High-Performance Capabilities for 1-Hop Containment of Network Attacks

Tilman Wolf, *Senior Member, IEEE*, Sriram Natarajan and Kamlesh T. Vasudevan

***Abstract*—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. We present a capabilities system that uses packet credentials based on Bloom filters. The credentials are fixed length (independent of the number of routers that are traversed by the packet) and can be verified by routers with a few simple operations. 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. We present an analysis of our design and a practical protocol implementation that can effectively limit unauthorized traffic with only a small per-packet overhead.**

***Index Terms*—network security, off-by-default network, Bloom filter, data path processing.**

I. 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. A major shortcoming, however, is the difficulty of providing inherent security guarantees. To provide authentication, confidentiality, integrity, and availability, a number of additions have been designed, developed, and deployed. These approaches range from cryptographic operations on end-systems and routers (e.g., SSL, VPN tunnels) to dedicated traffic monitoring and access control (e.g., firewalls, intrusion detection systems) to defenses against denial of service (DoS) attacks (e.g., anomaly detection, rate limiting). These extensions provide point solutions in the defense against specific attacks, but do not address security at an architectural level.

Tilman Wolf is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA, 01054, USA, e-mail: wolf@ecs.umass.edu. Sriram Natarajan is with NTT Multimedia Communications Laboratories, Inc., San Mateo, CA 94401, USA, e-mail: sriram@nttmcl.com. Kamlesh Vasudevan is with Acme Packet, Bedford, MA 01730, USA, e-mail: kamleshtv@gmail.com. This work was done while at the University of Massachusetts Amherst.

This material is based upon work supported by the National Science Foundation under Grant Nos. 0447873, 0626690, 0952524, and 1111276. This material is based upon work under subcontract #069153 issued by BAE Systems National Security Solutions, Inc. and supported by the Defense Advanced Research Projects Agency (DARPA) and the Space and Naval Warfare System Center (SPAWARSYSCEN), San Diego under Contract No. N66001-08-C-2013.

Manuscript received May 19, 2011; revised March 4, 2012 and September 26, 2012; accepted December 26, 2012.

In some communication scenarios, however, high levels of inherent and verifiable security are essential (e.g., financial transactions, military communication, remote medical procedures, etc.). To provide security guarantees for such scenarios, it is necessary to design entirely new network architectures that overcome the shortcomings of the current Internet. These specialized networks can then be deployed on a separate infrastructure (e.g., dedicated military network) or as a virtual network in the future Internet (using network virtualization [1]).

An important question that we address in this paper is how to design a network that can inherently guarantee that only authorized traffic is transmitted. 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.
- 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.

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 paper, 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." 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. The specific contributions of this paper are:

- A design of a deny-by-default architecture that uses data path credentials to verify packet permissions on every hop. Credentials are used both in the data path and the control plane to overcome denial-of-capabilities vulnerabilities.
- 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. We show a security analysis of how well data path credentials can defend against attacks in unicast, multicast, and network coding settings. The results allow us to determine a suitable tradeoff between credentials overhead and security performance.
- 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 that the overhead and performance degradation from using credentials is limited.

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.

The remainder of this paper is structured as follows. Section II presents related work. Section III states the security model, introduces the general architecture for credential-based data paths, and describes its *qualitative* security properties. The specific design of scalable credentials using Bloom filter data structures is then discussed in Section IV. Section V then describes the *quantitative* security properties of these credentials and shows how security requirements are met. Section VI discusses the implementation of data path credentials in a practical protocol and Section VII presents evaluation results from a prototype running on Emulab. Section VIII summarizes and concludes this paper.

II. Related Work

Security in computer systems and networks has traditionally been equaled to information assurance as defined by the CIA triad of confidentiality, integrity, and authenticity [2]. More recent models have added availability, control, and utility [3]. In this paper, we focus on assurance issues related to authentication and availability, which are based on the attack model described in Section III-A and can be complemented with cryptographic solutions for confidentiality and integrity.

Capabilities-based networks, which are one example of specialized networks that focus on security, have been discussed by Anderson et al. [4], Yaar et al. [5], and Yang et al. [6] in the context of DoS attacks. Previously, similar ideas have been proposed by Estrin et al. [7] for controlling packet flows in networks.

A system design to avoid denial of service attacks based on the idea of capabilities from [4] has been proposed by Yang et al. [6] 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) [8]. 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 [5], 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 VII-D). 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 [9], 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. [10] 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. [11]. 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 III-E).

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 [12]. Recently, Parno et al. developed a solution to DoC attacks by using proof-of-work to limit an attacker's capability requests [13]. 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 V-C, 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 [14] or deterministic [15]. 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 [16]. 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 [17]. 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 [18] and found a number of applications in network systems [19], [20]. We adapt Bloom filters for the use with what we call credentials. These credentials are derived from cryptographic hash functions such as SHA-1 [21]. The use of hash functions for packet authentication has been proposed by Tsudik in [22], 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 [23], but are not applicable in our work as we need fixed-length credentials to limit packet header sizes.

Some initial ideas on this topic have been published in our prior work, which includes an overview of the architecture [24] (which was developed in the context of the IAMANET project [8]), two short papers on the use of Bloom filters [25], [26], and some initial results of the prototype system [27].

It has been difficult to envision how novel security architectures can be deployed in the context of the existing Internet. The concept of router virtualization [1] has made it conceivable to deploy domain-specific network architectures in parallel to the existing best-effort Internet. Currently, several specific systems [28], [29] are being developed that could support the types of data path operations we propose in this paper.

III. Data Path Credentials

We begin the discussion of our system with a description of security requirements and attacker capabilities. Then we introduce the overall network 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 IV.

A. Security Model

In this work, we consider domain-specific networks that are dominated by strict security requirements. Examples are networks used for financial transactions, military communication, remote medical procedures, etc. 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.

1) Security Requirements: We consider the following security requirements:

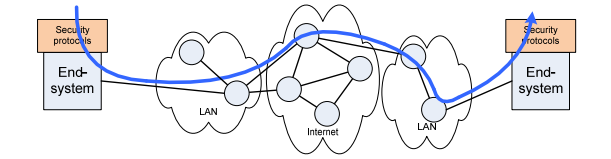
- Prevention of unauthorized network access and traffic injection: Only authorized users should be able to establish a connection in the network and send traffic.
- Detection of packet header spoofing: An unauthorized user should not be able to establish a connection by impersonating another entity.
- 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.

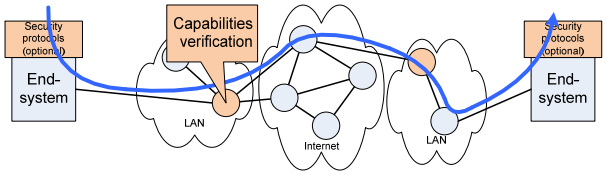
2) Attacker Capabilities: The capabilities of a potential attacker are assumed to be the following: (1) ability to read any packet traversing an attacked router; (2) ability to modify any packet traversing an attacked router; and (3) ability to send any packet from the attacked router.

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.

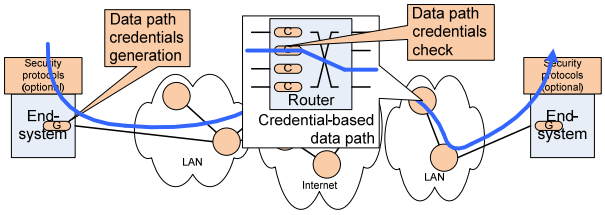
The limitations on the attacker's capability are necessary to keep the discussion of attack scenarios and security requirements within scope. Related work has addressed techniques



(a) Security in Existing Internet Architecture.



(b) Existing Capabilities-Based Network Architectures.



(c) Credential-Based Data Path Architecture.

Fig. 1. Comparison of Security Architecture of the Internet, Capabilities-Based Networks, and Credential-Based Networks.

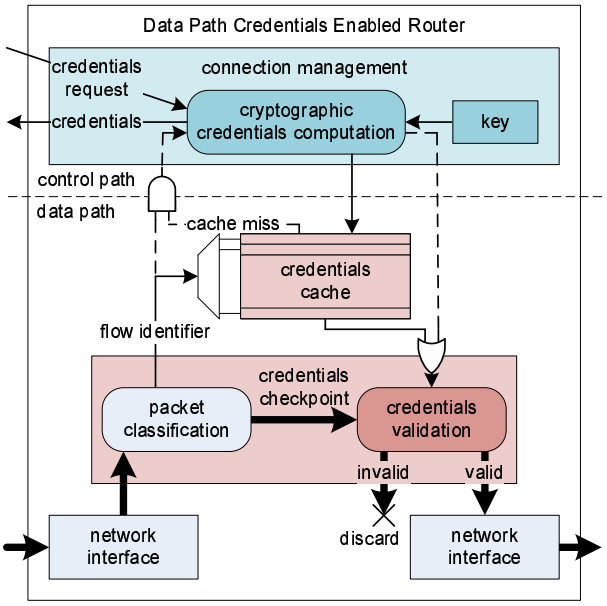


Fig. 2. Design of a Router System with Data Path Credentials.

that can provide environments where such assumptions are reasonable. For example, secure key generation and storage can be achieved with a cryptographic co-processor [30] and black holing can be circumvented by using multi-path routing and network coding [31]. The assumption that an attacker cannot partition the network is necessary to ensure that some valid communication between nodes is possible.

B. Network Architecture

With security requirements in place, we now turn to the system design and a discussion of how we can achieve security in the data path. The network architecture that we propose is depicted in Figure 1(c) and compared to conventional data networks in Figure 1(a) and existing capabilities-based networks in Figure 1(b). 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 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 [4], edge routers in [9], or LAN switches in [11]), 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.

C. Router Architecture

The system architecture of a router that implements data path credentials is shown in Figure 2. 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 IV. 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 2) 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 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 V-C.

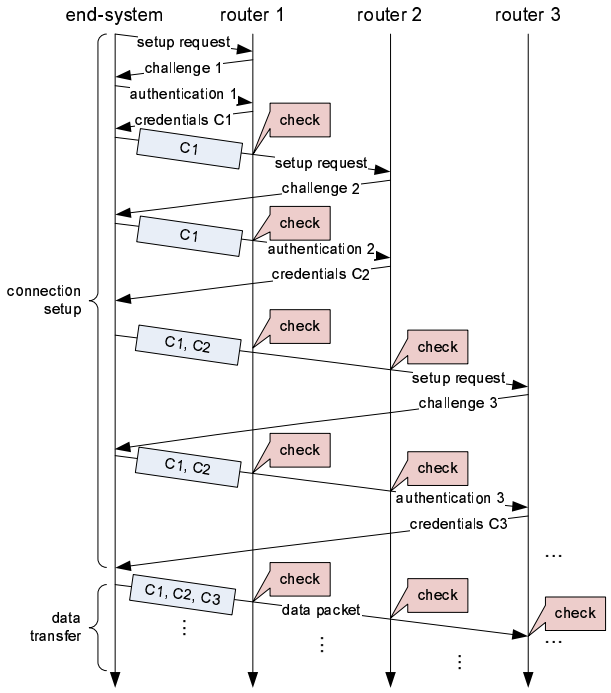


Fig. 3. Connection Setup to Establish Credentials.

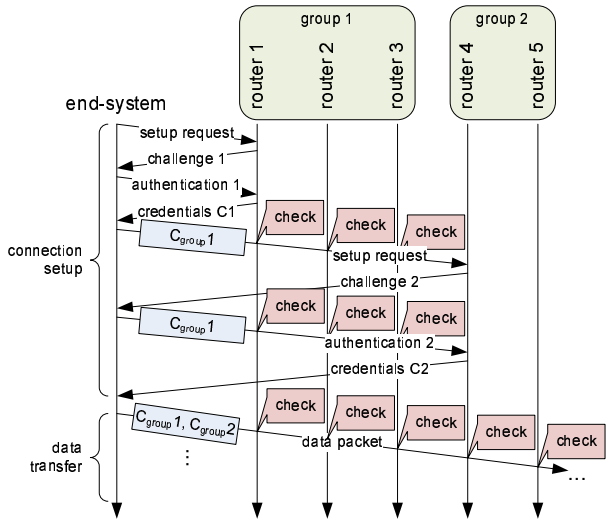


Fig. 4. Connection Setup using Group Credentials.

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 III-E).

D. Connection Management Scenarios

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 [10]. The control path of such networks pose as potential target for denial of service attacks [12]. In our design, connection setup is performed as an incremental process (as shown in Figure 3 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 only target routers 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).

1) Unicast: An example of the connection establishment process for *unicast* is sketched in the space-time diagram shown in Figure 3. 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 eligible 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 3) 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.

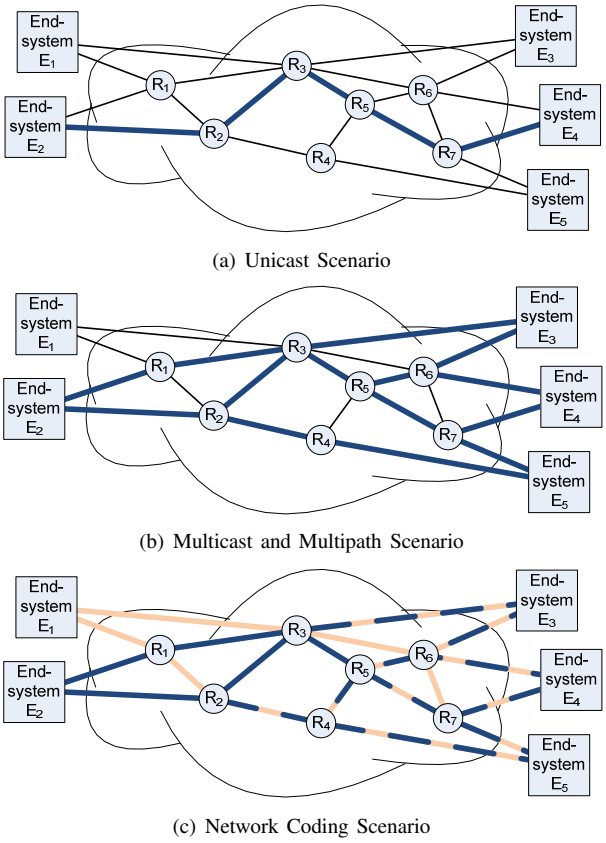
Clearly, having to exchange several messages with every router along a path in order to establish a single connection is a costly proposition. There are two approaches that can reduce this overhead:

- 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 reduced significantly (as illustrated in Figure 4). More details on these credentials can be found in Section IV-E.

2) Multicast and Network Coding: 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 Figure 5. In unicast, (Figure 5(a)), 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 5(b)). 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.

Network coding [32] is a recently proposed approach to improving end-to-end data transmissions in wireless networks.



(a) Unicast Scenario

(b) Multicast and Multipath Scenario

(c) Network Coding Scenario

Fig. 5. Connection Setup to Establish Credentials.

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 5(c)). 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 IV, we show that 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 V provides quantitative performance and security tradeoffs for each of these scenarios.

E. Authentication, Authorization 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 [33] to more broadly defined identities [34]. 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) [35] or a more complex federated trust model [36]. 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 [37]) to determine the access privileges of a particular entity. To support dynamic updates, a policy-based management systems [38] 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 [39].

Authentication, authorization, and access control decisions are made by routers during connection setup. Figure 3 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.

IV. 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.

A. 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 [5]) 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.

B. Bloom Filter Based Credentials

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.

1) Bloom Filters: We briefly review the concept of Bloom filters to provide context for our work. Bloom filters can be used to store membership information [18]. Specifically, a Bloom filter is a bit array that can store m bits. Using k different hash functions $h_1(x) \ldots h_k(x)$, an element x is mapped to k bit positions 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 V.

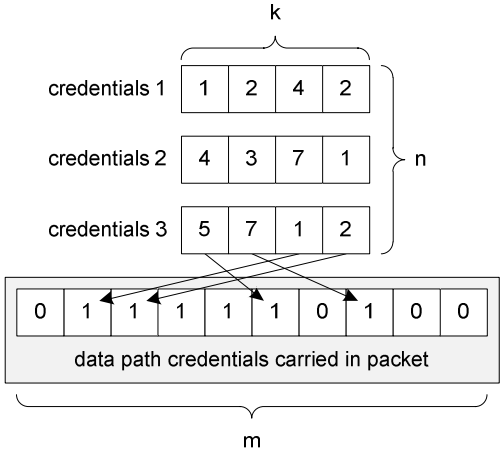


Fig. 6. Credentials Data Structure. This example shows three credentials that are aggregated to a set of 1's in the Bloom filter data structure.

2) 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 in Section III, 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 6.

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

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

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 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 V).

C. 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 [21]) 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 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 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 2. 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 III-A. A more detailed discussion on how security requirements are met is provided in Section V-D.

D. 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 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:

$$d(c) = \frac{1}{m} \sum_i c[i]. \tag{2}$$

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 V, 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.

E. Group Credentials

As illustrated in Figure 4, 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.

V. Security Analysis

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.

A. 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) [40], 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 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.

1) 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 in the Appendix):

$$b(m,k,n) = m \cdot \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right). \quad (3)$$

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:

$$f_{unicast}(m,k,n) = (d_{max})^{b(m,k,n)}. \quad (4)$$

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

$$b(m,k,n) \leq d_{max}. \quad (5)$$

2) 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

$$b(m,k,2^h-1) \leq d_{max}. \quad (6)$$

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:

$$f_{multicast}(m,k,n) = 1 - (1 - f_{unicast}(m,k,h))^{n+1}. \quad (7)$$

3) 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

$$b(m,k,(h-1) \cdot 2^h - 1) \leq d_{max}. \quad (8)$$

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:

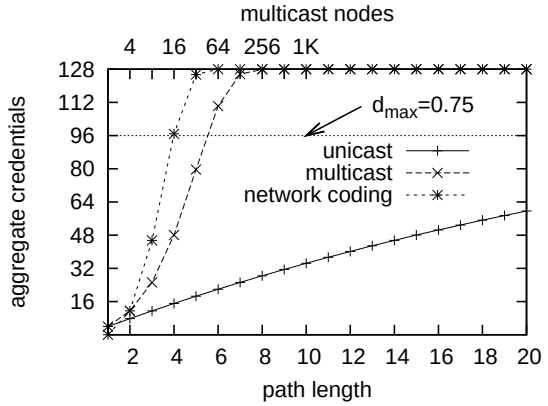
$$f_{network\ coding}(m,k,n) = \left(f_{multicast}(m,k,\frac{n+1}{2}) \cdot f_{unicast}(m,k,1)\right)^{\frac{n+1}{2}}. \quad (9)$$

B. Results

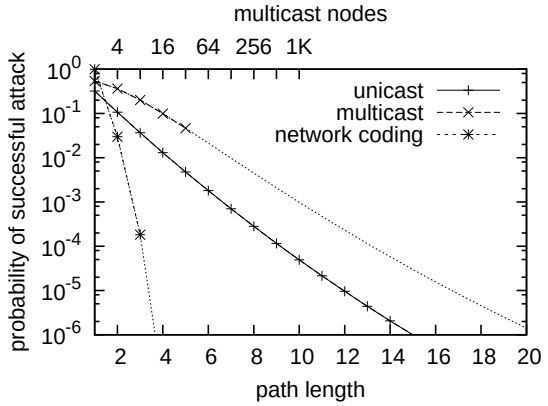
To illustrate the equations derived in the previous section in the context of a realistic system configuration, we provide security performance results of a few specific system configurations.

Figure 7(a) 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.

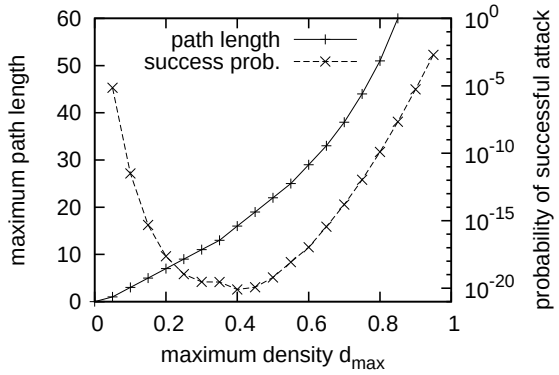
The false positive rates that correspond to this example are shown in Figure 7(b). Data points are only shown for



(a) Bits Set in Aggregate Credentials.



(b) Probability of Successful Attack Transmission.



(c) Maximum Path Length and Probability of Successful Attack for Different Density Limits in Unicast.

Fig. 7. Evaluation of Data Path Credentials (size m=128, number of hash functions k=4).

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,

TABLE I
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 II
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

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 7(c) 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 7(b)). For larger values of d_{max}, a large number of bits may 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 I and II. The maximum path length (assuming $d_{max} = 0.75$) is shown in Table I. The probability of a false positive end-to-end transmission with randomly generated credentials is shown in Table II. 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.

C. 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 [12]. 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 3). 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 [41]. 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.

D. 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 III-A. 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.

VI. 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.

A. Data Path Credentials Protocol

The header for DPCP is shown in Figure 8. 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

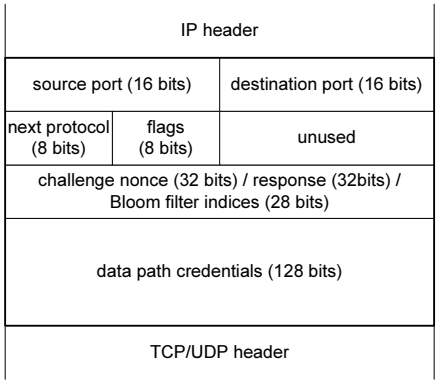


Fig. 8. Data Path Credentials Protocol Header.

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.

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.

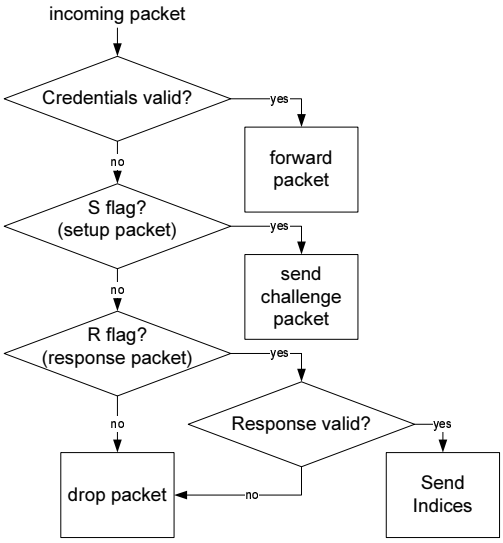


Fig. 9. Decision Diagram for DPCP Processing on Router.

B. 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 9). 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-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.

C. 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 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.

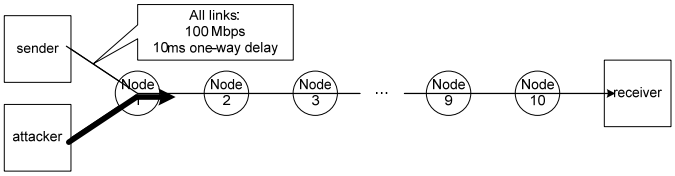


Fig. 10. Experimental Emulab Setup for Evaluation of DPCP.

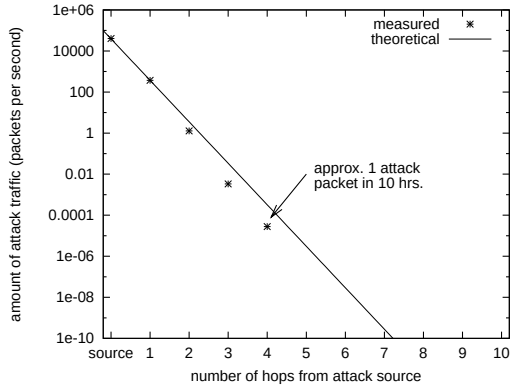


Fig. 11. Containment of Attack traffic with DPCP.

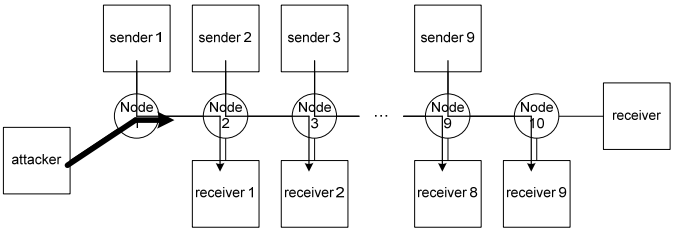


Fig. 12. Experimental Emulab Setup for Evaluation of Cross-Traffic Performance.

TABLE III
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%

VII. EVALUATION OF EMULAB IMPLEMENTATION

We have implemented the DPCP protocol on Emulab [42]. 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 10. Each link has a bandwidth of 100 Mbps and a one-way delay of 10ms.

A. Attack Containment

The most important result from our prototype implementation is that DPCP successfully contains attack traffic. Figure 11 shows the amount of traffic that passes each hop with an attacker sending traffic with random credentials (with the highest 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 12. We compare two different system configurations:

- 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 III. 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.

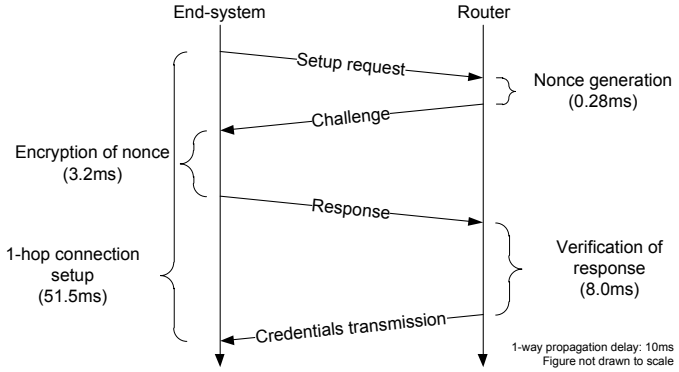


Fig. 13. Breakdown of Connection Setup Time for Single Hop.

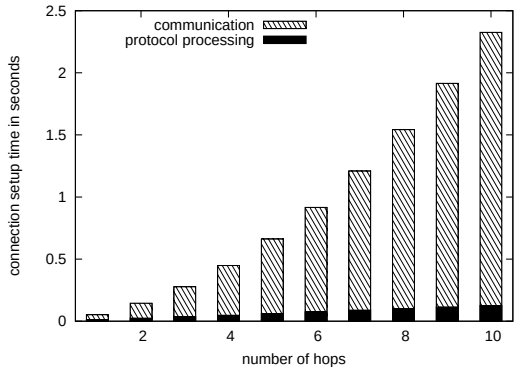


Fig. 14. Time for Connection Setup in DPCP.

B. Connection Setup Overhead

Clearly, there are additional costs in using DPCP over conventional TCP. In Figure 13, 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 III-D, such a setup is required for each hop along the path of a connection. Figure 14 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.

C. Throughput Comparison

The overall performance comparison of DPCP vs. conventional TCP is shown in Table IV. 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.

TABLE IV
FILE TRANSFER PERFORMANCE.

File size	File transfer time 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 V
FLOW SETUP PERFORMANCE. RESPONSE VERIFICATION AND CREDENTIALS GENERATION PER SECOND.

Processor	Response verification 512-bits	1024-bits	Credentials generation
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

D. 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 V. The table shows the achievable flow 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 [43]. 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 2) depends on the

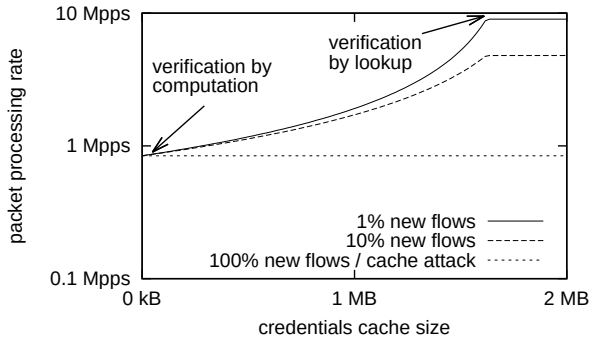


Fig. 15. Packet Processing Rate for Different Credential Cache Sizes. Total number of active flows is 100,000. Maximum credential lookup rate is 10 million per second.

number of active flows in a system. Each flow record requires 17 bytes of data for a 5-tuple flow identifier and credential indices (assuming $k = 4$). Figure 15 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 V) 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.
- 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.

VIII. Summary and Conclusions

In this paper, we have presented a capabilities-based network protocol that uses data path credentials to closely control traffic on every hop. We have shown that credentials that are based on Bloom filter data structures can efficiently implement capabilities and can provide probabilistic guarantees on permitting only valid traffic to traverse the network. We have provided a detailed security analysis that allows a quantitative evaluation of the capabilities of the system for unicast, multicast, and network coding uses. 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.

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 next-generation Internet architecture.

References

[1] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the Internet impasse through virtualization," *Computer*, vol. 38, no. 4, pp. 34–41, Apr. 2005.

[2] W. Stallings, *Cryptography and Network Security*, 4th ed. Prentice Hall, 2006.

[3] D. Parker, *Computer Security Handbook*, 4th ed. New York, NY: Wiley, Apr. 2002, ch. 5 – Towards a New Framework for Information Security.

[4] T. Anderson, T. Roscoe, and D. Wetherall, "Preventing Internet denial-of-service with capabilities," *SIGCOMM Computer Communication Review*, vol. 34, no. 1, pp. 39–44, Jan. 2004.

[5] A. Yaar, A. Perrig, and D. Song, "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.

[6] X. Yang, D. Wetherall, and T. Anderson, "A DoS-limiting network architecture," *SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 241–252, 2005.

[7] D. Estrin, J. C. Mogul, and G. Tsudik, "Visa protocols for controlling interorganizational datagram flow," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 4, pp. 486–498, May 1989.

[8] S. Alexander, B. DeCleene, J. Rogers, and P. Sholander, "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.

[9] F. Huici and M. Handley, "An edge-to-edge filtering architecture against DoS," *SIGCOMM Compututer Communication Review*, vol. 37, no. 2, pp. 39–50, Apr. 2007.

[10] H. Ballani, Y. Chawathe, S. Ratnasamy, T. Roscoe, and S. Shenker, "Off by default!" in *Proc. of Fourth Workshop on Hot Topics in Networks (HotNets-IV)*, College Park, MD, Nov. 2005.

[11] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "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.

[12] K. Argyraki and D. Cheriton, "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.

[13] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu, "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.

[14] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 226–237, Jun. 2001.

[15] A. Belenky and N. Ansari, "IP traceback with deterministic packet marking," *IEEE Communications Letters*, vol. 7, no. 4, pp. 162–164, Apr. 2003.

[16] A. S. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-based IP traceback," in *Proc. of ACM SIGCOMM 2001*, San Diego, CA, Aug. 2001, pp. 3–14.

[17] K. Argyraki and D. R. Cheriton, "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.

[18] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.

[19] A. Broder and M. Mitzenmacher, "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.

[20] C. Esteve Rothenberg, C. A. B. Macapuna, M. F. Magalhães, F. L. Verdi, and A. Wiesmaier, "In-packet Bloom filters: Design and networking applications," *Computer Networks*, 2011.

[21] D. E. Eastlake and P. E. Jones, "US secure hash algorithm 1 (SHA1)," Network Working Group, RFC 3174, Sep. 2001.

[22] G. Tsudik, "Message authentication with one-way hash functions," *SIGCOMM Computer Communication Review*, vol. 22, pp. 29–38, Oct. 1992.

[23] P. S. Almeida, C. Baquero, N. Preguiça, and D. Hutchison, "Scalable Bloom filters," *Information Processing Letters*, vol. 101, no. 6, pp. 255–261, Mar. 2007.

[24] T. Wolf, "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.

[25] T. Wolf, "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.

[26] T. Wolf, "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.

[27] T. Wolf and K. T. Vasudevan, "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.

[28] J. S. Turner, "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.

[29] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, "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.

[30] S. W. Smith and S. Weingart, "Building a high-performance, programmable secure coprocessor," *Computer Networks*, vol. 31, pp. 831–860, Apr. 1999.

[31] J. Dong, R. Curtmola, and C. Nita-Rotaru, "Secure network coding for wireless mesh networks: Threats, challenges, and directions," *Computer Communications*, vol. 32, pp. 1790–1801, Nov. 2009.

[32] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[33] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[34] D. Recordon and D. Reed, "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.

[35] R. Perlman, "An overview of PKI trust models," *IEEE Network*, vol. 13, no. 6, pp. 38–43, Nov. 1999.

[36] R. Bhatti, E. Bertino, and A. Ghafoor, "An integrated approach to federated identity and privilege management in open systems," *Communications of the ACM*, vol. 50, no. 2, pp. 81–87, Feb. 2007.

[37] D. F. Ferraiolo and D. R. Kuhn, "Role-based access control," in *Proc. of 15th National Computer Security Conference*, Baltimore, MD, Oct. 1992, pp. 554–563.

[38] D. Agrawal, K.-W. Lee, and J. Lobo, "Policy-based management of networked computing systems," *IEEE Communications Magazine*, vol. 43, no. 10, pp. 69–75, Oct. 2005.

[39] Y.-H. Cheng, M. Raykova, A. Poylisher, S. Alexander, M. Eiger, and S. M. Bellovin, "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, Jul. 2009, pp. 174–177.

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

[41] A. Kuzmanovic and E. W. Knightly, "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.

[42] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*. Boston, MA: USENIX Association, Dec. 2002, pp. 255–270.

[43] K. Jang, S. Han, S. Han, S. Moon, and K. Park, "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.

APPENDIX

In this appendix, 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 6 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 3.

Tilman Wolf (M'02, SM'07) is professor of Electrical and Computer Engineering at the University of Massachusetts Amherst. He received a Diplom in informatics from the University of Stuttgart, Germany, in 1998. He also received a M.S. in computer science in 1998, a M.S. in computer engineering in 2000, and a D.Sc. in computer science in 2002, all from Washington University in St. Louis.

He is engaged in research and teaching in the areas of computer networks, computer architecture, and embedded systems. His research interests include Internet architecture, network processors, their application in next-generation Internet architectures, and embedded system security.

Dr. Wolf is a senior member of the IEEE and the ACM. He has been active as program committee member and organizing committee member of several professional conferences, including IEEE INFOCOM and ACM SIGCOMM. He is currently serving as treasurer for the ACM SIGCOMM society and on the steering committee of the IEEE/ACM Transactions on Networking and on the editorial board of IEEE Micro.

Sriram Natarajan is a Senior Researcher at NTT Multimedia Communications Laboratories. He received his B.E. in Electronics and Communication Engineering from Anna University, India. He also received his M.S. and Ph.D. in Electrical and Computer Engineering from University of Massachusetts Amherst. His research interests span the areas of software-defined networks, network security, network virtualization and future Internet architectures.

Kamlesh T. Vasudevan is currently a SQA Engineer at Acme Packet, Bedford, MA. He received his Masters degree in Electrical and Computer Engineering at University of Massachusetts Amherst and Bachelor's degree in Electronics and Communications Engineering from Anna University, India. His research interests include the areas of network security and next-generation Internet architectures.