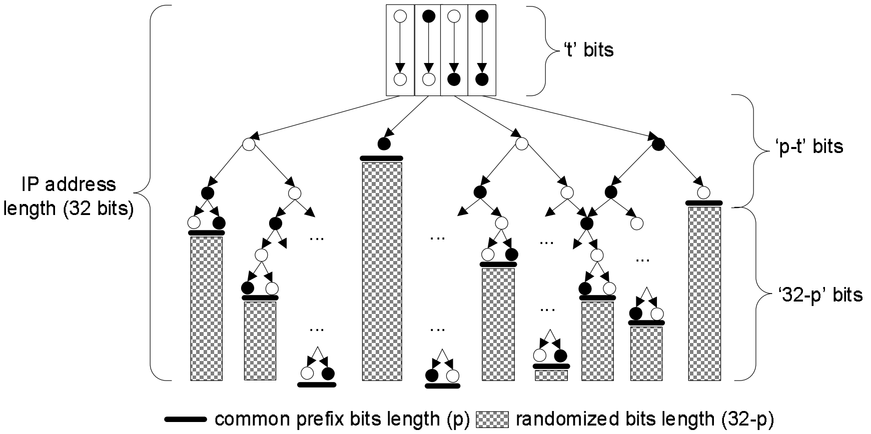
't' bits
'p-t' bits
IP address length (32 bits)
'32-p' bits
...
common prefix bits length (p)
randomized bits length (32-p)

Figure 3.8. Top-Hashing based EncrIP Function.

3.3.4 Top Hashing

One security concern with the above prefix tree is that nodes in T that are close to the root node can compromise the original/encrypted IP address mapping. For example, if the color of the top node in T is known, then the attacker can identify the first bit of all encrypted addresses in the address space. Therefore, maintaining prefix-properties at the top of the prefix tree leads to easy inference attacks. To address this problem, a technique call "top-hashing" [91] can be used to encrypt the first t bits of the address without using a tree but a $2^t \to 2^t$ cryptographic, collision-free hash function, as shown in Figure 3.8. Top-hashing removes any correlation between prefixes in the first t bits and thereby improves the level of security achieved in T.

The top-hashing scheme uses a cryptographic function to encrypt the first t bits in the original address (precomputed and stored in T), as shown in Figure 3.9. This ensures that the first t bits in the encrypted address are unique for each original address. A hash function clearly does not preserve prefixes, but as Figure 3.7 shows,

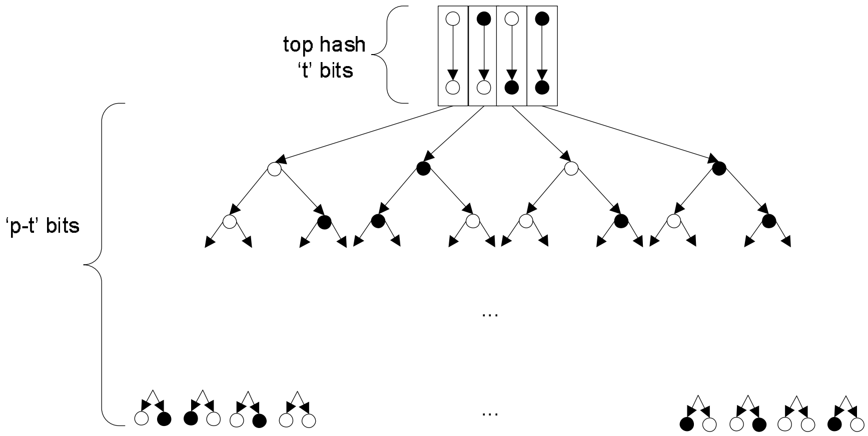

Figure 3.9. Top Hashing-based Prefix-Preserving Encryption.

0-3	4-7	8-13	14-15	16-18	19-31
Version	Length	DSCP	ECN	Total Length	
Extra field for Encrypted Source IP Address		Extra field for Encrypted Dest. IP Address		Flags	Fragment Offset
Time to Live		Protocol		Header Checksum	
Encrypted Source IP Address					
Encrypted Destination IP Address					
Data					

Figure 3.10. Encrypted IP Header.

there are no prefixes with a length less than 8 bits. Thus, values of up to $t = 8$ do not cause any practical problems.

3.3.5 EncrIP in IP Protocol Header

Since probabilistic encryption of addresses requires more space than the original 32 bits, a key question is where to store the additional data. As shown in Figure 3.10, we can use the 16-bit identification field to store two 8-bit extra fields – one for the source address and one for the destination address. The requirement for the identification field is that its value is unique for a given sender-destination pair within a maximum packet lifetime. Since we use probabilistic encryption, which generates pseudo-random values for this field, it can be argued that there is a probabilistic guarantee that the same value does not get repeated within a packet lifetime (except for high data rate flows, for which the conventional IP protocol has the same problem). Thus, *EncrIP can be implemented in the standard IP header without the need to use any options or redefining the use of any fields.*

3.4 Complete EncrIP Process

To realize EncrIP, there are several operations that need to be performed in the data plane and the control plane.

3.4.1 Data Plane

Based on the techniques described above, the encryption process for an address is as follows: (1) encrypt the first t bits of the original address using the top hashing scheme, (2) encrypt the next $p-t$ bits of the original address using the lookup operation in the encryption tree T, (3) encrypt the next $32-p$ bits of the IP address using the probabilistic encryption function, to output one of the 2^x possible encryptions of length $32-p+x$, (4) concatenate the encrypted t, $p-t$, $32-p+x$ bits to generate the final encrypted address consisting of 32 bits used in forwarding and x bits used in the extra field.

Correspondingly, the decryption process is as follows: (1) receive the encrypted address consisting of $32+x$ bits, (2) extract the first t bits and decrypt using the reverse top hashing function, (3) traverse the decryption tree T', which is derived from T, to determine the prefix length p and decrypt the next $p-t$ bits from the encrypted address, (4) extract the next $32-p+x$ bits in the address and decrypt using the probabilistic decryption function to retrieve the final $32-p$ bits in the original address, (5) concatenate the bits from the above steps to obtain the cleartext address.

3.4.2 Control Plane and Setup

To set up EncrIP in a virtual network, the VN operator only needs to enable the EncrIP gateways to perform address encryption for traffic that is sent to the untrusted network infrastructure and address decryption for traffic that is received. This requires that all EncrIP gateways compute the encryption tree T and its inverse T'. Since the tree construction can be done with a cryptographic function, the only requirements is that the gateways have a shared secret (or set up this shared secret through a secure protocol).

In addition, the network infrastructure needs to obtain routing information for the encrypted prefixes so that the forwarding information bases in routers can be set up. One of the great benefits of EncrIP is that this can be done very easily. The gateway nodes simply advertise the encrypted version of each prefix to the network infrastructure using conventional routing protocols (since these protocols cannot distinguish between a cleartext prefix and ciphertext prefix).

Thus, *EncrIP does not require any changes to the data plane operation or control plane operation of the network infrastructure or the end-systems.* All EncrIP functionality is implemented in EncrIP gateways.

3.4.3 Limitations

While EncrIP meets our security requirements to provide privacy in the network infrastructure, there are a few limitations:

- No support for flow-based operations in network infrastructure: Some systems need to identify which packets belong to the same flow to operate correctly (e.g., content inspection across packet boundaries). However, our requirement for session unlinkability inherently prevents such systems from being used in the network infrastructure.

- Changing prefix lengths: In dynamic networks, the length of prefixes can change in case a subnetwork is broken up into multiple parts. EncrIP needs to know prefix lengths to compute what portions of the address are encrypted with prefix-preserving encryption and with probabilistic encryption. Thus, the encryption tree needs to be updated when prefixes become longer.

- Profile-based inference attacks: We discuss in Section 3.5 how well EncrIP protects from statistical inference attacks, where an attacker tries to determine which packets belong to the same flow. There are profiling techniques that can determine the relationship and content of encrypted packets [23], which could be used to more effectively attack some EncrIP traffic.

We believe that these limitations do not present significant hurdles as they can be mitigated in any practical deployment of EncrIP.

3.5 Evaluation and Analysis

We are evaluating the effectiveness and performance of EncrIP in two ways. First, we present results from an analysis of EncrIP based on Internet routing information. Second, we present results from a prototype implementation. Our analysis focuses on the required gateway memory and prefix overhead as well as the security properties achieved by EncrIP. For our analysis, we consider the real world prefix length distributions obtained from the RouteViews project [2] (Routing Table Report for London, Japan and RouteViews-3 view, RIB date 11/05/11) and from the BGP routing table analysis report [31] (AS6447 and AS65000). The top-hash length is set to $t = 7$ bits, the prefix lengths vary within $p = 8 \ldots 32$ bits, and the encrypted address length is set to $32 + x = 40$ bits.

Table 3.3. Memory Requirements on EncrIP Gateways

Routing	Memory in MB			
Table	EncrIP	EncrIP-Fine	EncrIP-Mid	EncrIP-Coarse
London	5.32	3.96	3.62	3.60
Japan	5.38	4.02	3.80	3.68
Routeviews-3	5.41	4.13	3.87	3.71
AS65000	5.48	4.24	3.98	3.85
AS6447	5.66	4.48	4.07	3.92

Table 3.4. Prefix Overhead of Expansion Schemes

Routing	Number of Prefixes and Overhead)			
Table	EncrIP	EncrIP-Fine	EncrIP-Mid	EncrIP-Coarse
London	374K	786K (2.09×)	792K (2.11×)	3.17M (8.45×)
Japan	380K	854K (2.24×)	1050K (2.75×)	3.43M (9.00×)
Routeviews-3	384K	812K (2.11×)	830K (2.16×)	3.29M (8.57×)
AS65000	381K	858K (2.24×)	1063K (2.78×)	3.44M (9.02×)
AS6447	394K	857K (2.17×)	976K (2.47×)	3.46M (8.79×)

3.5.1 Gateway Memory and Prefix Overhead

An EncrIP gateway needs to maintain the encryption tree, which also contains the information of how long each prefix is. The memory requirements are calculated based on the prefix length distribution for each routing table report and for the different prefix expansion levels discussed in Section 3.3.3: EncrIP (no expansion), EncrIP-Fine, EncrIP-Mid, and EncrIP-Coarse. We determine the total memory required for T as follows: Each node in T holds 1-bit of information. The top hashing scheme totally consists of 2^t nodes and each top hash node lead to a subtree of size $2^{p-t} - 1$. Table 3.3 shows the total memory required by EncrIP gateways for the different prefix length distributions. From the table, we see that expansion reduces the amount of memory required since prefix length information can be stored more effectively.

However, prefix expansion comes at the cost of creating additional prefixes. Table 3.4 shows the number of prefixes that are present in the original EncrIP method and each of the expansion schemes (and the overhead compared to EncrIP). We observe that EncrIP-Fine and EncrIP-Mid require two to three times the number of

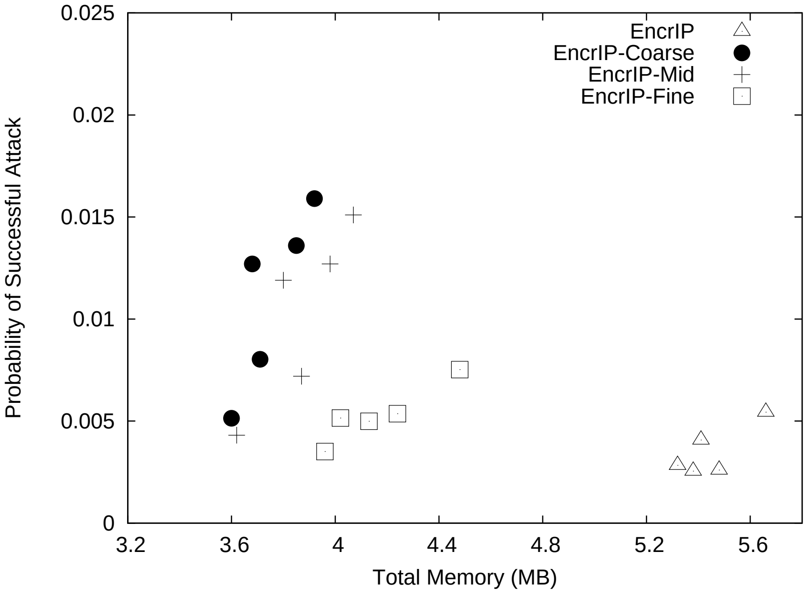

Figure 3.11. Attack probability vs. gateway memory.

prefixes that EncrIP requires, which may be acceptable in practice to save memory in the gateways.

3.5.2 Probability of Successful Attack

The goal of an attacker is to correctly map an encrypted IP address to the original IP address (i.e., violating the requirement of address privacy) or identifying separate packets that belong to the same flow (i.e., violating the requirement of session unlinkability). Since the latter is simpler for an attacker, we focus on attacking session unlinkability property of EncrIP. As the attacker does not know the encryption tree, the only practical attack is based on statistical inference. Thus, the attacker snoops the set of packets that share a common prefix of length p and then randomly guesses the encrypted-to-original address mapping in the remaining $m - p$ bits in the encrypted address. With this strategy, the probability of a successful attack depends on the probability of two encryptions of the same address being identical.

Since p bits in the encrypted address are known to the attacker (based on attacker strategy of snooping sets of packets with common prefix), the problem is reduced

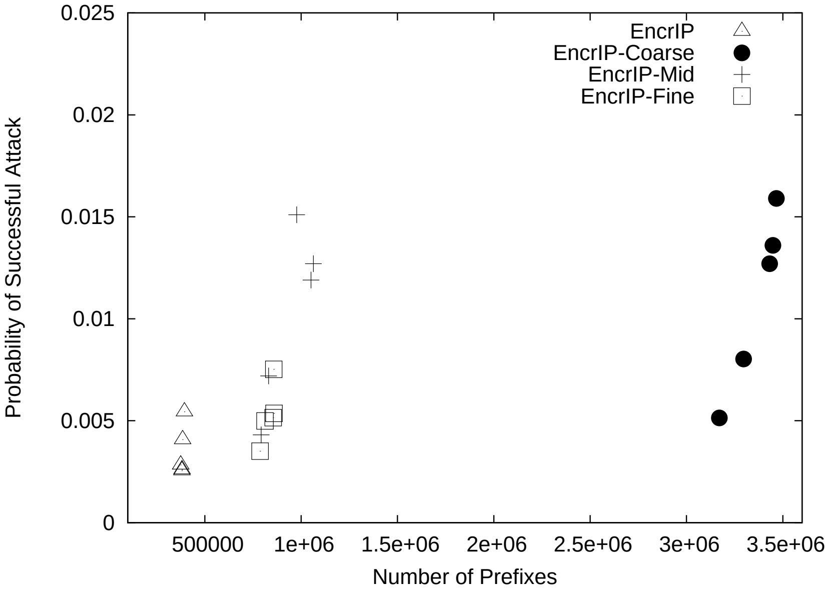

Figure 3.12. Attack probability vs. prefix overhead.

to guessing which of the remaining $2^{32-p+x} - 1$ addresses correspond to one of the remaining $2^x - 1$ representations of the original address. Thus, the probability of a successful attack is given by $P[\text{successful attack}] = \frac{2^x-1}{2^{32-p+x}-1}$.

The attack probability for different route tables and expansion schemes is shown in comparison to gateway memory requirements in Figure 3.11 and in comparison to prefix expansion overhead in Figure 3.12. We observe that the attack probability is lowest for EncrIP because no expansion is used and the range of bits on which probabilistic encryption can be applied is maximized. EncrIP, however, also requires most memory in the gateway. EncrIP-Fine and EncrIP-Mid, provide marginally lower protection from attacks, but require less memory while only increasing the number of prefixes by a factor of two. EncrIP-Coarse introduces a large prefix overhead without reducing the memory cost significantly over EncrIP-Mid. Therefore, EncrIP and EncrIP-Fine seem to be the best choices for a practical deployment.

The generalized probability of successfully identifying all the encrypted addresses mapped to the same original address is given by (where i varies from 1 to $2^x - 1$):

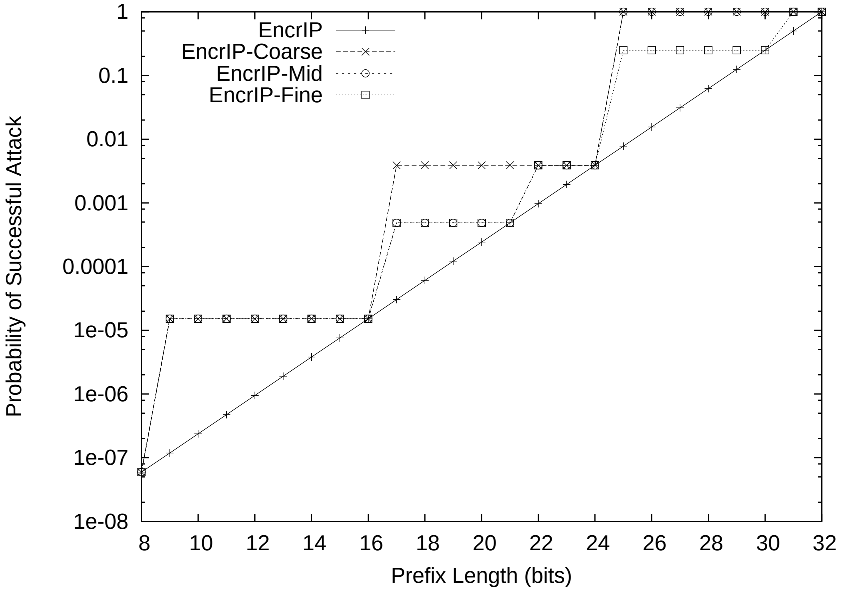

Figure 3.13. Probability of Successful Attack.

$$P(Attack)^{all} = \prod_i \left(\frac{2^x - i}{2^{32-p+x} - i} \right) \tag{3.2}$$

Figure 3.13 shows the probability of successful attack ($P(Attack)^2$) for varying prefix length.

3.5.3 Entropy

The entropy in our work determines the level of randomization provided by the encryption function. We assume that packets with common prefix length arrive equally likely from different sources and hence the rate at which the NI component receives an identical packet of prefix p is given by $\frac{1}{2^{m-p}}$. Therefore the entropy of the proposed encryption function is given by the Shannon entropy formula as follows:

$$H(X) = \sum_{i=1}^{2^{m-p}} P(X_i) * log_2 P(X_i) \tag{3.3}$$

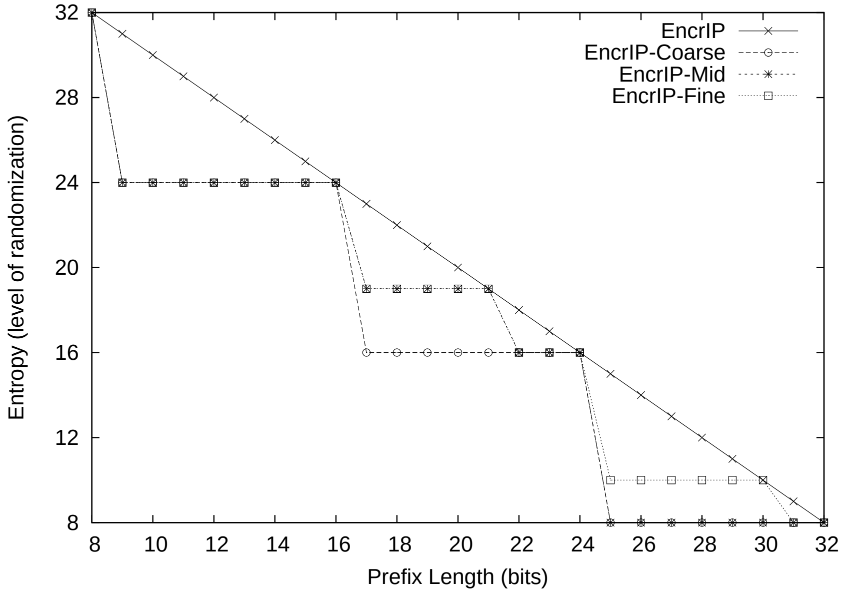

Figure 3.14. Entropy of EncrIP Function.

where $P(X_i) = \frac{1}{2^{m-p}}$. Figure 3.14 shows the entropy of the encryption function for varying prefix length and impact of entropy when prefix expansion is considered. For smaller common prefix bits, the level of randomization is higher and as the prefix size increases the entropy decreases linearly.

3.6 Prototype Implementation

We implemented EncrIP using the ProtoGENI experimental testbed [77]. For experimentation, we set up a network topology consisting of 11 virtual network end-system nodes, 3 gateway nodes, and 16 network infrastructure nodes. The encryption/decryption prefix trees were generated using SHA-1 and probabilistic encryption was implemented use OAEP. Forwarding tables on routers were set up using encrypted prefixes. EncrIP traffic used the standard IP header as described in Section 3.3.5.

The main result from the prototype system is that *EncrIP traffic was forwarded correctly between end-systems using unmodified routers in the network infrastructure.*

Table 3.5. Per-Packet Processing Latency on Gateway for EncrIP and IPsec

Operation	EncrIP	IPsec ESP-Tunnel mode (μs)		
	(μs)	3DES	AES128	AES256
Encryption	81	316	214	222
Decryption	80	293	198	201

We have also measured the processing latency and vulnerability to statistical inference attacks.

3.6.1 EncrIP Processing Latency

We have measured the per-packet processing time that a gateway spends to perform EncrIP encryption and decryption. The results are shown in Table 3.5. As a comparison, processing times are also shown for different variants of IPsec tunneling. Note that EncrIP is considerably faster since only addresses need to be modified. In addition, IPsec also incurs additional initial latency due to Internet Key Exchange (IKE) operations. Thus, in virtual networks, where address privacy and session unlinkability need to be achieved and data rates are high, EncrIP is clearly a better choice.

Note that these processing times are obtained from software-based router systems. In practice, high-performance network-processor-based implementations are expected to be considerably faster.

3.6.2 Statistical Inference Attacks

Finally, we evaluate the probability with which an attacker successfully can determine that addresses (or an address pair) from two different packets match. Since we use 8 extra bits per address, it can be expected that encrypted addresses need to be repeated every $2^8 = 256$ packets. When looking at address pairs, a repetition can be expected every $2^{(8+8)} = 65536$ packets. To verify this expectation with measurement, we initiated TCP connections from five different end-system nodes to five different

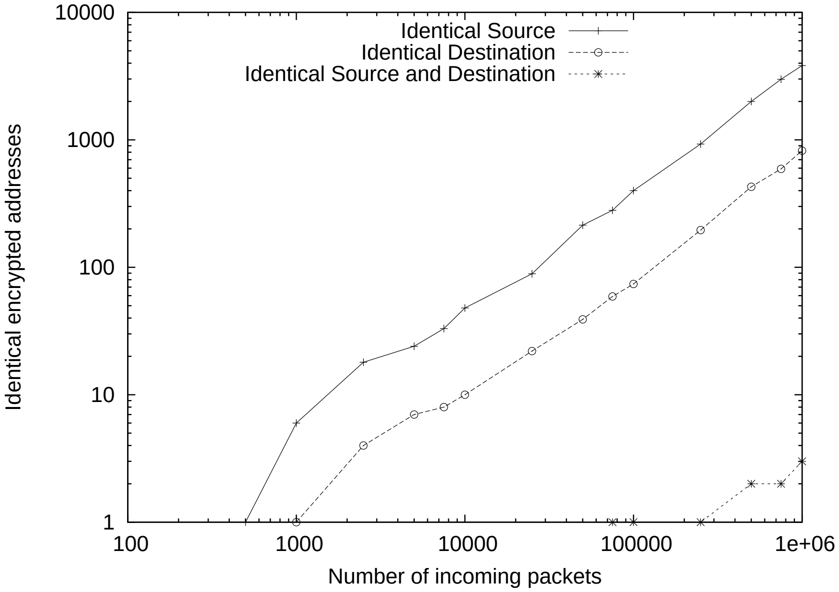

Figure 3.15. Statistical Inference Attack.

end-systems, each. At a router in the infrastructure, we recorded all incoming packets to determine the sets of identical addresses observed.

Figure 3.15 shows the total number of identical source address, destination addresses, and source and destination address pairs received for varying numbers of incoming packets. Since the source communicates to multiple destinations via the chosen core router (i.e., creating five flows with same source address), the number of identical source addresses received is higher than the number of identical destination addresses. For a total of 1 million incoming packets, we observed only 3 packets with the same source and destination address pair, which leads to a session linkability of less than 0.001%. In practical deployments with more active network connections and more short-lived flows, the probability of identifying packets belonging to the same session drops even lower. This confirms the effectiveness of EncrIP to provide address privacy and session unlinkability.

3.7 Related Work

The use of network virtualization introduces new security issues (e.g., privacy and integrity of hosted virtual networks, side-channel attacks between virtual networks, etc.) [75]. There have been several approaches to explicitly managing trust in this environment. A trust management framework that allows service providers (virtual networks) to identify trustworthy infrastructure providers based on their past experiences and feedback is proposed in [71]. To address accountability in hosting virtual networks, violation detection using processor extensions and network measurements to monitor service-level agreements are introduced in [61]. The lack of operational confidentiality when hosting virtual networks on third-party network infrastructure can be addressed using minimum disclosure routing [55], which extends Secure Multiparty Computation [29] to allow multiple NI providers to compute the routing decisions using only local routing information. These techniques are complementary to our work since they address the problem of trust, but do not present any solution for protecting operational information in the data plane of virtual network infrastructure.

While some attention has been devoted to developing security solutions to attacks and vulnerabilities in the Internet architecture, only recently attempts have been made to address the privacy concerns of end-users in the Internet architecture. Reference [115] shows the list of ISPs that introduce hidden traffic shaping techniques on peer-to-peer protocols. Such activities indicate the requirement to examine security issues, when hosting virtual networks on third-party network infrastructures. Current solutions to address the user privacy issue either propose a clean slate approach [68] or provide an overlay network over the current Internet architecture (Tor) [40]. Particularly, Tor promises to be an effective solution in anonymizing the source and destination addresses of the communicating entities. However, the technique has undesirable performance issues and is subjected to government crackdown in some countries [105]. Reference [90] introduces an Address Hiding Protocol (AHP) to anonymize the forwarding addresses by deploying the technique in cooperation with the ISPs. Reference [96] proposes an hash-based IP address anonymization technique that can improve the privacy issue in the current Internet architecture.

Secure tunneling protocol techniques (e.g., VPN) provide the required confidentiality of the data by encapsulating the packet payload. However, a detailed discussion on the limitations of such schemes were discussed earlier. Message Stream Encryption (MSE) protocol obfuscates the header and payload data to ensure the provision of confidentiality and authentication. To avoid biased management practices by ISPs, BitTorrent protocol versions introduced MSE based protocol encryption that enhances privacy and confidentiality [116]. However, [28] shows various potential vulnerabilities that can compromise the working of the MSE protocol.

Related to privacy in the data plane of a network is the problem of anonymization of network measurement traces. Prefix-preserving anonymization has been used to provide anonymity while still allowing analysis of traces [91, 127]. Anonymization of packet headers and payloads using trace transformations based on a programming environment for policy scripts was introduced in [83]. A cryptographic technique to scramble the host part of IP addresses in traces was introduced in [86]. An obfuscation algorithm using a many-to-one mapping between IP addresses and group-ID values was proposed in [94] to protect the sensitive data in network flows. While these anonymization techniques are effective with respect to network measurement and trace collection systems, they do not meet our security requirements – in particular session unlinkability. EncrIP does, however, borrow some ideas from prefix-preserving anonymization in order to allow use of commodity routers.

Our work, proposes an encryption technique that ensures the provision of confidential packet forwarding of network traffic in virtualized networks. We provide a platform where the NI components are oblivious to the VN forwarding address information, thereby improving the confidentiality and privacy in the system.

3.8 Summary

In this chapter, we have identified the need for providing privacy for the operational information in virtual networks that are hosted on a third-party infrastructure. We have presented EncrIP, an IP address encryption scheme, that uses prefix-preserving encryption and probabilistic encryption to hide address information while still allowing forwarding using unmodified IP headers and commodity routers. Our evaluation shows that gateways require only a few MB of memory to implement EncrIP and that in practice the probability of correctly identifying two packets belonging to the same flow is less than 0.001%. The implementation results show that EncrIP introduces low per-packet processing latency. Thus, EncrIP provides a practical solution to privacy in virtualized networks.

CHAPTER 4

CAPABILITIES-BASED VIRTUAL NETWORK INSTANCE

In this chapter, a capabilities-based virtual network instance is proposed which introduces data path credentials to authorize legitimate network access and curtail attack traffic. Some initial ideas on this topic have been published in prior work, which includes an overview of the architecture [121] (which was developed in the context of the IAMANET project [6]), two short papers on the use of Bloom filters [122, 123], and some initial results of the prototype system [125]. My contribution to this work is to provide detail analysis on the unicast, multicast and network coding scenarios, introduce the group credentials based network architecture, provide extensive security analysis, validate the security requirements by developing new prototype implementation, and perform experimental evaluation using Emulab that exhibits the robustness of the proposed technique against attack traffic.

4.1 Introduction

Capabilities-based networks present a fundamental shift in the security design of network architectures. Instead of permitting the transmission of packets from any source to any destination, routers deny forwarding by default. For a successful transmission, packets need to positively identify themselves and their permissions to the router. A major challenge for a high-performance implementation of such a network is an efficient design of the credentials that are carried in the packet and the verification procedure on the router. Recent proposals for capabilities-based networks have provided some ideas on the fundamental shift in the design philosophy of networks by moving from the Internet's "on-by-default" principle to an "off-by-default" assumption. In an off-by-default network, a connection needs to be explicitly authorized to reach an end-system rather than being allowed to connect to an end-system by default. Authorization is based on capabilities, which are tokens that represent authority for a particular operation. During the connection setup and data transfer, a connection's capabilities are validated along the connection path. Existing designs of these capabilities-based networks vary in terms of how capabilities are issued, where in the network capabilities are verified, and how the capabilities are implemented.

One key shortcoming of these approaches is that verification takes place only at one node (or a small number of nodes) in the network and thus malicious traffic can travel several hops, absorb resources, and possibly attack nodes before being filtered. In this proposal, we present a capabilities-based protocol that performs verification on every hop in the network. Our system is based on a novel design of capabilities, which we call "data path credentials (DPCP)." These credentials can be validated easily in the data path of routers and thus allow high-performance implementations. Therefore, we can check capabilities on every hop and effectively contain most attack traffic within one hop from its source.

An important question that is addressed in this work is how to design a network that can inherently guarantee that only authorized traffic is transmitted. Denial-of-service attacks and intrusion attacks, which are not authorized, should be contained as close to their source as possible. Blocking unauthorized traffic in the network serves two purposes:

- Protection of end-systems: End-systems may be vulnerable to DoS attacks and intrusion attacks. Eliminating attack traffic inside the network can help protect these systems.

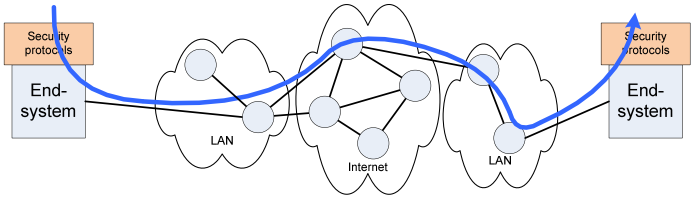

Figure 4.1. Security in Existing Internet Architecture.

- Protection of infrastructure: The effects of large amounts of DoS traffic in the network can have detrimental effects on otherwise unrelated traffic since link resources are shared. Eliminating DoS traffic inside the network can reduce these effects.

Thus, it is important that (1) attack traffic does not reach the end-system and that (2) attack traffic is squelched as close to the source as possible. The latter is one of the main distinctions of our work, which focuses on 1-hop containment, i.e., the elimination of most attack traffic within a single hop from the source.

The overall work shows that it is feasible to implement networks that use a capabilities-based approach to verify packets on every hop and thus inherently can limit the impact of malicious traffic. We begin the discussion of our system with a description of the overall network and router architecture for our capabilities system, which we call "data path credentials," and explain the system design and operation in more detail. The specific design of credentials is discussed in Section 4.3.

4.1.1 Network Architecture

The network architecture that we propose is depicted in Figure 4.3 and compared to conventional data networks in Figure 4.1 and existing capabilities-based networks in Figure 4.2. Nodes shown in blue indicate intermediate nodes in the forwarding path that do not have any inherent security provision. Nodes shown in orange indicate capabilities-based verification mechanisms. The key idea is to augment network traffic with credentials that can be audited in the data path on every hop. Each router

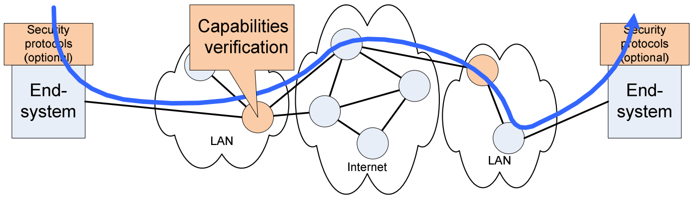
Security protocols (optional)
End-system
Capabilities verification
LAN
Internet
LAN
Security protocols (optional)
End-system

Figure 4.2. Existing Capabilities-Based Network Architectures.

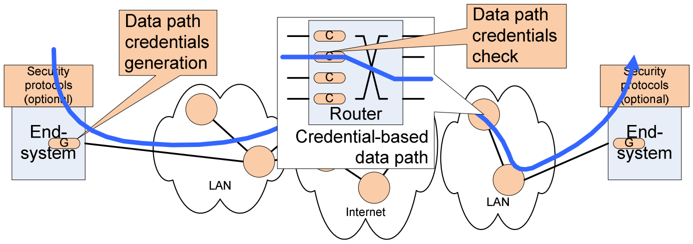
Data path credentials generation
Security protocols (optional)
End-system
G
C
C
C
C
Router
Credential-based data path
Data path credentials check
LAN
Internet
LAN
Security protocols (optional)
End-system
G

Figure 4.3. Credential-Based Data Path Architecture.

performs a credential check and thus can positively identify traffic that is eligible for forwarding. Attack traffic with invalid credentials is discarded.

This approach contrasts to the traditional Internet architecture insofar that security protocols are not constrained solely to end-systems (e.g., cryptographic protocols) or isolated routers (e.g., firewalls or intrusion detection systems). Instead, all routers along the data path of a connection participate in validating traffic and thus defending against attacks. In addition, end-system security protocols can provide orthogonal security features of confidentiality and integrity. In comparison to existing capabilities-based networks, packet validation is not limited to just a few nodes along the path (e.g., “verification points” in [9], edge routers in [59], or LAN switches in [30]), but performed everywhere. This increases the responsiveness of the network to attacks. To illustrate the operation of credentials in the data in more detail, we turn to the functionality implemented on routers.

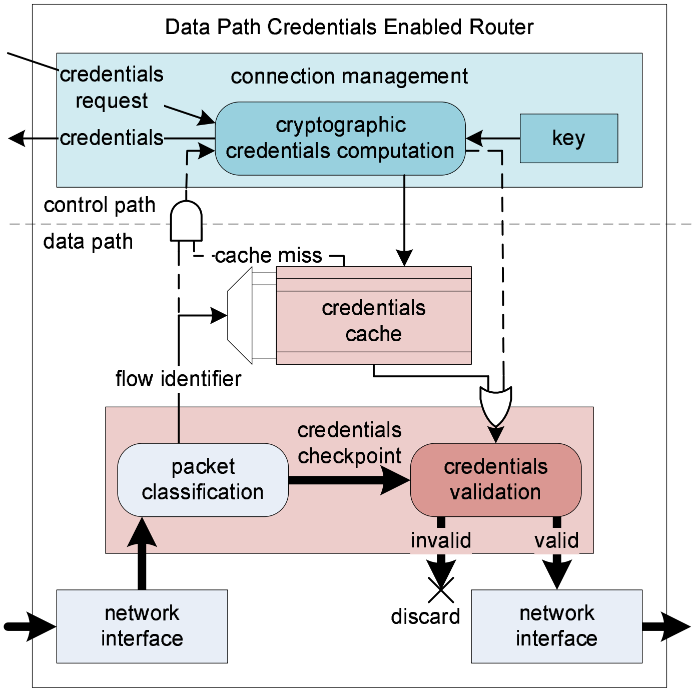

Figure 4.4. Design of a Router System with Data Path Credentials.

4.1.2 Router Architecture

The system architecture of a router that implements data path credentials is shown in Figure 4.4. To simplify the explanation, conventional packet forwarding functions, which remain unchanged, are not shown.

In the control path, connections are managed and credentials are created. An end-system can request credentials for a particular flow. These credentials are then computed based on the flow characteristics and the router's cryptographic key. More details on this process are discussed in Section 4.3. The resulting credentials are then transmitted back to the end-system and stored in the local credentials cache.

In the data path, packet headers are augmented to carry the credentials provided by the sending end-system. When a packet is received on the router for forwarding, the packet is first classified to identify which flow it belongs to. Then the credentials that were generated by the router are retrieved from the credentials cache. If the credentials match those in the packet, the packet is considered valid and thus forwarded. If the credentials do not match, then the packet is discarded.

If credentials cannot be found in the local credentials cache, it may be due to the limited size of the cache or due to an invalid packet. It is possible to trigger a credentials recomputation (dashed lines in Figure 4.4) before discarding the packet. This process may increase the systems vulnerability to denial of service (DoS) attacks since the cryptographic computation of credentials is an expensive operation. It is therefore important that that the cache is sufficiently large and that new credential requests take priority over recomputations. A more extensive discussion on how to defend against DoS attacks can be found in Section 4.5.1.

Since the router classifies packets by connection, a possible extension of our work could differentiate packets from different connections. For example, if quality of service (QoS) is considered, then QoS parameters and flow state can be stored in conjunction with credentials information. After a packet has been identified as belonging to a particular flow and containing valid credentials, it could be forwarded through the router based on its QoS requirements (e.g., by placing it in the high-priority output queue). These QoS parameters would need to be configured at the time of connection setup based on policies (see Section 4.2.4).

4.2 Connection Management

Connection management is an important aspect of our architecture since it addresses one of the key problems that appeared in prior designs of networks using capabilities [15]. The control path of such networks pose as potential target for denial of service attacks [10]. In our design, connection setup is performed as an incremental process (as shown in Figure 4.5 and further explained below). An end-system cannot send a credentials request to a router unless it has valid credentials for the entire path up to that router. Thus, any DoS attempt on the control infrastructure can

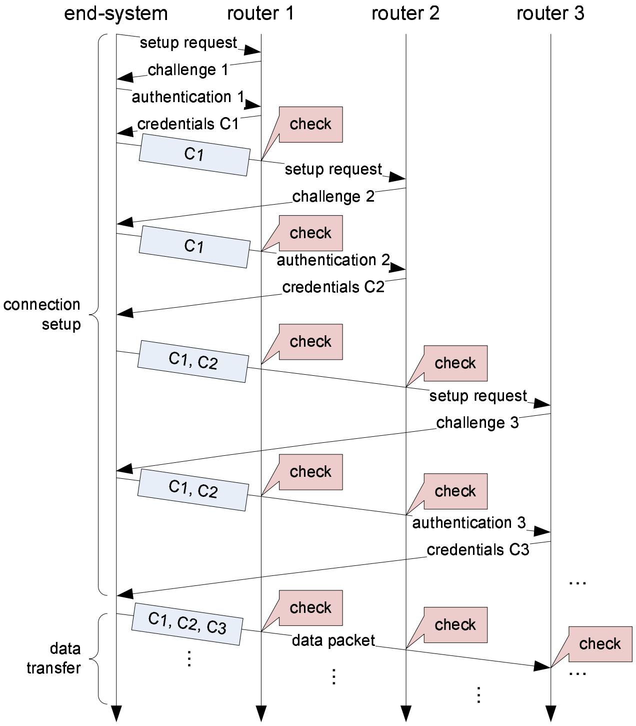

Figure 4.5. Connection Setup to Establish Credentials.

only target router immediately neighboring the attacker. A propagation of the DoS attack is not possible (unless the source can properly identify itself as an authorized end-system, in which case the DoS attack can be traced back and squelched by other means).

In this work, we consider three commonly used communication paradigms to illustrate the generality of the data path credentials concept: unicast, multicast, and network coding.

4.2.1 Unicast

An example of the connection establishment process for *unicast* is sketched in the space-time diagram shown in Figure 4.5. In this scenario, an end-system sends a request to establish a connection to the first of three routers in order to obtain credentials. The router may challenge the end-system to authenticate itself and may negotiate access policies. Once the router has determined that the end-system is eli-

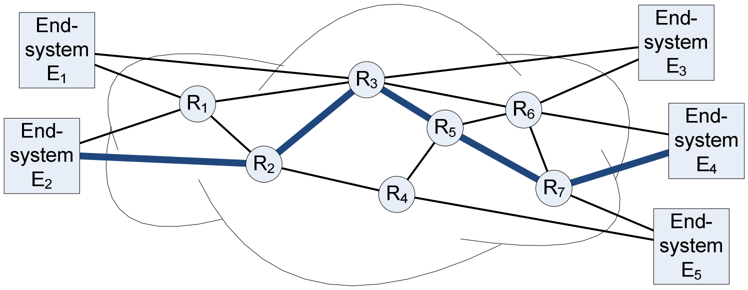

Figure 4.6. Connection Setup: Unicast Scenario.

gible to transmit data across the router, it provides credentials C1. These credentials need to be included in all further transmissions that traverse the router including the credentials request to router 2. Finally, the set of all credentials (C1, C2, and C3 in the example shown in Figure 4.5) is then carried in each data packet. Each data packet is checked on every router. If the set of credentials contains the correct instance for a particular router, the packet is forwarded. If the credentials do not match, the packet is discarded.

Clearly, having to exchange several messages with every router along a path in order to establish a single connection is a costly proposition. There are two approaches that can reduce this overhead:

- Credential Reuse: If multiple connections are established between two end-systems (either in parallel or within a given time window), credentials could be reused.

- Group Credentials: It is possible to create group credentials (e.g., for all routers within an autonomous system) where any router can issue credentials that are valid to traverse any set of routers in that group. In such a case, the number of credential requests per connection can be reduces significantly (as illustrated in Figure 4.10). More details on these credentials can be found in Section 4.3.6.

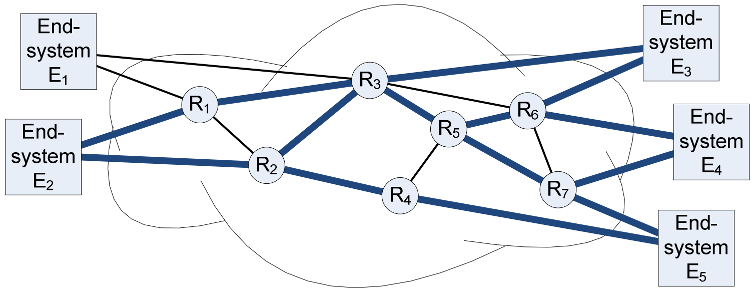

Figure 4.7. Connection Setup: Multicast and Multipath Scenario.

4.2.2 Multicast

To illustrate the versatility of the credentials-based data path design, we also consider credentials in usage scenarios that go beyond unicast: multicast (and multipath) and network coding. These communication modes are illustrated in the following section. In unicast, (Figure 4.6), the set of credentials includes only those along the path from source to destination. In *multicast/multipath*, packets get duplicated inside the network (as illustrated in Figure 4.7). With the duplication of a packet, its credentials get duplicated, too. Thus, the source needs to include credentials for all routers that may be traversed along the multicast tree / multipath graph.

4.2.3 Network coding

Reference [4] introduces a recently proposed approach to improving end-to-end data transmissions in wireless networks. Packets traverse multiple paths to the destination and may be coded together with other transmissions. At the receiver these operations are reversed to obtain the original packets. In such a usage scenario, data path credentials from both sources need to be combined (shown as dashed lines in Figure 4.8). While some routers may overlap in the coded packets' paths, flow identifiers are different and thus credentials are different. Therefore, all credentials from both sources need to be added into the set of credentials carried by the packet.

Clearly, there are cases where a large number of credentials are necessary to guarantee successful forwarding by all participating routers. In Section 4.3, we show that

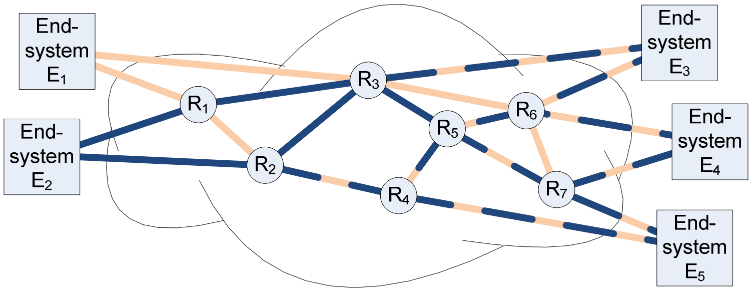

Figure 4.8. Connection Setup: Network Coding Scenario.

our design of credentials only requires constant space for most practical scenarios. In particular, we do not require space that increases linearly with the number of credentials that need to be provided. Thus, the above scenarios can be implemented efficiently. The security analysis in Section 4.4 provides quantitative performance and security tradeoffs for each of these scenarios.

4.2.4 Identity Management, Authentication and Access Control

For a realistic deployment of the proposed architecture, it is important to consider how identities are managed, how authentication is performed, and how access control is used to authorize network access. For our credential-based data path architecture, it is possible to use existing concepts and protocols to address these issues. Our proposed capabilities-based network is agnostic about the specifics of how identities and policies are implemented and managed.

In order for policies to be implemented in a meaningful way, it is necessary to have well-defined identities. In networks, there are many possible ways of defining identities, ranging from using network-specific identifiers for identity-based cryptography [25] to more broadly defined identities [93]. In our system, our only assumption about identities is that identities have key pairs for public-key/private-key cryptography available. We also assume that public keys are properly distributed through a Public Key Infrastructure (PKI) [85] or a more complex federated trust model [22].

In our work, we use the term “end-system” and “user” interchangeably to identify an entity that is the source or sink of a network connection.

To implement policies, we assume a conventional access control system (e.g., role-based access control [53]) to determine the access privileges of a particular entity. To support dynamic updates, a policy-based management systems [3] could be used for interactions between a policy manger and managed resources. In our context, the policy controller could be used to express which entities have access to a particular network of routers (i.e., managed resources) that implement these policies. These policies could be expressed using the Common Information Model (CIM) standardized by the Distributed Management Task Force (DMTF). A specific policy system in the context of access control to networks is Zodiac [34].

Authentication, authorization, and access control decisions are made by routers during connection setup. Figure 4.5 shows only a single exchange of packets for this process, but it is conceivable that a more extensive exchange takes place to establish access privileges. Once this process has been completed, data path credentials are used to enforce these network access policies by validating each packet on each hop of the route. Thus, the overhead for policy verification is limited to the connection setup phase.

4.3 Credentials Design

With the concept of credentials in the data path introduced in the previous section, we turn to the question of what these credentials look like specifically.

4.3.1 Requirements

The requirements for credentials are driven by several conflicting needs:

1. Security Requirement: In order to provide a secure network infrastructure, it is crucial that credentials are only available to authorized traffic in the network. Therefore, credentials should be difficult to fake.

2. Performance Requirement: Since credentials need to be validated for every packet on every router, it is necessary that credentials can be validated with low computational requirements.

3. Size Requirement: Since credentials for every router along the path of a connection need to be carried in each packet header, it is crucial that total size of all credentials is limited.

While the first requirement can be addressed by traditional cryptographic solutions, it is the second and third requirements that pose a novel set of challenges. As networks connect an increasing number of embedded devices (both as end-systems and as intermediate hops), power constraints are becoming increasingly important. Cryptographic operations require several orders of magnitude more operations than conventional packet processing and thus need to be limited to the initial connection setup.

An implication from the third requirement is that it is not practical to simply chain all credentials in the header of the packet. A limit on the header size would constrain the maximum hop count along a path (or the size of the multicast tree). Therefore, we seek a solution where credentials can be represented by a single fixed-length data structure. In addition, chained credentials (e.g., as used in SIFF [128]) are also vulnerable to an attack where routers are incrementally probed (similar to how traceroute works). In each step, an attacker only needs to try a few possible bit combinations until a router can be bypassed and the next one can be probed. Thus, chained credentials may not meet the first requirement.

4.3.2 Bloom Filters based Credentials Data Structure

To meet the above requirements, we introduce a credentials data structure that is based on Bloom filters. The main idea is that this data structure can maintain multiple credentials at the same time. When the packet is transmitted, each router can check if its own credentials are present in the data structure and thus validate the packet.

We briefly review the concept of Bloom filters to provide context for our work. Bloom filters can be used to store membership information [24]. Specifically, a Bloom filter is a bit array that can store m bits. Using k different hash functions $h_1(x) \dots h_k(x)$, an element x is mapped to k bit position in the array. An empty Bloom filter data structure starts with all array values set to 0. When adding element x, the bits corresponding to the hash function values for element x are set to 1. As multiple elements are added, it is possible (and intended) that set bits overlap (i.e., are combined with a logical OR function). When performing a check for membership of an element, the hash functions for the element are computed and it is checked if the according bits in the array are set. Only if all of these bits are set to 1, the element is reported to be a member of the set. The membership test is of a probabilistic nature and false positives are possible (i.e., elements that are not members of the set may be reported to be members), but false negatives are not (i.e., elements that are members of the set will never be reported as not being members). One of the properties of a Bloom filter is that it is not possible to perform a reverse operation where the list of members is extracted from the Bloom filter data structure.

Since the data structure allows that set bits from different elements can collide in the array, it is possible that an element that is not a member of the set may be reported as being a member. This occurs when the hash functions of this element map to bits that have been set by other members in the array (i.e., k collisions). The probability of this occurring increases as more members are added to the set (i.e., n

increases and thus more bits are set). By using a larger array (i.e., larger m) this probability can be decreased. We derive the exact value for this probability in the security analysis in Section 4.4.

4.3.3 Credentials Aggregation

To use the Bloom filter data structure as data path credentials for packets that traverse the network, we store credentials from each router along the path. As explained previously, the source node of a connection negotiates permission to transmit data across a router during connection setup. When router j ($1 \leq j \leq n$) permits transmission, it provides the source with its router credentials r_j. Router credentials are the set of indices $r_j[i]$ ($1 \leq i \leq k$) of bits that are set in the Bloom filter array. The credentials from all routers along the path are then superimposed (i.e., logical OR operation) in the Bloom filter data structure. This creates aggregate credentials c (consisting of a single bit array of size m) that are sent with each data packet. This process of creating credentials is illustrated in Figure 4.9.

When receiving a packet with aggregate credentials c, router j can then check the value of all bits that were provided in router credentials r_j. If the aggregate credentials are valid, then

$$\prod_i c[r_j[i]] = 1, \tag{4.1}$$

where the product is the equivalent of a logical AND operation. If the aggregate credentials do not contain the router credentials of a particular router, it is likely that one of the bits in credentials c does not contain a 1 at one of the router credentials' bit positions. Thus the validation of the aggregate credentials fails. This argument, of course, is of a probabilistic nature. A router may accept a packet that does not have correct credentials with the same probability as a false positive appears in the Bloom filter. However, packets are only successfully delivered to a destination if *all* routers let them pass. Thus, a packet with invalid credentials would need to encounter a false

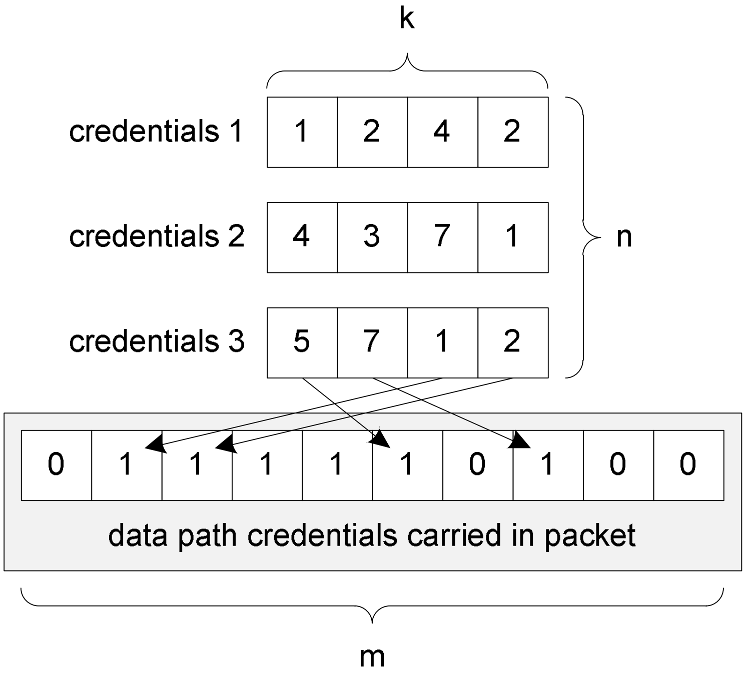

Figure 4.9. Credentials Data Structure. This example shows three credentials that are aggregated to a set of 1's in the Bloom filter data structure.

positive on every router along the path. This probability decreases geometrically with the number of hops in the path and thus is practically very small (see Section 4.4).

4.3.4 Credentials Security

The security of the network architecture depends on the security of the credentials. That is, it should not be practically feasible to generate fake credentials for attack traffic. In the context of Bloom filter credentials, it should therefore be difficult to guess which bits are set by any given router. We can achieve this by using cryptographically strong hash functions (e.g., SHA-1 [43]) where router j uses k secret keys $s_j[i], 1 \leq i \leq k$. The cryptographic hash function $h_i(s_j[i], f)$ uses router j's key for bit index k to determine which bits to set in the aggregate credentials. It is important that this function also uses flow identifier f (e.g., based on a 5-tuple hash) as an input to avoid attacks where credentials from an authenticated connection are used by a different connection. If router credentials are used by a different connection, the validation step (Equation 4.1) fails.

Credentials based on cryptographic hash functions and flow identifiers ensure the following properties:

- Data path credentials for different flows are different (even if they traverse the same set of routers) because the use of f as parameter in the hash function creates different router credentials.

- Data path credentials for flows that traverse different routers are different, because a different set of router credentials (each depending on s_j) are superimposed in the data path credentials.

- Data path credentials are difficult to fake since the result of the cryptographic hash function h_i cannot be guessed without availability of keys s_j. Also, credentials cannot be reversed to obtain hash keys.

- While the generation of credentials is computationally expensive ($n \times k$ cryptographic hash operations), credential check operations are simple. Credentials can be checked by performing k lookups in credentials c to verify Equation 4.1. Note that this requires that each router remembers the router credentials r_j for a particular flow. This is done by maintaining the credentials cache shown in Figure 4.4. If the credentials for a flow cannot be found in this cache, the router credentials can be recalculated (using s_j and f) at a higher computational cost.

- Data path credentials are of small and constant size since all router credentials r_j can be superimposed into a single Bloom filter data structure.

With these key properties of data path credentials, it is possible to provide security features on the network architecture level as discussed in Section 4.4.1. A more detailed discussion on how security requirements are met is provided in Section 4.5.2.

4.3.5 Density Limit

One important observation regarding credentials as described above is that there exists a very simple attack to circumvent a credentials check: an attacker could set all bits in the credentials to 1. Such credentials would always satisfy Equation 4.1, no matter what secret keys or flow identifiers are used. This is clearly an undesirable property.

In order to make data path credentials immune to this attack, we introduce one additional concept to our Bloom filter. We define a "density" metric $d(c)$ that reflects the number of 1's in credentials c as a fraction of the total size:

$$d(c) = \frac{1}{m} \sum_i c[i]. \tag{4.2}$$

To consider credentials valid, we require that the density is equal or below a certain threshold: $d(c) \leq d_{max}$. If the density is higher, we assume the credentials to be invalid and thus reject the packet. If the threshold is chosen to be too low, even valid credentials may be rejected. In Section 4.4, we derive an equation that allows us to estimate the expected number of set bits in the credentials data structure based on the number of routers involved.

4.3.6 Group Credentials

As illustrated in Figure 4.10, providing credentials for groups of routers can reduce the connection setup overhead. To implement such credentials, all routers in the group simply share the same secret keys. Router credentials issued by any router in the group sets the correct bits in the aggregate credentials to ensure that all other routers in the group let the packet pass. If the end-system is not aware of the grouping of routers, it negotiates router credentials with each router individually. Since the router credentials from all routers in the group are the same, the resulting aggregate credentials will have the same bits set.

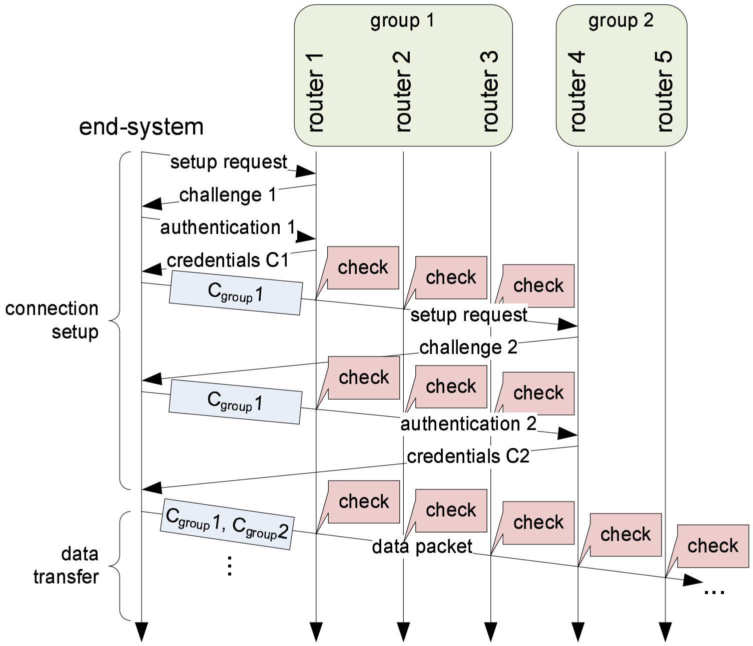

Figure 4.10. Connection Setup using Group Credentials.

4.4 Security Analysis

The goal in this thesis is to develop a virtual network instance that is dominated by strict security requirements. As explained above, we expect such networks to be deployed in parallel to the existing Internet (either through virtualization or through use of a dedicated infrastructure). Thus, we can design security requirements that are necessary for such domain-specific networks, but would be infeasible or inefficient in the conventional Internet.

4.4.1 Security Requirements

We consider the following security requirements:

- Prevention of unauthorized network access and traffic injection: Only authorized users should be able to establish a connection in the network and send traffic.

- Detection of packet header spoofing: An unauthorized user should not be able to spoof packet headers to impersonate another entity.

- Isolation of denial-of-service attacks: Sources of denial-of-service attacks should be identifiable to isolate the attack.

- Intrusion prevention: Connections to end-systems should only be allowed on explicitly specified ports (e.g., to avoid port-scans).

- Extrusion prevention: Connections from end-systems should be controllable to deny extrusion attempts (i.e., security breaches where sensitive data is transmitted from within a network).

In the context of these security requirements, our main focus is on authorization and availability by ensuring that only packets that have been positively identified are forwarded in the network. While this ties in closely with access control, confidentiality, and integrity, we discuss how these latter issues are already addressed through the use of existing key management and cryptographic solutions.

4.4.2 Attacker Capabilities

The capabilities of a potential attacker are assumed to be the following: (1) ability to read any packet traversing an attacked router; (2) ability to modify any packet traversing an attacked router; and (3) ability to send any packet from the attacked router.

Additionally, we constrain the capabilities of the attacker as follows: (1) an attacker does not have access to secret key material associated with an identity other than themselves; (2) an attacker cannot drop all or a subset of network traffic on a router (i.e., black holing); (3) an attacker's access to links and nodes is limited such that the network cannot be partitioned.

The limitations on the attacker's capability are necessary to keep the discussion of attack scenarios and security requirements within scope. Related work has addressed techniques that can provide environments where such assumptions are reasonable. For example, secure key generation and storage can be achieved with a cryptographic coprocessor [103] and black holing can be circumvented by using multi-path routing and network coding [41]. The assumption that an attacker cannot partition the network is necessary to ensure that some valid communication between nodes is possible.

With an understanding of the general concept of data path credentials and the specific design of credentials based on Bloom filters, we evaluate the quantitative security properties of this architecture. It is important to quantify these security properties in order to evaluate specific system configurations. In this section, we analyze the probabilistic guarantees that Bloom filters provide and present results in the context of specific usage scenarios. We also discuss the design's resilience against DoS attacks and how security requirements are met.

4.4.3 Probability of Successful Attack

The main goal of our data path credentials architecture is to identify valid traffic and thus not allow the transmission of attack traffic. Since a Bloom filter can yield false positives, it is possible that traffic with forged credentials may pass through the network. This false positive probability can be exploited by an attacker. Thus we need to obtain a quantitative understanding on how likely this attack is for different system configurations. This problem is related to the Generalized Birthday Problem (GBP) [117], but differs in that the GBP only considers a single false positive. In the case of data path credentials, we need false positives on every hop of the path for a successful end-to-end attack.

The best attack from an attacker's point of view is to send credentials with as many bits set as possible. The more bits are set, the more likely the validation step described in Equation 4.1 succeeds. The limit to the number of bits set in credentials is given by the maximum density d_{max}. Since it is not possible to set all bits in the credentials, the attacker needs to decide which ones to set. The credential system uses cryptographic hash functions, for which it is reasonable to assume that they generate pseudo-random outputs. Thus, there is no structure that the attacker can exploit. Thus, choosing a random set of bits for the attack credentials is as good a choice as any other combination.

- **Unicast**:

  In the *unicast* scenario, attack traffic needs to traverse n hops from source to destination and pass credential checks on each hop. When validating credentials, a router checks if all k bits of its credentials are set. To reach the destination, all bits set (in valid credentials) are checked at least once. Thus, an attacker has to be able to create forged credentials with at least those bits set. The number of bits set, $b(m, k, n)$, in a Bloom filter of size m with k hash functions and n stored items is (as derived below):

  $$b(m, k, n) = m \cdot \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right). \tag{4.3}$$

  Here, we derive the estimated number of bits, $b(m, k, n)$, that are set in a Bloom filter of size m with k hash functions when n elements have been entered (see Figure 4.9 for an illustration of the parameters). We first determine the probabilities for 0's and 1's in the Bloom filter data structure. The probability that a bit is not set by a single hash function depends on the size (i.e., m) of the Bloom filter data structure:

  $$P[\text{bit not set by single hash function}] = 1 - \frac{1}{m}.$$

  When using k hash functions, the probability that a bit is set by none of these hash functions is

$$P[\text{bit not set by } k \text{ hash functions}] = \left(1 - \frac{1}{m}\right)^{k}.$$

Note that for this analysis we assume that hash functions yield independent and uniformly distributed hashes (as it is the case for cryptographic hash functions). With n elements in the Bloom filter (i.e., n router credentials aggregated in credentials c), a bit in c is not set with probability

$$P[\text{bit not set by } n \text{ elements}] = \left(1 - \frac{1}{m}\right)^{kn}.$$

Accordingly, the probability that a bit is set to 1 in the aggregate credentials is:

$$P[\text{bit set by } n \text{ elements}] = 1 - \left(1 - \frac{1}{m}\right)^{kn}.$$

This probability applies to all m bits in the Bloom filter. Thus, the expected number of bits set to 1 in a Bloom filter of size m with k hash functions and n elements is thus

$$b(m,k,n) = m \cdot \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right),$$

which is the result used in Equation 4.3.

A false positive transmission (i.e., a successful attack) occurs when among the bits set by the attacker (limited by d_{max}), there is the correct set of $b(m,k,n)$ bits that is checked by the set of routers along the path. The probability for this event is:

$$f_{unicast}(m,k,n) = (d_{max})^{b(m,k,n)}. \tag{4.4}$$

The expected number of bits set in the credentials data structure also gives an estimate on the longest path that can be supported by a particular configuration. If the expected number of bits set exceeds the density threshold, then even valid credentials may be rejected (i.e., false negative). Valid configurations of m, k, and n must meet

$$b(m, k, n) \leq d_{max}. \tag{4.5}$$

- **Multicast**:

In the *multicast/multipath* scenario, the source needs to aggregate router credentials from all routers along all paths to all destinations. For simplicity, we assume that multicast is performed along a balanced and complete binary tree where each node corresponds to a router that duplicates the packet and sends it to two more nodes. The height of the tree, h, relates to the number of leaf nodes (i.e., multicast destinations), l, as follows: $2^{h-1} < l \leq 2^h$ or $h = \lceil \log_2 l \rceil$. The number of internal nodes in such a tree corresponds to the number of routers n that are encountered when multicasting: $n = 2^h - 1$ or $n = 2^{\log_2 l} - 1 = l - 1$. Thus, $2^h - 1$ router credentials have to be aggregated in the Bloom filter and the resulting number of bits set in the credentials is $b(m, k, 2^h - 1)$. This limits the set of valid configurations to

$$b(m, k, 2^h - 1) \leq d_{max}. \tag{4.6}$$

The exponential increase in the number of aggregated credentials requires a much larger Bloom filter data structure. However, the size for this data structure grows less than exponential due to overlapping hash indices. To determine the probability of false positive transmission of attack traffic we need to consider all $l = 2^h = n + 1$ multicast paths. The false positive probability is then:

$$f_{multicast}(m, k, n) = 1 - (1 - f_{unicast}(m, k, h))^{n+1}. \tag{4.7}$$

- **Network Coding**:

  For *network coding*, the analysis is similar to that of multicast. Each connection starts sending aggregate credentials with $2^h - 1$ router credentials. When generating a network coded packet, routers in the network aggregate credentials from multiple connections. Thus, by the time a packet reaches its destination, it may have gained credentials on every but the last hop along the path. (Note: For simplicity, we assume that coding is done only across two packets at any node.) Thus, there may be a total of $n = (h-1) \cdot 2^h - 1$ credentials combined in the packet. Thus, m and k need to be chosen suitably such that

$$b(m, k, (h-1) \cdot 2^h - 1) \leq d_{max}. \tag{4.8}$$

  To determine the false positive probability, we need to consider how many packets have to be received by a node such that the network coding can be reversed. If we code packets on each of $h-1$ hops, then 2^{h-1} packets need to be received by the receiver for successful decoding. This corresponds to successfully achieving $h-1$ false positive $(h-1)$-hop multicast transmissions and a 1-hop unicast transmission. (Due to the structure of network coding, one hop along the path cannot be multicast.) Since a tree of height $h-1$ has $\frac{n+1}{2}$ nodes, the false positive probability is then:

$$f_{network\,coding}(m, k, n) = \\ \left(f_{multicast}(m, k, \frac{n+1}{2}) \cdot f_{unicast}(m, k, 1)\right)^{\frac{n+1}{2}}. \tag{4.9}$$

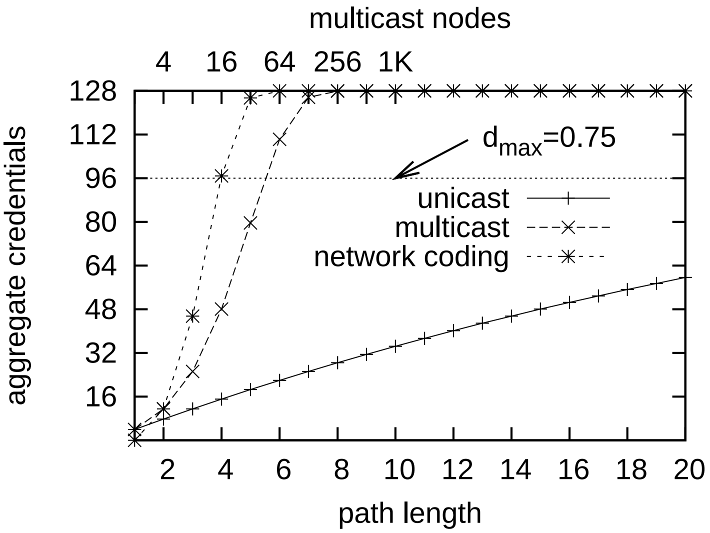

Figure 4.11. Bits Set in Aggregate Credentials.

4.5 Security Performance Evaluation

To illustrate the equations derived in the previous section in the context of a realistic system configuration, we provide security performance results of a few specific system configurations.

Figure 4.11 shows the number of bits set (as determined by $b(m, k, n)$) for credentials of size 128 bits and four hash functions. As the path length increases, the number of bits set by credentials also increases. For multicast and network coding, the increase is much steeper than for unicast since many more credentials are aggregated due to the larger number of paths and destinations. When the maximum density (in this example $d_{max} = 0.75$) is exceeded, credentials are rejected by all routers. Thus, credentials of a particular size should only be used in network configurations where the number of set bits is expected to be below d_{max}. For our example, unicast can support 44 hops, multicast can support 5 hops (32 destinations), and network coding can support 3 hops (8 destinations). It is important to note that this is a very conservative estimate for multicast and network coding as we assume a complete binary tree. In a real system, the number of destination nodes for a given path length is expected to be lower and thus longer paths can be supported.

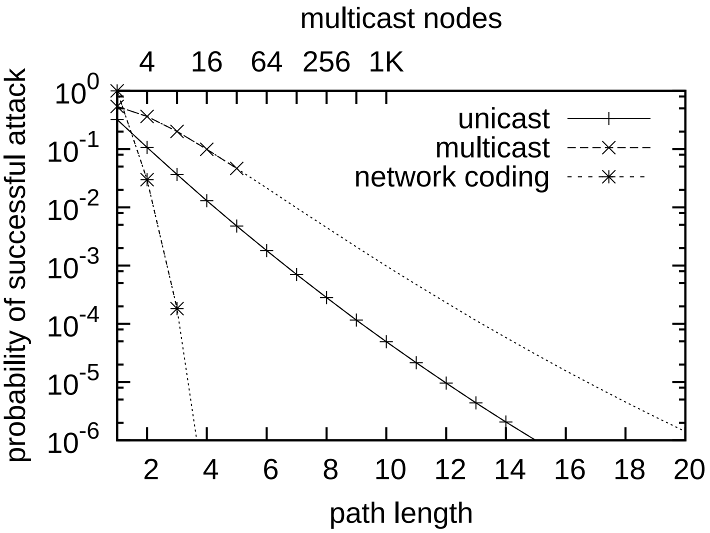

Figure 4.12. Probability of Successful Attack Transmission.

The false positive rates that correspond to this example are shown in Figure 4.12. Data points are only shown for those cases where the maximum density is not exceeded (the dotted lines continue beyond this limit to illustrate the overall trends). The decreasing trend with an increasing number of hops is due to repeated credential checks. The more routers are traversed, the less likely it becomes that random attack credential are validated repeatedly as false positives. The false positive rate for unicast decreases more quickly than that of multicast. Multicast has more source-to-destination paths and thus more opportunities to create a false positive transmission. For network coding, the opposite happens. Since coded packets can only be decoded when other coded packets are received, the probability of false positives drops even faster than with unicast.

Clearly, the maximum path length and the false positive probability depend on the choice of d_{max}. Figure 4.13 shows the trend of the maximum path length for unicast, which increases as expected with increasing values of d_{max}. The probability of false positive transmission of attack traffic shows a clear minimum around $d_{max} = 0.4$. For lower values of d_{max}, the overall path length is so short that high false positives may occur (as shown in Figure 4.12). For larger values of d_{max}, a large number of bits may

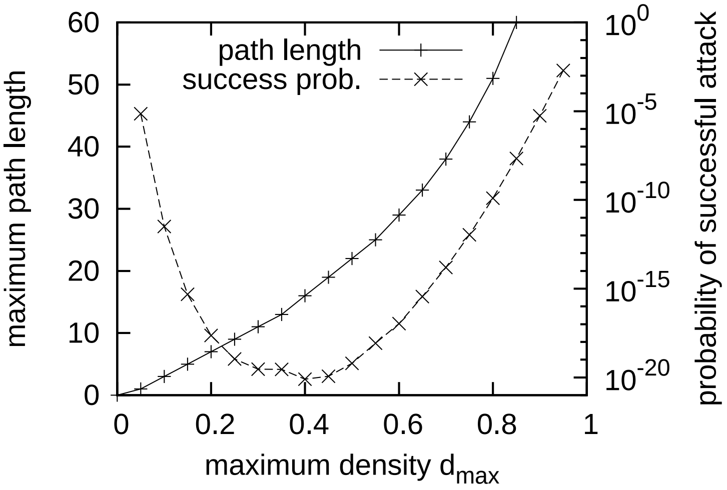

Figure 4.13. Maximum Path Length and Probability of Successful Attack for Different Density Limits in Unicast.

be set in credentials, which also leads to higher false positive rates. For multicast and network coding, similar trends can be observed (not shown). For a practical implementation, a balance between longer paths and less attack vulnerability needs to be found that is suitable for all types of connections.

The capabilities of different configurations for m and k are shown Tables 4.1 and 4.2. The maximum path length (assuming $d_{max} = 0.75$) is shown in Table 4.1. The probability of a false positive end-to-end transmission with randomly generated credentials is shown in Table 4.2. In general, a smaller number of hash functions works well as it does not cause the Bloom filter to fill up as quickly. However, fewer hash functions also mean that fewer bits are checked on each router. This effect can be seen in the multicast case where high attack success probabilities appear for $k = 2$.

Overall, the results show that data path credentials can detect attack traffic within a small number of hops, even when a large number of paths are involved (e.g., in the multicast scenario). The overhead for implementing such capabilities depends on the chosen size of the credentials, but could be practically as small as 64 bits for unicast over up to 20 hops.

Table 4.1. Maximum Path Length for Different Configurations of Credentials Size (n) and Number of Hash Functions (k).

k	n											
	unicast				multicast				network coding			
	32	64	128	256	32	64	128	256	32	64	128	256
2	21	44	88	177	4	5	6	7	3	3	4	5
4	10	22	44	88	3	4	5	6	2	3	3	4
8	5	11	22	44	2	3	4	5	2	2	3	3
16	2	5	11	22	1	2	3	4	1	2	2	3

Table 4.2. Attack Success Probability for Different Configurations of Credentials Size (n) and Number of Hash Functions (k).

k	n											
	unicast				multicast				network coding			
	32	64	128	256	32	64	128	256	32	64	128	256
2	.001	.000	.000	.000	.662	.507	.361	.243	.043	.040	.004	.000
4	.001	.000	.000	.000	.284	.125	.047	.016	.034	.000	.000	.000
8	.001	.000	.000	.000	.098	.018	.002	.000	.001	.001	.000	.000
16	.003	.000	.000	.000	.050	.003	.000	.000	1.00	.000	.000	.000

4.5.1 Denial of Service Attacks

One important issue to consider in the context of capability-based networking is the susceptibility of the system to denial of service attacks. Argyraki and Cheriton have argued that capabilities simply shift vulnerabilities from the data path to the control path [10]. In systems where no data path checks are performed as part of the control setup, this problem indeed exists. In our architecture, however, it is possible to contain DoS attacks such that they do not affect the entire network. We require that control traffic also obtain credentials for setting up capabilities further down the path (as shown in Figure 4.5). If a malicious node tries to attack the control infrastructure, it is limited to attacking only immediate neighbors. Since these systems do not issue suitable credentials without proper authentication (which is usually not the case in DoS attacks – otherwise independent service limitation and policing can be employed), the attacker cannot reach other routers farther away. Thus, the DoS attack is contained to immediately surrounding nodes and decreases geometrically in strength as the probability of multiple false positive credential checks decreases along the path. This difference in design is an important aspect for a practical deployment and successful use of data path credentials.

Beyond brute force denial of service attacks, more complex attacks based on low bandwidth, well-timed traffic injections have been proposed [65]. These types of attacks are generally difficult to detect in any network. In our architecture, these attacks can be avoided if they are launched from a non-authorized end-system (as this traffic gets dropped due to failed credentials checks). If the attack is launched by an authorized end-system, there is no apparent defense other than through access control.

4.5.2 Validation of Security Requirements

With an understanding of the qualitative and quantitative security properties provided by the credential-based data path architecture, we can revisit the security requirements stated in Section 4.4.1. Using credentials, a router can audit every packet in the data path and validate that it indeed is eligible to be forwarded. Credentials identify a packet in terms of its source and destination (e.g., machine, user) and its path (i.e., set of all credentials). The connection setup process can be used to enforce policies (e.g., access control) in order to control what network communication is permitted. With a suitable choice of credentials size and maximum path length, the probability of a successful attack due to false positives in the Bloom filter can be kept diminishingly small.

Thus, data path credentials can be effectively used to meet the stated security requirements:

- Prevention of unauthorized network access: All end-systems need to identify themselves properly to all routers along the path, where access permissions can be validated. If an end-system does not properly identify itself or does not have the appropriate permissions, no suitable credentials are issued. Thus, packets sent by such an end-system are dropped at or close to the first router.

- Detection of packet header spoofing: Credentials are issued based on the end-system identity. If packet headers (i.e., source addresses) are spoofed, then a router looks up the wrong set of credentials (due to a different flow identifier) and drops the spoofed packet. This happens for any spoofing of header fields used in the flow classification process.

- Partial prevention of traffic injection: Based on prevention of unauthorized access and header spoofing, it is not possible for an attacker to inject traffic from any node that is not along the path of an existing connection. However, it is possible to inject traffic if a node along the path is compromised since the credentials used in a connection are visible and thus can be copied for the attack traffic. The impact of this type of traffic injection can be mitigated through the use of end-system security protocols that require end-system keys to correctly encrypt packet payloads.

- Isolation of denial-of-service attacks: One of the main benefits of data path credentials is the ability to limit the impact of denial of service attacks. The use of credentials ensures that either the source of traffic can be identified (since correct credentials have been issued) or attack traffic can be isolated (since packets without correct credentials are dropped at or close to the first hop).

- Partial intrusion prevention: Data path credentials can be extended to not only include routers but also the receiving end-system. The system can limit, for example, to which port traffic can be sent. Thus, data path credentials act comparably to a firewall. This approach, however, does not protect against intrusion on a port that is "open" on the end-system.

- Partial extrusion prevention: In order to prevent an end-system from transmitting data to particular destinations, routers along the path can be configured to deny this communication (i.e., to not issue credentials when requested). This approach, however, does not prevent extrusion via relay nodes.

In addition, data path credentials can be used in conjunction with conventional security and cryptographic protocols (e.g., IPSec, SSL) to further improve network security.

4.6 Protocol Implementation

We have designed and prototyped a protocol that implements the functionality of data path credentials as discussed above. This *Data Path Credentials Protocol (DPCP)* is located between the network layer and the transport layer of the Internet protocol stack. In principle, it is possible to integrate data path credentials with a new network layer protocol. However, we do not consider the redesign of the Internet Protocol in this work. Instead, we leverage the functionality of the existing IP protocol and add DPCP on top of it.

4.6.1 Data Path Credentials Protocol

The header for DPCP is shown in Figure 4.14. The overall header is 28 bytes in size and is based on a Bloom filter configuration of $m = 128$ and $k = 4$. The fields in the header are used as follows:

- Port numbers: The first 32 bits are port numbers identical to how they are used in TCP and UDP. These fields are mere copies of the values that are carried in the layer 4 header of the packet (which is assumed to be either TCP or UDP to allow for packet classification). It can be debated if it is a “cleaner” design to copy these values or to let DPCP read the layer 4 protocol header. If the compactness of the header is important, these fields could be omitted in an optimized implementation.

- Next protocol: This 8-bit field indicates the layer 4 protocol header in the packet. This field is identical to the next protocol field in the IP header. (For our implementation, the IP header indicates 253 for DPCP, which is reserved for experimental protocols.)
- Flags: The flags are used in our protocol implementation to indicate packet types used during connection setup. The types used are the following:
  - Setup flag (S): Indicates a packet containing a setup request.
  - Challenge flag (C): Indicates a packet containing a challenge to an end-system.
  - Response flag (R): Indicates a packet containing a response to a challenge.
  - Credentials flag (I): Indicates a packet containing Bloom filter indices.
- Setup field: This field can be used in different ways for establishing connections. The use is identified by the flags.
  - Challenge nonce (32 bits): This nonce is used by the router to challenge the end-system.
  - Response (32 bits): The end-system sends this encrypted nonce to prove its identity.
  - Bloom filter indices (28 bits): These four indices (each with a size of $\log m = \log 128 = 7$ bits) indicate the bits that are set by a router as its credential in the Bloom filter.
- Data path credentials: This 128-bit field is the Bloom filter that carries all router credentials.

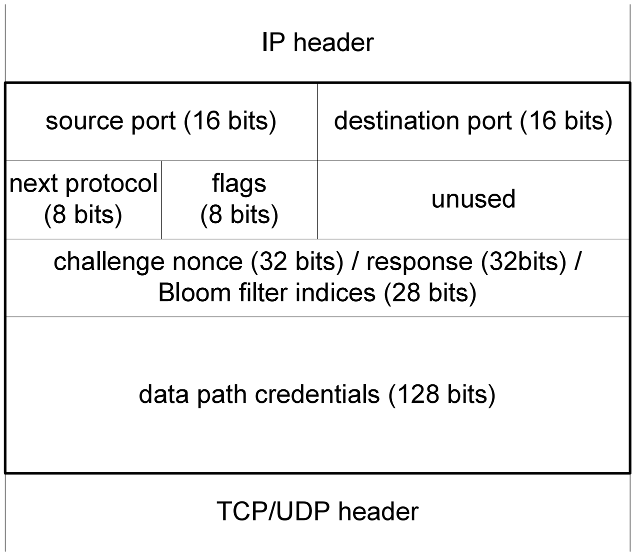
IP header
source port (16 bits)
destination port (16 bits)
next protocol (8 bits)
flags (8 bits)
unused
challenge nonce (32 bits) / response (32bits) / Bloom filter indices (28 bits)
data path credentials (128 bits)
TCP/UDP header

Figure 4.14. Data Path Credentials Protocol Header.

Note that the Bloom filter indices are separate from the Bloom filter itself. The reason for this design is that during connection setup, the Bloom filter is used to permit communication to and from the router from which credentials need to be obtained. The credential itself is carried in the Bloom filter indices. Since these indices are not used after connection setup, an optimized implementation may use two different header formats for setup and data transfer.

4.6.2 Router Processing

When DPCP packets arrive on a router, it needs to be distinguished if they belong to a connection setup request to this router or if they should be forwarded. Note that a router needs to forward connection requests that are directed at other routers along the path. To make this distinction, the router simply checks if the data path credentials carried in the packet are valid (see Figure 4.15). If they are, then there is no need to further consider the packet locally. (Note that this is also true in case the credentials of previous routers in the path accidentally cause a false positive in the Bloom filter.) In case of a setup packet, the appropriate challenge is sent to the end-

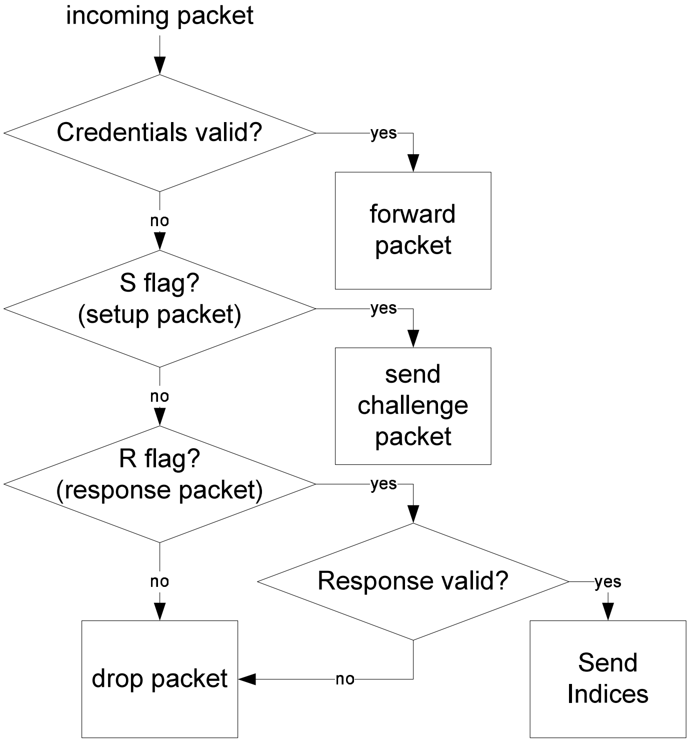

Figure 4.15. Decision Diagram for DPCP Processing on Router.

system. In case of a response, it is verified and the Bloom filter indices are returned to the sender. In all other cases, the packet is dropped.

4.6.3 Bidirectional Verification

One important practical consideration is that most communication in the Internet is bidirectional. Even in the setup phase, the challenge and credential packets need to reach the sender. To avoid that routers have to set up (and store) credentials for this return path, we set up the system to use the same data path credentials for both directions of traffic. (Note that we assume symmetric routes.) Thus, the challenge and credential packets simply carry the same data path credentials that were carried in the original packet.

To implement this type of bidirectional verification, it is necessary to modify the flow identification process. In our system, the classification result of a 5-tuple of a connection in one direction should match that of a connection in the opposite

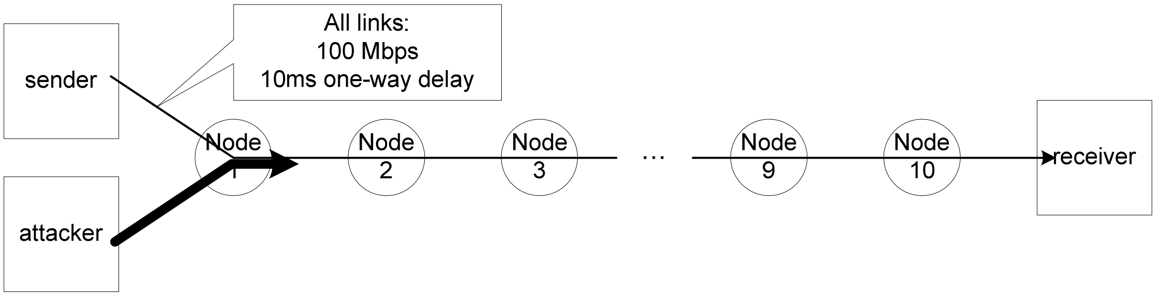

Figure 4.16. Experimental Emulab Setup for Evaluation of DPCP.

direction. We achieve this by sorting the IP addresses and port numbers before providing them to the classifier. Thus, when the IP addresses and port numbers are swapped on the return path, the sorting ensures that the classification result is still the same.

4.7 Evaluation of Emulab Implementation

We have implemented the DPCP protocol on Emulab [119]. We modified the Linux kernel (version 2.6.21.4) of Emulab end-systems and routers to implement the DPCP protocol operations. In our implementation we set the credential size to 128 bits (m=128) and the density threshold to $d_{max} = 0.3125$ (which corresponds to 40 out of 128 bits). For challenge/response authentication, every node was set up with a key pair and public-key/private-key cryptographic operation were performed using 1024-bit RSA-based encryption. To determine the Bloom filter indices for credentials for a flow, routers performed four SHA-1 hash functions (i.e., k=4). For our experiments, we used the simple 10-hop topology shown in Figure 4.16. Each link has a bandwidth of 100 Mbps and a one-way delay of 10ms.

4.7.1 Attack Containment

The most important result from our prototype implementation is that DPCP successfully contains attack traffic. Figure 4.17 shows the amount of traffic that passes each hop with an attacker sending traffic with random credentials (with the highest

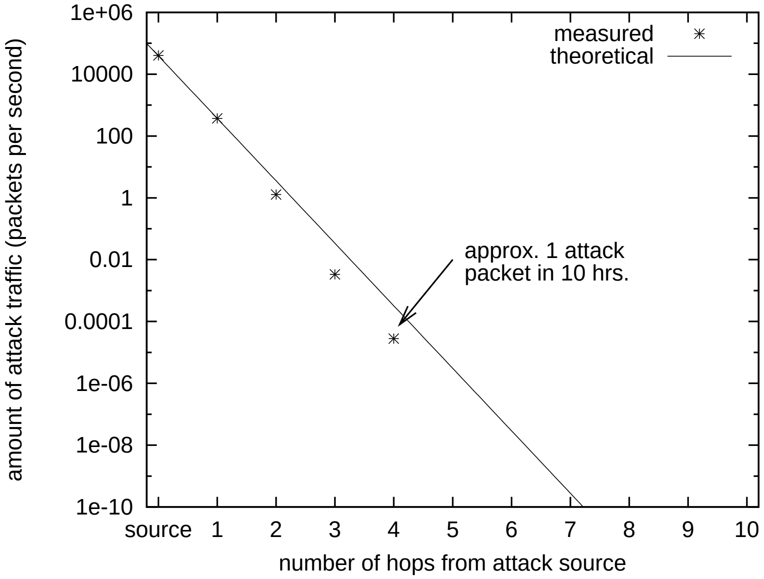

Figure 4.17. Containment of Attack traffic with DPCP.

permissible density of d_{max}). Since the attacker does not know which credentials are acceptable, only packets that have the correct bits set in the Bloom filter by coincidence are forwarded. The attack source sends at a high rate of approximately 40,550 packets per second. The first hop only forwards on average 372 packets per second (0.92% of the attackers traffic). This traffic is further reduced at the next hop (1.3 packets per second or 0.003% of original attack traffic), etc. During the 10-hour experiment to generate these data, only a single packet out of 1.46 billion made it past the fourth hop.

This result shows that the proposed protocol is highly effective in containing unauthorized traffic. The vast majority of all attack traffic (99.08%) is contained within a single hop of the attack source. Practically no traffic reaches any node that is 5 hops or more away from the attacker.

One key question is what impact this 1-hop containment has on other traffic in the network compared to existing credentials-based systems. Therefore, we consider the throughput performance of TCP cross-traffic as shown in Figure 4.18. We compare two different system configurations:

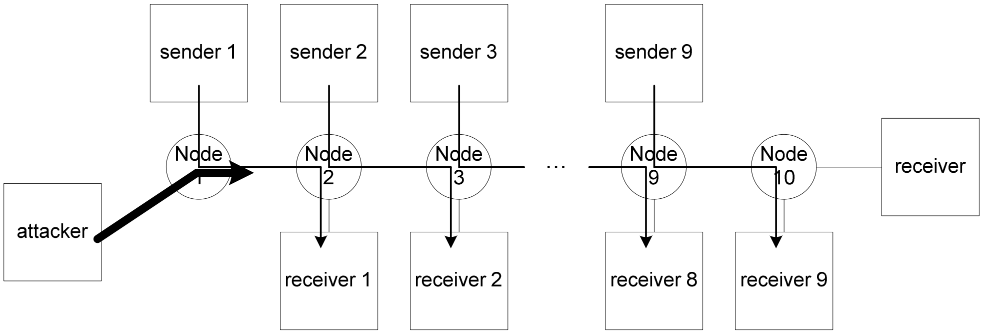

Figure 4.18. Experimental Emulab Setup for Evaluation of Cross-Traffic Performance.

- Existing credential-based network ("conventional"): In this setup, Node 9 performs credential checks and removes all attack traffic. This configuration corresponds to credential-based networks that only verify traffic at some nodes in the network.

- Data path credentials with 1-hop containment ("DPCP"): In this setup, credentials are verified with DPCP at every node in the network.

We assume Node 1 is the source of attack traffic that is destined to the receiver. The TCP throughput performance of cross-traffic on each link is shown in Table 4.3. In the conventional credential-based network, attack traffic affects the throughput performance of benign traffic all the way to the point where the credential check is performed. In our experiment, all connections on links between Node 1 and Node 9 drop 1–4% of their baseline performance. In contrast, DPCP can filter most traffic within the first hop and only traffic on the link between Node 1 and Node 2 is significantly affected. Even on the link between node 2 and Node 3, 95% of baseline performance can be achieved. For links farther away from the attack, 99–100% of baseline performance is achieved. These results show very clearly the ability of our system to protect the network infrastructure from DoS attacks, in addition to protecting end-systems.

Table 4.3. Comparison of Cross-Traffic Performance for Conventional and DPCP Credential-Based Networks.

Cross-traffic	Baseline	Conventional		DPCP	
link	Mbps	Mbps	% of basel.	Mbps	% of basel.
1–2	78.2	1.07	1%	1.04	1%
2–3	79.1	1.15	1%	74.9	95%
3–4	78.8	1.75	2%	77.9	99%
4–5	77.2	1.95	3%	76.5	99%
5–6	79.4	2.60	3%	79.1	100%
6–7	74.0	2.86	4%	73.4	99%
7–8	72.9	2.83	4%	72.6	100%
8–9	72.8	2.86	4%	71.8	99%
9–10	71.2	70.0	98%	71.5	100%

4.7.2 Connection Setup Overhead

Clearly, there are additional costs in using DPCP over conventional TCP. In Figure 4.19, we show the breakdown of the setup time for a single hop. The total time for verification of identities and the generation of credentials takes 11.5 ms in addition to the communication delay. As discussed in Section 4.2, such a setup is required for each hop along the path of a connection. Figure 4.20 shows the time required for a complete connection setup over different numbers of hops. This cryptographic processing delay grows linearly with the number of hops and is unavoidable in any protocol that verifies identities on every hop of the path. The communication delay grows quadratically as challenge and response messages have to traverse an increasing number of hops.

While the non-linear growth of connection setup time is clearly undesirable, it is important to note that it is necessary to ensure 1-hop containment and protection from denial-of-capabilities attacks. Also, this setup cost is only a one-time cost for a connection; it can be amortized over the lifetime of a longer connection and still lead to good throughput performance as can be seen in the next results.

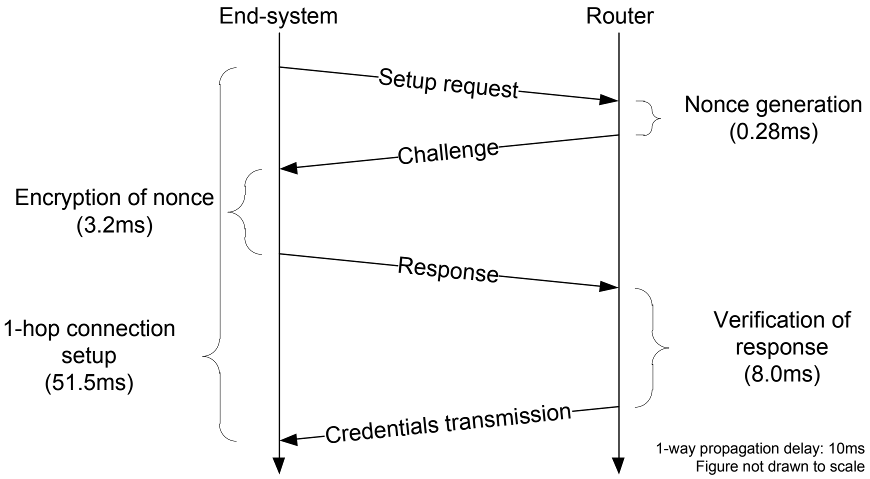

Figure 4.19. Breakdown of Connection Setup Time for Single Hop.

4.7.3 Throughput Comparison

The overall performance comparison of DPCP vs. conventional TCP is shown in Table 4.4. The table lists the completion times for file transfers of different length. Clearly, short file transfers are dominated by the setup cost in DPCP and thus lead to significantly lower performance. However, longer file transfers are only slower by 6%. This reduction in throughput is due to the additional computation that each router performs when verifying credentials. Clearly, such a small overhead is outweighed by the benefits of 1-hop containment shown above.

4.7.4 Flow Rate Performance

A key concern is the rate at which new connections can be established. The main limitation in the data plane is the speed at which cryptographic operations in the connection setup can be performed. In the control plane, there is a limitation due to policy-related computations (i.e., relating identities used in the connection setup with policy rules). We assume that the latter can be precomputed (and recomputed when updates happen) and thus can be performed efficiently.

To demonstrate the rate at which the cryptographic computations necessary in the data plane can be performed, we show Table 4.5. The table shows the achievable flow

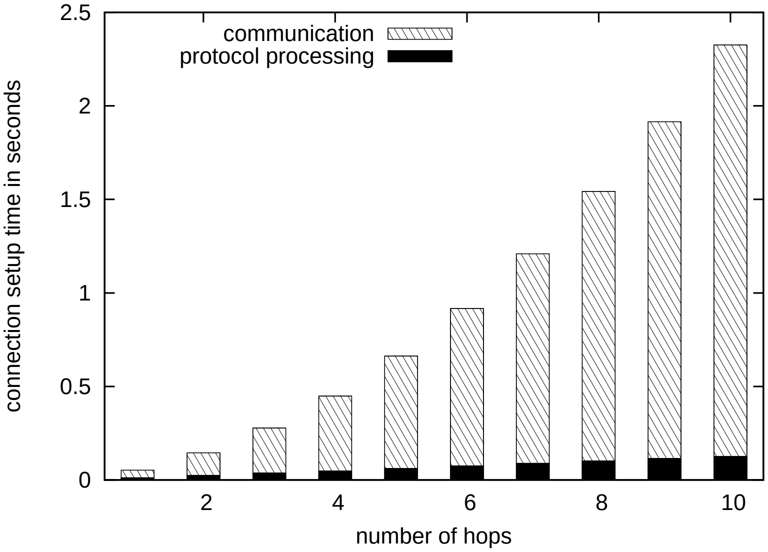

Figure 4.20. Time for Connection Setup in DPCP.

rate performance in a router for different types of processor systems. We have separated the response verification step and the credentials generation step since response verification only occurs once per connection. Credentials verification occurs at least once, depending on how effectively the credentials can be cached. It can be seen that the response verification, which consists of an RSA decryption operation, dominates the processing overhead for connection setup. With conventional processor hardware, multiple tens of thousands of connection setups per second can be supported. Using specialized hardware, such as a graphics processing unit, hundreds of thousands of connection setups per second can be achieved [60]. These rates are sufficient even for core network traffic (and they are expected to continue to increase with future generations of processors). The credentials generation step, which corresponds to SHA-1 computations, can be performed multiple hundreds of thousand times per second and thus does not pose a bottleneck.

The memory that needs to be maintained on a router (i.e., in the credentials cache shown in Figure 4.4) depends on the number of active flows in a system. Each flow record requires 17 bytes of data for a 5-tuple flow identifier and credential indices

Table 4.4. File Transfer Performance.

	File transfer time		
File size	TCP	DPCP	Overhead
10B	0.71s	2.35s	3.30×
100B	0.71s	2.35s	3.30×
1kB	0.72s	2.37s	3.29×
10kB	1.34s	3.22s	2.40×
100kB	2.65s	4.28s	1.61×
1MB	7.91s	10.2s	1.28×
10MB	16.1s	19.4s	1.20×
100MB	48.3s	53.9s	1.11×
1GB	495s	525s	1.06×

Table 4.5. Flow Setup Performance. Response verification and credentials generation per second

	Response verification		Credentials generation
Processor	512-bits	1024-bits	
Intel Xeon	7,491	2,250	351,049
AMD Dual Core 1210	18,582	7,160	461,874
Intel Core 2 Duo	34,780	11,097	660,019
Intel Pentium 4	61,967	17,859	842,909

(assuming $k = 4$). Figure 4.21 shows the effect of the credentials cache size on the maximum packet processing rate that can be achieved in the system. The results in the figure assume 100,000 active flows. The results are based on a system that can compute 842,909 credentials per second in case of a cache miss (from Table 4.5) or look up 10 million credentials per second in its cache in case of a hit. The three lines show the results for different percentages of new flows. A new flow percentage of 100% corresponds to a denial of capabilities attack, where no cache hit is achieved. The figure shows several important results:

- Even without a cache (i.e., size of zero), the system can handle over 800,000 packets per second. In this case, all credentials are verified by computing the indices from scratch.

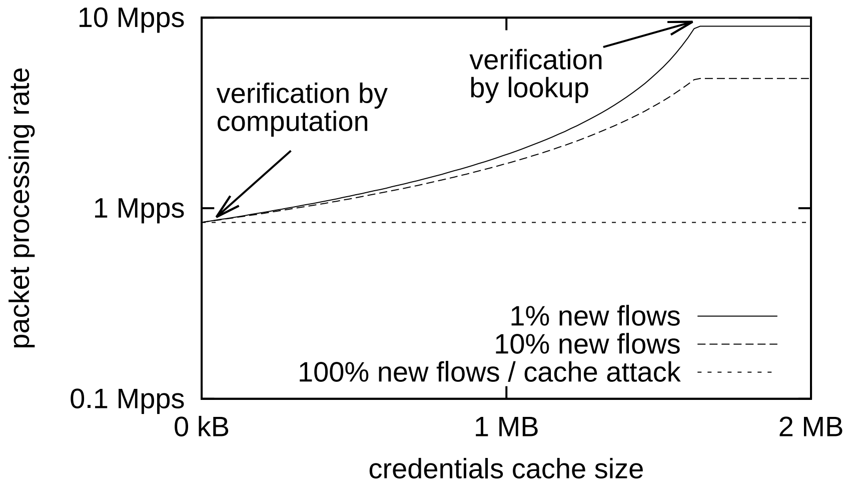

Figure 4.21. Packet Processing Rate for Different Credential Cache Sizes. Total number of active flows is 100,000. Maximum credential lookup rate is 10 million per second.

- Even for a cache attack, where all packets belong to new flows, the system can handle over 800,000 packets per second.

- For cache sizes over 1MB, the system can handle multiple million packets per second. Most verifications can be achieved by lookup and some are done by computation (depending on the percentage of new flows).

Thus, these results show that credential cache memory size can be traded off for computation. More computations are necessary and the throughput degrades when cache hits are smaller (due to small cache size or many new flows), but the system continues to function. In contrast, larger cache sizes can increase the system throughput to the limit that can be achieved by the credentials lookup mechanism. A system with a hundred thousand active flows requires less than two Megabytes of memory to operate in this regime. With current technology, this memory requirement is feasible to implement in a practical system.

4.8 Related Work

Capabilities-based networks, which are one example of specialized networks that focus on security, have been discussed by Anderson et al. [9], Yaar et al. [128], and Yang et al. [130] in the context of DoS attacks. Previously, similar ideas have been proposed by Estrin et al. [48] for controlling packet flows in networks. A system design to avoid denial of service attacks based on the idea of capabilities from [9] has been proposed by Yang et al. [130] where capabilities are used to authorize and verify traffic. In capabilities-based networks, traffic needs to be authorized before it is transmitted, thus providing more control than in the current Internet, where all traffic is allowed by default. In previously proposed capabilities-based networks, capabilities are validated by one or a small number of nodes along the path. In our design, every node validates packets to ensure 1-hop containment of malicious traffic. Early detection and elimination of attack traffic is important to limit its effects on valid traffic that shares the same networking resources. This 1-hop containment is important in conventional networks, but even more so in networks with limited bandwidth resources, such as mobile ad-hoc networks (MANET) [6]. In our architecture, we can identify and squelch most malicious traffic within one hop from its source and thus avoid the consumption of network resources as this traffic is forwarded to its target.

Another capabilities-based system, SIFF [128], classifies network traffic into privileged and unprivileged traffic, where the legitimate (privileged) traffic establishes connection using a capability exchange handshake. However, the length of the capabilities carried in the legitimate network traffic is not constant, which is inconvenient in network protocols. Also the computational overhead for the verification step on the router is high since it requires the computation of a keyed hash function (which is equivalent to the worst-case performance of our system as discussed in Section 4.7.4).

In addition, the security achieved by SIFF is lower than in our system since the capability bits used by each router are much fewer than in our system.

There also exist other variations of network systems that aim to squelch attack traffic. A router-based approach to DoS protection is proposed by Huici and Handley [59], where IP encapsulation is used to tunnel traffic between edge networks. DoS floods can be identified and squelched at the decapsulation point using access control mechanisms. Similarly, Ballani et al. [15] have proposed the use of access control rules to allow individual end-systems to inform the network about which traffic they want (or do not want) to receive. A network architecture to limit LAN traffic is Ethane proposed by Casado et al. [30]. Ethane provides fine-grained network access control for enterprise networks. Access is controlled by per-flow entries in the forwarding table of an edge switch. This design works well for the enterprise scenario, but falls short if access permissions are issued by an entity that does not have direct control over the forwarding table of a switch. Our architecture provides a separation between entities that issue credentials and those that enforce them. Ethane also addresses the general issue of policies and access permissions. In our work, we assume a suitable policy controller to be in place and focus on the issue of how to enforce access control in the data path (see Section 4.2.4).

It has been pointed out that capabilities systems are susceptible to denial of capabilities (DoC) attacks, i.e., denial of service attacks on the capability-granting subsystem [10]. Recently, Parno et al. developed a solution to DoC attacks by using proof-of-work to limit an attacker's capability requests [84]. While our approach is conceptually similar to previous off-by-default architectures, we introduce several novel ideas to make this general idea a practical reality. We consider computationally efficient credentials and show how they can be applied to unicast, multicast, and network coding scenarios. In Section 4.5.1, we show that in our architecture DoC attacks can be isolated to affect only routers close to the source of attack and thus limit the impact on the overall network. A complex proof-of-work scheme is not required since the DoC defense is inherently part of the network design.

Packet marking has been proposed as an alternate mechanism to provide defenses against DoS attacks by tracing back the path of malicious traffic. The marking process can be probabilistic [97] or deterministic [18]. Packet marking allows the identification of a traffic source even if an attacker spoofs protocol addresses. Traceback can also be achieved by extending routers to maintain records of packets that have been forwarded [104]. These audit trails can be examined to determine the source of a packet. Once malicious sources have been identified, they can be actively filtered as proposed in [11]. The process of packet marking, traffic analysis, and explicit blocking is reactive rather than proactive as in the case of capabilities-based networks.

The data path credentials that we propose in this work are based on Bloom filters. Bloom filters were introduced by Burton Bloom in 1970 [24] and found a number of applications in network systems [27,47]. We adapt Bloom filters for the use with what we call credentials. These credentials are derived from cryptographic hash functions such as SHA-1 [43]. The use of hash functions for packet authentication has been proposed by Tsudik in [108], but not in the context of Bloom filters, which require less storage space. We further expand the credentials data structure to consider the density of set bits in the Bloom filter (i.e., the fill level). Scalable Bloom filters have been proposed to circumvent the fill level problem [7], but are not applicable in our work as we need fixed-length credentials to limit packet header sizes. The concept of router virtualization [8] has made it conceivable to deploy domain-specific network architectures in parallel to the existing best-effort Internet. Currently, several specific systems [17, 109] are being developed that could support the types of data path operations we propose in this chapter.

4.9 Summary

In this chapter, a capabilities-based network protocol was presented that uses data path credentials to closely control traffic on every hop. The credentials were based on Bloom filter data structures can efficiently implement capabilities and can provide probabilistic guarantees on permitting only valid traffic to traverse the network. A detailed security analysis that allows a quantitative evaluation of the capabilities of the system for unicast, multicast, and network coding uses was provided. The results show that adding credentials with as few as 64 bits to packets can reduce the probability that attack traffic can reach its destination to a fraction of a percent. In addition, results from a practical implementation of the protocol on Emulab show that less than one percent of attack traffic passes the first hop and the performance overhead can be as low as 6% for large file transfers.

CHAPTER 5

SUMMARY AND FUTURE WORK

In this dissertation, security issues in network virtualization were explored, which points toward an interesting and important new area in network security. In particular, we systematically discuss the relationship between all entities and explore what potential attacks can be launched between each pair of participating entities. In this context, we discuss security requirements and attacker capabilities that underly each network entity. This work presents several ideas and two practical solutions to solve privacy, confidentiality, authorization, and availability of hosted virtual networks.

5.1 Summary

To summarize, the following defense mechanisms were proposed that contribute to the goal of secure network virtualization in the future Internet:

- **EncrIP:** To address the privacy and confidentiality issue, EncrIP, an IP address encryption scheme was presented. The technique uses prefix-preserving encryption and probabilistic encryption to hide address information while still allowing forwarding using unmodified IP headers and commodity routers. A detailed description on the security implication and performance overhead was quantified to show the effectiveness of the EncrIP technique. Our evaluation shows that gateways require only a few MB of memory to implement EncrIP and that in practice the probability of correctly identifying two packets belonging to the same flow is less than 0.001%. The implementation results confirm that the proposed technique introduces low latency and space overhead and is effective against statistical inference attacks. In summary, EncrIP provides a practical solution to privacy in virtualized networks.

- **DPCP:** Next, to guarantee inherent security features that support 1-hop containment of network attacks, DPCP, a capabilities-based virtual network instance was presented that uses high-performance data path credentials. Credentials based on Bloom filter data structure was presented that can efficiently implement the required capabilities and can provide probabilistic guarantees on permitting only valid traffic to traverse the network. A detailed security analysis that allows a quantitative evaluation of the capabilities of the system for unicast, multicast, and network coding was discussed. The experimental evaluation shows that, less than one percent of attack traffic passes the first hop and the performance overhead can be as low as 6% for large file transfers. Since router system can be easily extended to generate and validate credentials in the data path, we believe this design provides a practical solution to provide inherent security capabilities in the network virtualization architecture.

5.2 Future Work: Packet Forwarding Misbehavior Detection

One possible research direction that can be considered for the future work is discussed here. The network entities are subjected to both external and internal attacks and hence it is in the interest of all participating entities to consider the problem of accountability (e.g., violations, anomalies, tampering of resources etc) in the network virtualization architecture. One instance is when the NI components perform the packet forwarding functionality on-behalf of the hosted virtual network. Here, network traffic is forwarded by router components that belong to different administrative entities. In such a scenario, it is important for the VN to identify if the NI components are performing the packet forwarding as specified.

To address the above challenge, we discuss a forwarding misbehavior detection system that identifies if third-party NI components deviate from the expected forwarding behavior. Any malicious or compromised router can drop packets (i.e., black hole attack) or fraction of packets (i.e., gray hole attack) by injecting additive malicious losses, thereby compromising the forwarding behavior. Also, it is important to differentiate between malicious packet loss behavior and congestion-based network conditions. In addition, the technique should ensure that the proposed mechanism has high positive detection rates with effective accuracy and low performance overhead.

5.2.1 Challenges

The following technical challenges should be addressed by the proposed system: Are network infrastructure components exhibiting packet forwarding misbehaviors? How can the system verify or monitor the compromised entity with low false positives (high accuracy)? Does the monitoring mechanism introduce a detection or a prevention scheme in identifying the compromised network entity? Can the monitoring mechanism introduce a verification module that can guarantee the privacy of the VN and NI components? How can the user identify both the control and data plane performance of the network? What are the financial constraints for such identifications?

5.2.2 Related Work

Current compromised router identification mechanisms either involve distributed monitoring approach or propose an ack-based adversary identification technique. A monitoring mechanism that uses conservation of flow principle was introduced in [26]. A failure detector mechanism that uses a combination of source routing, authentication and reliability mechanisms to identify the compromised router component was introduced in [14]. Similarly, [81] uses traffic validation schemes between the source and the intermediate routers to detect faulty or compromised routers. One concern with these distributed monitoring schemes is that the technique incurs significant communication and storage overhead in determining the compromised node.

To optimize performance overhead, trajectory sampling [42] technique for traffic measurements was extended for the malicious router detection problem by [67]. Similarly, “Network confessional” [13] verifies the forwarding performance of network entities using delayed sampling technique. To localize the identification of the compromised entity, a fault localization protocol with lower performance overhead was introduced by [16] and a packet-dropping adversary identification mechanism was proposed in [134]. To identify incorrect packet forwarding behavior, [73] subdivides the problem into traffic characterization and violation detection and takes countermeasure to isolate the compromised component.

While current malicious router detection techniques either focus on optimizing the performance overhead or the detection accuracy, in virtualized networks we require a technique with high detection accuracy, low performance overhead, and localization of the packet-forwarding adversary. To provide such a technique, we introduce a controller-based detection system that can be effective in the network virtualization architecture.

5.2.3 Security Model

We begin our discussion with the security model, explaining the requirements and attacker capabilities in the system. The following set of security requirements ensure the secure processing of network traffic by routers: 1) NI components should forward packets as specified by the VN provider, 2) VN entity should be able to identify the NI entities that introduce malicious packet forwarding behavior, and 3) Malicious traffic originating from the compromised node should be inferred and discarded.

The possible attacker capabilities that can be launched from a compromised NI component are assumed to be the following: 1) The attacker can selectively drop legitimate network traffic, which introduces malicious packet loss behavior by exploiting the congestion control mechanism, 2) Arbitrary network traffic (data and control packets) can be injected from the compromised router, 3) Data packets can be modified to introduce anomalous forwarding and routing behavior, and 4) To hide the adversarial performance, false or compromised proof can be generated for the verification process. Based on the above security model, it is important to design a verification mechanism that ensures the avoidance of such attacks with the following properties:

- Detection Accuracy: The fundamental challenge in developing a forwarding misbehavior detection technique is to provide an accurate detection mechanism with low false positive (identifying congestion-based packet loss as malicious packet drop) and false negative (incorrectly disregarding a malicious packet loss as congestion-based packet loss) detection rates.

- Performance Overhead: Another important challenge is to provide a technique that is effective with respect to the communication overhead required to detect a malicious forwarding behavior (packet loss) and the storage overhead introduces in the system.

- Localization: Also, since a virtual network is hosted on multiple network infrastructure entities, it is important to localize the identification of the compromised component and avoid unjust blaming of NI entities.

5.2.4 Controller-based Detection System

To address the above challenges, a controller-based misbehavior detection system can be introduced as shown in Figure 5.1. The technique of using witness-based forwarding misbehavior detection in wireless networks was shown in [129]. Unlike the witness-based model, which chooses the witness nodes from the set of observing neighbor nodes, we require a secure VN-based controller entity that monitors the packet

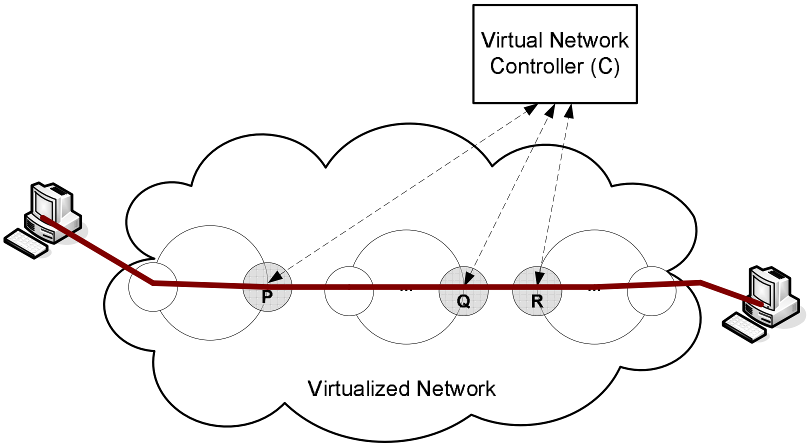

Figure 5.1. Forwarding Misbehavior Detection System

forwarding functionality of the hosted NI components. The system consists of a VN controller node (C), that evaluates the forwarding performance of the NI components between end systems. Particularly, the detection process should include the following steps: 1) Detection Setup, 2) Forwarding Proof Collection, and 3) Controller-based verification.

Detection Setup: Here we discuss an instance of how the detection setup connection is first established. At a particular time t, the controller C dynamically monitors nodes belonging to a single network infrastructure to verify the packet forwarding behavior. At this instance, the controller initiates the verification process at the output node of upstream NI and the input node of the downstream NI. To avoid excessive communication and storage overhead, rather than collecting the forwarding behavior information from each router in the forwarding path, the VN can perform the detection mechanism at the input and output border routers of each NI in the path. With this setup, If the output router belonging to a particular NI exhibits significant loss, the VN can raise a misbehavior detection alarm to the NI entity and hold the NI responsible for the irregularities emanating from its administrated router components.

For example, in figure 5.1 at a given instance of time, the VN controller C decides to monitor set of routers belonging to Q. In order to evaluate the forwarding performance of Q, the controller initiates the verification process by gathering the forwarding proofs sent to the input of Q from P and the forwarding proofs from the output of Q to R.

Forwarding Proof Collection: Verifying the data path performance of the NI components require collecting information about the packet forwarding performance. When the verification process is initiated by C, the forwarding proof (sample aggregate) data are collected from Q and evidence aggregates are collected from its neighboring NIs (P and R) to verify the forwarding behavior. To avoid preferential treatment to sample packets (i.e., the node decides to forward sample packets correctly and drop packets that are not considered in the sample set), the proof collection mechanism should use a hash-based delay sampling technique. The technique ensures that an adversarial node does not know a-priori if a packet is required for verification analysis and hence preferential treatment to sampled packets can be avoided.

To ensure that all three routers sample the same forwarding proof information, it is important to consider the following: 1) All three routers should initiate the sampling process at the same time, 2) Gather same forwarding proof information, and 3) Send/Update the controller C with the required information for verification. An explicit signaling for each of these process can be sent from the controller to the three routers, however, an efficient mechanism (less communication overhead) is to avoid sending any signaling. To gather the forwarding proof information, we can define set of hash functions based on packet identifiers (e.g., packet headers) to ensure that all three routers provide the identical (correct) forwarding proof information.

Controller-based Verification: When the controller receives set of forwarding proofs from the three entities (P, Q, and R) the system initiates the verification process to identify forwarding misbehaviors. Each proof contains sampled tuples and sample count that were considered during the sampling process. The verification mechanism can perform a comparison on the sample aggregates and count value which can then determine if the router's current state is compromised or not. The performance of the verification process can be improved by comparing the samples using tuple aggregates rather than checking each tuple in the sample aggregate, as shown in [64].

5.3 Conclusion

Considering the network virtualization architecture, it is hard to pin-point the compromised section of the network, since the virtual network may be operated at different layers and be hosted on multiple third-party infrastructure components along the end-to-end path. Therefore, to support **accountability** when hosting virtual networks on third-party infrastructures, we require a misbehavior detection system that dynamically monitors the forwarding performance and identify compromised (malicious) network infrastructure components. An initial design of such a system was discussed here. It is important to consider dynamic monitoring of network entities irrespective of the trust management and service level agreements between them. This helps in identifying possible security violations that can be encountered in the system. The technical challenges and practical deployment of the above solution can be looked into as a potential research topic in the network virtualization architecture.

BIBLIOGRAPHY

[1] Abbott, Timothy G., Lai, Katherine J., Lieberman, Michael R., and Price, Eric C. Browser-based attacks on tor. In *Proceedings of the 7th international conference on Privacy enhancing technologies* (Berlin, Heidelberg, 2007), PET'07, Springer-Verlag, pp. 184–199.

[2] Advanced Network Technology Center, University of Oregon. *Route Views Project Page*, 2003. http://www.routeviews.org/.

[3] Agrawal, Dakshi, Lee, Kang-Won, and Lobo, Jorge. Policy-based management of networked computing systems. *IEEE Communications Magazine 43*, 10 (Oct. 2005), 69–75.

[4] Ahlswede, Rudolf, Cai, Ning, Li, Shuo-Yen Robert, and Yeung, Raymond W. Network information flow. *IEEE Transactions on Information Theory 46*, 4 (July 2000), 1204–1216.

[5] Ahmed, Khurshid, Wenxuan, Zhou, Matthew, Caesar, and P. Brighten, Godfrey. Veriflow: Verifying network-wide invariants in real time. In *Hot Topics in Software Defined Networking (HotSDN)* (August 2012).

[6] Alexander, Scott, DeCleene, Brian, Rogers, Jason, and Sholander, Peter. Requirements and architectures for intrinsically assurable mobile ad hoc networks. In *Proc. of the 2008 IEEE Conference on Military Communications (MILCOM)* (San Diego, CA, Nov. 2008).

[7] Almeida, Paulo Sérgio, Baquero, Carlos, Preguiça, Nuno, and Hutchison, David. Scalable Bloom filters. *Information Processing Letters 101*, 6 (Mar. 2007), 255–261.

[8] Anderson, Thomas, Peterson, Larry, Shenker, Scott, and Turner, Jonathan. Overcoming the Internet impasse through virtualization. *Computer 38*, 4 (Apr. 2005), 34–41.

[9] Anderson, Tom, Roscoe, Timothy, and Wetherall, David. Preventing Internet denial-of-service with capabilities. *SIGCOMM Computer Communication Review 34*, 1 (Jan. 2004), 39–44.

[10] Argyraki, Katerina, and Cheriton, David. Network capabilities: The good, the bad and the ugly. In *Proc. of Fourth Workshop on Hot Topics in Networks (HotNets-IV)* (College Park, MD, Nov. 2005).

[11] Argyraki, Katerina, and Cheriton, David R. Active Internet traffic filtering: real-time response to denial-of-service attacks. In *ATEC'05: Proceedings of the USENIX Annual Technical Conference 2005 on USENIX Annual Technical Conference* (Anaheim, CA, Apr. 2005), pp. 135–148.

[12] Argyraki, Katerina, Maniatis, Petros, and Singla, Ankit. Verifiable network-performance measurements. In *Proceedings of the 6th International COnference on Emerging Networking Experiments and Technology (CoNEXT)* (Philadelphia, PA, Dec. 2010).

[13] Argyraki, Katerina, Maniatis, Petros, and Singla, Ankit. Verifiable network-performance measurements. In *Proceedings of the 6th International COnference* (New York, NY, USA, 2010), Co-NEXT '10, ACM, pp. 1:1–1:12.

[14] Avramopoulos, Ioannis, Kobayashi, Hisashi, Wang, Randolph, and Krishnamurthy, Arvind. Highly secure and efficient routing. In *IN PROC. IEEE INFOCOM 2004, HONG KONG* (2004).

[15] Ballani, Hitesh, Chawathe, Yatin, Ratnasamy, Sylvia, Roscoe, Timothy, and Shenker, Scott. Off by default! In *Proc. of Fourth Workshop on Hot Topics in Networks (HotNets-IV)* (College Park, MD, Nov. 2005).

[16] Barak, Boaz, Goldberg, Sharon, and Xiao, David. Protocols and lower bounds for failure localization in the internet. In *Proceedings of the theory and applications of cryptographic techniques 27th annual international conference on Advances in cryptology* (Berlin, Heidelberg, 2008), EUROCRYPT'08, Springer-Verlag, pp. 341–360.

[17] Bavier, Andy, Feamster, Nick, Huang, Mark, Peterson, Larry, and Rexford, Jennifer. In VINI veritas: realistic and controlled network experimentation. In *SIGCOMM '06: Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (Pisa, Italy, Aug. 2006), pp. 3–14.

[18] Belenky, Andrey, and Ansari, Nirwan. IP traceback with deterministic packet marking. *IEEE Communications Letters 7*, 4 (Apr. 2003), 162–164.

[19] Bellare, Mihir, Ristenpart, Thomas, Rogaway, Phillip, and Stegers, Till. Format-preserving encryption. *IACR Cryptology ePrint Archive 2009* (2009), 251.

[20] Bellare, Mihir, and Rogaway, Phillip. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security* (1993), Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, Eds., ACM, pp. 62–73.

[21] Bellare, Mihir, and Rogaway, Phillip. Optimal asymmetric encryption. In *EUROCRYPT* (1994), Alfredo De Santis, Ed., vol. 950 of *Lecture Notes in Computer Science*, Springer, pp. 92–111.

[22] Bhatti, Rafae, Bertino, Elisa, and Ghafoor, Arif. An integrated approach to federated identity and privilege management in open systems. *Communications of the ACM 50*, 2 (Feb. 2007), 81–87.

[23] Bissias, George, Liberatore, Marc, Jensen, David, and Levine, Brian Neil. Privacy Vulnerabilities in Encrypted HTTP Streams. In *Proc. Privacy Enhancing Technologies Workshop (PET)* (May 2005), pp. 1–11.

[24] Bloom, Burton H. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM 13*, 7 (July 1970), 422–426.

[25] Boneh, Dan, and Franklin, Matthew. Identity-based encryption from the Weil pairing. *SIAM Journal on Computing 32*, 3 (2003), 586–615.

[26] Bradley, K.A., Cheung, S., Puketza, N., Mukherjee, B., and Olsson, R.A. Detecting disruptive routers: a distributed network monitoring approach. *Network, IEEE 12*, 5 (sep/oct 1998), 50 –60.

[27] Broder, A., and Mitzenmacher, M. Network applications of Bloom filters: A survey. In *Proc. of the 40th Annual Allerton Conference on Communication, Control, and Computing* (Allerton, IL, Oct. 2002), pp. 636–646.

[28] Brumley, Billy Bob, and Valkonen, Jukka. Attacks on message stream encryption. In *Proceedings of the 13th Nordic Workshop on Secure IT Systems—NordSec '08* (October 2008), Hanne Riis Nielson and Christian W. Probst, Eds., pp. 163–173.

[29] Canetti, Ran, Feige, Uriel, Goldreich, Oded, and Naor, Moni. Adaptively secure multi-party computation. In *STOC* (1996), Gary L. Miller, Ed., ACM, pp. 639–648.

[30] Casado, Martin, Freedman, Michael J., Pettit, Justin, Luo, Jianying, McKeown, Nick, and Shenker, Scott. Ethane: taking control of the enterprise. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications* (Kyoto, Japan, Aug. 2007), pp. 1–12.

[31] Centre for Advanced Internet Architectures. *BGP Routing Table Analysis Reports.* http://http://bgp.potaroo.net/.

[32] Chasaki, Danai, and Wolf, Tilman. Design of a secure packet processor. In *Proc. of ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ANCS)* (San Diego, CA, Oct. 2010).

[33] Chasaki, Danai, Wu, Qiang, and Wolf, Tilman. Attacks on network infrastructure. In *Proc. of Twentieth IEEE International Conference on Computer Communications and Networks (ICCCN)* (Maui, HI, Aug. 2011).

[34] Cheng, Yuu-Heng, Raykova, Mariana, Poylisher, Alex, Alexander, Scott, Eiger, Martin, and Bellovin, Steve M. The Zodiac policy subsystem: A policy-based management system for a high-security MANET. In *Proc. of IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY)* (London, United Kingdom, July 2009), pp. 174–177.

[35] Chowdhury, N. M. Mosharaf Kabir, and Boutaba, Raouf. A survey of network virtualization. *Computer Networks 54*, 5 (Apr. 2010), 862–876.

[36] Chowdhury, N.M.M.K., and Boutaba, R. Network virtualization: state of the art and research challenges. *Communications Magazine, IEEE 47*, 7 (july 2009), 20 –26.

[37] Chowdhury, N.M.M.K., Rahman, M.R., and Boutaba, R. Virtual network embedding with coordinated node and link mapping. In *INFOCOM 2009, IEEE* (april 2009), pp. 783 –791.

[38] Clark, Christopher, Fraser, Keir, Hand, Steven, Hansen, Jacob Gorm, Jul, Eric, Limpach, Christian, Pratt, Ian, and Warfield, Andrew. Live migration of virtual machines. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2* (Berkeley, CA, USA, 2005), NSDI'05, USENIX Association, pp. 273–286.

[39] Desai, Vikram, Natarajan, Sriram, and Wolf, Tilman. Packet forwarding misbehavior detection in next-generation networks. In *IEEE International Conference on Communications (ICC) 2012* (Ottawa, Canada, June 2012).

[40] Dingledine, Roger, Mathewson, Nick, and Syverson, Paul. Tor: the second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13* (Berkeley, CA, USA, 2004), SSYM'04, USENIX Association, pp. 21–21.

[41] Dong, Jing, Curtmola, Reza, and Nita-Rotaru, Cristina. Secure network coding for wireless mesh networks: Threats, challenges, and directions. *Computer Communications 32* (Nov. 2009), 1790–1801.

[42] Duffield, N.G., and Grossglauser, M. Trajectory sampling for direct traffic observation. *Networking, IEEE/ACM Transactions on 9*, 3 (jun 2001), 280 –292.

[43] Eastlake, Donald E., and Jones, Paul E. US secure hash algorithm 1 (SHA1). RFC 3174, Network Working Group, Sept. 2001.

[44] Eatherton, Will. The push of network processing to the top of the pyramid. In *Keynote Presentation at ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ANCS)* (Princeton, NJ, Oct. 2005).

[45] El Gamal, Taher. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proc. of of CRYPTO 84 on Advances in Cryptology* (Santa Barbara, CA, 1985), vol. 196 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 10–18.

[46] Electronic Frontier Foundation. *FCC Rules Against Comcast for BitTorrent Blocking.* `http://www.eff.org/deeplinks/2008/08/fcc-rules-against-comcast-bit-torrent-blocking`.

[47] Esteve Rothenberg, Christian, Macapuna, Carlos Alberto Braz, Magalhães, Maurício Ferreira, Verdi, Fábio Luciano, and Wiesmaier, Alexander. In-packet Bloom filters: Design and networking applications. *Computer Networks* (2011).

[48] Estrin, Deborah, Mogul, Jeffrey C., and Tsudik, Gene. Visa protocols for controlling interorganizational datagram flow. *IEEE Journal on Selected Areas in Communications 7*, 4 (May 1989), 486–498.

[49] Evans, Nathan S., Dingledine, Roger, and Grothoff, Christian. A practical congestion attack on tor using long paths. In *Proceedings of the 18th conference on USENIX security symposium* (Berkeley, CA, USA, 2009), SSYM'09, USENIX Association, pp. 33–50.

[50] Facebook, Inc. *Facebook Key Facts.* `http://newsroom.fb.com/content/default.aspx?NewsAreaId=22`. Accessed: May 21, 2012.

[51] Feamster, Nick, Gao, Lixin, and Rexford, Jennifer. How to lease the Internet in your spare time. *SIGCOMM Computer Communication Review 37* (Jan. 2007).

[52] Feldmann, Anja. Internet clean-slate design: what and why? *SIGCOMM Computer Communication Review 37*, 3 (July 2007), 59–64.

[53] Ferraiolo, David F., and Kuhn, D. Richard. Role-based access control. In *Proc. of 15th National Computer Security Conference* (Baltimore, MD, Oct. 1992), pp. 554–563.

[54] Freedom to Thinker. *Three Flavors of Net Neutrality.* `https://www.freedom-to-tinker.com/blog/felten/three-flavors-net-neutrality`.

[55] Fukushima, M., Hasegawa, T., Hasegawa, T., and Nakao, A. Minimum disclosure routing for network virtualization. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on* (april 2011), pp. 858 –863.

[56] Gentry, Craig. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Theory of computing* (New York, NY, USA, 2009), STOC '09, ACM, pp. 169–178.

[57] Goldwasser, Shafi, and Micali, Silvio. Probabilistic encryption. *Journal of Computer and System Sciences 28*, 2 (Apr. 1984), 270–299.

[58] Hsiao, Hsu-Chun, Kim, Tiffany Hyun-Jin, Perrig, Adrian, Yamada, Akira, Nelson, Samuel C., Gruteser, Marco, and Meng, Wei. LAP: Lightweight anonymity and privacy. In *Proceedings of the IEEE Symposium on Security and Privacy* (May 2012).

[59] Huici, Felipe, and Handley, Mark. An edge-to-edge filtering architecture against DoS. *SIGCOMM Compututer Communication Review 37*, 2 (Apr. 2007), 39–50.

[60] Jang, Keon, Han, Sangjin, Han, Seungyeop, Moon, Sue, and Park, KyoungSoo. SSLShader: cheap SSL acceleration with commodity processors. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation (NSDI)* (Boston, MA, Mar. 2011).

[61] Keller, Eric, Lee, Ruby B., and Rexford, Jennifer. Accountability in hosted virtual networks. In *Proc. of the First ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)* (Barcelona, Spain, Aug. 2009), VISA '09, pp. 29–36.

[62] Keller, Eric, Yu, Minlan, Caesar, Matthew, and Rexford, Jennifer. Virtually eliminating router bugs. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies* (New York, NY, USA, 2009), CoNEXT '09, ACM, pp. 13–24.

[63] Kent, S., and Atkinson, R. Security architecture for the Internet protocol. RFC 2401, Network Working Group, Nov. 1998.

[64] Kompella, Ramana Rao, Levchenko, Kirill, Snoeren, Alex C., and Varghese, George. Every microsecond counts: tracking fine-grain latencies with a lossy difference aggregator. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication* (New York, NY, USA, 2009), SIGCOMM '09, ACM, pp. 255–266.

[65] Kuzmanovic, Aleksandar, and Knightly, Edward W. Low-rate TCP-targeted denial of service attacks: the shrew vs. the mice and elephants. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications* (Karlsruhe, Germany, Aug. 2003), pp. 75–86.

[66] Le Blond, Stevens, Manils, Pere, Chaabane, Abdelberi, Kaafar, Mohamed Ali, Castelluccia, Claude, Legout, Arnaud, and Dabbous, Walid. One bad apple spoils the bunch: exploiting p2p applications to trace and profile tor users. In *Proceedings of the 4th USENIX conference on Large-scale exploits and emergent threats* (Berkeley, CA, USA, 2011), LEET'11, USENIX Association, pp. 2–2.

[67] Lee, Sihyungo, Wong, T., and Kim, H.S. Secure split assignment trajectory sampling: A malicious router detection system. In *Dependable Systems and Networks, 2006. DSN 2006. International Conference on* (june 2006), pp. 333 –342.

[68] Liu, Vincent, Han, Seungyeop, Krishnamurthy, Arvind, and Anderson, Thomas. Tor instead of ip. In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks* (New York, NY, USA, 2011), HotNets '11, ACM, pp. 14:1–14:6.

[69] Matsumoto, Makoto, and Nishimura, Takuji. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul. 8* (January 1998), 3–30.

[70] McKeown, Nick, Anderson, Tom, Balakrishnan, Hari, Parulkar, Guru, Peterson, Larry, Rexford, Jennifer, Shenker, Scott, and Turner, Jonathan. OpenFlow: enabling innovation in campus networks. *SIGCOMM Computer Communication Review 38*, 2 (Apr. 2008), 69–74.

[71] Mekouar, Loubna, Iraqi, Youssef, and Boutaba, Raouf. Incorporating trust in network virtualization. In *Proceedings of the 2010 10th IEEE International Conference on Computer and Information Technology* (Washington, DC, USA, 2010), CIT '10, IEEE Computer Society, pp. 942–947.

[72] Mendonca, Marc. In pursuit of privacy on a public internet. Diplom thesis, University of Calicornia Santa Cruz, USA, March 2012. `http://escholarship.org/uc/item/4k69t6p`.

[73] Mizrak, Alper Tugay, Cheng, Yu-Chung, Marzullo, Keith, and Savage, Stefan. Fatih: Detecting and isolating malicious routers. In *DSN '05: Proceedings of the 2005 International Conference on Dependable Systems and Networks (DSN'05)* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 538–547.

[74] Natarajan, S., Huang, Xin, and Wolf, T. Efficient conflict detection in flow-based virtualized networks. In *Computing, Networking and Communications (ICNC), 2012 International Conference on* (30 2012-feb. 2 2012), pp. 690 –696.

[75] Natarajan, S., and Wolf, T. Security issues in network virtualization for the future internet. In *Computing, Networking and Communications (ICNC), 2012 International Conference on* (30 2012-feb. 2 2012), pp. 537 –543.

[76] Natarajan, Sriram, and Wolf, Tilman. Encrypted packet forwarding in virtualized networks. In *Proceedings of the 2011 ACM/IEEE Seventh Symposium on Architectures for Networking and Communications Systems* (Washington, DC, USA, 2011), ANCS '11, IEEE Computer Society, pp. 213–214.

[77] National Science Foundation. *Global Environment for Network Innovation.* `http://www.geni.net/`.

[78] Newswatch Blog. *China attacks VPNs with DNS poisoning.* `http://tvnewswatch.blogspot.com/2011/06/china-attacks-vpns-with-dns-poisoning.html`.

[79] Nextgov: Technology and the Business of Government. *Federal Network Security Breaches Spike 650 percent.* http://www.nextgov.com/nextgov/ng_20111003_6771.php. Accessed: June 2, 2012.

[80] Oberheide, Jon, Cooke, Evan, and Jahanian, Farnam. Exploiting Live Virtual Machine Migration. In *BlackHat DC Briefings* (Washington DC, February 2008).

[81] Padmanabhan, Venkata N., and Simon, Daniel R. Secure traceroute to detect faulty or malicious routing. *SIGCOMM Computer Communications Review 33* (2002), 77–82.

[82] Panchenko, Andriy, Pimenidis, Lexi, and Renner, Johannes. Performance analysis of anonymous communication channels provided by tor. In *Proceedings of the 2008 Third International Conference on Availability, Reliability and Security* (Washington, DC, USA, 2008), ARES '08, IEEE Computer Society, pp. 221–228.

[83] Pang, Ruoming, and Paxson, Vern. A high-level programming environment for packet trace anonymization and transformation. In *Proc. of the ACM SIGCOMM Conference* (Karlsruhe, Germany, Aug. 2003), pp. 339–351.

[84] Parno, Bryan, Wendlandt, Dan, Shi, Elaine, Perrig, Adrian, Maggs, Bruce, and Hu, Yih-Chun. Portcullis: protecting connection setup from denial-of-capability attacks. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications* (Kyoto, Japan, Aug. 2007), pp. 289–300.

[85] Perlman, Radia. An overview of PKI trust models. *IEEE Network 13*, 6 (Nov. 1999), 38–43.

[86] Peuhkuri, Markus. A method to compress and anonymize packet traces. In *Proc. of First ACM SIGCOMM Internet Measurement Workshop* (San Francisco, USA, Nov. 2001), pp. 257–260.

[87] Phillip, Porras, Seungwon, Shin, Vinod, Yegneswaran, Martin, Fong, Mabry, Tyson, and Guofei, Gu. A security enforcement kernel for openflow networks. In *Hot Topics in Software Defined Networking (HotSDN)* (August 2012).

[88] Planetlab Consortium. *An open platform for developing, deploying, and accessing planetary-scale services.* http://www.planet-lab.org/.

[89] Privacy Rights Clearinghouse. *Identity Theft & Data Breaches.* http://www.privacyrights.org/Identity-Theft-Data-Breaches.

[90] Raghavan, Barath, Kohno, Tadayoshi, Snoeren, Alex C., and Wetherall, David. Enlisting isps to improve online privacy: Ip address mixing by default. In *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies* (Berlin, Heidelberg, 2009), PETS '09, Springer-Verlag, pp. 143–163.

[91] Ramaswamy, Ramaswamy, and Wolf, Tilman. High-speed prefix-preserving IP address anonymization for passive measurement systems. *IEEE/ACM Transactions on Networking 15*, 1 (Feb. 2007), 26–39.

[92] Reader's Digest. *How To Hide Anything.* `http://www.rd.com/home/how-to-hide-anything/`.

[93] Recordon, David, and Reed, Drummond. OpenID 2.0: a platform for user-centric identity management. In *Proc. of the Second ACM Workshop on Digital Identity Management (DIM)* (Alexandria, VA, Nov. 2006), pp. 11–16.

[94] Riboni, Daniele, Villani, Antonio, Vitali, Domenico, Bettini, Claudio, and Mancini, Luigi V. Obfuscation of sensitive data in network flows. In *INFOCOM* (2012), Albert G. Greenberg and Kazem Sohraby, Eds., IEEE, pp. 2372–2380.

[95] Ristenpart, Thomas, Tromer, Eran, Shacham, Hovav, and Savage, Stefan. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM conference on Computer and communications security* (New York, NY, USA, 2009), CCS '09, ACM, pp. 199–212.

[96] Sakakima, K., Ata, S., and Kitamura, H. Anonymous but traceable ip address-based communication system. In *Networks and Communications, 2009. NETCOM '09. First International Conference on* (dec. 2009), pp. 259 –264.

[97] Savage, Stefan, Wetherall, David, Karlin, Anna, and Anderson, Tom. Network support for IP traceback. *IEEE/ACM Transactions on Networking 9*, 3 (June 2001), 226–237.

[98] Security News Daily. *2011 Set to Be Worst Year Ever for Security Breaches.* `http://www.securitynewsdaily.com/2011-worst-year-ever-security-breaches-0857/`.

[99] Security News Daily. *Computer Virus Swipes Data from Japan's Space Agency.* `http://www.securitynewsdaily.com/japan-space-agency-computer-virus--1495/`.

[100] Security News Daily. *Gangs Are Eavesdropping on Police Radios Via Smartphone Apps.* `http://www.securitynewsdaily.com/gang-members-police-radios-1476/`.

[101] Security News Daily. *Russian Hackers Shut Down Illinois Water Plant Not.* `http://www.securitynewsdaily.com/top-security-stories-2011-1445/10/`.

[102] Sherwood, Rob, Gibb, Glen, Yap, Kok-Kiong, Appenzeller, Guido, Casado, Martin, McKeown, Nick, and Parulkar, Guru. Flowvisor: A network virtualization layer. Technical report, Department of Computer Science, Stanford University, 2009.

[103] Smith, Sean W., and Weingart, Steve. Building a high-performance, programmable secure coprocessor. *Computer Networks 31* (Apr. 1999), 831–860.

[104] Snoeren, Alex S., Partridge, Craig, Sanchez, Luis A., Jones, Christine E., Tchakountio, Fabrice, Kent, Stephen T., and Strayer, W. Timothy. Hash-based IP traceback. In *Proc. of ACM SIGCOMM 2001* (San Diego, CA, Aug. 2001), pp. 3–14.

[105] Technology Review, MIT. *China Cracks Down on Tor Anonymity Network.* `http://www.technologyreview.com/web/23736/?a=f`.

[106] Tor Project Blogs. *Ethiopia Introduces Deep Packet Inspection.* `https://blog.torproject.org/blog/ethiopia-introduces-deep-packet-inspection`.

[107] Trostle, Jonathan, Matsuoka, Hosei, Tariq, Muhammad Mukarram Bin, Kempf, James, Kawahara, Toshiro, and Jain, Ravi. Cryptographically protected prefixes for location privacy in ipv6. In *Proceedings of the 4th international conference on Privacy Enhancing Technologies* (Berlin, Heidelberg, 2005), PET'04, Springer-Verlag, pp. 142–166.

[108] Tsudik, Gene. Message authentication with one-way hash functions. *SIGCOMM Computer Communication Review 22* (Oct. 1992), 29–38.

[109] Turner, Jonathan S. A proposed architecture for the GENI backbone platform. In *Proc. of ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ANCS)* (San Jose, CA, Dec. 2006), pp. 1–10.

[110] Turner, Jonathan S., Crowley, Patrick, DeHart, John, Freestone, Amy, Heller, Brandon, Kuhns, Fred, Kumar, Sailesh, Lockwood, John, Lu, Jing, Wilson, Michael, Wiseman, Charles, and Zar, David. Supercharging PlanetLab: a high performance, multi-application, overlay network platform. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications* (Aug. 2007), pp. 85–96.

[111] Turner, Jonathan S., and Taylor, David E. Diversifying the Internet. In *Proc. of IEEE Global Communications Conference (GLOBECOM)* (Saint Louis, MO, Nov. 2005), vol. 2.

[112] University of Massachusetts, Amherst. *University of Massachusetts: Computer Intrusion.* `http://www.umass.edu/computerintrusion/`. Accessed: May 21, 20012.

[113] University of Utah. *Network Emulation Testbed.* `http://www.emulab.net/`.

[114] US Department of Health and Human Services. *Health Information Privacy.* `http://www.hhs.gov/ocr/privacy/hipaa/administrative/breachnotificationrule/breachtool.html`, June 2012.

[115] VUZE. *Bad ISPs.* `http://wiki.vuze.com/w/Bad_ISPs`.

[116] VUZE. *Message Stream Encryption.* `http://wiki.vuze.com/w/Message_Stream_Encryption`.

[117] Wagner, David. A generalized birthday problem. In *Proc. of the 22nd Annual International Cryptology Conference on Advances in Cryptology* (Santa Barbara, CA, Aug. 2002), vol. 2442 of *Lecture Notes in Computer Science*, pp. 288–303.

[118] Wang, Yi, Keller, Eric, Biskeborn, Brian, van der Merwe, Jacobus, and Rexford, Jennifer. Virtual routers on the move: live router migration as a network-management primitive. *SIGCOMM Comput. Commun. Rev. 38* (August 2008), 231–242.

[119] White, Brian, Lepreau, Jay, Stoller, Leigh, Ricci, Robert, Guruprasad, Shashi, Newbold, Mac, Hibler, Mike, Barb, Chad, and Joglekar, Abhijeet. An integrated experimental environment for distributed systems and networks. In *Proc. of the Fifth Symposium on Operating Systems Design and Implementation* (Boston, MA, Dec. 2002), USENIX Association, pp. 255–270.

[120] Wolf, Tilman. Challenges and applications for network-processor-based programmable routers. In *Proc. of IEEE Sarnoff Symposium* (Princeton, NJ, Mar. 2006).

[121] Wolf, Tilman. A credential-based data path architecture for assurable global networking. In *Proc. of the 2007 IEEE Conference on Military Communications (MILCOM)* (Orlando, FL, Oct. 2007).

[122] Wolf, Tilman. Design of a network architecture with inherent data path security. In *Proc. of ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ANCS)* (Orlando, FL, Dec. 2007), pp. 39–40.

[123] Wolf, Tilman. Data path credentials for high-performance capabilities-based networks. In *Proc. of ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ANCS)* (San Jose, CA, Nov. 2008), pp. 129–130.

[124] Wolf, Tilman, and Tessier, Russell. Design of a secure router system for next-generation networks. In *Proc. of Third International Conference on Network and System Security (NSS)* (Gold Coast, Australia, Oct. 2009).

[125] Wolf, Tilman, and Vasudevan, Kamlesh T. A high-performance capabilities-based network protocol. In *Proc. of Fifth Workshop on Secure Network Protocols (NPSec) held in conjunction with Seventeenth IEEE International Conference on Network Protocols (ICNP)* (Princeton, NJ, Oct. 2009).

[126] Wu, Qiang, Shanbhag, Shashank, and Wolf, Tilman. Fair multithreading on packet processors for scalable network virtualization. In *Proc. of ACM/IEEE Symposium on Architectures for Networking and Communication Systems (ANCS)* (San Diego, CA, Oct. 2010).

[127] Xu, Jun, Fan, Jinliang, Ammar, Mostafa H., and Moon, Sue B. Prefix-preserving IP address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In *Proc. of 10th IEEE International Conference on Network Protocols (ICNP'02)* (Paris, France, Nov. 2002), pp. 280–289.

[128] Yaar, Abraham, Perrig, Adrian, and Song, Dawn. SIFF: A stateless internet flow filter to mitigate DDoS flooding attacks. In *Proc. of IEEE Symposium on Security and Privacy* (Oakland, CA, May 2004), pp. 130–143.

[129] Yang, Sookhyun, Vasudevan, Sudarshan, and Kurose, Jim. Witness-based detection of forwarding misbehaviors in wireless networks. In *Wireless Mesh Networks (WIMESH 2010), 2010 Fifth IEEE Workshop on* (june 2010), pp. 1 –6.

[130] Yang, Xiaowei, Wetherall, David, and Anderson, Thomas. A DoS-limiting network architecture. *SIGCOMM Computer Communication Review 35*, 4 (2005), 241–252.

[131] Yin, Dong, Unnikrishnan, Deepak, Liao, Yong, Gao, Lixin, and Tessier, Russell. Customizing virtual networks with partial fpga reconfiguration. *SIGCOMM Computer Communication Review 41* (Jan. 2011), 125–132.

[132] YouTube, Inc. *YouTube Statistics.* `http://www.youtube.com/t/press_statistics`. Accessed: June 18, 2012.

[133] ZDNet, Inc. *Wikileaks has been under DDoS attack for the last three days.* `http://www.zdnet.com/blog/security/wikileaks-has-been-under-ddos-attack-for-the-last-three-days/12219`.

[134] Zhang, Xin, Jain, Abhishek, and Perrig, Adrian. Packet-dropping adversary identification for data plane security. In *Proceedings of the 2008 ACM CoNEXT Conference* (New York, NY, USA, 2008), CoNEXT '08, ACM, pp. 24:1–24:12.