International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) e-ISSN: 2395-5325
Volume 2, Issue 1 (January '2016)

A STUDY ON DELIVERING PROTOCOL FOR MULTI ADMINISTRATIVE DOMAIN MANET

B.Nathiya,
M.Phil Scholar,
Department of Computer Science and Applications,
Vivekanandha College of Arts and Science College for Women (Autonomous),
Elayampalayam, Namakkal, Tamilnadu.

N.Kohila,
Assistant Professor,
Department of Computer Science and Applications,
Vivekanandha College of Arts and Science College for Women (Autonomous),
Elayampalayam, Namakkal, Tamilnadu.

Abstract: Mobile Ad-hoc Networks (MANETs) enable highly dynamic networks of nodes to be formed nodes that are moving in a non-deterministic pattern, for example in a battlefield scenario. It is Multi Administrative Domain (MAD) if each network node belongs to an independent authority that is each node owns its resources and there is no central authority owning all network nodes. One of the main obstructions in designing Service Advertising, Discovery and Delivery (SADD) protocol for MAD MANETs is the fact that, in an attempt to increase their own visibility, network nodes tend to flood the network with their advertisements.

Keywords: Mobile Ad-hoc Networks, Multi Administrative Domain, Advertising, Discovery and Delivery

I.INTRODUCTION

MANETs consisting of mobile devices such as laptops, cell-phones, PDAs are more and more widespread. This opens up opportunities for many interesting applications. Peer to Peer is the way content can be advertised and exchanged between peers thus supporting entertainment as well as emergency management applications. Protocols supporting the above activities are often called Service Advertising, Discovery and Delivery (SADD) protocols. SADD protocols have been extensively studied [1, 2].However, to the best of our knowledge, all SADD protocols proposed in the literature target SAD-MANETs. Unfortunately, SADD protocols designed for SAD-MANETs may not work for MAD-MANETs. For example, a WiFi MANET consisting of handheld de- vices each belonging to a different user is indeed a MAD-MANET. If network nodes behave selfishly they may deviate from the specified protocol if this is at their advantage. Thus, a selfish PDA (user) may refuse to forward packets (to save energy) or may decrease its backoff time (to increase its bandwidth) or may increase its advertisement frequency (to increase its own visibility). Hence, a SADD protocol for MAD-MANETs must deploy suitable countermeasures to protect the network from node misbehaviors that may eventually kill any networking activity.

II. NODE BEHAVIOR

In order to design protocols for MAD-MANETs, reasonable hypotheses on node behaviors are needed. Here are some well known classes of node behaviors. Malicious nodes are willing to spend their resources just to damage the net- work. For example, malicious nodes may perpetrate a Denial of Service (DoS) attack by flooding the network (or part of it) with their messages. Malicious nodes may do so even if this will use up all of their energy without actually doing them any real service. Doing them any real service. Selfish nodes act in their best interest. For example, rather than spending its energy flooding the network with messages without getting any reward, a selfish node may refuse to forward packets (to save energy) or may decrease its backoff time after a collision (to increase its bandwidth). One may think that altruistic nodes do not exist. However it is just a matter of fact that most nodes (agents) in a MAD-MANET are indeed altruistic. That is one can often safely assume that most (although not all) network nodes are altruistic.

Assuming that all nodes are malicious is a too pessimistic hypothesis. No protocol for MAD-MANETs can be designed under such a hypothesis. As for MAD-MANETs the typical approach is to assume that most nodes are selfishor altruistic [3]. In some cases malicious node can be tolerated fol-lowing the approach in [38]. As for SADD protocols, the main obstruction to overcome is flooding. In fact, all nodes in the network will be eager to broadcast their advertisements (otherwise they would not participate in the protocol to begin with). This may result in flooding which, in turn, leads to a Denial of Service (DoS) attack. In fact, if too many nodes flood the network with their advertisements eventually no one will be able to send anything (DoS attack). Note that flooding is an attractive misbehavior for malicious as well as selfish nodes. In fact a selfish node may be interested in increasing its advertising frequency to increase its visibility. On the other hand, a malicious node may increase its advertising frequency since it is an easy way to flood the network and carry out a DoS attack. Of course, flooding is not the only possible misbehavior for nodes in a MAD-MANET. We note, however, that flooding is something that any node participating in the protocol will desire to do and, last but not least, can easily do by simply changing a protocol parameter (namely, the advertisement frequency). This is not the case with other attacks. For example, packet dropping may be desirable for a node, but usually requires some nontrivial work on the protocol implementation.

III.OVERVIEW OF MULTI ADMINISTRATIVE DOMAIN AND SADD

The advertisement and the profile define the services (e.g. content, resources, consultancy, etc). The full service description (just full description in the following) gives full information about the advertised service. Here are examples of full descriptions. If the advertising node is offering a movie then the full description will be the advertised movie file. If the advertising node is offering, say, piano lessons,

then the full description will be a file with the maestro address. Each node periodically broadcasts its advertisement to the network nodes. All nodes cooperate in spreading the advertisement by forwarding it to their neighbors (advertising phase). Upon receiving an advertisement, a node uses its own profile to evaluate its interest in the received advertisement. If the received advertisement is considered interesting, the interested node starts a unicast transmission asking the advertising node for the full description (discovery phase). Finally, the advertising node delivers such full description using again a unicast transmission (delivery phase).

To limit collisions, nodes should avoid forwarding recently forwarded packets. In a SAD-MANET this is typically done by endowing packets with a sequence number field [4][5]. However, in a MAD-MANET sequence numbers do not work since a misbehaving node may broadcast many times the same packet (advertisement) using higher and higher sequence numbers. In this way, packets coming from that node will appear to other nodes as new packets and, accordingly, will be always forwarded. This increases bandwidth usage and visibility of the misbehaving node, but decreases bandwidth and visibility of all other nodes.

To counteract such misbehavior, nodes may store in RAM the forwarded messages. In this way, each node will be able to detect if an incoming message (from a certain node) is new or recently seen (and processed). However, usually handheld devices do not have enough RAM to effectively store all recently seen messages. The problem of SADD protocols for SAD-MANETs, that is, they do not account for selfish or malicious misbehaviors of nodes. On the other hand, by exploiting the obedient nature of SAD-MANET nodes, the previously mentioned protocols propose routing schemas much more sophisticated than ours.

To the best of our knowledge no previously published paper addresses the problem of designing a SADD protocol for MAD-MANETs. There are however protocols for MAD networks (i.e. networks consisting of selfish nodes). An example is the BAR Gossip [6] protocol which allows an altruistic (i.e. following the protocol) broadcaster to stream data to a pool of possibly selfish or even malicious clients. Other approaches rely on specific domain policies and architectures [7].Flooding of advertisements can be considered as a particular case (an easy one to carry out) of Denial of Service (DoS) attack. For this reason we will compare our protocol with those striving to counteract DoS attacks in MANETs. [8]

The SEAD protocol in [9] makes use of elements from a one-way hash chain to provide authentication for both the sequence number and the metric in each entry. The SRP protocol in [10] is based on multiple routes and relies exclusively on the mutual authentication of the end nodes (source and destination). In a probabilistic routing approach is proposed to drive the on-demand discovery process and to reduce the control overhead. Note that the above secure protocols do not solve our problem since they are not able to prevent flooding of legitimate messages (advertising) from legitimate nodes participating in the protocol. The Ariadne protocol enables to secure the routing discovery phase and ensures that all forwarded packets follow the secure route. Here, request-flooding attacks are considered and the proposed solution consists of a rate limit for each node the route requests it is asked to relay. Even if mitigating the effect of flooding at the network layer, this solution does not solve our problem. Indeed, following the approach in [11] the more advertisement messages a node sends, the more it will get broadcasted at the expense of the nodes sending less advertisements. As a result, in our framework the Ariadne approach not only is not sufficient to discourage advertising flood, but encourages it since nodes that are not flooding may never see their advertisement broadcasted.

MANETs have been highly vulnerable to attacks due to the dynamic nature of their network infrastructure. In [12] the authors present an authentication, authorization and security assessment strategy in which, once a device enters a MANET, it is immediately taken out if considered dangerous by the infrastructure. None of the aforementioned approaches can be applied to our context since the DoS attack we are considering here (advertisement flood) does not fall in the class of attacks studied in [12]. Since in a MANET nodes are usually constrained by limited computation resources, selfish nodes may refuse to cooperate.

IV.PROTOCOL DESIGN FOR MAD-SADD

a) Communication Environment

As for the link layer, wireless communication links between network nodes are implemented using the WiFi (IEEE 802.11b) protocol. As for the transport layer, nodes (processes) communicate using the UDP protocol. As for our protocol, MAD-SADD, it is an application layer protocol. Note that since MAD- SADD takes care of routing we do not use the routing protocols provided at the network layer level.

b) Message header

MAD-SADD messages are organized into packets. A packet is organized as a record consisting of administrative fields (header) and a data field. The administrative fields are the following: 1. **time:** packet time stamp; 2. **source address**:ID of the node (see assumption 4 of Sect. 2); 3. **destination address**: ID of final destination node; 4. **packet from address**: ID of node from which the packet has been received; 5. **next hop address**: ID of node to which thepacket will be forwarded; 6**. seq number**: sequence number; 7. **type**: packet type (i.e. short-description, query-unicast, data-info, data-info-ack, data, ack,no-route).

c) Advertising Phase: Broadcast

In our scenario, each node is eager to transmit its own advertisement. In order to meet this requirement, each node periodically sends its own advertisement in broadcast to all the reachable neighbors (see Figure 1). This happens with a fixed advertisement frequency f, i.e. a node broadcasts its own advertisement every 1/f seconds. The advertisement must fit into one packet. For this reason, we also call it short description (of the service). Moreover short-description is also the type of a packet containing a short description (i.e. an advertisement). A node receiving a short description will request (in unicast) the full description of the advertisement only if it is interested in the advertised service. As an

example, in Figure1 node a (advertising node) broadcasts its advertising, reaching the two nodes within its transmission range.

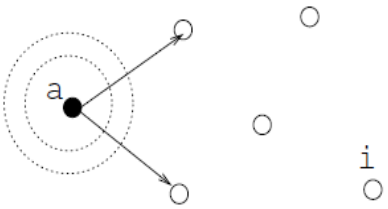

Figure 1: MAD-SADD execution -Advertising broadcast

d) Advertising Phase: Forwarding

When receiving a short-description packet, a node is required to forward it in broadcast to all the reachable neighbors Figure 2. Because of assumptions 1 and 3 of Sect. 2 we can assume that when requested a node will actually forward a packet, without modifying it. However, if all nodes forward in broadcast all the advertising they receive, there would be many collisions, resulting in poor overall performances. in this phase we cannot use sequence numbers as it is usually done in MANETs. We address this problem by forwarding only new (i.e. not recently received) short- description packets. In this way, we reduce the number of forward operations for each node, thus saving energy and decreasing the number of collisions. In order to decide if a newly arrived short-description packet m was already received in the past or is new, we proceed as follows. Each node maintains a cache BF with the signatures of the received short descriptions. The data structure used to implement BF is a Bloom filter. Thus, when node i (interested node) receives from node j a short description packet m, node i checks whether (the signature of) m is already in BF (m is old) or not (m is new). If m is in BF, then i checks if m was recently forwarded. More specifically, let f be the advertising frequency. If the difference between the time stamp of m and the time stamp of the last message from j is less than 1/f, then m is discarded, otherwise it is forwarded. This means that j cannot send to i message m more than once within 1/f seconds. Of course j can send to i other messages within the same period of time. The above approach allows us to avoid advertisement flooding. Note that, since the Bloom filter stores advertisement signatures, we may have false positives, i.e. we may decide that a short description is old when it is indeed new. However, this is very unlikely to happen. As an example, in Figure 2 the advertising is propagated through the network.

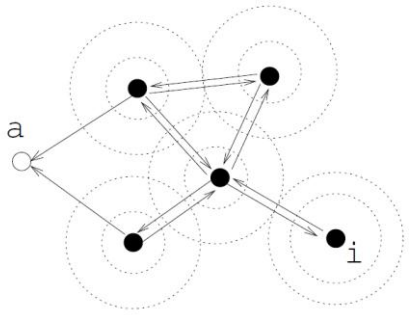

Figure 2: MAD-SADD execution -Advertising forward

f) Discovery Phase

When node i receives a new advertisement message m (i.e. m is not in BF), it evaluates its interest for advertisement m using its profile. For example, this can be done by using a data mining algorithm (e.g. cosine similarity [13]) when messages are free format or exploiting the message format (e.g. as in [14]). Of course any of the above approaches can be used within MAD-SADD. To carry out our simulations here we use a free format for profiles and advertisements, and cosine similarity to evaluate advertisements. If node i deems m to be interesting, it discards any other incoming short-description packet and behaves as follows. First of all, node i stores in a k-dimensional vector, say undo-vector, the entry values of BF for the k indexes computed by BF hash functions on argument m. Then i inserts the signature of m into BF. Finally, i starts a unicast communication with the advertiser of m, call it a. Namely, using a unicast transmission i asks a for the full description M of the advertisement m. The Bloom filter BF is cleared when the message is added, where is the maximum number of insertions BF has been designed for. In this way, BF will contain only recently seen advertisements, thus allowing to forward old advertisements to newly arrived nodes. Here, we simply remark that, once the intermediate nodes between i and a are selected, all such nodes will cooperate in forwarding the query-unicast packet from i to a. As an example, in Figure 3 node i sends a query-unicast packet to a.

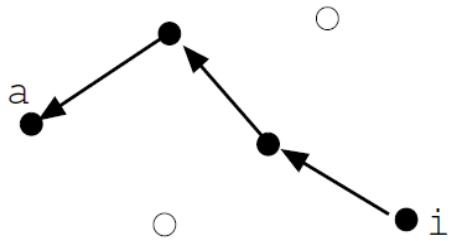

Figure 3: MAD-SADD execution -Discovery

g) Delivery Phase: Communication Setup

When an advertising node a receives a query-unicast packet q from an interested node i, q is immediately served, if a is not busy, otherwise q is stored in a queue Q queries. In order to serve a query-unicast packet coming from a node i, the first operation to carry out is the setup of the communication between i and a. This is done in the following way. Node a sends to i a data-info packet, containing the total length of M and the number of data packets which are needed to completely transmit M. Then, node i sends to a a data-info-ack message. All the intermediate nodes in the path between i and a simply forward these packets. Note that during this phase other queries unicast may be received by a. These queries are all enqueued in Q queries. When a receives the data-info-ack packet, a can start sending M. To this end, M is

divided into segments (data packets). Each of these data packets is then stored in a queue Q segments.

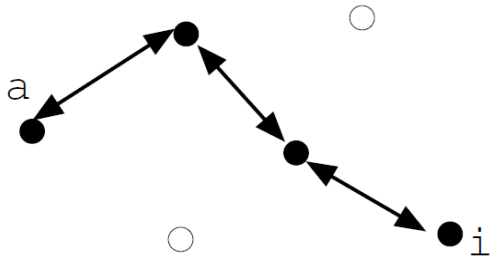

Figure 4: MAD-SADD execution -Delivery

h) Delivery Phase: Sending the Full Description

Once queue Q segments contains all the needed data packets, each one of them is sent to the requiring node i. Namely, for each data packet p in Q segments, asends p to i and waits for the ack for p from i. Once the ack packet is received, the next packet from Q segments is considered. When Q segments becomes empty, Q queries is checked again. If Q queries is not empty, i.e. if there are pending unicast queries for a, then node a extracts a new unicast query from Q queries and serves it as explained above. In this way, we force nodes to complete the transmission of full descriptions, before serving new unicast queries. As for the data-info and data-info-ack packets, all the intermediate nodes inthe path between I and a simply forward the data and ack packets. As an example, in Figure 4 node a sends to i a data-info packet, which is acknowledged by i with a data-info-ack packet. Finally, a sends the data packets to i. Each data packet is properly acknowledged by i. At this point, the unicast communication is finished and node i can consider new incoming short-description packets. we are concerned, no selfish misbehavior takes place during a unicast communication. Thus, as usual, we can use sequence numbers to avoid forwarding many times the same data packets.

i) Handling Link Failures

Since nodes are mobile, links may fail. For example, a node may leave the net- work or run out of energy. Note that link failures may cause problems only during the unicast transmissions that are during the discovery and delivery phases. If a node in the transmission chain is not able to deliver a packet (either data or ack) to the next hop, it notifies this link failure to the previous hop with a no-route packet, which will be sent back in the chain to the source. As usual, a receiving node uses a timeout to detect transmission failures. Furthermore, upon a failure, node i (i.e. the one that has received the interesting advertisement m) restores the previous status of the Bloom filter BF by using the values stored in the undo-vector during the discovery phase.

One may wonder why we need to restore the previous values of BF in case of communication failure. Suppose that we do not use the undo-vector of and we simply store new advertisements in BF. Then a node may never get the full description of an advertisement it is interested in. Here is how. Node a broadcasts its advertisement m. Node i receives m, stores it in BF and, finding m interesting, sends a query-unicast to a asking for a full description M of m. The unicast communication fails so i never gets M from a. After a while (depending on the advertisement frequency) node a will broadcast advertisement m again. Now i have m in BF so it will not check its interest for m and will just forward m. Thus, as long as m stays in BF, node I will never consider m anymore, so missing the full description of an advertisement i is actually interested in. For this reason, upon a failure, node i must erase m from its BF. This is achieved by using the undo-vector to restore the previous values of BF entries modified by storing m in BF.

j) Routing Protocol

While routing is not needed during broadcast communication, when unicast communication takes place we have to address the problem of reaching a specified destination. Our main contribution concerns the mechanism to thwart node selfish behavior, thus avoiding "advertising flooding". Thus, as for routing, we may use any of the many routing protocols available (e.g. AODV [15] and DSR [16]). However, by exploiting the advertising and discovery (unicast) phases of MAD- SADD we use here a sort of "lightweight" AODV during the delivery (unicast) phase. In fact, we can piggyback routing information in MAD-SADD packets thus avoiding the overhead of additional routing messages. Namely, our protocol extracts routing information during both the broadcast and the unicast phase. This is done by extracting the information contained in the headers of short-description and query-unicast packets in order to set up message routing. An example will clarify the matter.

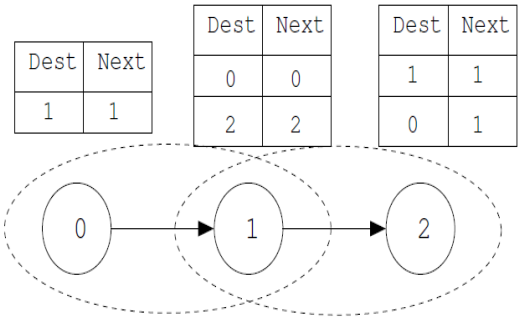

Figure 5: Routing: advertising Phase

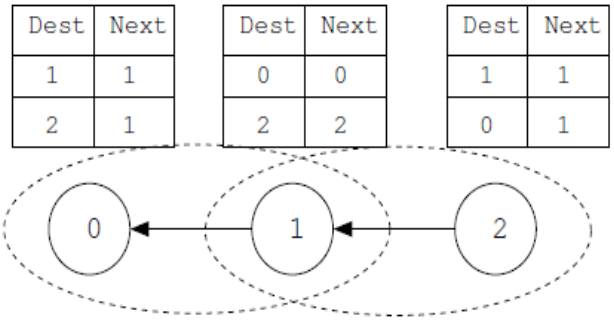

Figure 6: Routing: discovery phase

Consider the situation shown in Figure 5. Node 0 broadcasts its short-description packet, which can be received by node 1 only. Upon receiving the packet, node 1 updates its routing table adding a link to node 0. Then, node 1 broadcasts the same message. Since node 2 is reachable from node 1, node 2 can now receive the short-description packet and update its routing table as well. Note that Figure 5 shows the routing tables after 2 has forwarded the advertisement too. The routing tables of Figure 5 are partial and do not contain enough information to establish a bidirectional unicast tunnel between 0 and 2. In fact, whenever node 2 queries node 0 for detailed data, the latter is not able to reroute the requested information towards node 2, since the routing able in 0 does not contain 2 as a possible destination. This problem can be solved by considering also the information exchanged during the unicast phase. Namely, each node i receiving a query-unicast packet p extracts from p the source address s and the previous hop address h. Then, I adds to its routing table an entry with s as destination and h as next hop. For example, in Figure 6 node 0 adds the last entry (destination = 2, next hop =) by extracting this information from the query-unicast packet forwarded by 1.

REFERENCES

[1] Helal, S., Desai, N., Verma, V., Lee, C., Konark - a service discovery and delivery protocol for ad-hoc networks, WCNC 2003, vol. 3, (2003), 2107– 2113.

[2] Ratsimor, O., Chakraborty, D., Joshi, A., Finin, T., Allia: alliance-based service discovery for ad-hoc environments, WMC '02, ACM Press (2002), 1–9.

[3] Butty´an, L., Hubaux, J.P., Security and Cooperation inWireless Networks: Thwarting Malicious and Selfish Behavior in the Age of Ubiquitous Computing, Cambridge University Press, (2007)

[4] Li, H.C., Clement, A., Wong, E.L., Napper, J., Roy, I., Alvisi, L., Dahlin, M., Bar gossip, OSDI '06: USENIX Operating Systems Design and Implementation, USENIX Association (2006), 191–204.

[5] Thomas, G., Capacity of the wireless packet collision channel without feedback. IEEE Transactions on Information Theory 46(3), (2000), 1141–1144.

[6] Liu, A., Yu, H., Li, L., An energy-efficiency and collision-free mac protocol for wireless sensor networks, Proc. of Vehicular Technology Conference 2005, vol. 2, IEEE (2005), 1317–1322.

[7] Durresi, A., Zhang, P., Durresi, M., Barolli, L., Architecture for mobile heterogeneous multi domain networks, Mob. Inf. Syst., 6, 1, (2010), 49–63.

[8] Jin, X., Zhang, Y., Pan, Y., Zhou, Y., Zsbt, A novel algorithm for tracing dos attackers in manets, EURASIP Journal on Wireless Communications and Networking, vol. 2006, (2006), 1–9.

[9] Hu, Y.C., Johnson, D.B., Perrig, A., Sead: Secure efficient distance vector routing for mobile wireless ad hoc networks, In Ad Hoc Networks Journal, 1(1), (2003), 175–192.

[10] Papadimitratos, P., Haas, Z.J., Secure data transmission in mobile ad hoc networks, WiSe '03: Proceedings of the 2nd ACM workshop on Wireless security, ACM (2003), 41–50.

[11] Hu, Y.C., Perrig, A., Johnson, D.B., Ariadne: A secure on-demand routing protocol for ad hoc networks, Wireless Networks, 11(1-2), (2005), 21–38.

[12] Palmieri, F., Fiore, U., Castiglione, A., Automatic security assessment for next generation wireless mobile networks, Mob. Inf. Syst., 7, 3, (2011), 217–239.

[13] Hand, D., Mannila, H., Smyth, P., Principles of Data Mining, The MIT Press (2001)

[14] Helal, S., Desai, N., Verma, V., Lee, C., Konark - a service discovery and delivery protocol for ad-hoc networks, WCNC 2003, vol. 3, (2003), 2107– 2113.

[15] AODV. http://tools.ietf.org/html/rfc3561 (2007)

[16] DSR. http://tools.ietf.org/html/rfc4728 (2007)

[17] IEEE: IEEE standard 802.11. IEEE (1999)

[18] Aiyer, A.S., Alvisi, L., Clement, A., Dahlin, M., Martin, J.P., Porth, C., Bar fault tolerance for cooperative services, SOSP '05, (2005), 45–58. [38] Li, H.C., Clement, A., Wong, E.L., Napper, J., Roy, I., Alvisi, L., Dahlin, M., Bar gossip, OSDI '06: USENIX Operating Systems Design and Implementation, USENIX Association (2006), 191–204.

[19] Aad, I., Hubaux, J.P., Knightly, E., Impact of denial of service attacks on ad hoc networks, IEEE Transactions on Networking 16(4) (2008), 791–802.