Misuse Detection in Consent-based Networks

Mansoor Alicherry[1] and Angelos D. Keromytis[1]

[1] Bell Labs, Alcatel-Lucent, Murray Hill, NJ, USA
[2] Department of Computer Science, Columbia University, New York, USA

Abstract. Consent-based networking, which requires senders to have permission to send traffic, can protect against multiple attacks on the network. Highly dynamic networks like Mobile Ad-hoc Networks (MANETs) require destination-based consent networking, where consent needs to be given to send to a destination in any path. These networks are susceptible to multipath misuses by misbehaving nodes.
In this paper, we identify the misuses in destination-based consent networking, and provide solution for detecting and recovering from the misuses. Our solution is based on our previously introduced DIPLOMA architecture. DIPLOMA is a *deny-by-default* distributed policy enforcement architecture that can protect the end-host services and network bandwidth. DIPLOMA uses capabilities to provide consent for sending traffic. In this paper, we identify how senders and receivers can misuse capabilities by using them in multiple paths, and provide distributed solutions for detecting those misuses. To that end, we modify the capabilities to aid in misuse detection and provide protocols for exchanging information for distributed detection. We also provide efficient algorithms for misuse detection, and protocols for providing proof of misuse. Our solutions can handle privacy issues associated with the exchange of information for misuse detection. We have implemented the misuse detection and recovery in DIPLOMA systems running on Linux operating systems, and conducted extensive experimental evaluation of the system in Orbit MANET testbed. The results show our system is effective in detecting and containing multipath misuses.

Keywords: network capability, MANET security, misuse detection

1 Introduction

Consent-based networking is emerging as a "clean-slate" design for providing security against multiple attacks in the Internet [12, 5, 17, 4]. In consent-based networking, a sender needs to have permission to send traffic to a destination. Consent-based architectures may support permission to send to a destination on a particular path (*path-based*) or on any path (*destination-based*). In path-based consent architectures, every node (or realm) in the path from a source to a destination need to give consent to send traffic. This gives the nodes control over the traffic passing through them, making it suitable for networks like the Internet, where there are multiple providers (or administrative domains) and the paths are mostly static. In destination-based consent architectures, permission is

given to send traffic to a destination on any of the available paths; intermediate nodes honor those permissions and forward the traffic. This architecture is useful for networks where the paths are dynamic, as in mobile ad-hoc networks.

In a consent-based system, senders are given the permission to send traffic in the form of verifiable proofs of consent (capabilities). The nodes perform bandwidth enforcement by rate controlling the bandwidth used for the flows that are part of the capability. In destination-based consent architectures, it is possible to use the capability to reach a destination on multiple paths. In those cases, not all traffic corresponding to a capability may go through a node. Hence, any single node may not be able to enforce the bandwidth constraints of the capability. Furthermore, a node that has authority over multiple destination nodes may assign permission to reach those destinations in a single capability. Hence, the same capability may be used for multiple unrelated flows. Even if all the traffic passes through a node, the node may be unable to enforce the bandwidth constraints across unrelated flows due to high processing required to account for traffic across the flows. It is also possible for certain nodes to collude with senders allowing for larger bandwidth than the one allocated in the capability. When misuse prevention is not feasible, we need a detection mechanism. Once misuse is detected, the capability may be revoked or temporarily not honored, or the node misusing the capability may be isolated.

Recently proposed DIPLOMA [4, 3] is a destination-based consent architecture for MANETs based on the concept of network capabilities [5]. A capability is a cryptographically sealed token of authority that has associated rights. DIPLOMA capabilities propagate both access control rules and traffic-shaping parameters that should govern a node's traffic. All the nodes in the path from a source to a destination can verify and enforce the capability. The architecture is based on deny-by-default paradigm, where nodes can only access the services and hosts they are authorized for by the capabilities given to them. Thus, DIPLOMA provides two main features: access control of the end-host services, and protection of network bandwidth.

In this paper, we identify the sources of misuse in DIPLOMA and provide solutions for detecting those misuses. A misuse may constitute either the use of a capability in multiple paths to a destination, or the use of the same capability to multiple destinations. The detection of misuse may be done based on the information locally available to the node (local detection), or based on the information exchanged among the nodes (distributed detection).

To provide solutions for detecting misuses, we enhance the capability establishment protocol to enable nodes to detect the misuses. We also describe the protocols for communicating the information about the flows going through the nodes to enable distributed detection. We also provide efficient algorithms for detecting misuses.

The node detecting a misuse should be able to provide the proof of the same, so that other nodes can take action based on the misuse. Our solution can provide the proof of the misuse, so that rogue nodes cannot exploit the misuse detection

algorithms itself. Our solution also handles privacy issues associated with the exchange of information about the flows.

We implemented our algorithms in the Orbit lab testbed [1]. We show that the algorithms require minimum processing and memory. We also show that the amount of information exchanged for the misuse detection algorithm is minimal. We also conduct extensive experiments on capability misuses, and show that our system effectively detects and contains these misuses.

2 Misuses in consent-based architectures

The misuses in consent-based architectures depend on the type of the architecture and the resources it is trying to protect. It depends whether the consent is for a particular path or on any path to the destination. The misuse also depends on whether the consent has any bandwidth constraint or it is just an access constraint (i.e. unlimited bandwidth).

In architectures like network capabilities [5, 16, 17] and visas [7], consent is given to access the receiver in any path. In network capabilities, all the nodes from a source to destination participate in the protocol. In visas, on other hand, only the source and destination networks are involved in the protocol. In ICING architecture [12], consent is given for a particular path. ICING also requires that all the intermediate nodes from the source to destination give explicit consent for the packet to go through. In DIPLOMA, consent is given for any path, but enforcement is done at all the intermediate nodes from source to destination.

Another factor that influences the misuse is whether the consent based architecture depends on the trusted nature of the routers or intermediate nodes and security of the communication medium. If the protocol assumes trusted intermediate nodes, it may be possible to overcome the protection by compromising the routers. In general, protocols designed for wired networks assume trusted routers. DIPLOMA is designed for wireless networks where the routers may not be trusted and the communication medium is broadcast in nature.

A consent-based architecture may provide only access control or may additionally provide bandwidth limitations. It is easier to enforce bandwidth constraints on the path-based architectures. In destination based architecture like DIPLOMA, which also provides bandwidth constraints, it is a challenge to enforce the bandwidth constraints due to use of multiple paths to a destination; the constraints has to be enforced across all the paths. The focus of this paper is to detect and recover from misuses involving multiple paths.

3 DIPLOMA Overview

We assume wireless ad-hoc network setting where the nodes have limited mobility. In DIPLOMA architecture, the resources needed to access a service are allocated by the *group controller(s)* (GCs) of the MANET. Group controllers are nodes responsible for maintaining the group membership for a set of MANET nodes, and *a priori* authorize communications within the group. This means that GCs do not participate in the actual communications, nor do they need

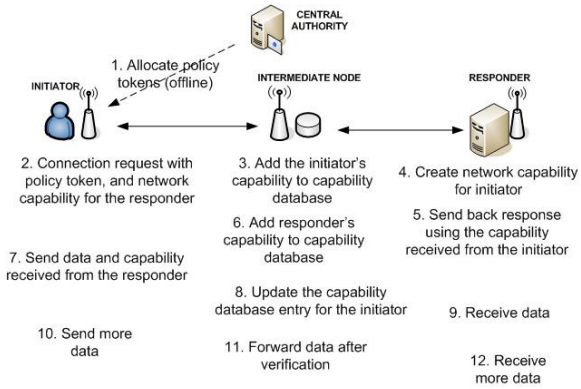

Fig. 1. System overview

to be consulted by nodes in real time; in fact, if they distribute the appropriate policies ahead of time, they need not even be members of the MANET. In some cases, the GC may be reachable through a high-energy-consumption, high-latency, low-bandwidth long-range link (*e.g.,* a satellite connection); interactions in such an environment should be kept to a minimum, and only for exceptional circumstances (*e.g.,* for revoking access for compromised nodes). The resource allocation by GC to a node is represented as a credential (capability) called *policy token*, and it can be used to express the services and the bandwidth a node is allowed to access. They are cryptographically signed by the GC, which can be verified any node in the MANET.

When a node (initiator) requests a service from another MANET node (responder) using the policy token assigned to the initiator, the responder can provide a capability back to the initiator. This is called a *network capability*, and it is generated based on the resource policy assigned to the responder and its dynamic conditions (*e.g.,* level of utilization).

Figure 1 gives a brief overview of DIPLOMA. All nodes in the path between an initiator to a responder (*i.e.,* nodes relaying the packets) enforce and abide by the resource allocation encoded by the GC in the policy token and the responder in the network capability. The enforcement involves both access control and bandwidth allocation. A responder accepts packets (except for the first) from an initiator only if the initiator is authorized to send, in the form of a valid network capability. It accepts the first packet only if the initiator's policy token is included. An intermediate node will forward the packets from a node only if they have an associated policy token or network capability, and if they do not violate the conditions contained therein. Possession of a capability does not imply resource reservation; they are the maximum limits a node can use. Available resources are allocated by the intermediate nodes in a fair manner, in proportion to the allocations defined in the capability.

The capability need not be contained in all packets. The first packet carries the capability, along with a transaction identifier (TXI) and a public key. Subsequent packets contain only the TXI and a packet signature based on that

public key. Intermediate nodes cache policy tokens and network capabilities in a *capability database*, treating them as soft state. A capability database entry contains the source and the destination addresses, TXI, the capability, public key for the packet signature and packet statistics. Capability retransmissions update the soft state of intermediate nodes when the route changes due to node mobility. The soft state after a route change is also updated using an on-demand query for the capability database entry from the upstream nodes.

3.1 Misuses in DIPLOMA

In this section, we identify ways of misusing capabilities in destination-based consent systems like DIPLOMA. These includes simultaneous use of a capability on multiple paths to get more than allocated bandwidth, or misusing a policy to create network capabilities more than the policy is entitled to.

Misuse of policy tokens: Policy tokens are capabilities allocated by the group controllers to the nodes to access the services running on other nodes in MANET. The node for which the policy token is allocated is called *owner* of that policy token. A policy token contains the owner, the destination node, the type of service, the allocated bandwidth, and the signature of the group controller. The destination field of a policy token may correspond to a specific host or a group of hosts. A sender (*i.e.,* the owner) can send traffic to multiple receivers simultaneously using a policy token that has authorization to access those receivers. While accessing multiple receivers, the sender should not exceed the total bandwidth allocated to that policy token. A misbehaving sender may try to exceed this allocation by deliberately communicating with multiple receivers without satisfying the overall bandwidth constraints of the capability. We call this misuse as *concurrent-destination misuse*.

Another way to misuse the capabilities is to use multiple paths to the receiver. The sender may use the same capability on multiple paths, and may claim the bandwidth allocation of the capability in each of the paths. This way the sender can bypass the bandwidth enforcement that is performed by the intermediate nodes. Though the receiver can easily detect this kind misuse, it might be collaborating with the sender to receive a larger bandwidth. We call this misuse as *multi-path misuse*.

Misuses of network capabilities: Network capabilities are the capabilities issued by the receiver nodes to the senders that authorize sending traffic to those receivers. They are similar to policy tokens, except that the destination field cannot be arbitrary; it has to be the receiver that issued the capability. The capabilities also need to contain a signed policy issued by the group controller authorizing the receiver to issue such a capability.

Nodes can misuse a network capability in two ways: either by a receiver issuing more than it is entitled to, or by a sender sending more than the network capability. A sender could misuse the network capability by sending the capability over multiple paths. This is same as the multi-path misuse. Note that concurrent destination misuse is not possible with network capabilities, since those capabilities have fixed destination.

A receiver creating network capabilities may perform another form of misuse. The receiver needs to conform to the policy while creating network capabilities. A policy puts an upper bound on the amount of bandwidth a receiver may allocate to capabilities simultaneously. A receiver might not abide by this policy, and might allocate more network capabilities than it is entitled to. We call this misuse as *policy misuse*.

4 System Architecture

Figure 2 shows the architecture of the misuse detection system in DIPLOMA. The DIPLOMA engine, which is responsible for packet processing and capability enforcement, collects the information about the capabilities going through the node and provides them to the misuse detection engine. This information is stored in the local records table. The detection engine may also receive the information about the communication flows and the associated capabilities from other nodes, which are stored in the external records table. The detection engine periodically runs the misuse detection algorithm described in Section 5 on these records. Whenever the algorithm detects a misuse, it informs the local DIPLOMA engine as well as the misuse detection engines of the other nodes. The DIPLOMA engine makes use of this information while accepting the capabilities for connection establishment and packet forwarding.

Based on where the flow information is obtained, the misuse detection algorithm is classified as local or distributed.

Local detection: In many cases, we can detect capability misuse using the local information a node has, received either through the packets passing through that node, or by listening to the channel and snooping on the packets in its neighborhood. For example, a receiver can detect any misuse by a sender directed towards it. Nodes in the sender's neighborhood may be able to hear all the packets by listening to the channel. In those cases, a neighboring node will be able to detect any misuse by a sender, and provide a proof of the misuse.

Distributed detection: When it is not possible to detect the misuse based on local information, we resort to distributed detection. For example, if the sender is using a directional antenna, then its neighboring nodes may not be able to hear all the packets it has sent. A misuse may be targeted towards multiple receivers; hence, a single receiver cannot detect it. Even if a misuse involves a single receiver, the receiver may be colluding with the sender and may not report the misuse. In distributed detection, the nodes periodically exchange information about the flows passing through them. The misuse detection algorithm is run using the combination of the information a node collected locally and what it received from other nodes. In distributed detection, one or more nodes in the MANET are designated as *verifier nodes*. All the other nodes (called *collector nodes*) send information about the flows going through them to one or more of these verifier nodes. The verifier nodes run the misuse detection algorithm, and inform the collector nodes about any misbehaving nodes and the associated capabilities, along with proof of misuse.

Figure 3 illustrates the types of the detection methods that are useful in various misuse scenarios. Whether distributed detection is required or not is

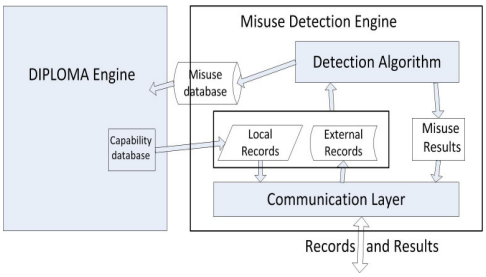

Fig. 2. Misuse detection architecture

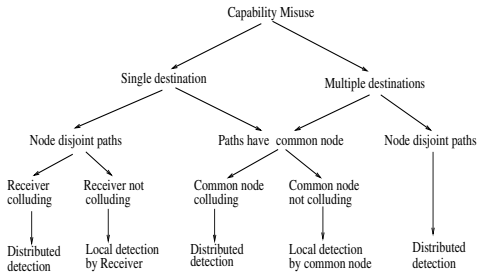

Fig. 3. Various types of capability reuse and detection algorithm

dictated by whether there exists any common node in the misuse paths, and whether the common node is willing to co-operate. A common node can run the detection algorithm based on the local information alone to identify, and report the misuse. For the misuse involving multiple paths to a single destination, the receiver is always a common node. If all the common nodes are colluding with the sender, then a distributed detection is required.

Our solution consists of the following: a capability-encoding scheme that aids detection, protocols for exchanging information for distributed detection, and detection algorithms.

4.1 Capability encoding

In the DIPLOMA architecture, it is permissible to use a capability for multiple communication sessions concurrently. For example, a node possessing a policy token to communicate with a group of destinations may be simultaneously communicating with multiple nodes in that destination group. Similarly, it is possible to use a policy authorizing the issue of the network capability to create multiple capabilities simultaneously. For example, a node may be receiving packets from multiple source nodes, and may want to allocate network capabilities to those senders based on a single policy authorizing the allocations. Both of these concurrent uses of policies are valid as long as the nodes do not use (or allocate) more bandwidth than allowed by the policies. When the nodes split the bandwidth of a policy into multiple capabilities, we need protocols that enable other nodes to check if these capabilities are within the limit.

While sending the policy token or while creating receiver capabilities, the owner has to decide on how to split the available bandwidth. The protocol allows dividing the available bandwidth into 32 or 64 equal sized slots, which are represented using a bitmask of 32 or 64 bits. We call this bitmask as the *allocation vector*. A bit in the allocation vector is set, if the corresponding bandwidth slot is used. The allocation vector is included in the capability request packets, as well as on the network capabilities created by the receivers. For a policy token that a sender is using to communicate with multiple destinations, the allocation vector on a capability request indicates the portion of the available bandwidth allocated to that communication session. When a sender uses multiple paths to reach a destination, the allocation vectors on the capability requests on each of

the paths indicate the portion of the available bandwidth from the capability allocated to that path. If a receiver node creates multiple network capabilities, based on a policy, the allocation vector field in the capability indicates the portion of the available bandwidth allocated to that capability. Note that there could be multiple bits set in the allocation vector indicating bandwidth allocation proportional to number of bits set in the vector.

It is permissible to allocate the same bandwidth slot to different capabilities, derived from the same policy, at different times. Every capability request and network capabilities contain a start time stamp and a validity duration, which indicates the time until they are valid. To extend its validity, the owner needs to create a new request. Hence, a misuse constitute the existence of two capability requests for the same capability, or two network capabilities for the same policy that has a common bit set in their allocation vector at the same time.

Our misuse detection algorithms do not depend on the data structure used for dividing the allocated bandwidth. Allocation vectors have easy representation, and allow for easy unions and intersection operations using bitwise operators. If finer granularity is needed in dividing the bandwidth, one could use other representations like slab allocation.

4.2 Communication protocol

When a sender node wants to communicate with a receiver node, it uses the *capability request* packet to inform the intermediate nodes about the capability that will be used for the communication [4]. This request is signed by the sender's private key. To avoid any non-repudiation of the capability request by the sender, the nodes participating in the misuse detection are required to store the capability request and the signature. The capability request have many information that are not necessary for the misuse detection. Storing this information puts undue burden on the misuse detection nodes. One way to solve this problem is to sign the information that are essential for misuse detection (called *misuse detection block*), separately from that which are not useful for misuse detection (called *capability establishment block*). Unfortunately, this requires senders to perform two expensive signature operations. As a compromise between the amount of information stored by nodes and the processing needed at the sender, we sign the request in two steps. First, the sender computes the hash of the capability establishment block. Then, it signs the combination of this hash and the misuse detection block. To prove capability misuse by a sender, the nodes only need to keep track of the hash, the misuse detection block, and the signature.

A capability request packet contains the capability establishment block and misuse detection block. A node can verify the capability request by first computing the hash of the capability establishment block, and verifying the signature for the combination of hash and the misuse detection block. A valid signature indicates that the packet was not tampered with. We also use a similar scheme for signing the networks capabilities created by the receiver nodes.

Information exchange: For the distributed detection, the detector nodes send information about the communication sessions and the capabilities passing through them to the verifier nodes. The detector nodes use the underlying

MANET unicast or multicast routing protocol to reach the verifier nodes. The information required to detect misuse consists of the identity of the sender node, the transaction identifier, serial number and the issuer of the capability, the allocation vector, the time stamps, and the previous and next nodes in the path. This information about the communication session is called a *record*. A node can send multiple records in a packet. The nodes sign the packet using their private keys. A similar record is also send for the network capabilities for detecting misuses in them. The algorithms used for detection of the misuse by the senders, and the receivers are similar. Hence, we will deal only with the sender misuse in rest of the paper.

5 Detection Algorithms

In this Section, we describe the DIPLOMA misuse detection algorithms that are used for both local and distributed detection. Then we describe how a verifier node can provide a proof of misuse. Finally, we provide solution for handling the privacy issues in our misuse detection architecture.

Recall that there is a misuse if there are two communication sessions that use the same capability and have a common bit set in the allocation vectors with overlapping validity periods. Hence, the goal of the algorithm is to find such communication sessions. To that end, the algorithm first groups the records corresponding to a communication session. This is because there could be multiple records for a communication, received from different collector nodes. Once the records of communication sessions are grouped together, the algorithm look at records across the communication sessions to identify misuse.

In DIPLOMA, a communication session can be uniquely identified by the (transaction identifier, sender identity) pair. If the sender uses multiple paths to a destination, the sender is required to use different transaction identifiers for each path.

The misuse detection algorithm has two phases. In the first phase, it removes the duplicate records for each communication sessions from the collection of records it gathered locally and from other nodes. It also detects if a sender uses the same transaction identifier on multiple paths. The output of the first phase is a set of records, consisting of at most one record for a transaction identifier per sender. This phase is not required if all the records are obtained from the capability database of the local DIPLOMA engine, as the engine already prevents duplicates. In the second phase, the algorithm detects if there is any misuse on the filtered records output by the first phase.

5.1 Phase 1 - Duplicate removal and multipath detection

The removal of duplicate records for a communication session is performed by sorting the records based on (sender, transaction id) pair and keeping only one record per pair. However, this step will not detect use of the same transaction identifier by a sender on multiple paths. To detect the multiple path misuse, we use the following property of the paths.

If the same transaction id is used in two different paths to a destination, then there will be two records for it that has a common previous node or a common

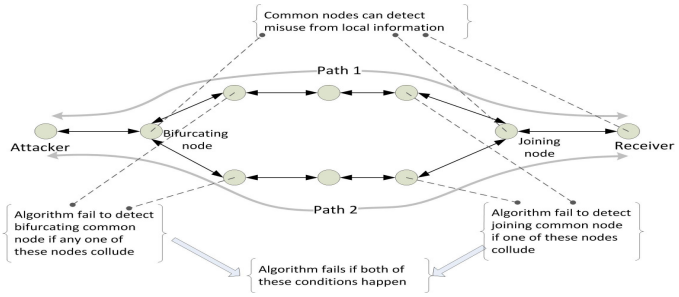

Fig. 4. Properties of multiple paths in aiding misuse detection

next node. This is because since the source and the destination nodes are common in both paths, the paths need to bifurcate at some node and join at another node. If the paths are bifurcating at any node other than the sender, or joins anywhere other than the receiver, then the DIPLOMA engine at the common nodes in the path can detect the misuse during the connection establishment stage. If the paths are node disjoint and the receiver is not colluding with the misbehaving sender, then the receiver can detect misuse. If all the common nodes are colluding with the sender, then the phase 1 algorithm can detect the misuse looking for the common nodes. This is depicted in Figure 4.

Algorithm 1 describes the phase 1 algorithm. It goes through the records corresponding to the same (sender, transaction id) pair and verifies that all the records use the same capability, allocation vector and time stamps. It also stores the previous nodes and the next nodes of each record in temporary arrays. The presence of duplicates in these arrays indicates a misuse.

Analysis: The algorithm will fail to detect a multipath misuse if certain nodes collude with the sender and do not provide the relevant records to the verifier. If the common nodes, including the receiver, collude with the sender, then local detection of the multipath misuse will fail. If at least one of the nodes in the path next to the common node colludes, where the forking of the paths has occurred, then the algorithm will fail to detect that common node. Similarly if one of the nodes before the common node at which the joining of the paths take place, then also the algorithm will fail to detect the common next node. The algorithm will fail to detect a multipath to a destination, when it cannot detect both the common previous and next nodes. This is depicted in Figure 4. It is still possible for a verifier to detect that it has not received records from some of the nodes (which may be colluding with the sender), because of the existence of two path fragments (as opposed to one path) for the transaction identifier. However, we cannot use this against the sender, because of the possibility of packet losses, or the possibility of a node deliberately not sending the records to the verifier.

5.2 Phase 2 - Reuse of the capability detection

The second phase, the algorithm detects misuse of capabilities across the communication sessions. Recall that a misuse is identified by a common bit set in the allocation vectors at overlapping validity periods. The input to phase 2 is the records output by phase 1. Hence, there is only one record per communication

Algorithm 1 Duplicate-record removal & multi-path detection

```
1: L_i ← List of all (source node, transaction id) pair
2: L_u ← NULL {Output list of unique records}
3: for all id ∈ L_i do
4:    L_r ← List of records for id
5:    H_prev, H_next ← NULL
6:    Add first record of L_r to L_u
7:    for all rec ∈ L_r do
8:       if attributes of rec different from head(L_r) then
9:          print Misuse. Different attributes for same transaction
10:      else if prevhop(rec) ∈ H_prev or nexthop(rec) ∈ H_next then
11:         print Misuse. Same transaction in different paths
12:      end if
13:      Add prevhop(rec) to H_prev and nexthop(rec) to H_next
14:   end for
15: end for
16: return L_u
```

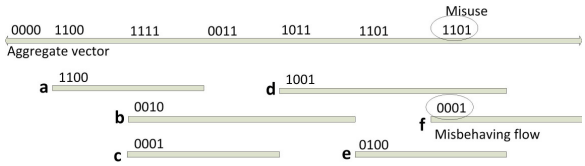

Fig. 5. Computation of aggregate allocation vector and misuse detection using interval graphs.

session. The algorithm goes through all the records corresponding to each of the capabilities and detects misuse.

The algorithm treats the records as an interval graph, where each record corresponds to an interval for which they are active. There is an allocation vector associated with each interval, which is the allocation vector of the corresponding record. We define the *aggregate allocation vector* at any point of time as the union of the allocation vectors of the intervals passing through it. There is misuse at any point in time if the intersection of any two intervals passing through it is not empty. This is depicted in Figure 5.

Once the interval graph is formed, we can detect any misuse in linear time in the number of intervals. The algorithm goes through the end points of the intervals in increasing time and updates the aggregate allocation vector. At the beginning of the algorithm, this vector is set to NULL. Whenever the algorithm considers the beginning of an interval, it checks if the intersection of the aggregate allocation vector and the allocation vector of that interval is non-empty. If it is not empty, then there is misuse. Otherwise, the allocation vector of that interval is added to the aggregate vector. Similarly, when the algorithm considers the end of an interval, its allocation vector is subtracted from the aggregate allocation vector. If there are both entering and leaving intervals at any point, then the leaving operation is considered before the entering operation. This is because the

new entering interval could use slots from the leaving interval, without causing misuse.

Figure 5 illustrates the computation of aggregate allocation vector and the misuse detection. There are six flows labeled as $a, b, \ldots, f$. They are represented as the intervals in which they are active. Their corresponding allocation vectors are also shown. For simplicity of illustration, we use the allocation vector of 4 bits. The vector on the top line shows the aggregate allocation vector when the flows enter or leave the system. The aggregate vector at any point is the union of the allocation vectors of the interval going through that point. There is no misuse for flows $a, b, \ldots, e$. The flow f uses one of the slots of flow d, hence there is misuse. The aggregate allocation vector before f entered the system was 1101. The flow f uses one bandwidth slot. The sender assigned it the slot 0001, which is a reuse of the existing slot. If the sender had assigned it the slot 0010, then there would not be any misuse. Hence, it is important that senders allocate the right bandwidth slot for flows.

Creating an interval graph from the records is performed by sorting the endpoints of the interval. In fact, our algorithm maintains two sorted lists: one for the starting points of the intervals and the second for the ending points of the intervals. The algorithm is given in Algorithm 2.

Algorithm 2 Phase 2 - checking for reuse of bandwidth slots

```
1: L_i ← List of all (capability id, issuer) pair
2: for all id ∈ L_i do
3:    L_r ← List of records for id
4:    L_s ← Records in L_r sorted on start time
5:    L_e ← Records in L_r sorted on end time
6:    aggregate ← φ
7:    while L_s not empty do
8:       time_s ← starttime(head(L_s))
9:       time_e ← endtime(head(L_e))
10:      if time_e ≤ time_s then
11:         rec ← head(L_e)
12:         L_e ← L_e − rec
13:         aggregate ← aggregate − allocvector(rec)
14:      else
15:         rec ← head(L_s)
16:         L_s ← L_s − rec
17:         if aggregate ∩ allocvector(rec) ≠ NULL then
18:            print Misuse. Reuse of bandwidth slots.
19:         end if
20:         aggregate ← aggregate ∪ allocvector(rec)
21:      end if
22:   end while
23: end for
```

5.3 Privacy issues

In the protocol presented so far, the detector nodes send the information about all the flows going through them to the verifier. Even though the records contain only the sender node identity and does not have the receiver identities of the flow, it is still possible to deduce the receiver identity by following the path using the previous and next hop information. Hence, the verifier can know about the source and destination of all the flows. Another privacy concern is the knowledge about the number of flows a sender is sending, even if the verifier is not interested in knowing the receivers.

We can modify the protocol to honor privacy, and still detect the misuse as follows. The detection algorithm continues to function even if all the fields in the records, except the time stamps and the allocation vector, were encrypted with a key that is common across all the flows corresponding to a capability. The detector nodes can create such a key by taking a known function (*e.g.,* hash) of the capability. Since the flows are going through the detector nodes, they know about the complete capability associated with the flow. However, verifier knows only about their serial numbers, and cannot recreate the keys. Hence, verifier cannot decrypt the records for the flows not going through it. In this scheme, the verifier can still get information about all the flows that use any of the capabilities passing through it.

6 Experimental evaluation

In this section, we study the effectiveness of the misuse detection algorithms. The experiments were conducted in the Orbit Lab Testbed [1]. The algorithms were implemented on DIPLOMA systems running on Debian Linux with kernel 2.6.30. We analyzed memory, bandwidth and processing overheads, and found them to be minimal; due to lack of space we omit the results.

6.1 Effectiveness in Containing attacks

Now we study the effectiveness of misuse detection algorithm in detecting and containing the attacks. We used the topology given in Figure 6, created by assigning non-overlapping channels of 802.11b and 802.11a to the links [3].

In this set of experiments, there are four flows: two by good nodes and two by the attacker. Each of the senders is allocated a capability of 4 Mbps. All the flows were created using UDP *iperf*. The good nodes, 2 and 5, send traffic at 4 Mbps each to the destinations nodes 7 and node 4 respectively. We denote those flows as flow 1 and flow 2 respectively and call them good flows. Flow 1 takes the path $2-8-6-7$ and flow 2 takes the path $5-8-3-4$. These flows are started at time 0 seconds and last for 120 seconds. The attacker, which is the node 1, sends flows to nodes 4 and 7, which we denote by flows 3 and 4. We call these attack flows. Each of the attack flows are 12 Mbps, even though the attacker has one capability with the allocated bandwidth of 4 Mbps, which can be used for either destination. Flow 3 is started at time 30 seconds and takes the path $1-2-3-4$. Flow 4 is started at time 60 seconds and takes the path $1-5-6-7$. Both the flows last for 120 seconds, and use the same capability with

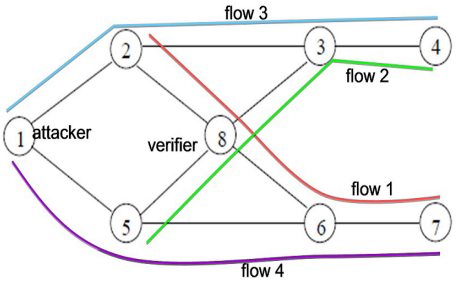

Fig. 6. Topology to study the performance of detection algorithm

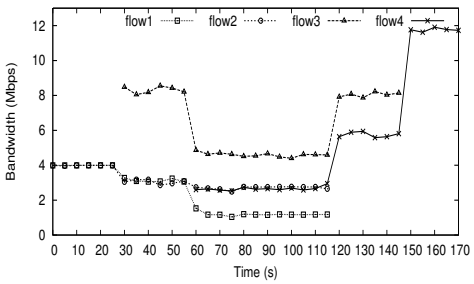

Fig. 7. Bandwidth of flows in a system that does not require consent to send

all the bits in the allocation vector set. Hence, the attacker launches two types of attacks. First, it is sending higher bandwidth than that is allocated in the capability, which starts at time 30 seconds. Secondly, it uses the same capability to talk to multiple destinations simultaneously using the same bandwidth slot. This attack starts at time 60 seconds.

We conduct three sets of experiments. The first is called the *original*, and does not require any consent for sending the traffic. This scheme cannot protect against both the attacks. Then we use the consent-based scheme, where DIPLOMA requires capabilities for sending traffic. This DIPLOMA without misuse detection, can handle the first bandwidth hogging attack but cannot prevent reuse of the capability across the flows. Finally, we use DIPLOMA with misuse detection to handle both types of attacks. For each of the experiments, we report the bandwidth of each of the flow over a period of time. All experiments were run 6 times, and we show the average bandwidth. The iperf servers (receivers) measured the bandwidths at 5 second intervals.

Figure 7 shows the results for the original system. In this system, until 30 seconds, where there are only two flows from the good nodes, both the flows get their requested 4 Mbps bandwidth. At 30 seconds, when the first attacker flow (flow 3) arrives, the bandwidth of the good flows drop as the attacker flow takes up most of the bandwidth. The attacker receives a bandwidth between 8 and 8.5 Mbps, whereas the bandwidth of good flows drop to 3 Mbps. At 60 seconds, when the second attack flow arrives (flow 4), the bandwidth of the good flows further drops along with the bandwidth of the first attacker flow. The bandwidth of the flow 1 and flow 2 drops to 1.1 Mbps and 2.7 Mbps respectively. The bandwidth of first attacker flow drops to 4.6 Mbps. The new attacker flow receives 2.6 Mbps. This trend continues until 120 seconds, when the good flows end. At that point, the first attacker flow bandwidth recovers to its previous levels (8 Mbps) and the second attacker flow receives a higher bandwidth of 5.8 Mbps. At 150 seconds, when the first attacker flow ends, the second attacker flow receives 11.8 Mbps, which is close to the requested bandwidth of that flow.

Figure 8 shows the results for the DIPLOMA scheme when misuse detection is not in effect. Until 30 Seconds, where the attacker had not started sending any traffic, the good flows 1 and 2 get the bandwidth that are allocated in

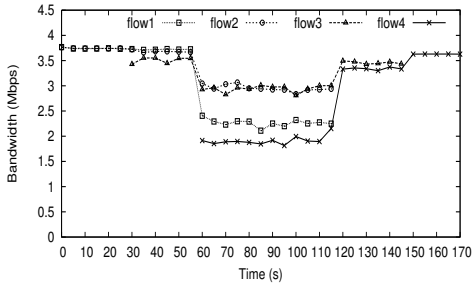

Fig. 8. Bandwidth of the flows in DIPLOMA without misuse detection

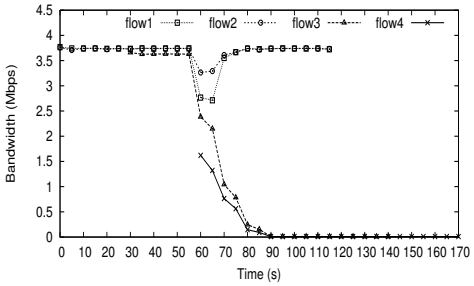

Fig. 9. Bandwidth of the flows in DIPLOMA with misuse detection

their capability. The bandwidth reported by the flows is 3.74 Mbps, which is slightly less than the allocated 4 Mbps due to the additional headers present in DIPLOMA packets. At 30 seconds, when the attacker starts sending the first attack flow (flow 3) at the rate of 12 Mbps, the bandwidth of the good flows drops only slightly to 3.71 Mbps. The attacker gets a bandwidth of 3.5 Mbps, which is closer to its allocated bandwidth. Hence, the consent-based DIPLOMA scheme is able to protect the good flows and contain the attacker to its allocated bandwidth. At 60 seconds, when the second attack flow (flow 4) starts, the bandwidth of the good flows and the attacker drops. This drop for the good flow is not as drastic as the original scheme. Here the bandwidth of the flow 1 drops to 2.3 Mbps and that of flow 2 drops to 3 Mbps. The existing attacker flow drops to 2.9 Mbps, and the new attacker flow receives 1.9 Mbps bandwidth. This drop in bandwidth is due to limited available bandwidth on the network. The attacker is reusing the capability at this point, and DIPLOMA ends up honoring the same capability in two node disjoint paths. At 120 s, the genuine flow ends. At that point, the first attack flow bandwidth moves back to its original level of 3.5 Mbps. The second attacker flow bandwidth ends up at 3.3 Mbps. This increase is due to the freed up capacity from the good flows. At 150 seconds, the first attack flow ends and the second attack bandwidth increases slightly to 3.6 Mbps. Even then, the bandwidth of the individual attack flows does not go above the allocated bandwidth of 4 Mbps, because DIPLOMA enforces the bandwidth. Therefore, in DIPLOMA without the misuse detection, an attacker cannot go above the allocated bandwidth in a single path. However, it can bypass that check by sending traffic to multiple destinations in disjoint paths, if the capability permits it. When this happens, genuine traffic is affected due to capacity sharing.

Figure 9 shows the results for the DIPLOMA system with misuse detection. The behavior of the system is same as DIPLOMA without misuse detection, until the misuse happens at 60 seconds. At 60 seconds, when the attacker sends the second flow (flow 4), which constitutes a capability reuse, the behavior changes from DIPLOMA without the misuse detection. In this case, the nodes send the record to the verifier (node 8), which detects the misuse. In this experiment, the period at which nodes send the record is 10 seconds. Hence, the verifier detects the misuse before 70 seconds, and informs the forwarding nodes. The forwarding

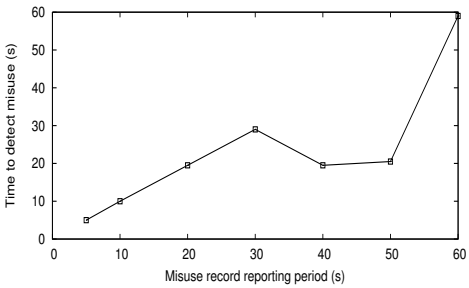

Fig. 10. Time to detect the misuse for different record reporting periods

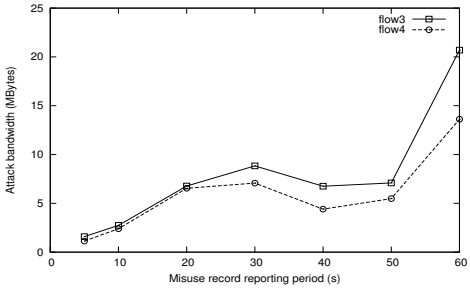

Fig. 11. Additional attack traffic after misuse for different record reporting periods

nodes starts to block the attack flows. The drop of the attack bandwidth is gradual due the nature of our implementation [3]. The bandwidth of the good flows drops to 2.8 Mbps and 3.3 Mbps for a short duration (10 seconds) at 60 seconds while the misuse detection and recovery takes place. At 85 seconds, the recovery is complete and the bandwidth of the good flows moves back to the levels before the reuse attack. Even after the genuine flow ends at 120 s, the attacker flows continue to be blocked due to their misuse action. This continues even after the attacker stops the misuse at 150 seconds, when the first attacker flow ends. Hence, DIPLOMA can effectively contain capability misuse.

6.2 Speed of detecting misuse

Next, we study how fast our scheme can detect the misuses and communicate with the affected nodes. The time to detect the misuse will depend on the frequency at which records are sent to the verifiers. Hence, we study the misuse detection speed as a function of that period, using the same set of flows as in the previous experiment.

Figure 10 plots the time required at the detector nodes to get the misuse notification after the misuse happened for various record reporting periods. For the periods up to 30 seconds, the time to detect is the same as the period. Hence, the misuse is detected as soon as the record is received to the verifier. For the periods 40 and 50 seconds the time to detect misuse was less than the period, and for 60 seconds the detection time was the same as the period. This is because for 40 and 50 seconds experiments, the start of the attack and the start of the period may not have been synchronized. Hence, it is possible for the detectors to send the record to the verifier in time less than the period after the attack has happened. The verifier detects the misuse upon receiving the records.

6.3 Attack bandwidth after misuse

Now we study the amount of additional traffic the attacker is able to send after the misuse. This quantity depends on how fast the system is able to detect the misuse and take action against the attacker. Hence, we study the additional traffic as a function of record reporting period. We use the same flows and attack scenario as the previous experiment and measure the data received at the receiver for the attack flows (flows 3 and 4), after the misuse has started.

Figure 11 plots the amount of traffic received at the receivers for the attack flows after the misuse, for different record reporting periods. Up to 30 seconds, the attack traffic increases as the record reporting time increases. This is because the misuse traffic will be treated as the legitimate traffic and allowed to pass through until the misuse is detected; and the misuse detection time is proportional to the record reporting period. Note that even if the attacker is trying to send the traffic at 12 Mbps, the throughput the flows receive is less than 4 Mbps due to bandwidth enforcement by the intermediate nodes. In this set of experiments, the flow 3 had slightly higher bandwidth than flow 4, similar to the experiment in Figure 9. For 40 and 50 seconds, the attack traffic drops is less than that of 30 seconds, as the misuse is detected before the complete period as explained in the previous experiment.

7 Related Work

The concept of capabilities was used in operating system for securing resources [15]. Follow-on work investigated the controlled exposure of resources at the network layer using the concept of "visas" for packets [7], which is similar to network capabilities. More recently, network capabilities were proposed to prevent DDoS attacks [5]. We extend the concept to MANETs and use it for both access control and traffic shaping [4, 3]. All these works represent destination-based consent architectures. Path-based consent architectures in the context of Internet have also been proposed [12].

Intrusion detection systems (IDS) for MANETs is an active area of research [10]. In the Local Intrusion Detection Architecture [2] communities of nodes are formed, which exchange various security data and intrusion alerts. The nodes can also place mobile agents in the other nodes, to do a specific mission in an autonomous and asynchronous manner. Distributed IDS architecture [18] is another proposed IDS architecture for MANETs. It uses a local detection engine, with the input from the local data collection, and a co-operative detection engine, with the input from the neighboring nodes to detect intrusions.

Solutions are also proposed to specifically detect attacks on routing protocols based on the protocol specification. [8] detects violations in the protocol specification based on an extended finite state automation (EFSA) of AODV. Distributed Evidence-Driven Message Exchange Intrusion Detection Model (DEMEM) [13] detects inconsistency among the routing messages in OLSR. [6] models the attacks on AODV protocol using attack tree and identify the damages. Proposals are also made when to isolate a misbehaving node based on the criticality of that node in maintaining the connectivity [14]. There is also a rich literature on detecting DoS and node replication attacks in sensor networks [9, 11].

8 Conclusions and Future Work

We identified sources of misuse in destination-based consent architectures and provided a distributed solution for detecting misuses. Our solution is demonstrated for DIPLOMA, a consent-based architecture for MANETs. DIPLOMA is a *deny-by-default* distributed policy enforcement architecture based on capabilities. We provided capability encodings and protocols for exchanging information

for distributed misuse detection, and presented efficient algorithms for misuse detection. We showed through experimental evaluations that our algorithms are effective in identifying and containing the misuse. In the future, we plan to measure the energy consumed for detecting misuses and the energy savings due to preventing attacks, and to look at misuse in path-based consent architectures.

Acknowledgements This work was supported in part by the National Science Foundation through Grant CNS-07-14277. Any opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

References

1. Orbit Lab Test-bed. *http://www.orbit-lab.org*.
2. P. Albers, O. Camp, B. Jouga, L. Me, and R. Puttini. Security in ad hoc networks: A general id architecture enhancing trust based approaches. *IEEE Network, WIS*, 2002.
3. M. Alicherry and A. D. Keromytis. DIPLOMA: Distributed Policy Enforcement Architecture for MANETs. *International Conference on Network and System Security*, September 2010.
4. M. Alicherry, A. D. Keromytis, and A. Stavrou. Deny-by-Default Distributed Security Policy Enforcement in Mobile Ad Hoc Networks. *SecureComm*, 2009.
5. T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial-of-Service with Capabilities. *Proc. of Hotnets-II*, 2003.
6. P. Ebinger and T. Bucher. Modelling and analysis of attacks on the manet routing in aodv. *LNCS*, 2006.
7. D. Estrin, J. C. Mogul, and G. Tsudik. Visa protocols for controlling interorganizational datagram flow. *IEEE J. on Selected Areas in Comm.*, May 1989.
8. Y. Huang and W. Lee. Attack analysis and detection for ad hoc routing protocols. *RAID*, 2004.
9. J. McCune, E. Shi, A. Perrig, and M. Reiter. Detection of denial-of-message attacks on sensor network broadcasts. *IEEE Security and Privacy*, 2005.
10. A. Mishra, K. Nadkarni, and A. Pactha. Intrusion detection on wireless ad hoc networks. *IEEE Wireless Communications*, 2004.
11. B. Parno, A. Perrig, and V. Gligor. Distributed detection of node replication attacks in sensor networks. *IEEE Security and Privacy*, 2005.
12. A. Seehra, J. Naous, M. Walfish, D. Mazires, A. Nicolosi, and S. Shenker. A policy framework for the future internet. *ACM Workshop on Hot Topics in Networks (HotNets)*, October 2009.
13. C. Tseng, S. Wang, C. Ko, and K. Levitt. Demem: Distributed evidence-driven message exchange intrusion detection model for manet. *RAID*, 2006.
14. S. Wang, C. H. Tseng, K. Levitt, and M. Bishop. Cost-sensitive intrusion responses for mobile ad hoc networks. *RAID*, 2007).
15. E. Wobber, M. Abadi, M. Burrows, and B. Lampson. Authentication in the Taos Operating System. *ACM Trans. on Computer Systems*, 12, February 1994.
16. A. Yaar, A. Perrig, and D. Song. SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In *In IEEE Symposium on Security and Privacy*, 2004.
17. X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *In Proceedings of ACM SIGCOMM*, pages 241–252, 2005.
18. Y. Zhang and W. Lee. Intrusion detection for wireless ad-hoc networks. *MOBICOM*, 2000.