ODSBR: An On-Demand Secure Byzantine Resilient Routing Protocol for Wireless Ad Hoc Networks

BARUCH AWERBUCH
Johns Hopkins University
REZA CURTMOLA
Johns Hopkins University
DAVID HOLMER
Johns Hopkins University
CRISTINA NITA-ROTARU
Purdue University
and
HERBERT RUBENS
Johns Hopkins University

Ad hoc networks offer increased coverage by using multi-hop communication. This architecture makes services more vulnerable to internal attacks coming from compromised nodes that behave arbitrarily to disrupt the network, also referred to as Byzantine attacks. In this work we examine the impact of several Byzantine attacks performed by individual or colluding attackers.

We propose ODSBR, the first on-demand routing protocol for ad hoc wireless networks that provides resilience to Byzantine attacks caused by individual or colluding nodes. The protocol uses an adaptive probing technique that detects a malicious link after $\log n$ faults have occurred, where n is the length of the path. Problematic links are avoided by using a route discovery mechanism that relies on a new metric that captures adversarial behavior. Our protocol never partitions the network and bounds the amount of damage caused by attackers. We demonstrate through simulations ODSBR's effectiveness in mitigating Byzantine attacks. Our analysis of the impact of these attacks versus the adversary's effort gives insights into their relative strengths, their interaction and their importance when designing multi-hop wireless routing protocols.

1. INTRODUCTION

Ad hoc wireless networks provide several benefits over traditional wireless local area networks (WLANs). They offer increased coverage by using multi-hop communication, i.e. all the nodes take part in the process of forwarding packets. Ad hoc networks do not require a fixed infrastructure such as base stations and routers. Thus, they are self-organizing and can easily be deployed in situations where no fixed infrastructure is present, such as emergency deployments, natural disasters, military battle fields, and rescue missions.

In spite of the above benefits, the wireless communication medium has particularities that create a complex, unpredictable and challenging environment. High error rates, variable and unpredictable characteristics of the signal strength and propagation fluctuations with time and environment frequently result in broken links. The mobility of the nodes and the constrained power resources contribute to the complexity of the environment. In addition, security is more challenging in ad hoc networks because the open wireless medium is more susceptible to attacks and the multi-hop cooperative communication makes services more vulnerable to attacks coming from within the network.

A key component of ad hoc wireless networks is an efficient routing protocol. In the context of ad hoc networks, routing protocols must converge quickly and use battery power efficiently. Thus, traditional proactive routing protocols using periodic updates (link-state and distance vectors [Kurose and Ross 2000]) are less suitable for ad hoc wireless networks because they constantly consume power and bandwidth throughout the network, regardless of the presence of network activity. In addition, they are not designed to track topology changes occurring at a high rate. On-demand routing protocols [Perkins and Royer 2000; Johnson et al. 2001] are more appropriate for wireless environments because they initiate a route discovery process only when data packets need to be routed. Discovered routes are then cached until they go unused for a period of time or break because the network topology changes.

Wireless routing protocols often operate in adversarial environments. Several attacks can come from outsiders or nodes that do not possess the credentials to participate in the protocol. These attacks include eavesdropping and injection or modification of control and data packets and are usually prevented by using encryption, authentication, and integrity mechanisms. Unfortunately, due to their increased susceptibility to theft and software vulnerabilities, wireless devices can be easily compromised and controlled by an adversary. In this case, attacks no longer come from outside the network, but from within the network. Once an adversary has compromised a node, it gains access to the cryptographic material stored on that node, thus rendering useless security mechanisms based exclusively on authentication. Inside attacks are more difficult to address since a compromised device can exhibit arbitrary (malicious) behavior. Such attacks are also known as Byzantine [Lamport et al. 1982] attacks and protocols able to provide service in their presence, are often referred to as Byzantine resilient protocols. Some examples of Byzantine node behavior include: advertising false routing information, trying to redirect routes, or simply droping packets. Wireless networks are also vulnerable to wireless specific attacks such as flood rushing and wormhole attacks, particularly dangerous when they are performed by Byzantine adversaries. When adversaries collude, they can perform stronger attacks, which allow them to control a significant number of the paths in the network.

1.1 Our contribution.

The goal of this work is to provide routing survivability under an adversarial model where any intermediate node or group of colluding nodes perform Byzantine attacks. While some existing work provides protection against specific attacks that may be conducted by a single Byzantine node against different routing components, no other existing work provides an ad hoc wireless routing protocol for coping with a large set of attacks available to a set of colluding Byzantine attackers and targeting both route discovery and data forwarding.

A Byzantine fault occurring along a path may be attributed to a specific node using expensive and complex Byzantine agreement; however, this is provably impossible [Lamport

et al. 1982] under certain circumstances, for example when a majority of the nodes are malicious. We circumvent this obstacle by avoiding the assignment of "guilt" to individual nodes. Instead, whenever the endpoints of a link disagree, we deduce that at least one of them is faulty, therefore the link is considered faulty and should be avoided. Our method ensures that as long as a fault-free path exists between two nodes, they can communicate reliably even if an overwhelming majority of the network acts in a Byzantine manner. We focus on attacks at the network layer and do not consider attacks against the MAC or physical layers. More specifically, our contributions are:

—We present a detailed description of several general (black hole) and wireless-specific (flood rushing, wormhole and overlay network wormhole) Byzantine attacks and analyze their mechanisms and interaction. The identified attacks can not be addressed by using only authentication mechanisms.

—We propose ODSBR, a secure robust on-demand routing protocol which is resilient to strong adversarial attacks, including those mentioned above and performed by colluding Byzantine attackers. Using an adaptive probing technique, our protocol identifies a faulty link after $\log n$ faults have occurred, where n is the length of the path between two endpoints. Because the locations of the attackers are learned, our protocol provides adversary avoidance in arbitrary network configurations. Problematic links are avoided by using a newly proposed metric that captures failures and adversarial behavior.

—We provide an analysis that shows that ODSBR bounds the damage an attacker or group of colluding attackers can cause to the network. Our link monitoring mechanism limits the amount of damage in the case of dynamic adversaries that alternate between good and bad behavior, while never completely disconnecting nodes from the network.

—We developed a protocol independent Byzantine attack module for the NS2 [ns2] simulator in order to simulate the attacks considered in this work. We believe the module is a helpful tool for the secure routing research community.

—We demonstrate through simulations the effects of the considered attacks on the AODV [Perkins and Royer 2000] protocol. Our results quantify the damage caused by the attacks and provide insights into identifying those which result in the greatest network disruption while requiring the least number of adversarial participants. We consider the performance of AODV to be representative of both insecure routing protocols and existent secure routing protocols (such as Ariadne [Hu et al. 2002], SEAD [Hu et al. 2002], ARAN [Sanzgiri et al. 2002], SRP [Papadimitratos and Haas 2002], SDT [Papadimitratos and Haas 2003] and Afora[Lee 2002]). Although some of them [Lee 2002; Hu et al. 2002; Papadimitratos and Haas 2003] address a weaker subset of the Byzantine attacks we address, they do not provide protection against the entire set of strong colluding Byzantine attacks we investigate.

—We show through simulations how ODSBR mitigates the above identified attacks. Analysis of the results gives insights into the survivability of the routing service while under attack and indicates the main factors contributing to the effectiveness of the attacks: flood rushing and strategic adversarial positioning.

The rest of the paper is organized as follows. We describe the network and security models as well as the attacks considered in this work in Section 2. We present our protocol, ODSBR, in Section 3 and provide an analysis in Section 4. We present experimental

results that demonstrate the effectiveness of ODSBR in addressing the considered attacks in Section 5. We overview related work in Section 6 and conclude our paper in Section 7.

2. NETWORK AND SECURITY MODEL

2.1 Network Model

This work relies on a few specific network assumptions. Our protocol requires bi-directional communication on all links. This is also required by most wireless MAC protocols, including 802.11 [IEEE 1999], to operate correctly. We focus on providing a secure routing protocol, which specifically addresses threats to the ISO/OSI network layer. We do not address attacks against lower layers such as the MAC or the physical layer. We assume that the physical layer uses jamming-resilient techniques such as direct sequence spread spectrum (DSSS) or frequency hopping spread spectrum (FHSS) (as in the case of 802.11). Though MAC protocols can detect packet corruption, we do not consider this a substitute for cryptographic integrity checks [Stone and Partridge 2000].

2.2 Security Model and Considered Attacks

We assume that the network is not open. Specifically, there exists a public-key infrastructure administered by a Certificate Authority (CA). A distributed cluster of peer CAs sharing a common certificate and revocation list can be deployed to improve the CA's availability. The public keys are used to protect the route discovery phase and to establish shared key used in other phases of our protocol.

We consider only the source and destination to be trusted, and assume that there exists a non-adversarial path between source and destination. Nodes are authenticated using public key-based techniques in the route discovery phase and symmetric key-based techniques in subsequent phases of the protocol. Messages that can not be authenticated are discarded. Any intermediate node on the path between the source and destination may exhibit Byzantine behavior. The goal of our protocol is to detect Byzantine behavior and avoid it, or bound its effect on the overall system. We assume that an intermediate node can exhibit such behavior either alone or in collusion with other nodes.

We do not address resource consumption attacks where a node receives a high number of messages with incorrectly authenticated packets. This is a general problem with protocols performing authentication by using digital signatures.

We define *Byzantine behavior* as any action by an authenticated node performed at the network layer that results in disruption or degradation of the routing service (i.e. the attacker does not have control over the MAC or physical layers). Attacks like eavesdropping, fabricating or modifying packets can be prevented by traditional encryption, authentication and integrity mechanisms. More complex attacks include manipulating the route discovery phase of the protocol to control the path establishment. Examples of such attacks are making a path appear either longer or shorter than it is. One simple way to do this is by modifying the information that propagates on the packet, such as the hop count or the list of nodes on the path. These attacks can also be prevented by using more sophisticated authentication and integrity techniques.

Unfortunately, there are attacks that can not be prevented by authentication mechanisms alone. An example is a basic Byzantine attack referred to as a *black hole attack* where the adversary drops data packets (entirely or selectively), while still participating in the routing protocol. As a result, whenever an adversarial node is selected on a path, data will be lost

partially or entirely on that path. Another example is the *flood rushing* attack. The attack exploits the flood duplicate suppression technique used by many wireless routing protocols. If an attacker succeeds in rushing an authenticated flood through the network before the flood traveling through a legitimate route, then the legitimate version will be ignored and only the adversarial version will be propagated. This attack may result in establishing many adversarial controlled paths. Authentication techniques can not prevent the attack, since once a node is compromised and under adversarial control, all the cryptographic keys are available to the attacker. Thus, the attacker can generate messages that appear to be authentic.

Besides attacks that can be performed individually by one malicious node, we consider stronger forms of Byzantine attacks that involve colluding malicious nodes which coordinate their actions. Examples of such attacks are the *Byzantine wormhole* attack and its stronger variant the *Byzantine overlay network wormhole*. In a *Byzantine wormhole* attack two colluding adversaries cooperate by tunneling packets between each other in order to create a shortcut (or wormhole) in the network. This tunnel can be created by using a private communication channel, such as a pair of radios and directional antennas, or by using the existing ad hoc network infrastructure. The adversaries can then use the low cost appearance of the wormhole in order to increase the probability of being selected on paths, and then attempt to disrupt the network by selectively dropping the data packets or to perform traffic analysis. Note that for a Byzantine wormhole, the wormhole link exists between two compromised (adversarial) nodes, while in a traditional wormhole two honest nodes are tricked into believing that there exists a direct link between them. Wormhole detection techniques [Hu et al. 2003a; Hu and Evans 2004; Eriksson et al. 2006] proposed against traditional wormholes, are ineffective in the case of Byzantine wormholes due to the trust of the wormhole link end points (which are adversarial).

A *Byzantine overlay network wormhole* attack is a more general (and stronger) variant of the previous attack and occurs when several nodes are compromised and form an overlay network. By tunneling packets through the overlay network, the adversaries make it appear to the routing protocol that they are all neighbors, which considerably increases their chances of being selected on routes and facilitates further attacks.

We do not consider general attacks such as Sybil and node replication attacks. Also, preventing traffic analysis is not the goal of this work, which instead focuses on survivable routing.

3.   ODSBR DESCRIPTION

ODSBR is an on-demand source routing ad hoc wireless routing protocol, designed to cope with a wide class of Byzantine attacks outlined in Section 2.2. The design of ODSBR is centered around the impossibility of distinguishing between failures and malicious behavior. Thus, ODSBR addresses both failures and attacks within an unified framework. A fault is defined as any disruption that results in significant loss or delay. Upon detection of the attack (the number of lost packets becomes higher than a threshold value), ODSBR enters a probing mode with the goal of discovering the attack location. As a result of this probing procedure, the location of the adversary can be narrowed down to a single link (the guilt is assigned to a link, since it is theoretically impossible [Lamport et al. 1982] to indicate a node).

Unlike other protocols, ODSBR does not use number of hops as path selection metric.

Number of hops was shown not to be an appropriate metric in multi-hop wireless networks in non-adversarial environments [Lundgren et al. 2002; De Couto et al. 2003]. Under a Byzantine adversarial model selecting the shortest path will not guarantee that such a path is adversary free. The ODSBR approach to this problem is defining a new metric that captures reliability and adversarial behavior based on past history and selecting the shortest path returned from its route discovery process according to this reliability metric. The metric is represented by a list of link weights where high weights correspond to low reliability. Each node in the network maintains its own list, referred to as a *weight list*, and dynamically updates that list when it detects faults. Faulty links are identified using a secure adaptive probing technique that is embedded in the regular packet stream. These faulty links are avoided using a secure route discovery protocol that incorporates the reliability metric. More specifically, our routing protocol can be separated into three successive phases, each phase using as input the output from the previous:

—*Route discovery in an adversarial environment:* Double flooding, per node flood verification, and forwarding rules guarantee that the route discovery process will always find the lowest cost path according to our reliability metric. However, this path is not guaranteed to be adversarial-free until the weight of adversarial links has been increased sufficiently such that the lowest cost path is fault free.

—*Byzantine fault detection:* This component discovers problematic links on the path from the source to the destination using as input the full path and outputting a faulty link. The source requires cryptographic proof, in the form of secure acknowledgments, that packets are delivered successfully and uncorrupted to the destination. Probes and secure acknowledgments from intermediate nodes locate faults along the path down to the nearest link. Due to the structure of the probing scheme, an adversarial node can only cause a fault to be localized to one of its adjacent links, and does not have the power to arbitrarily incriminate other links in the network. Also, as probes are cryptographically coupled to every data packet, it is impossible to escape detection without delivering the majority of packets correctly (dropping less than an allowable threshold). Although a malicious node can send acknowledgments and still not forward packets, the fault will be detected because other honest nodes, including the destination, will indicate to the source that data was not received.

—*Link weight management:* One goal of our protocol is fault avoidance. This is achieved by the route discovery phase based on weights associated with each link. The link weight management component of the protocol maintains a weight list for links discovered by the fault detection algorithm and uses a multiplicative increase scheme to penalize links. In addition, our protocol uses a rehabilitation mechanism that, on one hand, limits the amount of damage an attacker or group of attackers can cause and, on the other hand, reduces the impact of false positives by never completely disconnecting nodes from the network. Links which have been identified as faulty will be eventually reset back to "non-faulty" as long as enough good traffic has gone through them. This prevents transient failures from becoming permanent black marks. Our link weight management scheme amortizes the number of lost packets caused by adversaries over enough successful packets, and maintains the overall loss rate bounded even in the case of dynamic adversaries that alternate between good and bad behavior.

3.1 Route Discovery

Whenever a source must communicate with a destination node for which it does not have a path already cached, it needs to establish a path. Many on-demand routing protocols achieve this by flooding a route request from the source to the destination and by using a reverse unicast from the destination to the source. The best path is selected based on a metric propagated in the request packet. This metric is either a hop count as in AODV, or the identifiers of the nodes on the path, as in DSR. Thus, the path selection metric does not capture failure or adversarial behavior.

During the route discovery, the following attacks can take place. An attacker can drop either the route request or the route reply. As a result, even if a non-adversarial path exists between the source and destination, no path will be established. In addition, in the absence of authentication and integrity mechanisms, any node can insert packets or modify forwarded packets. Thus, arbitrary – including shorter or longer – paths can be established, or cache poisoning may occur. Finally, an adversary can perform either a flood rushing attack or a wormhole attack that will allow him to obtain control of paths in the network.

Our protocol uses the following mechanisms to address the above identified attacks and limitations of existing metrics used to select paths. The protocol floods both the route request and the response in order to ensure that if a fault free path exists in the network, then a path will be established even in the presence of adversaries.

The initial flood guarantees that the route request reaches the destination. The response flood guarantees that the response will reach back to the source in spite of adversaries dropping packets. Note that the established path may contain adversarial nodes that did not show any malicious behavior yet.

Our protocol protects the route discovery phase from both outside and Byzantine attackers by requiring the source to digitally sign the initial route request flood. This prevents unauthorized nodes from initiating resource consuming route requests, and limits excessive route requests from a Byzantine node. A node verifies the signed route request before forwarding it. Any route request that fails verification is dropped immediately, preventing the request from flooding through the network.

We propose a new metric that captures failures and adversarial behavior. This metric is represented by weights assigned to each link, where low values denote high reliability. The route discovery protocol chooses a route that is a minimum weight path between the source and the destination. This path is found by accumulating the cost hop-by-hop and forwarding the flood only if the new cost is less than the previously forwarded minimum cost. At the completion of the route discovery protocol, the source is provided with a complete path to the destination. In our protocol, the actual path discovery occurs during the route response flood, as opposed to during the route request flood as in other on-demand protocols. This reduces the cost of route requests to unreachable destinations and allows the use of the destination's learned weight list. The accumulated path is protected from outside adversaries by having each node appending its identifier and signing the response. Every node verifies the entire content of the packet so far before forwarding the reply. This ensures that only authenticated nodes can become part of the path. If only the source verifies all of the weights and signatures, then adversaries could propagate low cost fabricated responses and block correct responses from ever reaching the source. Therefore, each intermediate node must verify the weights and the signatures carried by a response, in order to guarantee that a path will be established. Our protocol does not consider caching

because this will prevent the fast avoidance of links with high losses caused by failures or malicious actions.

To mitigate flood rushing attacks, our route discovery protocol processes all duplicate response flood packets and if a valid flood packet with a lower metric is received, an additional re-broadcast is scheduled. The advantage of this technique is that even if an adversary performs a successful "rush" in an attempt to be selected on the path, the adversarial variant of the flood will be shortly overridden by the legitimate flood with a lower path cost. Any adversary that manages to bypass the protection mechanisms and starts creating damage, will be detected by the fault location algorithm and then avoided when a new path is selected.

The following five steps describe in details the route discovery protocol:

I. Request Initiation. The source creates and signs a request that includes the destination, the source, a sequence number, and a list of detected malicious links and their weights. The source then broadcasts this request to its neighbors. The source's signature allows the destination and intermediate nodes to authenticate the request and prevents an adversary from creating a valid route request.

II. Request Propagation. The request propagates to the destination via flooding which is performed by intermediate nodes as follows. When receiving a request, an intermediate node first verifies the source signature on the request and checks its list of recently seen requests for a matching request (one with the same destination, source and request sequence number). If there is no matching request in its list, and the source's signature is valid, it stores the request in its list and rebroadcasts the request. If there is a matching request, the node does nothing.

III. Request Receipt / Response Initiation. Upon receiving a new request from a source for the first time, the destination verifies the authenticity of the request, and creates a response that contains the source, the destination, a response sequence number and the combined link weight list (both the source and the destination weight lists are combined). The destination then signs the entire response and broadcasts it.

IV. Response Propagation. When receiving a response, the node computes the total weight of the path by summing the weight of all the links (carried in the response) on the specified path to this node. If the total weight is less than any previously forwarded matching response (same source, destination and response sequence number), the node verifies the signatures of the response header and of every hop listed on the packet so far. If the entire packet is verified, the node appends its identifier to the end of the packet, signs the appended packet, and broadcasts the modified response. This response propagation functions similarly to the single source Bellman-Ford algorithm.

V. Response Receipt. When the source receives a response, it performs the same verification as the intermediate nodes as described in the response propagation step. If the path in the response is better than the best path received so far, the source updates the route used to send packets to that specific destination.

Note that an adversary can still influence the path selection by creating what we refer to as *virtual links*. Colluding adversaries may create virtual links by forming wormholes, as described in Section 2.2, or any other type of shortcuts in the network. A virtual link can also be created by deleting one or more hops from the end of the route response. ODSBR's approach to mitigating Byzantine wormholes is motivated by the observation that the primary attack when a wormhole exists is the dropping of packets that attempt to

travel through the wormhole, rather than the wormhole formation. A wormhole attack will appear to ODSBR as a faulty link between two nodes. ODSBR mitigates the attack not by preventing the formation of the wormhole, but by detecting it and increasing its weight. Once the wormhole's link weight has been increased sufficiently, ODSBR will avoid it and select the next best alternate path. This strategy does not require any additional hardware or capabilities to function, and it works equally well for both Byzantine and traditional wormholes. We present a detailed analysis of the effect of virtual links in Section 4.

3.2 Byzantine Fault Detection

Once a path is established, data can flow between source and destination. In this case, attacks that may occur are either on the data content (i.e. modification, injection, eavesdropping) which we assume are addressed at the transport or application layer, or on the data delivery (i.e. selective dropping of data). ODSBR provides protection against nodes that selectively drop traffic. Our detection algorithm is based on using authenticated acknowledgments of the data packets. If a valid acknowledgment is not received within a timeout, our protocol assumes that the packet was lost. This definition of loss includes both malicious and non-malicious causes. For example a loss can be caused by packet drop due to buffer overflow, packet corruption due to interference, a malicious attempt to modify the packet contents, or just a drop of the packets by intermediate nodes.

A network operating "normally" exhibits some amount of loss. We define a *threshold* that sets a bound on what is considered a tolerable loss rate. In a well behaved network the loss rate should stay below the threshold. A *fault* is defined as a loss rate greater than or equal to the threshold. The value of the threshold also specifies the amount of loss that an adversary can create without being detected. Hence, its value should be chosen as low as possible, while still greater than the normal loss rate. The threshold value is determined by the source, and may be varied independently for each route.

Our protocol is designed to have minimum overhead under normal conditions: only the destination is required to send an acknowledgment when no faults have occurred. If losses exceed the threshold, the protocol attempts to locate the faulty link by requiring a dynamic set of intermediate nodes, in addition to the destination, to send acknowledgments to the source. Moreover, the fault detection protocol is built to primarily use symmetric key cryptography, so costly asymmetric operations are avoided on a per packet basis.

Normal topology changes occur frequently in ad hoc wireless networks. Although our detection protocol locates faulty links caused by these changes, an optimized mechanism for detecting them would decrease the overhead and detection time. Any of the mechanisms used by the route maintenance of the DSR protocol [Johnson et al. 2001], (i.e. MAC layer notifications), can be used as an optimized topology change detector. When a node receives notifications from such a detector, it reacts by creating a route error message containing the identifiers of the broken link and of the node that reports the error. The signed message is then propagated along the path back to the source. Upon receipt of an authenticated route error message, the source passes the faulty link to the link weight management component. As mentioned above, an intermediate node exhibiting Byzantine behavior can always incriminate one of its links, so adding a mechanism that allows it to explicitly declare one of its links faulty does not weaken the security model.

Note that the detection mechanism can itself be attacked or exploited by an adversary, so it must be designed accordingly. Our protocol embeds probe request information in the data packet stream. This avoids attacks in which malicious nodes drop probe requests
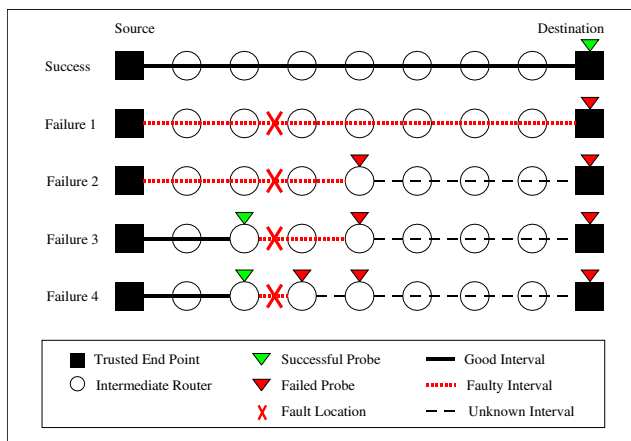
Fig. 1.   Byzantine Fault Detection: an Example

for particular nodes making those nodes appear as being malicious. An intermediate node exhibiting Byzantine behavior can always incriminate one of its links by not sending the acknowledgment, but by doing so it also incriminates itself. Although a malicious node can send acknowledgments and still not forward data packets, the fault will be detected because other honest nodes, including the destination, will indicate to the source that data was not received. Below we provide more details about the detection mechanism and the way it is protected against adversarial behavior.

Fault Detection Mechanism. Our fault detection protocol requires the destination to return an acknowledgment to the source for every received data packet. The source keeps track of the number of recent losses (acknowledgments not received over a window of recently sent packets). If the losses violate the acceptable threshold, the protocol registers a fault between the source and the destination and starts a binary search on the path, in order to identify the faulty link. A simple example is illustrated in Figure 1.

The source controls the search by specifying on data packets the list of intermediate nodes that must send acknowledgments in addition to the destination. We refer to the set of nodes required to send acknowledgments as probed nodes, or for short *probes*. The list of probes is specified on legitimate traffic. Thus, an adversary is unable to drop traffic without also dropping the list of probes and eventually being detected.

The list of probes defines a set of non-overlapping intervals that cover the whole path, where each interval covers the sub-path between the two consecutive probes that form its endpoints. When a fault is detected on an interval, the interval is divided in two by inserting a new probe. This new probe is added to the list of probes appended to future packets. The process of sub-division continues until a fault is detected on an interval that corresponds to a single link. In this case, the link is identified as being faulty and is passed to the link weight management component, which will update the metric associated with the link. The path sub-division process is a binary search that proceeds one step for each fault detected. This results in the detection of a faulty link after $\log n$ faults have occurred, where n is the length of the path.

In Figure 1, in the absence of adversaries, the path between the source and destination

---

seq,data,ID_1,ID_2,ID_3,...,ID_p,$HMAC_{dest}$,$HMAC_p$,...,$HMAC_3$,$HMAC_2$,$HMAC_1$
where $HMAC_i$ is computed with K_{source,ID_i} and p is the number of probes

---

Fig. 2. Probe Specification

contains initially a single good interval. Assume the third node from the source becomes adversarial and starts dropping data packets. If enough data packets are dropped and the loss threshold is reached, then the source will register a fault somewhere on this path and the fault detection mechanism kicks in. The source will start adding probes on the path in order to localize the faulty link: a probe will be added in the middle of the interval (the fourth node) which will fail to respond because the adversarial node drops packets and probes are embedded in data packets. If the adversary stops dropping data packets now, then the protocol will successfully manage to deliver an amount of packets just below the threshold. Otherwise, a probe will be added on the second node, which will send a successful acknowledgement and then another probe will be added on the third node, which will fail. Consequently, the source has managed to identify the faulty link between the second and the third nodes.

The probes are specified by listing the identifiers of the probed nodes in path order on each packet. This list is protected from tampering by appending an HMAC [HMA 2002] (computed with the shared key between the source and that node) of the entire packet so far for every probed node in the reverse order they are specified on the packet (see Figure 2). A node can detect if it is required to send an acknowledgment by checking the list for its identifier. If the node finds its identifier, it verifies the last HMAC and removes it from the packet. A node discards packets that do not have the correct number of remaining HMACs, and a probe discards packets if the last HMAC does not verify. The reverse ordered HMACs prevent the adversary from incriminating other links by successfully tampering with the node list (i.e. removing specific nodes). If an adversary attempts to modify the list, it will incriminate one of its own links.

ODSBR can deal with a mobile adversary along the path, since probes once started are not retired immediately. In the worse case, all intermediate nodes may act as probes. An adversary can always drop packets just below the threshold if it knows the threshold value. This is a problem with all deterministic protocols and can be avoided by using randomization in the threshold selection.

Acknowledgment Specification. For each successfully received data packet, the destination generates an acknowledgment containing the sequence number of the data packet and an HMAC for authentication. Each probe appends its own HMAC over the entire acknowledgement packet accumulated so far, and forwards it along the reverse path towards the source. Timeouts are set at every probe so that if the acknowledgment is not received from the previous probe, the node gives up waiting and generates its own acknowledgment packet. In order for the protocol to locate the fault, only the node immediately preceding the fault location times out and generates an acknowledgment. This is accomplished by using a staggered timeout scheme. A simple technique for implementing the staggered timeout scheme is to use a fixed parameter that represents an upper bound on the amount of time it takes for a packet to traverse a single link. A node may then compute its timeout by multiplying this parameter by the number of hops needed to traverse the remainder of the path to the destination and return back to the current position along the path.

When the source receives the acknowledgment packet, it attempts to verify the accumu-

lated HMACs starting from the end of the packet. The first HMAC, if any, that does not verify, defines the end of the interval on which a loss is registered, i.e. the protocol registers a loss on the interval between the last valid HMAC and the first encountered invalid HMAC. If the source times out the acknowledgment, a loss is registered on the interval between the source and the first probe. The result of this acknowledgment creation, timeout, and verification technique, is that if an adversary drops, modifies, or delays either an acknowledgment or data packet, then a loss will be registered on the interval containing the affected link.

Interval and Probe Management. Let ρ be the acceptable threshold loss rate. By using the above probe and acknowledgment specifications, a loss is attributed to an interval between two probes when the source successfully received and verified an acknowledgment from the closer probe, but did not from the further probe. When the loss rate on an interval exceeds ρ, the interval is divided in two.

Maintaining probes adds overhead to our protocol, so it is desirable to retire probes when they are no longer needed. The mechanism for deciding when to retire probes is based on the loss rate ρ and the number of lost packets. The goal is to bound the aggregate loss rate. Each interval has an associated counter C that specifies its lifetime. Initially, there is one interval with a counter of zero (i.e. there are no losses). When a fault is detected on an interval with a counter C, a new probe is inserted which *divides* the interval. Each of the two new intervals have their counters initialized to $\mu/\rho + C$, where μ is the number of losses that caused the fault. The counters are decremented for every acknowledgment that is successfully received, until they reach zero. When the counters of both intervals on either side of a probe reach zero, the probe is retired, *joining* the two intervals. This results in a loss rate bounded to ρ. If the adversary attempts to create a higher loss rate, the algorithm will be able to identify the faulty link.

Shared Key Establishment. Our detection mechanism makes extensive use of pairwise symmetric keys shared between the source and each node along the path. Pre-distributing and refreshing shared keys would be impractical for network maintainers. We propose a technique for on-demand creation of these keys using the assumed public key infrastructure. When the source needs to probe a node with which it does not already share a key, it generates a new key for that node, and encrypts it with that node's public key. The encrypted key is embedded in the probe list on outgoing data packets. Digital signatures are used to authenticate the packets before the shared key is established. Once the shared key is established, acknowledgments sent by the node can be authenticated using an HMAC.

In order to ensure reliable delivery of the shared key to the intended node even in the case where there are active adversaries disrupting the path, the source continues to attach the encrypted key for every outgoing data packet, until a verifiable acknowledgment (that uses the shared key) is received from the node. Note that since the shared key establishment is seamlessly integrated with the adaptive probing, multiple keys can be established simultaneously even as the protocol dynamically changes the set of probed nodes.

It may be desirable to generate the key shared by the source and destination using a fully authenticated contributory key exchange instead of the above technique. This is because the source-destination key will likely be used to encrypt data, and thus perfect forward secrecy [1] would be desirable. This property is not necessary for intermediate nodes as their

---

[1]Informally, perfect forward secrecy [Menezes et al. 1996] demands that the compromise of long-term keys

shared keys are not used to protect data confidentiality, but only for authentication and integrity checks. An authenticated Diffie-Hellman contributory key exchange between the source and the destination can either be piggy-backed on the route request and response floods, or conducted immediately after the path has been established.

3.3 Link Weight Management

An important aspect of our protocol is its ability to avoid faulty links in the process of route discovery by using a new metric that captures faulty and adversarial behavior. The metric is represented by link weights. The decision to identify a link as faulty is made by the detection component of the protocol. The link management scheme sets the link weights using the history of faults that have been detected on links.

The goal is to penalize faulty links and to reward good behavior, to prevent completely disconnecting nodes from the network and transient failures from becoming permanent marks. In addition, the identification of a link as faulty only increases the weight of that link; As a result, even if all the links in the network were identified as faulty once, the network would continue to operate as before, just with higher "distance" metrics for a given path. Also, the detection of a faulty link along a path uses separate loss tracking for each probe segment, so intentional dropping in the first part of the path will not jeopardize the second part of the path.

The link weight management protocols operates as follows. When a fault is registered on a link, the link's weight is doubled. This ensures that the protocol will eventually avoid selecting paths containing that link during future route discoveries. The technique we use for rehabilitating the link weights is similar to the one we use for retiring probes (see Section 3.2). In addition to the weight, a counter is associated with each identified faulty link. This counter represents the remaining time before the link weight will be reset back to its initial value (non-faulty status). If μ is the number of packets dropped while identifying a faulty link and ρ is the threshold loss rate, then the link's counter is increased by μ/ρ. Each non-zero counter is reduced by $1/m$ for every successfully delivered packet, where m is the number of links with non-zero counters.

4. ANALYSIS

In this section we specify how ODSBR addresses the Byzantine attacks identified in Section 2, focusing on the black hole, flood rushing, Byzantine wormhole and overlay of Byzantine wormholes attack.

4.1 Black Hole

The ODSBR protocol uses end-to-end acknowledgments from the destination to detect the presence of a black hole attack. Upon detection of the attack (the number of lost packets becomes higher than a threshold value), ODSBR enters a probing mode with the goal of discovering the attack location. As a result of this probing procedure, the location of the adversary can be narrowed down to a link (the guilt is assigned to a link, since it is theoretically impossible [Lamport et al. 1982] to indicate a node). The weight of a blamed link is doubled, which ensures that the protocol will avoid selecting paths containing that link during future route discoveries. As a result, if there exists an adversarial-free path to the destination, ODSBR will eventually find it within a bounded amount of packet loss.

---

should not lead to the compromise of any previously used session keys.

In addition, ODSBR can deal with a mobile adversary along the path, since probes once started are not retired immediately. In the worse case, all nodes may act as probes.

An adversary can drop packets just below the threshold in an attempt to avoid detection. However in this case the protocol has successfully "scared" the adversary into predominantly behaving correctly. In addition, randomized threshold selection can help reduce the effectiveness of this adversarial strategy. One of the main advantages of ODSBR's approach to mitigating black holes is that the locations of the attackers are learned, thus enabling adversary avoidance in arbitrary network configurations.

4.2 Flood Rushing

The route discovery phase of the ODSBR protocol has several features which help mitigate the effects of flood rushing. The protocol performs hop-by-hop authentication and integrity checking of route discovery flood packets. This prevents an invalid variant of the flood from propagating through the network. Using only end-to-end authentication (source and destination only) will not prevent an invalid variant from propagating and blocking the valid flood.

In addition, ODSBR processes all duplicate response flood packets and if a valid flood packet with a lower metric is received, an additional re-broadcast is scheduled. The advantage of this technique is that even if an adversary performs a successful "rush" in an attempt to be selected on the path, the adversarial variant of the flood will be shortly overridden by the legitimate flood with a lower path cost. The method will increase the protocol overhead because nodes affected by the rushing adversary need to re-broadcast the flood packet more than once. Finally, any adversary that manages to bypass these two protection mechanisms and starts creating damage will be detected by the fault location algorithm and then avoided when a new path is selected.

4.3 Byzantine Wormhole

ODSBR's approach to mitigating Byzantine wormholes is motivated by the observation that the primary attack when a wormhole exists is the dropping of packets that attempt to travel through the wormhole, rather than the wormhole formation. A wormhole attack will appear to ODSBR as a faulty link existing between two nodes. ODSBR mitigates the attack not by preventing the formation of the wormhole, but by detecting it and increasing its weight. Once the wormhole's link weight has been increased sufficiently, ODSBR will avoid it and select the next best alternate path. This strategy does not require any additional hardware or capabilities to function, and it works equally well for both Byzantine and traditional wormholes. The number of packets lost and the amount of time taken to find an adversarial-free path, will be proportional to the number of wormhole links that create paths shorter than the legitimate route. ODSBR's ability to mitigate the wormhole attack will be reduced if many wormhole links are present.

4.4 Byzantine Overlay Network of Wormholes

The convergence of ODSBR is slowed if many adversaries exist in the network and cooperate to create an overlay of Byzantine wormholes. However, the amount of damage that the attackers can create is bounded. The number of packets lost and the amount of time taken to find an adversarial-free path, will be proportional to the number of wormhole links that create paths shorter than the legitimate route. As a result, ODSBR's ability to mitigate the wormhole attack will be reduced if many wormhole links are present. We provide an upper

bound on the number of packets lost because of adversarial behavior, namely behavior that reduces the packet transmission success rate below some threshold.

Let q^- and q^+ be the total number of lost packets and successfully transmitted packets, respectively. Ideally, $q^- - \rho \cdot q^+ \leq 0$, where ρ is the transmission success rate, slightly higher than the original threshold. This means the number of lost packets is a ρ-fraction of the number of transmitted packets. While this is not quite true, it is true "up to an additive constant", i.e. ignoring a bounded number ϕ of packets lost. Specifically, we prove that there exists an upper bound ϕ such that:

$$q^- - \rho \cdot q^+ \leq \phi \tag{1}$$

Assume that there are N nodes, k of which are adversarial, $k < N$. We denote by $\tilde{E}$ the set of links controlled by adversarial nodes. The maximum size of $\tilde{E}$ is kN.

Consider a faulty link e, convicted j_e times and rehabilitated a_e times. Then, its weight, w_e, is at most n, where n is the upper bound of the length of a non-faulty path in the network. If a link reaches the weight of n, it is effectively priced out of the network as it is more expensive than any possible non-faulty path. By the algorithm, w_e is given by the formula:

$$w_e = 2^{j_e - a_e} \tag{2}$$

The number of convictions is at least q^-/μ, so

$$\frac{q^-}{\mu} - \sum_{e \in \tilde{E}} j_e < 0 \tag{3}$$

Also, the number of rehabilitation operations is at most $\frac{q^+}{\mu/\rho}$, so

$$\sum_{e \in \tilde{E}} a_e - \frac{q^+}{\mu/\rho} < 0, \tag{4}$$

where μ is the number of lost packets that exposes a link as faulty. Thus

$$\frac{q^-}{\mu} - \frac{q^+}{\mu/\rho} \leq \sum_{e \in \tilde{E}} (j_e - a_e) \tag{5}$$

From Eq. (2) we have $j_e - a_e = \log w_e$. Therefore:

$$\sum_{e \in \tilde{E}} (j_e - a_e) = \sum_{e \in \tilde{E}} \log w_e \tag{6}$$

By combining Eq. (5) and (6), we obtain

$$q^- - \rho \cdot q^+ \leq \mu \sum_{e \in \tilde{E}} \log w_e \leq \mu \cdot kN \cdot \log n \tag{7}$$

and since $\mu = b \log n$, where b is the number of lost packets per window, Eq. (5) becomes

$$q^- - \rho \cdot q^+ \leq b \cdot kN \cdot \log^2 n \tag{8}$$

Therefore, the amount of disruption a dynamic adversary can cause to the network is bounded. Note that kN represents the number of links controlled by an adversary. If there are no adversarial nodes Eq. (8) becomes the ideal case where $q^- - \rho \cdot q^+ \leq 0$.

5.   EXPERIMENTAL RESULTS

In this section we show how AODV [Perkins and Royer 2000], a well-known routing pro-tocol for ad hoc wireless networks, reacts under several Byzantine attack scenarios. In addition, we conducted simulations of the same attacks against ODSBR in order to show its effectiveness in mitigating the attacks. Our experiments seek to identify attacks which result in the greatest network disruption while requiring the least number of adversarial participants. Although a number of secure ad hoc routing protocols exist, out of which some [Lee 2002; Hu et al. 2002; Papadimitratos and Haas 2003] address a weaker subset of the Byzantine attacks we address, they do not provide protection against the entire set of strong colluding Byzantine attacks we investigate.

Each data point in the figures of this section is averaged over 30 different random en-vironments and over all destination nodes. AODV and ODSBR are simulated in the same set of random environments in order to generate paired statistics (a standard method of sta-tistical variance reduction). A paired T-test analysis of all our data shows that the largest p-value for any set is .0083. Therefore, the observed performance differences between AODV and ODSBR are statistically significant with a confidence of over 99%.

5.1   ODSBR Implementation

We implemented our protocol using the NS2 [ns2] network simulator. We assumed the protocol uses RSA [Rivest et al. 1978; DSS 2006] with 1024-bit keys for digital signatures operations, AES [AES 2001] with 128-bit keys for symmetric encryptions and HMAC [HMA 2002] with SHA1 as the message authentication code. The impact of these cryp-tographic operations is represented by adjusting the packet size as if the packet actually contained authenticating data (e.g. digital signatures or MACs, and by introducing packet delay accordingly as if CPU time was spent performing cryptographic operations.

For practical reasons, we implemented a less complex fault detection phase. Instead of the binary search probing technique, we use only two states: a "non-probing" state where only the destination returns acknowledgments, and a "probing" state where all intermedi-ate nodes also return acknowledgments. The protocol operates in the non-probing state until a loss threshold violation occurs and a fault is detected. If in the probing state, the source node successfully delivers enough packets and the loss rate goes below a specified threshold, then the source node returns to the non-probing state. Preliminary experiments we conducted showed that when the total number of hops is relatively small, the cost of enabling all the probes at once is low. In this case the two-state technique reduces the amount of time necessary to identify a link and considerably simplifies the protocol imple-mentation. However, for large networks, probing all intermediate nodes on a path will be more expensive and the general binary search probing technique should be used.

The performance of the implementation is influenced by the values of several parame-ters: the loss threshold rate, the timeout allowed for a packet to traverse a link and the size of the sliding window necessary to keep track of the packet loss history. After conducting a series of experiments with different sets of parameters, the values were chosen as fol-lows: loss threshold rate – 10%, link timeout – 250 milliseconds and sliding window size – 100 packets. We tuned these parameters conservatively in order to ensure that the protocol will operate in a wide range of environments. Although the simulations in this work were conducted with 50 nodes, these values were tuned for efficient operation with up to 100 nodes.
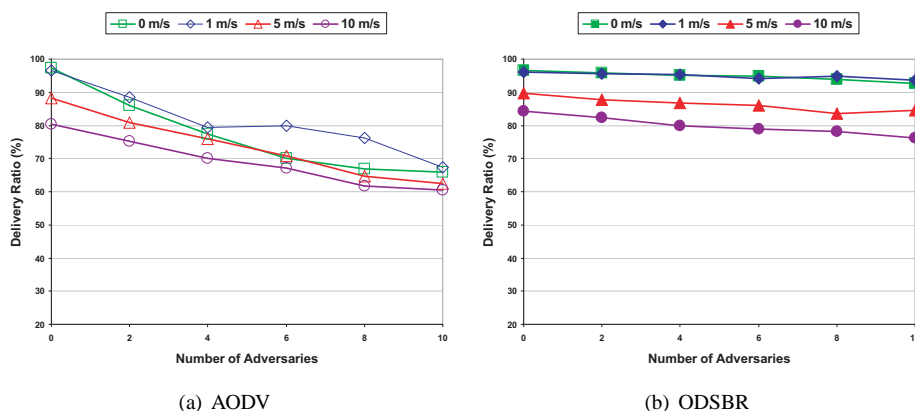
(a) AODV   (b) ODSBR

Fig. 3.   Black Hole Attack: Random Placement Configurations

5.2   Simulation Setup

We performed simulations using the NS2[ns2] network simulator. Nodes in the network were configured to use 802.11 radios with a bandwidth of 2 Mbps and a nominal range of 250 meters. Fifty nodes were randomly placed within a 1000 by 1000 meter square area. We varied the nodal speed between 0 and 10 meters/second. In addition to these 50 nodes, up to 10 adversarial nodes were added to the simulations, depending on the considered attack configuration. A traffic load of 10 constant bit rate (CBR) flows was used to simulate data communication through the ad hoc network. An aggregate load of 0.1 Mbps was offered to the network by having each flow send 256 byte packets at approximately 4.9 packets per second. The simulation time was 300 seconds for each simulation and the results were averaged over 30 random seeds.

We used a modified random way-point mobility model that addresses the concerns raised in [Yoon et al. 2003] about the validity of the standard random way-point model. Nodes select a speed uniformly distributed between 10% and 90% of the given "max" speed to achieve more steady mobility and ensure that the average speed does not drop drastically over the course of the simulation. In addition, 300 virtual seconds of mobility are generated before the start of the simulation such that when the simulation starts, nodes are already in motion. This allows the average speed and node distribution to stabilize before the simulation starts.

In order to simulate the considered Byzantine attacks, we developed a protocol-independent Byzantine attack simulation module for NS2. This module provides the capability to simulate the black hole, Byzantine wormhole, and Byzantine overlay network wormhole attacks without modifying the routing protocol.

5.3   Black Hole Attack

We simulate a black hole attack by dropping any data packet sent by the routing agent. Routing protocol control packets are unaffected. On a real device, depending on the routing protocol implementation, performing a black hole attack may be as simple as deactivating IP forwarding.

We evaluate the delivery ratio by using as a baseline the case where no black holes exist in the network. We then increase the number of adversarial nodes, randomly placed in the
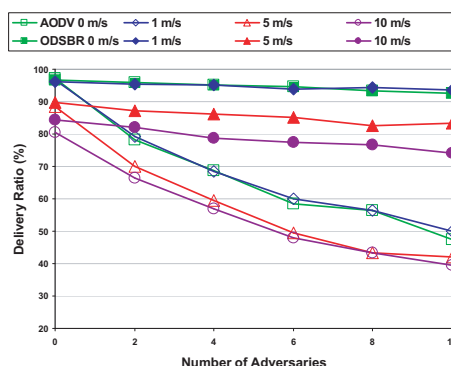
Fig. 4.   Flood Rushing Attack Combined with Black Hole Attack

network, and evaluate the effect this increase has on the delivery ratio. Figure 3 shows the delivery ratio of the AODV and ODSBR protocols as a function of the number of adversarial nodes, for different levels of mobility. We note that the delivery ratio of AODV does decrease as the number of adversaries increases, but a large number of adversarial nodes is required in order to cause a significant network disruption. For example, approximately 10 adversarial nodes are required to drop the delivery ratio of AODV below 70%. This happens because there is no effort by the adversary to get itself selected on many paths and although a number of compromised nodes exist in the network, there is no coordination between them when performing the attack. We conclude that AODV can sustain attacks consisting of a small number of uncoordinated black holes.

ODSBR remains basically unaffected by attacks at low mobility, maintaining a delivery ratio of about 95%, even in the presence of 10 adversarial nodes. At higher mobility, we see a slight decrease (about 8%, from 84% to 76%) in the delivery ratio because node mobility causes some paths to be broken, and some packets will be lost before ODSBR reacts and readjusts the path.

5.4   Flood Rushing Attack

The total network damage that can be caused by a black hole attack is directly related to the likelihood of an adversary being selected as part of the routing paths in the network. Therefore, any additional attack that helps a node to be selected on many paths can increase the impact of a regular black hole attack. In general, these type of attacks occur during the route discovery protocol. The flood rushing attack is an example of such an attack. We performed simulations examining the impact of black hole attacks when combined with flood rushing.

We simulated flood rushing attacks as follows. During the propagation of a normal flood packet, each node waits a small randomized delay before re-transmitting it. These delays are designed to reduce the number of collisions and in some protocols to help ensure that the shortest paths are selected. Eliminating the extra delay is the simplest mechanism available to provide an adversary a time advantage over the normal flood.

Figure 4 shows the delivery ratio of AODV and ODSBR as a function of the number of adversarial nodes randomly placed within the simulation area, at different mobility values. It can be observed from Figures 3(a) and 4 that the delivery ratio for AODV is significantly

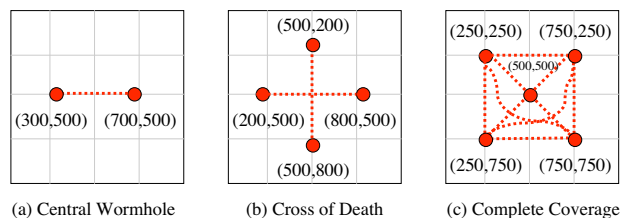(a) Central Wormhole   (b) Cross of Death   (c) Complete Coverage

Fig. 5. Wormhole Network Configurations: adversarial location shown in coordinates (x,y)

affected by flood rushing. The intensity of the attack is due to the greater number of paths that an adversary succeeds in getting selected on. No coordination exists between the attackers. The attack is very effective and lowers AODV's delivery ratio to about 40% when 10 adversaries are present in the network (as opposed to about 70% when no flood rushing was present).

The impact of flood rushing on ODSBR is almost unnoticeable. At low mobility, ODSBR delivers over 90% of the packets (as opposed to 95% when no flood rushing was present), even in the presence of 10 adversaries. This indicates that the technique used by OSDBR to process lower path cost floods instead of discarding them is effective against flood rushing.

5.5 Byzantine Wormhole Attacks

In the previously examined attacks, malicious nodes do not coordinate their actions. A coordinated attack can be much stronger, particularly if adversaries have knowledge of the network topology and/or traffic patterns. This can allow them to select strategic locations that can increase the effectiveness of an attack. For example, an adversary may locate itself in the vicinity of a specific target, or between two nodes that communicate frequently, or position itself such that it can hear all the communication on the network. A Byzantine wormhole attack, or simply a *wormhole*, is an example of an attack that requires coordination between two attackers. A wormhole can be used either to increase the effectiveness of a black hole directly, or as an effective tool in conducting flood rushing attacks, by allowing an adversary to jump several hops ahead of the legitimate flood through the wormhole. In addition, the placement of the wormhole in the network can increase the number of adversarial controlled paths.

Below we examine coordination among attackers and placement of attackers as possible strategies for increasing the effectiveness of an attack. We simulated the most effective wormhole attack by assuming that communication through the wormhole tunnel has no latency and has unlimited bandwidth. Several wormholes are placed in the network, but no coordination exists between them (coordinated wormholes are studied in Section 5.6). We examine the effect of wormhole placement by considering three configurations which we refer to as *random placement*, *central wormhole* and *cross of death* (Figure 5). In all cases, we evaluated the impact of the wormhole attack both by itself, and when combined with flood rushing.

Random Placement. The first configuration we consider is a set of wormholes randomly placed in the network. Figure 6(a) presents results for AODV and ODSBR in the presence of the wormhole attack alone, while Figure 6(b) presents results for the wormhole attack combined with flood rushing. When compared to the black hole attack with randomly placed adversaries (Figures 3 and 4), the same number of adversaries placed randomly, but now forming wormholes, can mount a more effective attack against AODV. This result

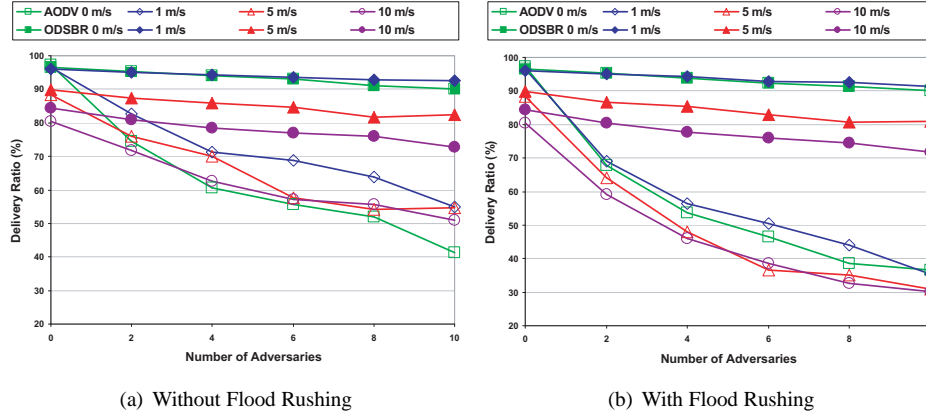(a) Without Flood Rushing   (b) With Flood Rushing

Fig. 6.   Wormhole Attack: Random Placement Configurations

is due to the fact that by using wormhole tunneling, the adversaries are selected as part of more routes and are thus able to drop more traffic and create more damage. When combined with flood rushing, the delivery ratio for AODV goes as low as 30% to 40%, depending on the mobility of the nodes.

In the case of ODSBR, the presence of wormholes has very little impact when compared with the black hole attack conducted by a number of attackers randomly distributed in the network (see Figures 3(b) and 6(a)). This indicates that ODSBR uses an effective mechanism in dealing with mobile adversaries that coordinate to create wormholes. It can be observed that adding flood rushing to increase the effectiveness of the attack does not make a difference for ODSBR, as expected from previous results (see Figures 6(a) and 6(b)).

Central Wormhole. Pictured in Figure 5(a), this configuration contains only two adversaries placed at coordinates (300,500) and (700,500) in the 1000 x 1000 m^2 area considered for our simulations. Since the nominal range is 250 m, this placement gives the wormhole a good coverage of the communication in the network.

The results presented in Figure 7(a) show the delivery ratio as a function of the mobility of the nodes, for AODV and ODSBR. In addition, the normal delivery ratios in the case of no adversaries are shown for reference. Although only one wormhole is present, the attack is considerably more effective than the black hole attack (see also Figures 3 and 4). For example, when flood rushing is enabled and two attackers coordinate to form a *central wormhole*, AODV's delivery ratio can drop as low as 41%, which is similar in strength to 10 randomly placed adversaries performing the black hole attack. This indicates that strategic positioning plays a significant role in the impact of an attack. The attacker needs to compromise only 2 nodes and then coordinate the attack.

For ODSBR, the wormhole at the specified location has a small effect, dropping the delivery ratio from about 80% in the case of 10 randomly placed adversaries, to about 70% in the case of the central wormhole, when nodes have a mobility of 10 m/s in both cases. This indicates that the placement of the wormhole did not allow it to control all of the paths, so adversarial-free paths existed in the network. Since there was only one wormhole, ODSBR found it and used alternative paths to successfully perform data forwarding.

Cross of Death. As seen in Figure 5(b), this configuration contains four adversaries

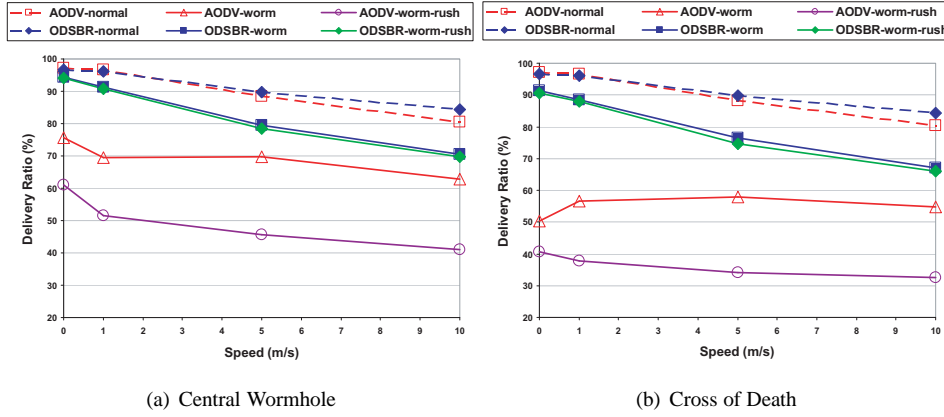(a) Central Wormhole   (b) Cross of Death

Fig. 7.   Wormhole Attack: Strategic Positioning Configurations

placed at coordinates (200,500), (800,500), (500,200), (500,800). They form two wormholes, in the shape of a cross. There is no coordination between the two wormholes.

The results presented in Figure 7(b) show the delivery ratio as a function of the mobility of the nodes, for AODV and ODSBR. As we expected, this is a more effective attack against AODV than the *central wormhole* attack, since the adversarial nodes are covering a larger area and are able to draw in (and drop) more traffic.

For ODSBR, the addition of one more wormhole was not problematic. Although each of the two wormholes had a good coverage, because of the lack of coordination among the four attackers, adversarial-free paths still exist in the network, so ODSBR manages to find them and use them as alternate paths.

Figures 6, 7(a) and 7(b) allow us to analyze the number of randomly placed adversaries required to inflict the same amount of damage as a strategically placed attack. It can be noted that for AODV, with mobility > 0 m/s, the *central wormhole* configuration inflicts slightly more damage than 4 randomly placed adversaries (2 random wormholes) and the *cross of death* inflicts slightly more damage than 8 such adversaries (4 random wormholes). For ODSBR, both the *central wormhole* and the *cross of death* cause more damage than 10 randomly placed adversarial nodes (5 wormholes). This indicates that the wormhole attack is more effective if the adversaries are strategically placed, rather than randomly placed.

5.6   Byzantine Overlay Network Wormhole Attack

In Section 5.5 we analyzed the case where the wormholes were just point-to-point tunnels between two adversaries. While this attack is strong and effective, an even stronger variant exists, when the attackers also coordinate the wormholes. More specifically, the attacker compromises a number of nodes and organizes them in an overlay network wormhole, or a *super-wormhole*. In a super-wormhole attack with n adversaries there exist essentially n^2 point-to-point tunnels between the adversaries.

In the following set of simulations a static wormhole configuration is placed within the network. We investigated three configurations which we refer to as *random placement*, *cross of death*, and *complete coverage* (see Figure 5). In all cases, we first evaluate the effect of the super-wormhole attack on the delivery ratio. We then combine the super-wormhole with flood rushing and examine the impact of the combined attack. We assume
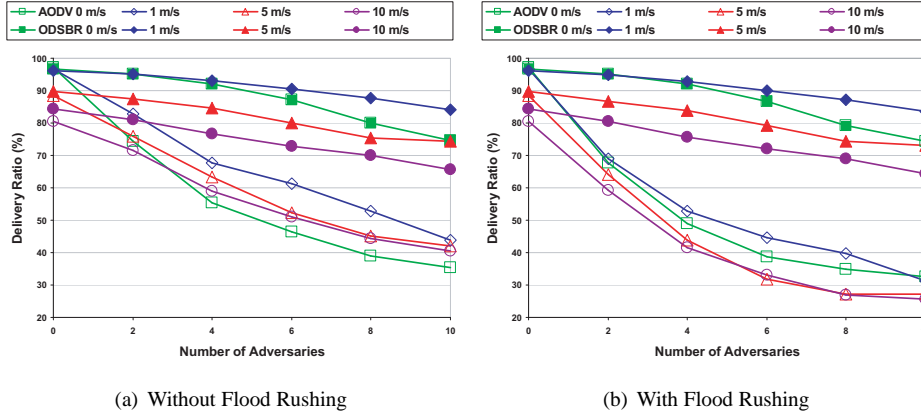
(a) Without Flood Rushing

(b) With Flood Rushing

Fig. 8.   Super-Wormhole Attack: Random Placement Configurations
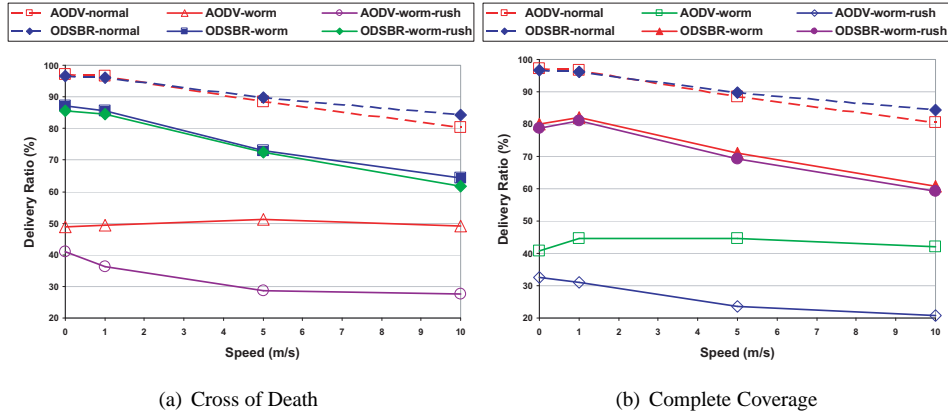

(a) Cross of Death

(b) Complete Coverage

Fig. 9.   Super-Wormhole Attack: Strategic Positioning Configurations

that communication through the super-wormhole tunnels is instantaneous.

Random Placement. In this configuration a set of up to 10 adversarial nodes are randomly placed in the network and form a super-wormhole. Figure 8 presents results for AODV and ODSBR for the super-wormhole attack, with and without flood rushing. In this case, both for AODV and ODSBR, the super-wormhole attack is more effective than the regular wormhole, though not by much. This leads us to believe that a super-wormhole created by randomly placed adversaries gives them little advantage over the case when the same number of adversaries create regular 1-to-1 wormholes.

Cross of Death. The same configuration as the *cross of death* in Section 5.5 was used, but with all four adversarial nodes connected in a super-wormhole configuration. The results presented in Figure 9(a) show the delivery ratio as a function of the mobility of the nodes, for AODV and ODSBR, both with and without flood rushing. In addition, the normal delivery ratios in the case of no adversaries are shown for reference.

Figures 9(a) and 8 help us determine the number of randomly placed adversaries required to inflict the same amount of damage as a strategically placed attack. We conclude that for AODV, with mobility > 0 m/s, the *cross of death* configuration inflicts slightly less

damage than a super-wormhole created by 8 randomly placed adversaries. For ODSBR, if mobility > 0 m/s, the *cross of death* causes about the same damage as a super-wormhole created by 9 randomly placed adversaries if flood rushing is not used, or 10 adversaries if flood rushing is enabled. Observe that the attack is slightly more effective than the *cross of death* with regular wormholes Figure 7(b)). This is because the additional tunnels created in the super-wormhole scenario are of limited strategic value in comparison to the primary tunnels.

Complete Coverage. The strength of the super-wormhole attack can be increased significantly if the adversaries are able to position themselves throughout the network such that they can hear any transmission that takes place in the network. We simulated the configuration shown in Figure 5(c), with five adversarial nodes placed at coordinates (250,250), (250,750), (500,500), (750,250), (750,750).

Observe the devastating effect of this attack in Figure 9(b). When combined with flood rushing, the delivery ratio of AODV drops as low as 20% in the presence of only five adversaries, while ODSBR still delivers 60% of the packets. Since the five adversarial nodes almost completely cover the entire ad hoc network, adding more adversaries will not significantly increase the effectiveness of the attack. A set of only five colluding adversaries strategically placed and launching a coordinated attack practically paralyze the considered ad hoc network when an insecure routing protocol is used. It may seem that a super-wormhole attack is not feasible in practice because it may require a large number of point-to-point tunnels established between the adversaries. However, our simulations show that only five adversaries can cause a major disruption in a network of 50 nodes, making this attack more practical and easier to mount.

5.7 Protocol Overhead

We conducted simulations to compare the overhead of ODSBR with that of AODV, in order to evaluate the cost of security. ODSBR's overhead has two components: (a) the route discovery overhead, and (b) the acknowledgement overhead (ODSBR requires a protocol acknowledgement for each successfully delivered data packet). The acknowledgement overhead is the price our protocol pays in order to offer Byzantine survivability. In real implementations, this overhead can be reduced by piggy-backing ODSBR acknowledgments on TCP acknowledgments. We emphasize that in our ns2 implementation and in all our simulations ODSBR acknowledgements are sent explicitly; Thus the shown delivery ratios fully take into account the effect of the acknowledgement overhead.

Figure 10(a) illustrates the route discovery overhead in a non-adversarial scenario. At all simulated levels of mobility, ODSBR transmits more routing packets per second than AODV. This is because ODSBR floods both the route request and the route reply, while AODV floods only the route request. ODSBR requires bidirectional flooding to guarantee route establishment in the presence of Byzantine adversaries. If the route reply was unicast, then an adversary on the reverse path could forward the request but drop the reply, thus preventing a route from being established, although a correct path existed in the network.

Next we present the route discovery overhead while under attack. Figure 10(b) depicts the overhead of the routing protocols as a function of the number of adversaries, when the adversaries execute a black hole (denoted with AODV-BH and ODSBR-BH) and a super-wormhole attack (denoted with AODV-SW and ODSBR-SW). In this scenario the nodes are under random way-point mobility with a maximum speed of 1 m/s. Observe that the routing overhead of ODSBR increases with the number of adversaries. This occurs

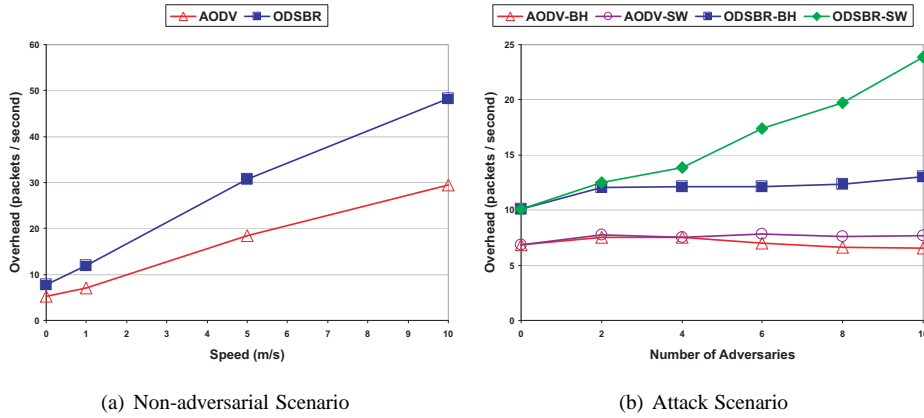(a) Non-adversarial Scenario   (b) Attack Scenario

Fig. 10.   ODSBR Route Discovery Overhead

as a result of the protocol actively detecting faults and readjusting the path to avoid them. The overhead of ODSBR increases proportionally to the number of faulty links in the network. Since a super-wormhole attack results in a larger number of faulty links than a black hole, we see a considerable difference in the routing overhead of ODSBR when detecting a super-wormhole. On the contrary, the overhead of AODV decreases slightly as the number of adversaries increases. Since the adversaries forward the AODV control packets successfully and only drop data packets, AODV is unable to detect that a fault is occurring. This results in a massive reduction in AODV's delivery ratio, while no additional routing messages are generated.

5.8   Discussion

In this section we provide a comparison of the simulation results previously presented, in order to determine the relative strength of the Byzantine attacks (see Figure 11). To evaluate the effects of these attacks in a mobile ad hoc network, we selected scenarios where the mobility of the nodes was 1 m/s. This value was chosen in order to better isolate the damage caused specifically by the Byzantine attacks as opposed to losses due to node mobility. Analysis of these results indicates that two main factors contribute to the effectiveness of the attacks at disrupting the AODV routing protocol: flood rushing and strategic adversarial positioning.

Flood Rushing. In Figure 11(a), the line labeled "Black Hole Rushing" shows the results of a random placement black hole attack with flood rushing enabled. Observe that by enabling flood rushing, this attack resulted in a much greater reduction in the delivery ratio as compared to the same attack without flood rushing. In addition, the flood rushing made this attack strong enough that it caused more damage than the random wormhole attack and comparable damage to the random super-wormhole attack. The black hole attack (a non-colluding attack and easier to execute), combined with flood rushing can create more damage than the wormhole attack (a colluding attack and harder to mount). This motivates the need to design routing protocols which are able to mitigate the flood rushing attack.

Strategic Positioning. The results indicate that the strength of the attacks can be significantly increased if the adversaries are strategically positioned. The point labeled "Complete Coverage" in Figure 11(a) illustrates the effectiveness of strategic positioning. This
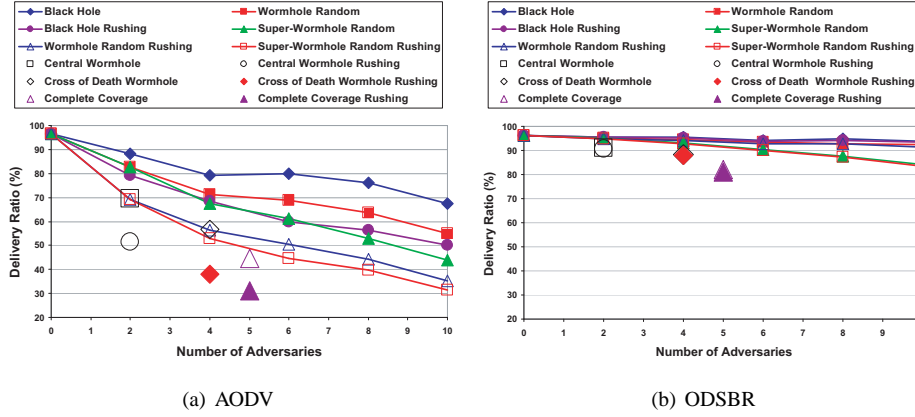
(a) AODV

(b) ODSBR

Fig. 11.   Attacks Comparison

is the result of a super-wormhole with adversaries arranged in a dominating set configuration. By being strategically placed, five adversaries are able to reduce the delivery ratio of AODV to just 45%, without using flood rushing. In comparison, six randomly placed adversaries executing a super-wormhole attack, are only capable of reducing the delivery ratio of AODV to 61%. This demonstrates the power of strategic positioning in crippling the performance of the AODV routing protocol.

When used together, flood rushing and strategic positioning can cause substantial damage to the routing protocol. To quantify the most effective attack, we define the relative strength of a particular attack configuration σ as:

$$\sigma = \frac{DR_{norm} - DR_{adv}}{DR_{norm} \cdot Num_{adv}} \tag{9}$$

where DR_{norm} and DR_{adv} are the delivery ratios in the absence and in the presence of adversaries respectively, and Num_{adv} is the number of adversaries. Intuitively, σ represents the amount of damage an attack can cause per adversary. The higher σ is, the greater the relative strength of the considered attack, since this indicates that a larger amount of damage can be inflicted by a smaller number of adversaries.

Note that in the "Complete Coverage Rushing" case the delivery ratio drops to 30%, while $\sigma = 13.6$. Although this point corresponds to an attack that results in the greatest reduction of AODV's delivery ratio, it is not the most effective attack from the adversary's perspective because five nodes need to be compromised. Alternatively, we can consider the point referred to as "Central Wormhole Rushing" in Figure 11. This attack is able to lower AODV's delivery ratio from 96.6% to 51.4%, while requiring only two colluding adversaries, thus $\sigma = 23.4$. In fact, this is the highest σ observed out of all the considered attacks. This colluding attack executed by only two adversaries combines both flood rushing and strategic positioning, inflicting the highest amount of damage with the least number of adversaries.

The delivery ratio in all the shown experimental results is averaged over several different environments and over all destination nodes. Although the overall average delivery ratio for AODV did not fall to 0 in any of the shown results, in some of the simulations and for some of the source-destination pairs, AODV's throughput may have been reduced to 0.

Figure 11(b) presents a summary of the 1 m/s simulation results for the ODSBR proto-

col. The first observation is that at this level of mobility, the ODSBR protocol was able to successfully deliver over 80% of the packets under all simulated attack scenarios. This validates the protocol's overall strategy for operation in a Byzantine environment. In particular, the results show that ODSBR is resilient against flood rushing attacks which we have shown are devastating to other existing on-demand protocols.

6. RELATED WORK

A key component of providing security services in an ad hoc wireless network is an effective public key infrastructure. Research results in this direction are as follows. Hubaux et al. [Hubaux et al. 2001] proposed a completely decentralized public-key distribution system similar to PGP [Zimmermann 1995]. Zhou and Haas [Zhou and Haas 1999] explored threshold cryptography methods in a wireless environment. Brown et al. [Brown et al. 2000] showed how PGP, enhanced by employing elliptic curve cryptography, is a viable option for wireless constrained devices.

Many routing protocols for MANETs [Perkins and Royer 2000; Johnson et al. 2001] use number of hops as metrics. Recently, number of hops was shown [Lundgren et al. 2002; De Couto et al. 2003] not to be a good metric for applications that focus on throughput and better metrics were proposed in [De Couto et al. 2003] and [Awerbuch et al. 2005]. However, both metrics do not take into consideration adversarial behavior.

Many vulnerabilities in network protocols are caused by the lack of message integrity and authentication mechanisms, which allows an attacker to alter or fabricate packets. Significant research in securing wired [Hauser et al. 1997; Smith et al. 1997; Kent et al. 2000] or ad hoc wireless [Hu et al. 2002; Hu et al. 2002; Sanzgiri et al. 2002; Papadimitratos and Haas 2002] routing protocols focused on this aspect. Papadimitratos and Haas showed in [Papadimitratos and Haas 2002] how impersonation and replay attacks can be prevented for on-demand routing by disabling route caching and providing end-to-end authentication using an HMAC [HMA 2002] primitive which relies on the existence of security associations between sources and destinations. Other significant works include SEAD [Hu et al. 2002] and Ariadne [Hu et al. 2002] that provide efficient secure solutions for the DSDV [Perkins and Bhagwat 1994] and DSR [Johnson et al. 2001] routing protocols. SEAD uses one-way hash chains to provide authentication, while Ariadne uses Tesla [Perrig et al. 2001] source authentication technique to achieve similar security goals. In [Sanzgiri et al. 2002] the authors focus on an analogous problem, providing end-to-end authentication for two well-known on-demand protocols: AODV [Perkins and Royer 2000] and DSR [Johnson et al. 2001], by using digital signatures for authentication. They also provide a protocol that guarantees minimum path selection using an onion-like encryption [Syverson et al. 1997] technique, where digital signatures and public cryptography encryption/decryption are performed and accumulated at each hop.

The problem of insider threats in routing protocols was less studied. In her seminal work [Perlman 1988] and more recently [Perlman 2005], Perlman defined the problem for link-state routing protocols and proposed the Network-layer Protocol with Byzantine Robustness (NPBR). A simplified version of the problem, namely nodes that simply drop data, was considered in [Cheung and Levitt 1997] and [Bradley et al. 1998] which use probing and local monitoring to address the problem. However, many details such as disguising probing packets from the adversary or dealing with colluding attackers are not discussed. Very recent work considered the problem for intra-domain protocols such as

BGP [Mizrak et al. 2005].

In the context of wireless networks relevant works are Watchdog [Marti et al. 2000], SDT [Papadimitratos and Haas 2003], Ariadne [Hu et al. 2002] and Afora [Lee 2002]. Watchdog exploits the fact that a node can overhear its neighboring nodes forwarding packets to other destinations. If a node does not overhear a neighbor forwarding more than a threshold number of packets, it concludes that the neighbor is adversarial. The scheme does not require any explicit network overhead or cryptography and is effective against the basic black hole attack in single rate fixed transmission power networks. However, it does not perform well when either power control or multi-rate (i.e. 802.11abg [802 1999a; 1999b]) are used, since their use will violate the assumption that the forwarding transmission is successfully overheard. In addition, the method is vulnerable to attacks from two consecutive and colluding adversaries where the first adversarial node does not report that the second did not forward the data.

SDT avoids the black hole attack by disseminating a packet across several disjoint paths. The method has relatively low overhead, converges quickly, and works effectively in a well connected ad hoc wireless network, where the number of disjoint paths is large. A multi-path approach to avoid the black hole attack is also used by Ariadne. Nodes that have several paths available assign a fraction of packets to be sent along each path. The disadvantage is that in a sparsely connected network, where the number of available disjoint paths is small, all of the discovered paths may contain an attacker and thus, the schemes will be less effective. Finally, in Afora nodes do not forward all route replies, but probabilistically drop some of them to increase resilience to attacks. Both Ariadne and Afora rely on an external feedback mechanism to notify them about the resilience of particular paths. Our approach is different in that we do not rely on external feedback mechanisms, instead we identify the problematic links on faulty paths and avoid them.

Rushing Attack Prevention (RAP) [Hu et al. 2003b] prevents the rushing attack by waiting for up to k flood requests and then randomly selecting one to forward, rather than always forwarding only the first one. To prevent a single attacker from bypassing the scheme by simply sending k requests, the RAP protocol incorporates secure neighbor discovery and secure route delegation schemes. However, these schemes have significant network overhead because multiple rounds of communication are required for every hop the route request propagates. In addition, RAP will be ineffective if the adversary has compromised k or more nodes.

A mechanism proposed to prevent wormholes is *Packet Leashes* [Hu et al. 2003a]. The authors suggest restricting the maximum transmission distance by using either a tight time synchronization (temporal leash) or location information (geographic leash). Temporal leashes require additional hardware, such as accurate clocks or GPS receivers. The protocol is effective at preventing the traditional wormhole attack, but is ineffective against the Byzantine variant because preventing the wormhole is the responsibility of its end points. In this case the end points are adversarial and cannot be trusted to follow the protocol correctly. Another method, proposed for ad hoc wireless sensor networks relies on directional antennas [Hu and Evans 2004]. The approach prevents wormholes by having each node maintaining accurate information about its neighbors. Messages coming from a node that is not perceived as a neighbor are ignored. The protocol is appropriate for sensors networks which in general have low mobility. However, maintaining neighbor information in mobile networks is more challenging and expensive. In addition, the protocol that maintains infor-

mation about neighbors can itself be subjected to wormhole attacks, particularly because it requires cooperation among nodes. More recently, Truelink [Eriksson et al. 2006] prevents non-Byzantine wormholes by using MAC layer acknowledgments to infer if a link exists or not between two nodes.

Several attacks relying on impersonation, lying and resource consumption were identified in [Jakobsson et al. 2003]. The building block relying on impersonation and lying are not effective against ODSBR. Although ODSBR does not specifically address resource consumption, it uses several mechanisms to limit the propagation of undesired requests.

Several methods relying on node cooperation to build reputation were proposed in [Buchegger and Boudec 2002; Michiardi and Molva 2002; Theodorakopoulos and Baras 2004]. The CONFIDANT protocol [Buchegger and Boudec 2002] avoids node misbehavior by using a reputation mechanism in which trust relationships are established between nodes based on direct observations and indirect observations reported by other nodes. The CORE protocol [Buchegger and Boudec 2002] takes a similar approach and uses the concept of reputation to enforce node cooperation. In contrast with the CONFIDANT protocol, CORE requires that reputation values received from indirect observations be positive, thus preventing malicious nodes from wrongfully accusing legitimate nodes. The adversarial model considered in [Buchegger and Boudec 2002; Michiardi and Molva 2002] is not as strong as the one considered in this paper, i.e. it does not offer protection against colluding Byzantine attackers. Specifically, the reputation mechanism can be itself exploited by colluding malicious nodes to wrongfully accuse correct nodes or increase each other's reputation by providing false observations about each other. The general framework for evaluating trust in ad hoc networks proposed in [Theodorakopoulos and Baras 2004] is interesting but suffers from several drawbacks. In particular the approach is expensive in practice because it uses an iterative algorithm involving many rounds of interactions. In addition the framework does not address the problem of gathering indirect opinions from non-neighboring nodes in a Byzantine adversarial environment.

Buttyan and Hubaux [Buttyan and Hubaux 2003] address the problem of "selfish" nodes and propose a mechanism based on *nuglets* to stimulate node cooperation in packet forwarding. They assume each node contains a tamper resistant security module which maintains a nuglet counter; a node needs to spend nuglets in order to send its own packets and is rewarded by forwarding packets for other nodes. The approach is interesting, but relies on specialized hardware and considers only selfish nodes.

7. CONCLUDING REMARKS

We focused on analyzing the ability of ad hoc routing protocols to provide correct service in the presence of failures and Byzantine attacks. We presented ODSBR, a secure on-demand routing protocol resilient to Byzantine failures caused by an adversary or group of colluding adversaries. Key components of our scheme are an adaptive probing technique that detects malicious links after $\log n$ faults have occurred, where n is the length of the routing path, and a new metric that captures adversarial behavior. Problematic links are avoided by the route discovery protocol using the newly proposed metric. Our protocol bounds logarithmically the total amount of damage that can be caused by an attacker or group of attackers.

We demonstrated through experiments that state-of-art routing protocols such as AODV are vulnerable to a wide range of Byzantine attacks. After examining several types of at-

tacks, we conclude that the most effective attack is the central wormhole combined with flood rushing: only two colluding adversaries were able to reduce AODV's delivery ratio to 51%. We showed that ODSBR was able to mitigate a wide range of Byzantine attacks; in particular, it was not significantly affected by flood rushing. Its performance only decreased when it needed to detect and avoid a large number of adversarial links.

Our experiments showed that flood rushing and strategic positioning of adversaries are the two most important factors for an effective attack against insecure on-demand protocols, particularly when adversaries collude. The flood rushing attack amplifies any attack it is combined with because it allows an attacker to have control on the route selection. Ad hoc routing protocols must be designed to take into consideration this attack.

REFERENCES

The network simulator - ns2. http://www.isi.edu/nsnam/ns/.

1999a. *IEEE Std 802.11a-1999*. http://standards.ieee.org/.

1999b. *IEEE Std 802.11b-1999*. http://standards.ieee.org/.

2001. *Advanced Encryption Standard (AES)*. Number FIPS 197. National Institute for Standards and Technology (NIST). http://csrc.nist.gov/encryption/aes/.

2002. *The Keyed-Hash Message Authentication Code (HMAC)*. Number FIPS 198. National Institute for Standards and Technology (NIST). http://csrc.nist.gov/publications/fips/index.html.

2006. *Digital Signature Standard (DSS)*. Number FIPS 186-3. National Institute for Standards and Technology (NIST). http://csrc.nist.gov/publications/drafts/fips_186-3/Draft-FIPS-186-3_March2006.pdf.

AWERBUCH, B., HOLMER, D., AND RUBENS, H. 2005. The medium time metric: High throughput route selection in multirate ad hoc wireless networks. *Kluwer Mobile Networks and Applications (MONET) Journal, Special Issue on Internet Wireless Access: 802.11 and Beyond*.

BRADLEY, K. A., CHEUNG, S., PUKETZA, N., MUKHERJEE, B., AND OLSSON, R. A. 1998. Detecting disruptive routers: A distributed network monitoring approach. In *Proceedings of IEEE Symposium on Security and Privacy*.

BROWN, M., CHEUNG, D., HANKERSON, D., HERNANDEZ, J., KIRKUP, M., AND MENEZES., A. 2000. PGP in constrained wireless devices. In *Proceeding of USENIX Security Symposium*. USENIX.

BUCHEGGER, S. AND BOUDEC, J.-Y. L. 2002. Performance analysis of the CONFIDANT protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks. In *Proc. of MobiHoc*. ACM, 226–236.

BUTTYAN, L. AND HUBAUX, J.-P. 2003. Stimulating cooperation in self-organizing mobile ad hoc networks. *Mob. Netw. Appl. 8,* 5, 579–592.

CHEUNG, S. AND LEVITT, K. 1997. Protecting routing infrastructures from denial of service using cooperative intrusion detection. In *New Security Paradigms Workshop*.

DE COUTO, D. S. J., AGUAYO, D., BICKET, J., AND MORRIS, R. 2003. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of ACM Annual International Conference of Mobile Computing (MOBICOM)*. San Diego, California.

ERIKSSON, J., KRISHNAMURTHY, S. V., AND FALOUTSOS, M. 2006. Truelink: A practical countermeasure to the wormhole attack in wireless networks. In *Proc. of ICNP '06*.

HAUSER, R., PRZYGIENDA, T., , AND TSUDIK, G. 1997. Reducing the cost of security in link-state routing. In *Proceedings of ISOC Symposium of Network and Distributed Systems Security (NDSS)*.

HU, L. AND EVANS, D. 2004. Using directional antennas to prevent wormhole attacks. In *Proceedings of ISOC Symposium of Network and Distributed Systems Security (NDSS)*.

HU, Y.-C., JOHNSON, D. B., AND PERRIG, A. 2002. SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*.

HU, Y.-C., PERRIG, A., AND JOHNSON, D. B. 2002. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proceedings of ACM Annual International Conference of Mobile Computing (MOBICOM)*.

HU, Y.-C., PERRIG, A., AND JOHNSON, D. B. 2003a. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *Proceedings of IEEE Conference of the IEEE Communications Society (INFO-COMM)*.

HU, Y.-C., PERRIG, A., AND JOHNSON, D. B. 2003b. Rushing attacks and defense in wireless ad hoc network routing protocols. In *Proceedings of ACM Workshop of Wireless Security (WiSe)*.

HUBAUX, J.-P., BUTTYAN, L., AND CAPKUN, S. 2001. The quest for security in mobile ad hoc networks. In *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*.

IEEE. 1999. *IEEE Std 802.11, 1999 Edition*. http://standards.ieee.org/catalog/olis/lanman.html.

JAKOBSSON, M., WETZEL, S., AND YENER, B. 2003. Stealth attacks on ad-hoc wireless networks. In *IEEE Vehicular Technology Conference*.

JOHNSON, D. B., MALTZ, D. A., AND BROCH, J. 2001. *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks. in Ad Hoc Networking*. Addison-Wesley, Chapter 5, 139–172.

KENT, S., LYNN, C., AND SEO, K. 2000. Secure border gateway protocol (s-bgp). *IEEE Journal on Selected Areas in Communication 18*, 4.

KUROSE, J. AND ROSS, K. 2000. *Computer Networking, a top down approach featuring the Internet*. Addison-Wesley Longman.

LAMPORT, L., SHOSTAK, R., AND PEASE, M. 1982. The byzantine generals problem. *ACM Trans. Program. Lang. Syst. 4*, 3, 382–401.

LEE, H. I. 2002. Afora: Ad hoc routing in the face of misbehaving nodes. Master's Thesis, MIT.

LUNDGREN, H., NORDSTRÖM, E., AND TSCHUDIN, C. 2002. Coping with communication gray zones in ieee 802.11b based ad hoc networks. In *Proceedings of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM)*. ACM Press, New York, NY, USA, 49–55.

MARTI, S., GIULI, T., LAI, K., AND BAKER, M. 2000. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of ACM Annual International Conference of Mobile Computing (MOBICOM)*.

MENEZES, A., VAN OORSCHOT, P., AND VANSTONE, S. 1996. *Handbook of Applied Cryptography*. CRC Press.

MICHIARDI, P. AND MOLVA, R. 2002. CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proc. of Sixth IFIP Communications and Multimedia Security Conference*.

MIZRAK, A., CHENG, Y.-C., MARZULLO, K., AND SAVAGE, S. 2005. Fatih: Detecting and isolating malicious routers. In *Proceedings of International Conference on Dependable Systems and Networks (DSN)*.

PAPADIMITRATOS, P. AND HAAS, Z. 2002. Secure routing for mobile ad hoc networks. In *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS)*. 27–31.

PAPADIMITRATOS, P. AND HAAS, Z. 2003. Secure data transmission in mobile ad hoc networks. In *Proceedings of ACM Workshop of Wireless Security (WiSe)*.

PERKINS, C. E. AND BHAGWAT, P. 1994. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*.

PERKINS, C. E. AND ROYER, E. M. 2000. *Ad hoc Networking*. Addison-Wesley, Chapter Ad hoc On-Demand Distance Vector Routing.

PERLMAN, R. 1988. Network layer protocols with byzantine robustness. Ph.D. thesis, MIT LCS TR-429.

PERLMAN, R. 2005. Routing with byzantine robustness. Tech. Rep. TR-2005-146, Sun Microsystems.

PERRIG, A., CANETTI, R., SONG, D., AND TYGAR, D. 2001. Efficient and secure source authentication for multicast. In *Proceedings of ISOC Symposium of Network and Distributed Systems Security (NDSS)*.

RIVEST, R. L., SHAMIR, A., AND ADLEMAN, L. M. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM 21*, 2 (Feb.), 120–126.

SANZGIRI, K., DAHILL, B., LEVINE, B. N., SHIELDS, C., AND BELDING-ROYER, E. 2002. A secure routing protocol for ad hoc networks. In *Proceedings of IEEE International Conference on Network Protocols (ICNP)*.

SMITH, B. R., MURTHY, S., AND GARCIA-LUNA-ACEVES, J. 1997. Securing distance-vector routing protocols. In *Proceedings of ISOC Symposium of Network and Distributed Systems Security (NDSS)*.

STONE, J. AND PARTRIDGE, C. 2000. When the CRC and TCP checksum disagree. In *Proceedings of SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*.

SYVERSON, P. F., GOLDSCHLAG, D. M., AND REED, M. G. 1997. Anonymous connections and onion routing. In *IEEE Symposium on Security and Privacy*.

THEODORAKOPOULOS, G. AND BARAS, J. S. 2004. Trust evaluation in ad-hoc networks. In *Proc. of ACM Workshop on Wireless Security (WiSe '04)*. 1–10.

YOON, J., LIU, M., AND NOBLE, B. D. 2003. Random waypoint considered harmful. In *Proceedings of IEEE Conference of the IEEE Communications Society (INFOCOMM)*. San Francisco, CA.

ZHOU, L. AND HAAS, Z. 1999. Securing ad hoc networks. *IEEE Network Magazine 13,* 6.

ZIMMERMANN, P. 1995. *The Official PGP User's Guide*. MIT Press.