An On-Demand Secure Routing Protocol Resilient to Byzantine Failures

Baruch Awerbuch, David Holmer, Cristina Nita-Rotaru and Herbert Rubens
Department of Computer Science
Johns Hopkins University
3400 North Charles St.
Baltimore, MD 21218 USA
{baruch, dholmer, crisn, herb}@cs.jhu.edu

ABSTRACT

An ad hoc wireless network is an autonomous self-organizing system of mobile nodes connected by wireless links where nodes not in direct range can communicate via intermediate nodes. A common technique used in routing protocols for ad hoc wireless networks is to establish the routing paths on-demand, as opposed to continually maintaining a complete routing table. A significant concern in routing is the ability to function in the presence of byzantine failures which include nodes that drop, modify, or mis-route packets in an attempt to disrupt the routing service.

We propose an on-demand routing protocol for ad hoc wireless networks that provides resilience to byzantine failures caused by individual or colluding nodes. Our adaptive probing technique detects a malicious link after $\log n$ faults have occurred, where n is the length of the path. These links are then avoided by multiplicatively increasing their weights and by using an on-demand route discovery protocol that finds a least weight path to the destination.

Categories and Subject Descriptors

C.2.0 [**General**]: Security and protection; C.2.1 [**Network Architecture and Design**]: Wireless communication; C.2.2 [**Network Protocols**]: Routing protocols

General Terms

Algorithms, Design, Reliability, Security, Theory

Keywords

ad hoc wireless networks, on-demand routing, security, byzantine failures

WiSe '02, September 28, 2002, Atlanta, Georgia, USA.

1. INTRODUCTION

Ad hoc wireless networks are self-organizing multi-hop wireless networks where all the hosts (nodes) take part in the process of forwarding packets. Ad hoc networks can easily be deployed since they do not require any fixed infrastructure, such as base stations or routers. Therefore, they are highly applicable to emergency deployments, natural disasters, military battle fields, and search and rescue missions.

A key component of ad hoc wireless networks is an efficient routing protocol, since all of the nodes in the network act as routers. Some of the challenges faced in ad hoc wireless networks include high mobility and constrained power resources. Consequently, ad hoc wireless routing protocols must converge quickly and use battery power efficiently. Traditional proactive routing protocols (link-state [1] and distance vectors [1]), which use periodic updates or beacons which trigger event based updates, are less suitable for ad hoc wireless networks because they constantly consume power throughout the network, regardless of the presence of network activity, and are not designed to track topology changes occurring at a high rate.

On-demand routing protocols [2, 3] 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.

Many of the security threats to ad hoc wireless routing protocols are similar to those of wired networks. For example, a malicious node may advertise false routing information, try to redirect routes, or perform a denial of service attack by engaging a node in resource consuming activities such as routing packets in a loop. Furthermore, due to their cooperative nature and the broadcast medium, ad hoc wireless networks are more vulnerable to attacks in practice [4].

Although one might assume that once authenticated, a node should be trusted, there are many scenarios where this is not appropriate. For example, when ad hoc networks are used in a public Internet access system (airports or conferences), users are authenticated by the Internet service provider, but this authentication does not imply trust between the individual users of the service. Also, mobile devices are easier to compromise because of reduced physical security, so complete trust should not be assumed.

Our contribution. We focus on providing routing survivability under an adversarial model where any intermediate node or group of nodes can perform byzantine attacks such as creating routing loops, misrouting packets on non-optimal paths, or selectively dropping packets (*black hole*). Only the source and destination nodes are assumed to be trusted. We propose an on-demand routing protocol for wireless ad hoc networks that operates under this strong adversarial model.

It is provably impossible under certain circumstances, for example when a majority of the nodes are malicious, to attribute a byzantine fault occurring along a path to a specific node, even using expensive and complex byzantine agreement. Our protocol circumvents this obstacle by avoiding the assignment of "guilt" to individual nodes. Instead it reduces the possible fault location to two adjacent nodes along a path, and attributes the fault to the link between them. 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.

Our protocol consists of the following phases:

- *Route discovery with fault avoidance.* Using flooding and a faulty link weight list, this phase finds a least weight path from the source to the destination.
- *Byzantine fault detection.* This phase discovers faulty links on the path from the source to the destination. Our adaptive probing technique identifies a faulty link after $\log n$ faults have occurred, where n is the length of the path.
- *Link weight management.* This phase maintains a weight list of links discovered by the fault detection algorithm. A multiplicative increase scheme is used to penalize links which are then rehabilitated over time. This list is used by the route discovery phase to avoid faulty paths.

The rest of the paper is organized as follows. Section 2 summarizes related work. We further define the problem we are addressing and the model we consider in Section 3. We then present our protocol in Section 4 and provide an analysis in Section 5. We conclude and suggest directions for future work in Section 6.

2. RELATED WORK

Secure routing protocols for ad hoc wireless networks is a fairly new topic. Although routing in ad hoc wireless networks has unique aspects, many of the security problems faced in ad hoc routing protocols are similar to those faced by wired networks. In this section, we review the work done in securing routing protocols for both ad hoc wireless and wired networks.

One of the problems addressed by researchers is providing an effective public key infrastructure in an ad hoc wireless environment which by nature is decentralized. Examples of these works are as follows. Hubaux et al.[5] proposed a completely decentralized public-key distribution system similar to PGP [6]. Zhou and Haas [7] explored threshold cryptography methods in a wireless environment. Brown et al.[8] showed how PGP, enhanced by employing elliptic curve cryptography, is a viable option for wireless constrained devices.

A more general trust model where levels of security are defined for paths carrying specific classes of traffic is suggested in [9]. The paper discusses very briefly some of the cryptographic techniques that can be used to secure on-demand routing protocols: shared key encryption associated with a security level and digital signatures for data source authentication.

As mentioned in [10], source authentication is more of a concern in routing than confidentiality. Papadimitratos and Haas showed in [11] 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 [12] primitive which relies on the existence of security associations between sources and destinations. Dahill et al.[16] focus on providing hop-by-hop authentication for the route discovery stage of two well-known on-demand protocols: AODV [2] and DSR [3], relying on digital signatures. Other significant works include SEAD [13] and Ariadne [4] that provide efficient secure solutions for the DSDV [14] and DSR [3] routing protocols, respectively. While SEAD uses one-way hash chains to provide authentication, Ariadne uses a variant of the Tesla [15] source authentication technique to achieve similar security goals.

Marti et al.[18] address a problem similar to the one we consider, survivability of the routing service when nodes selectively drop packets. They take advantage of the wireless cards promiscuous mode and have trusted nodes monitoring their neighbors. Links with an unreliable history are avoided in order to achieve robustness. Although the idea of using the promiscuous mode is interesting, this solution does not work well in multi-rate wireless networks because nodes might not hear their neighbors forwarding communication due to different modulations. In addition, this method is not robust against collaborating adversaries.

Also, relevant work has been done in the wired network community. Many researchers focused on securing classes of routing protocols such as link-state [10, 19, 20, 21] and distance-vector [22]. Others addressed in detail the security issues of well-known protocols such as OSPF [23] and BGP [24]. The problem of source authentication for routing protocols was explored using digital signatures [23] or symmetric cryptography based methods: hash chains [10], chains of one-time signatures [20] or HMAC [21]. Intrusion detection is another topic that researchers focused on, for generic link-state [25, 26] or OSPF [27].

Perlman [28] designed the Network-layer Protocol with Byzantine Robustness (NPBR) which addresses denial of service at the expense of flooding and digital signatures. The problem of byzantine nodes that simply drop packets (*black holes*) in wired networks is explored in [29, 30]. The approach in [29] is to use a number of trusted nodes to probe their neighbors, assuming a limited model and without discussing how probing packets are disguised from the adversary. A different technique, flow conservation, is used in [30]. Based on the observation that for a correct node the number of bytes entering a node should be equal to the number of bytes exiting the node (within a threshold), the authors suggest a scheme where nodes monitor the flow in the network. This is done by requiring each node to have a copy of the routing table of their neighbors and reporting the incoming and outgoing data. Although interesting, the scheme does not work when two or more adversarial nodes collude.

3. PROBLEM DEFINITION AND MODEL

In this section we discuss the network and security assumptions we make in this paper and present a more precise description of the problem we are addressing.

3.1 Network Model

This work relies on a few specific network assumptions. Our protocol requires bi-directional communication on all links in the network. This is also a requirement of most wireless MAC protocols, including 802.11 [31] and MACAW [32]. We focused on providing a secure routing protocol, which addresses threats to the ISO/OSI network layer. We do not specifically address attacks against lower layers. For example, the physical layer can be disrupted by jamming, and MAC protocols such as 802.11 can be disrupted by attacks using the special RTS/CTS packets. Though MAC protocols can detect packet corruption, we do not consider this a substitute for cryptographic integrity checks [33].

3.2 Security Model and Considered Attacks

In this work we consider only the source and the destination to be trusted. Nodes that can not be authenticated do not participate in the protocol, and are not trusted. Any intermediate node on the path between the source and destination can be authenticated and can participate in the protocol, but may exhibit byzantine behavior. The goal of our protocol is to detect byzantine behavior and avoid it. We define *byzantine behavior* as any action by an authenticated node that results in disruption or degradation of the routing service. We assume that an intermediate node can exhibit such behavior either alone or in collusion with other nodes. More generally, we use the term *fault* to refer to any disruption that causes significant loss or delay in the network. A fault can be caused by byzantine behavior, external adversaries, lower layer influences, and certain types of normal network behavior such as bursting traffic.

An adversary or group of adversaries can intercept, modify, or fabricate packets, create routing loops, drop packets selectively (often referred to as a *black hole*), artificially delay packets, route packets along non-optimal paths, or make a path look either longer or shorter than it is. All the above attacks result in disruption or degradation of the routing service. In addition, they can induce excess resource consumption which is particularly problematic in wireless networks.

There are strong attacks that our protocol can not prevent. One of these strong attacks, referred to as a *wormhole* [4], is where two attackers establish a path and tunnel packets from one to another. For example, the attackers can tunnel route request packets that can arrive faster than the normal route request flood. This may result in non-optimal adversarial controlled routing paths. Our protocol addresses this attack by treating the wormhole as a single link which will be avoided if it exhibits byzantine behavior, but does not prevent the wormhole formation. Also, we do not address traditional denial of service attacks which are characterized by packet injection with the goal of resource consumption.

Whenever possible, our protocol uses efficient cryptographic primitives. This requires pairwise shared keys[1] which are established on-demand. The public-key infrastructure used for authentication can be either completely distributed (as described in [5]), or Certificate Authority (CA) based. In the latter case, a distributed cluster of peer CAs sharing a common certificate and revocation list can be deployed to improve the CA's availability.

[1]We discourage group shared keys since this is an invitation for impersonation in a cooperative environment.

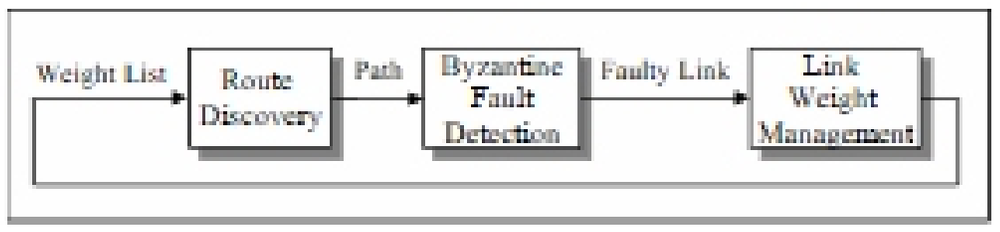

Figure 1: Secure Routing Protocol Phases

3.3 Problem Definition

The goal of this work is to provide a robust on-demand ad hoc routing service which is resilient to byzantine behavior and operates under the network and security models described in Sections 3.1 and 3.2. We attempt to bound the amount of damage an adversary or group of adversaries can cause to the network.

4. SECURE ROUTING PROTOCOL

Our protocol establishes a reliability metric based on past history and uses it to select the best path. 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 normal packet stream. These 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 (see Figure 1):

- *Route discovery with fault avoidance.* Using flooding, cryptographic primitives, and the source's weight list as input, this phase finds and outputs the full least weight path from the source to the destination.
- *Byzantine fault detection.* The goal of this phase is to discover faulty links on the path from the source to the destination. This phase takes as input the full path and outputs a faulty link. Our adaptive probing technique identifies a faulty link after $\log n$ faults occurred, where n is the length of the path. Cryptographic primitives and sequence numbers are used to protect the detection protocol from adversaries.
- *Link weight management.* This phase maintains a weight list of links discovered by the fault detection algorithm. A multiplicative increase scheme is used to penalize links which are then rehabilitated over time. The weight list is used by the route discovery phase to avoid faulty paths.

4.1 Route Discovery with Fault Avoidance

Our route discovery protocol floods both the route request and the response in order to ensure that if any fault free path exists in the network, a path can be established. However, there is no guarantee that the established path is free of