DAMON: A Distributed Architecture for Monitoring Multi-hop Mobile Networks

Krishna N. Ramachandran, Elizabeth M. Belding-Royer, Kevin C. Almeroth
Department of Computer Science
University of California
Santa Barbara, CA 93106
{krishna, ebelding, almeroth}@cs.ucsb.edu

***Abstract*— With the advent of small form-factor devices, protocol standardization, and robust protocol implementations, multi-hop mobile networks are witnessing widespread deployment. The monitoring of such networks is crucial for their robust operation. To this end, this paper presents *DAMON*, a distributed system for monitoring multi-hop mobile networks. DAMON uses agents within the network to monitor network behavior and send collected measurements to data repositories. DAMON's generic architecture supports the monitoring of a wide range of protocol, device, and network parameters. Other key features of DAMON include seamless support for multiple repositories, auto-discovery of sinks by the agents, and resiliency of agents to repository failures. We have implemented DAMON agents that collect statistics on data traffic and the Ad hoc On-demand Distance Vector (AODV) routing protocol. We have used our implementation to monitor an ad hoc network at the 58th Internet Engineering Task Force (IETF) meeting held November 2003 in Minneapolis, MN. In this paper, we describe the architecture of DAMON and report on the performance of the IETF network using monitoring information collected by DAMON. Our network monitoring system is available online for use by other researchers.**

I. Introduction

Multi-hop mobile networks are currently experiencing widespread deployment. This increase in deployment is driven by advances in hardware and software technology, standardization of protocols, and the emergence of new applications. Examples of these applications include “last-mile” Internet delivery, search and rescue, home networking, and distributed gaming.

For the robust operation of mobile networks, it is crucial that monitoring complements their increasing deployment. Monitoring offers several benefits to network operators, system designers, and researchers. It can provide network operators with valuable insight into the state of the network, which can in turn help them understand the network's topology and usage. Monitoring can also enable operators to perform critical tasks, such as fault detection/isolation, necessary for the robust operation of the network. In addition, operators can use network monitoring to check for compliance of system implementations with set standards. Compliance checks are important because mobile networks are typically formed by users who carry devices with heterogenous hardware and software supplied by different vendors. System designers and researchers can use monitoring to improve protocols and systems through the analysis of collected network state. This state can also help designers develop realistic data traffic, user mobility, and wireless propagation models. Network simulators, such as NS-2 [1] and GloMoSim [2], can then apply these models to simulate real-world network behavior more accurately [3].

Monitoring multi-hop mobile networks, however, is more challenging than monitoring wired or single-hop wireless networks. This is because of characteristics unique to mobile networks, such as the lack of a hierarchical network structure, network-wide device mobility, and routing challenges. As a result, monitoring a network from a controlled network element — a well-applied strategy in wired and single-hop wireless networks — cannot be utilized for multi-hop mobile networks. Consequently, popular monitoring tools that are based on protocols such as the Simple Network Management Protocol (SNMP) and the Internet Control Message Protocol (ICMP) have limited utility in mobile networks.

To overcome the challenges of monitoring mobile networks, tools must be *tailored* specifically for such networks. Our goal is to address the need for such a tool with *DAMON*. DAMON is based on a Distributed Architecture for MONitoring mobile networks. Its generic architecture supports the monitoring of a wide range of protocol, device and network parameters. DAMON uses agents present within the network to monitor and send information to a distributed set of monitoring sinks or repositories that store the monitored information. Its key features include seamless support for multiple repositories, auto-discovery of sinks by agents, and resiliency of agents to repository failures. These features enable network operators to deploy, configure, and maintain DAMON with little effort.

We have implemented DAMON for the Linux and Microsoft Windows operating systems. Our DAMON implementation collects key statistics concerning data traffic and the Ad hoc On-Demand Distance Vector Routing (AODV) protocol [4]. To demonstrate its utility, we used our implementation to monitor an ad hoc network at the Internet Engineering Task Force (IETF) meeting held in Minneapolis, MN from the 9th to the 14th of November 2003. The information gathered using DAMON has given us valuable insight into the performance of the deployed network.

The rest of this paper is organized as follows. In Section II, we present our goals and design choices for DAMON and discuss various challenges in meeting those goals. Section III describes how DAMON overcomes those challenges and presents DAMON's design in detail. In Section IV, we discuss our implementation of DAMON. In Section V, we present observations of the IETF network performance using monitoring information collected by DAMON. In Section VI, we review related work, and we conclude the paper in Section VII.

II. Goals, Design Choices, and Challenges

Mobile networking is an area of rapid technological advances in system design and networking protocols. For a monitoring solution to cope with the pace of development, a generic architecture is required that is capable of monitoring any aspect of a deployed network. Our goal is to design such a generic architecture. In meeting this goal, a monitoring solution must address characteristics unique to such networks. These characteristics make the design of the monitoring solution challenging. In this section, we first describe our goals and design choices for DAMON and then describe the challenges of meeting these goals.

A. Goals and Design Choices

Our design of a generic architecture for monitoring mobile networks is guided by three design choices. These choices relate to the pervasiveness of a monitoring solution, the number of monitoring *sinks* or repositories, and the temporal property of monitoring information. We elaborate on each of the three choices below.

Pervasiveness of the Monitoring Solution: The *pervasiveness* of a monitoring solution is determined by the level of participation of nodes in the monitoring effort. At one end of a *pervasiveness spectrum* lies a non-pervasive technique where collection of network state is done by a controlled network element that is not an active participant in the network. At the other end is a pervasive technique in which specialized agents hosted by nodes in the network collect network state. As an example of the former, consider a single-hop wireless network deployed using an access point. By positioning a controlled network element close to the access point, it can collect network state by sniffing the traffic flowing in the wireless medium. Such a non-pervasive technique, if applied in a multi-hop mobile network, can result in the collection of information that leads to an inaccurate analysis of the network state. This is because there could be communication among nodes that is not heard by a controlled network element. Consequently, monitoring multi-hop mobile networks requires a pervasive solution, where participant nodes actively collect network state and deliver the collected information to a repository. Our goal, therefore, is to design DAMON as a pervasive monitoring solution.

The pervasiveness aspect of such a solution, however, can be minimized by installing agents at certain *fully-capable* nodes — nodes where device parameters such as energy, disk space, and processing power can be dedicated for monitoring. This strategy, called *limited coverage*, is in contrast to a *complete coverage* strategy where agents are installed at every node. The limited coverage strategy can be effective in some network configurations. For instance, consider a multi-hop network that provides Internet connectivity to a community. Because all the traffic is likely to flow through a pre-determined, non-mobile set of fully-capable routers, these routers can also function as monitoring agents. Our goal is to make DAMON generic enough to support both the limited coverage and complete coverage strategies.

Number of Sinks: The information collected by monitoring agents is sent to sinks that serve as repositories for collected information. Depending on the size of the network, the functionality of a sink can either be centralized (*single sink*) or distributed (*multiple sinks*). A centralized monitoring sink is suitable for a small network. In large networks, however, a centralized sink can result in poor spatial reuse of the wireless medium, congestion of routes to the sink, and excessive load at the sink. For such networks, the repository functionality can be distributed among several sinks that optimally are either non-mobile or less mobile compared to other nodes. The monitoring sinks can then use back-channels to aggregate their stored information.

As an example of such a distributed sink setup, consider a community multi-hop network deployment where routers are equipped with multiple wired/wireless interfaces. An example of such an architecture is the Transit Access Point (TAP) architecture [5]. In such an architecture, selected wireless routers can then function as monitoring sinks by using their back-channels for the aggregation of the collected information.

Our goal for DAMON is to take advantage of multiple sinks if they are available. In this way, DAMON avoids the drawbacks associated with a centralized sink setup.

Temporal Property of Monitoring Information: The temporal property of monitoring information is determined by the monitoring requirements. For example, consider a requirement such as tracking network topology in real-time. If topology information from the network is not delivered for processing in a timely manner, the resulting view of the network can be inaccurate. On the other hand, another requirement could be to obtain a log of all packets forwarded by a node without any constraints on time.

Based on its temporal property, information collected from the network can be classified into two types: *time dependent information* and *time independent information*. Our goal is to design DAMON such that it distinguishes between these two types. In this way, the information can be processed seperately. Higher delivery priority can be given to time dependent information as compared to time independent information.

B. Challenges

In realizing the goals described above, several challenges unique to mobile networks must be addressed. The various challenges are briefly discussed below.

Device mobility: With mobility, a route to a destination typically changes. This can result in a sink that is reachable becoming unreachable after movement. Moreover, mobility can cause transient breaks in a connection between an agent and a sink, thereby preventing the delivery of collected information.

Resource Constrained Devices: Participant devices in mobile networks are typically resource constrained. These devices are characterized by low processing power, limited disk space, and low energy. The allocation of limited resources for monitoring can result in poor system performance.

Fluctuating Link Qualities: The dynamic characteristics of a wireless link, such as multi-path fading and interference from the environment, can result in widely varying fluctuations in its quality. Link quality fluctuations are likely to result in routing path challenges, which in turn can lead to breaks in established connections between an agent and a sink. This can interfere with the delivery of monitoring information.

Short-lived Network Connections: An effect of mobility and fluctuating link qualities is that connections between an agent and a sink can be *short-lived*. Short-lived connections can prevent the delivery of monitoring information.

III. DAMON Design

This section describes the DAMON design. DAMON uses several techniques to overcome the challenges discussed in Section II. An overview of the techniques is presented first, followed by a more detailed description.

A. Overview

DAMON uses an agent-sink architecture for monitoring mobile networks. Its agents are hosted by participant nodes in the network and discover the presence of sinks automatically. Furthermore, agents are *resilient* to monitoring sink failures, i.e., after the failure of a monitoring sink, because of mobility, network congestion or the sink itself crashing, agents automatically switch their choice of the monitoring sink and send collected information to a different sink. The auto-discovery and resilient nature of agents are facilitated by periodic *beacons* initiated by sinks.

DAMON classifies monitoring information as time dependent information and time independent information. Time dependent information is packaged into *Time Dependent Digests* (TDDs) before delivery to a sink. Examples of TDDs include the energy left on the device or the identity of a node's neighbors. Because such TDDs are typically small in size, DAMON allows multiple TDDs to be aggregated. Intermediate agents on a path from the source of the TDD to a monitoring sink can aggregate their TDDs with the TDD being forwarded. On the other hand, time independent information (packet logs for example) is likely to be larger in size than TDDs. It is, therefore, packaged into small-sized, sequence-numbered "chunks" called *Time Independent Digests* (TIDs). DAMON supports the retransmission of TIDs by using their sequence numbers to keep track of which TIDs have been successfully delivered to a sink. TDDs and TIDs are delivered to the sink using routes discovered by the network routing protocol.

By supporting sink auto-discovery and resiliency to sink failures, DAMON overcomes challenges associated with device mobility. By supporting the retransmission of digests, DAMON overcomes challenges associated with failed information delivery because of mobility, fluctuating link qualities, and short-lived connections. To minimize the overhead on resource constrained devices, DAMON agents can be installed on fully-capable devices if they are present.

The remainder of this section describes the DAMON design in more detail. Presented next is a description of the sink auto-discovery process, followed by a discussion of the agent framework.

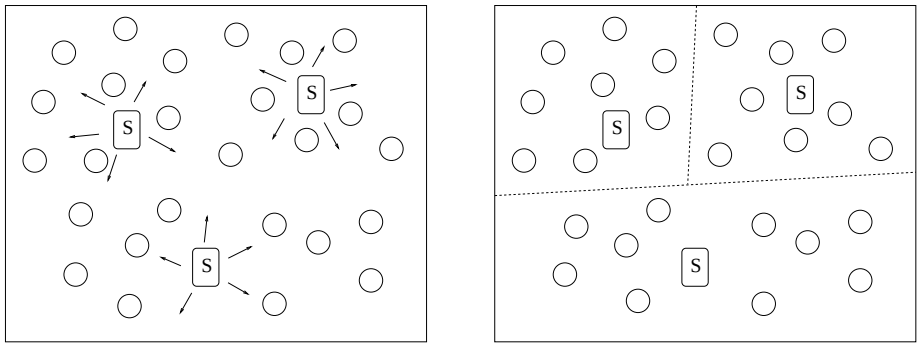
B. Sink Auto-discovery

The self starting and resilient nature of DAMON agents is facilitated by periodic beacons originated by monitoring sinks. These beacons, which are broadcast network-wide, serve to advertise the presence of a sink. Beacons also contain *agent-instructions*. Agent-instructions can be updated by the network operator to suit changes in monitoring requirements. This enables agents to automatically adapt to the new requirements.

The propagation of beacons is controlled by the agents. Upon receiving a beacon, an agent rebroadcasts the beacon in the following two cases: (1) the monitoring sink advertised in the beacon is the first sink learned by the agent, or (2) if the monitoring sink advertised in the beacon is the closest sink to the agent. Proximity to a monitoring sink is determined by using the *hop count* field carried in each beacon. The hop count gives the number of hops the beacon has traversed before reaching the agent. The proximity check performed by agent nodes prevents the needless flood of beacons in the presence of multiple monitoring sinks. After the proximity check is performed, the agent adds, in a table, an entry that corresponds to the sink advertised in the beacon. This table ranks all sinks the agent has learned based on proximity. The agent then chooses the first entry in its table as its *primary sink*. Each entry in the table has a lifetime associated with it, which is updated whenever a beacon is received. Lifetimes are used to identify and remove any stale entries in the table.

With mobility, proximity-based association may result in an uneven distribution of agents with sinks. We believe, however, that proximity based association is a simple and low overhead technique for associating agents with sinks. Figure 1 illustrates the sink auto-discovery process. In Figure 1(a), sinks broadcast beacons advertising their presence. Figure 1(b) illustrates the association of agents with its primary sinks.

In the sink auto-discovery process, the beaconing interval used by sinks affects how quickly agents discover sinks. There is a tradeoff between choosing a small beaconing interval for quick sink detection and the extra packet overhead due to frequent beacon flooding. Also, agents at the periphery of the transmission range of a sink or another agent can receive



(a) Sink Beaconing (b) Agent-Sink Associations

Fig. 1. Sink Auto-discovery.

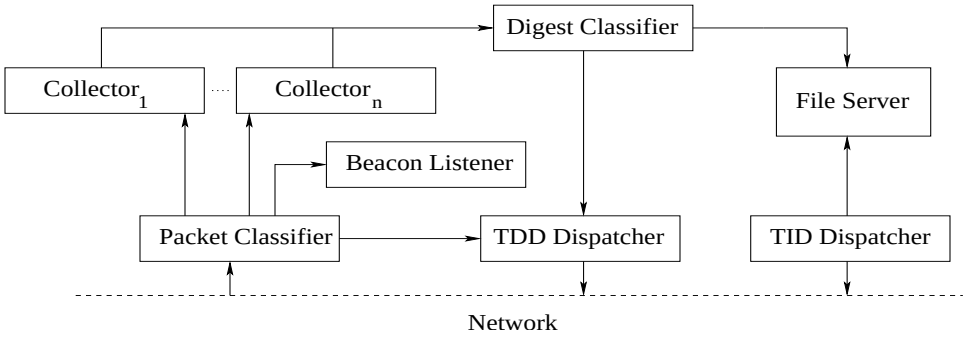


Fig. 2. Monitoring Agent Framework.

beacons intermittently. This can happen for example in areas called "gray-zones", where different packets are received with varying probabilities based on factors such as packet size and transmission rate [6]. Intermittent reception of beacons can result in unnecessary switching of the primary sink. To prevent this oscillation, an agent replaces its choice of the primary sink only when a predetermined number of successive beacons (three in our implementation) are successfully received.

C. Agent Framework

Figure 2 illustrates the architectural framework of the monitoring agent. When a packet relevant[1] to DAMON arrives from the network, it is first captured[2] and then delivered to the *Packet Classifier* module. The Packet Classifier categorizes packets based on their type and dispatches them to the appropriate *Packet Handlers*. Packet Handlers implement a specific operation on the received packet. For example, the *Beacon Listener* module handles sink beacons, whereas the *TDD Dispatcher* module handles TDDs received from other nodes. The TDD Dispatcher aggregates received TDDs with TDDs generated locally. The aggregation is simply the packaging of payloads from two different TDDs into the payload of a single TDD. Other proposed techniques for aggregation could also be applied [7], [8]. The aggregation operation is affected by criteria such as limits on digest size, the maximum transmission unit (MTU), as well as privacy and security. The *Collector* modules handle all other types of packets. A collector module implements a specific monitoring requirement such as summarization[3] of routing table information or link quality estimates in TDDs or TIDs.

Digests created by the Collector modules are classified by the *Digest Classifier* module. This module delivers digests either to the TDD Dispatcher for immediate delivery to the primary sink or to the *File Server* module. The File Server stores all digests on the local disk for later delivery to the primary sink. Digests stored by the File Server are then retrieved by the TID Dispatcher for transmission to the sink. The stored Digests are assigned monotonically increasing sequence numbers. The sequence numbers are used by the TID Dispatcher to track which digests have been successfully delivered to the sink. In addition to storing TIDs and assigning them sequence numbers, the file server ensures that the size of the stored digests does not exceed a certain limit to prevent disk overflow. Disk overflow can happen if an agent is unable to deliver digests to any sink because of sink failures. Overflow is prevented by using a simple First-In-First-Out (FIFO) policy to purge digests. This purging of digests can result in "gaps" in the monitoring information, which can make the interpretation of the collected information particularly challenging. To handle gaps, tools for the analysis of network state should be designed to minimize the impact of missing information.

[1]Relevance is dependent on the monitoring requirements.

[2]The capture of relevant packets is done by an implementation specific module on the host operating system that can receive network traffic.

[3]The format for summarizing information in digests is implementation specific.

IV. DAMON Implementation

Our goal for the DAMON implementation is to provide a monitoring solution for ad hoc networks utilizing the Ad hoc On-demand Distance Vector (AODV) routing protocol [4]. Specifically, we want our implementation to collect AODV and data traffic statistics to enable off-line analysis of network performance.

To aid the reader in understanding the implementation, an overview of AODV is presented first, followed by a detailed description of the DAMON implementation.

A. AODV Overview

AODV is an on-demand ad hoc routing protocol. For neighbor detection, AODV can use either broadcast HELLOs or link layer feedback. Route discovery is based on a route request, route reply cycle. Route discovery begins with a broadcast Route Request (RREQ) message. The RREQ contains the destination IP address for the requested route and the destination's last known sequence number. Destination sequence numbers in AODV are used to ensure loop-free operation. As the RREQ is propagated throughout the network, each intermediate node creates a *reverse route* entry towards the *originator* (source) of the RREQ. An intermediate node forwards only the first RREQ it receives from the originator. If the *destination-only* flag is set in the RREQ message, only the destination is allowed to issue a Route Reply (RREP). If the destination-only flag is not set in the RREQ, an intermediate node is allowed to issue an RREP provided it has an active route towards the destination.

The RREP message is unicast towards the source along the reverse route setup during RREQ propagation. As the RREP is propagated, intermediate nodes on the reverse route create a *forward route* entry for the destination node in their respective route tables. When an active route breaks, the node in the route that detects the break has the option of doing a *local repair* by finding another route towards the destination, or sending a Route Error (RERR) message towards the source to notify it of the break.

B. Implementation

We implemented DAMON using Perl. Our agent consists of two collector modules. The first collector, called the *AODV collector*, gathers AODV protocol information. This collector also provides topology data for tracking the network topology in real-time. The second collector, called the *data traffic collector*, gathers data traffic statistics. We implemented these two collectors for measuring metrics such as the packet delivery ratio, route discovery latency, and network throughput. Using such metrics, the performance of a network can be evaluated. More details about the two collectors are given below.

The AODV collector summarizes RREQ, RREP, RERR, and HELLO control messages in an *AODV-CONTROL* TID in the following manner. For each RREQ, RREP, and RERR that a node originates and receives, the collector records the UDP payload of the control packet and the time the packet was sent. HELLO payloads are not captured because neighbor connectivity information is recorded in the routing table snapshots. For each RREQ, RREP, and RERR that a node forwards and for each HELLO that a node receives, the collector increments a control-packet specific counter by one. Each minute, the counter values are stored by the collector in a *AODV-CONTROL* TID and then reset to zero. Additionally, the AODV collector records routing table updates (*deltas*), with the time the update occurs, in a *AODV-RT-TABLE* TID. The collector, in addition to creating TIDs, also creates TDDs called *AODV-NEIGHBOR* TDDs periodically (one minute in our implementation). The TDDs contain information about the node's neighbors and the quality of the link to each neighbor.

The traffic collector gathers traffic statistics from all data packets sent and received by the node. For each data packet, the collector records the IP source and destination fields, application protocol type, and packet size in a *DATA-STATISTICS* TID.

The two collectors package TIDs such that they do not exceed one megabyte each in size. This is done to prevent short-lived connections with a sink from preventing the delivery of digests. The agent periodically sends TIDs to the sink using TCP; in our implementation this occurs every ten minutes. TDDs, on the other hand, are sent using UDP. This is because the TDDs in our implementation do not require reliable delivery since they only carry neighbor information and are re-sent every 60 seconds.

V. Case Study: IETF Experiment

We used our DAMON implementation to monitor an ad hoc network deployed during the 58th Internet Engineering Task Force (IETF) meeting. The meeting was held in Minneapolis, MN from the 9th to the 14th of November 2003. The purpose of the network deployment was to enable the formation of a publicly accessible ad hoc network utilizing the AODV routing protocol in a heterogenous environment such as the IETF. The heterogenous environment was a result of meeting participants carrying hardware and software supplied by different vendors.

Our goals for the DAMON deployment were fourfold: (1) to validate the design and implementation of DAMON, (2) to track the topology of the IETF network in real-time, (3) to evaluate the performance of AODV in the IETF network, and (4) to observe realistic traffic and mobility patterns in the network.

In this section, we first describe the configuration of the IETF ad hoc network. We then discuss how we used information collected by DAMON to troubleshoot a network outage we encountered during the experiment. This section ends with a description of the traffic distribution in the ad hoc network as observed by DAMON.

A. Network Configuration

To enable attendees at the meeting to participate in the ad hoc network, we provided implementations of the AODV routing protocol for the Linux 2.4 and Microsoft Windows XP operating systems. For the IETF DAMON deployment, we adopted a complete coverage strategy to obtain the comprehensive state of the ad hoc

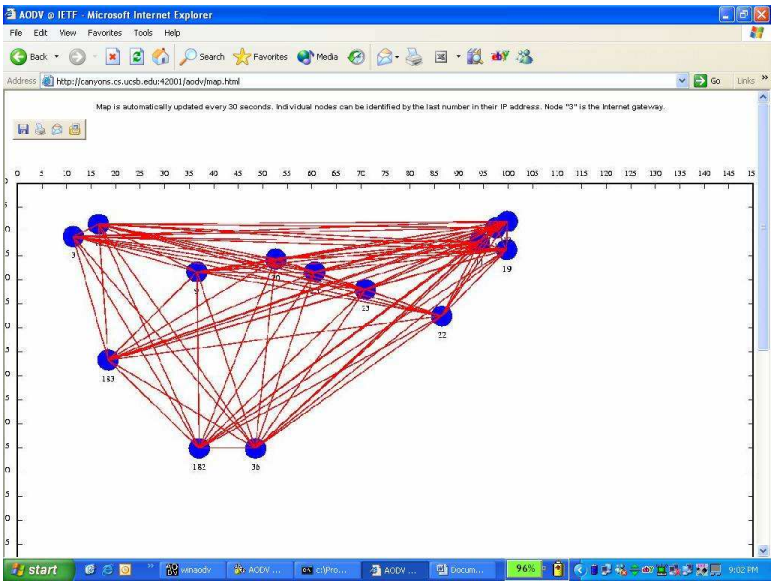


Fig. 3. Network topology at 10:00pm CST, 12th November 2003.

network. We, therefore, distributed the implementation of our DAMON agent along with the AODV implementation.

One node in the ad hoc network was configured to act as an Internet gateway. Hence nodes in the ad hoc network could obtain (possibly multi-hop) paths to the Internet. Internet connectivity was also provided at the conference via an IEEE 802.11b single-hop wireless network that consisted of 23 Cisco 1200 access points distributed on three floors of the meeting venue. The ad hoc network was co-located with these access points.

Because of the heterogenous nature of the ad hoc network, unidirectional links were a possibility. A unidirectional link is capable of transmitting packets in only one direction. Because AODV assumes that all links are bidirectional, unidirectional links have been shown to negatively impact the performance of a multi-hop network [9].

To prevent AODV from trying to utilize a path with one or more unidirectional links, each node ran a unidirectional link detector called *PUDL*. To avoid unidirectional links, PUDL estimates the reliability of each link between a node and its neighbors by periodically sending sequence numbered unicast probes in both directions on the link. It then measures the gap in the sequence numbers of received probes to estimate the link reliability. A link that is less reliable than a certain threshold (40% in our implementation) is classified as unidirectional. The AODV routing protocol then avoids any link classified as unidirectional.

Since we did not anticipate the formation of a large ad hoc network at the meeting, we decided to use only one sink in our DAMON deployment. Therefore, agent-to-sink associations were done using a static configuration in the agent implementation. The sink node was equipped with an additional network interface connected to a wired network. This interface connected the sink via the Internet to a server located at UC-Santa Barbara. This server was used as a backup repository for the information collected during the experiment. The server also used the AODV-NEIGHBOR TDDs sent by agents in the network to create a map of the network topology. This map was then made available in real-time on a web-site. Figure 3 shows one such map of the network topology taken as a screenshot on a Microsoft Windows machine that was a participant in the ad hoc network.

B. Troubleshooting Study

During one IETF session held between 13:00 - 15:30 CST on November 11th, nodes in the ad hoc network experienced intermittent connectivity with the Internet. Upon investigation, we noticed that PUDL was classifying a majority of each node's neighbors as unidirectional. Because neighbors were classified as unidirectional, stable routes to the Internet gateway were not formed, and consequently, connectivity with the gateway was unreliable.

To further understand the problem, we looked at the monitoring information collected by DAMON agents. Specifically, we focused on the period between 13:09 and 13:21 CST when there were seven nodes in the network. These seven nodes were co-located in the same conference room (an area of 18x30 square meters). Table I shows the percentage of AODV HELLO and PUDL Probe messages received by the agent running on the Internet gateway from nodes in the network during the 12 minute period. To calculate the percentages given in the table, we consider the periodicity with which HELLO and Probe packets are sent. The periodicity then lets us estimate the number of HELLO and Probe packets expected from a node in a given time period. The resulting estimates, along with the actual number of HELLO and Probe packets recorded by the agent, gives the percentages shown in the table.

Node ID	% Broadcast HELLO	% Unicast Probes
1	91.80	74.10
2	76.26	12.69
3	92.06	36.00
4	74.73	42.18
5	69.23	54.10
6	95.42	11.40
7	97.85	6.66

TABLE I
HELLO AND PUDL PROBES RECEIVED BY THE GATEWAY AGENT FROM EACH NEIGHBOR.

There are multiple observations that can be made from the data presented in the table. First, both broadcast and unicast messages suffer losses, and some nodes suffer significant losses compared to others. For example, only 69% of HELLOs from node 5 are recorded by the agent compared to 97% of HELLOs from node 7. Furthermore, a significantly higher percentage of broadcast messages are received than unicast messages. As an example, almost 98% of HELLOs from node 7 are received compared to only 6% of Probes. Finally, there is no correlation between the loss of HELLOs and Probes; i.e., predicting the loss of unicast messages by looking at the loss of broadcast messages, or vice-versa, can lead to inaccurate conclusions. For example, consider nodes 4 and 7. Node 4 delivers just 74% of its HELLOs compared to 97% by Node 7. However, 42% of Probes sent by Node 4 are received by the gateway compared to only 6% by Node 7.

One of the primary reasons for the high loss rate was because the nodes were operating in a heavily loaded (23 access points and several hundreds of users connected to those access points), and thus very noisy, environment. To maximize the network throughput, we had statically configured the AODV implementations to transmit data packets at the highest data rate of 11Mbps. High data rates, however, require a low bit error rate. The noisy and heavily loaded environment adversely affected the bit error rate. Because of this, the wireless hardware was unable to successfully deliver probes at 11Mbps.

Based on the above observations, we make the following conclusions:

- *Relying on thresholds to avoid unidirectional links can eliminate links that are necessary for connectivity*: We used the 40% threshold in various tests performed with our implementation before the IETF experiment. This threshold value, however, was not suitable for the noisy IETF environment. This resulted in poor network performance during the IETF session. Our experiences, therefore, suggest that a threshold scheme can result in satisfactory performance in one network environment, but can result in poor performance in a different environment. Consequently, alternate techniques for avoiding unidirectional links are needed.
- *Routing protocols should select routes based on how reliably a path delivers unicast packets*: Many current routing protocols use broadcast packets to discover routes. The above observations suggest that using a purely broadcast-based mechanism to discover routes can result in the selection of paths that poorly deliver unicast data packets. Routing protocols, therefore, should consider the reliability of unicast packet delivery during route discovery.

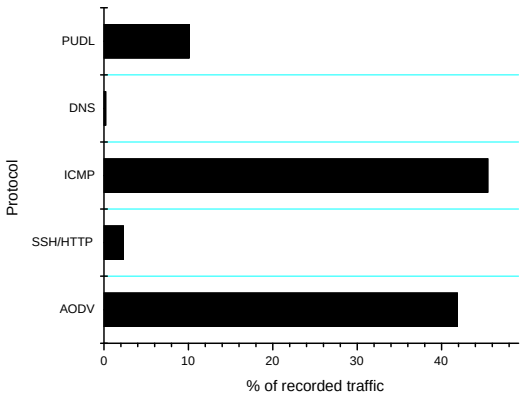


Fig. 4. Per-protocol Traffic Distribution with Unidirectional Link Filtering.

C. Traffic Distribution

In this section, we present our preliminary analysis of the traffic distribution during the IETF. Figure 4 shows per-protocol traffic distribution as monitored by an agent within the ad hoc network during the network outage previously described. The data presented in the figure is from a 50 minute log of monitoring information collected between 13:38 and 14:28 CST on November 11th. The figure shows that approximately 42% of all messages were AODV control packets. Almost 30% of the AODV control packets were HELLO messages; HELLOs made up approximately 13% of all monitored traffic. The remainder of the AODV control packets were RREQs (55%), RREPs (13%), and RERRs (2%). ICMP messages made up approximately 45% of the overall traffic. Of this percentage of ICMP messages, only 2.5% were ICMP Destination Unreachable packets. The rest of the ICMP messages were ICMP Request/Response packets. PUDL probes account for roughly 10% of the overall traffic. Because the node from which we collected this information was experiencing intermittent connectivity with the Internet gateway, only 2.3% of the overall traffic was TCP traffic carrying SSH/HTTP protocol packets.

Figure 5 shows the traffic distribution by protocol as monitored by an agent during a session held between 21:40 and 22:10 CST on 12th November 2003. For this session, we turned off link filtering at all nodes to prevent the elimination of links necessary for connectivity. The figure shows that approximately 66% of all packets were AODV control packets. Of all AODV packets, about 38% were HELLO packets. The remainder of the AODV control packets were RREQs (51%), RREPs (10%), and RERRs (1%). Approximately 33%

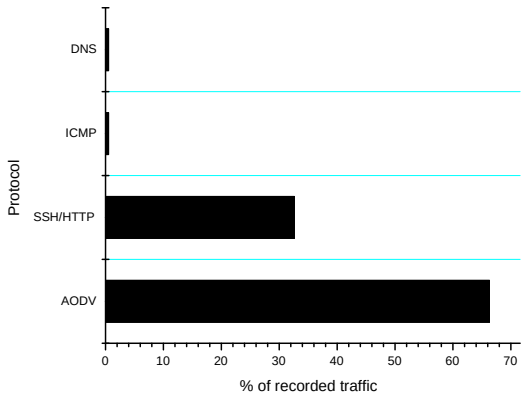


Fig. 5. Per-protocol Traffi c Distribution without Unidirectional Link Filtering.

of the overall traffic was composed of TCP traffic (SSH/HTTP protocol traffic).

Based on the traffic distributions, we can make three observations. First, the utilization of HELLO messages for neighbor detection can result in considerable traffic overhead. Alternate, low overhead techniques, such as detecting neighbors using link layer feedback, should be used. Second, the high number of RREQs recorded is because the ad hoc nodes were not configured with a default gateway setting. Consequently, each ad hoc node would issue a route request, even for external addresses, to which the gateway would subsequently send a route reply. In order to reduce the number of RREQs issued by a node, ad hoc nodes should send packets destined to external addresses to their default gateway instead of performing a route discovery. Third, user behavior can be predicted by the traffic distribution. For example, Figure 4 suggests that users use ICMP to check for network connectivity during a network outage, while Figure 5 suggests that users with connections to servers on the Internet do not use ICMP to check network connectivity. Based on such characterizations of user behavior and traffic distributions, it is feasible to create realistic user and traffic models that can enable network simulators model real-world network behavior more accurately.

VI. Related Work

A wide array of monitoring tools are available for wired networks. Early tools developed were *traceroute* and *ping*. Other, more recent tools make use of standardized management protocols such as the Simple Network Management Protocol (SNMP) to achieve sophisticated monitoring requirements. These tools can be classified into two main types: tools that rely on information from within the network, such as information collected from network routers (BGP state for example); and tools that rely on end-to-end monitoring to collect network state. Examples of the former are Rocketfuel [10] and MANTRA [11]. Example of the latter are ScriptRoute [12] and King [13]. Such tools have lead to several studies that give valuable insight into the performance of deployed protocols and networks [14], [15], [16].

In the area of monitoring infrastructured wireless networks, there is a general lack of monitoring tools that are available to the community. Some proprietary tools are supplied by access point vendors such as Cisco, Netgear, and Proxim. These tools are typically installed on the device itself and facilitate access to monitoring information via SNMP or HTTP. Nevertheless, numerous studies have analyzed the performance of such networks using these tools [17], [18], [19], [20].

In the area of sensor networks, other tools exist [7], [8] that come close to the functionality provided by DAMON. However, these tools are designed specifically for sensor networks. Furthermore, these tools do not have many of DAMON's features such as support of multiple sinks, sink auto-discovery, and resiliency to sink failures.

VII. Conclusion

The monitoring of mobile networks is crucial for their robust operation and should complement their increasing deployment. Monitoring offers several benefits to network operators, system designers, and researchers. The benefits range from the maintenance of deployed networks to the improvement of protocols and systems.

In this paper, we proposed DAMON, a monitoring system based on a distributed architecture for monitoring mobile networks. DAMON uses agents within the network to send collected information to monitoring sinks. Its key features include support for multiple sinks, sink auto-discovery, and the resiliency of agents to sink failures. These features enable network operators to deploy, configure, and maintain DAMON with little effort. We have implemented DAMON for the Microsoft Windows and Linux operating systems. We have made our implementation available to the community for research and deployment purposes[4]. We demonstrated DAMON's utility by reporting on the performance of an ad hoc network deployed at the 58th Internet Engineering Task Force meeting held in Minneapolis, MN.

As future work, we would like to develop a suite of analysis tools for interpreting information collected by DAMON. Also, our goal is use DAMON to monitor a large scale multi-hop network being deployed in the UC-Santa Barbara campus and at a student community on the outskirts of the campus.

Acknowledgment

This work is supported in part by an NSF Networking Research Testbeds (NRT) grant (ANI-0335302) and by Intel Corporation. We thank Intel for the development of the AODV Windows implementation.

References

[1] K. Fall and E. Varadhan. ns notes and documentation. In *http://www-mash.cs.berkeley.edu/ns/*, 1999.

[2] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks. In *Workshop on Parallel and Distributed Simulations*, Banff, Canada, May 1998.

[4]DAMON software: http://moment.cs.ucsb.edu/damon

[3] D. Kotz, C. Newport, R. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental Evaluation of Wireless Simulation Assumptions. In *ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Venice, Italy, October 2004.

[4] C. Perkins, E. Belding-Royer, and S. Das. Ad Hoc On-Demand Distance Vector (AODV) Routing. Internet Engineering Task Force (IETF), RFC 3561, July 2003.

[5] R. Karrer, A. Sabharwal, and E. Knightly. Enabling Large-scale Wireless Broadband: The Case for TAPs. In *Workshop on Hot Topics in Networks*, Cambridge, MA, November 2003.

[6] H. Lundgren, E. Nordstrom, and C. Tschudin. Coping with Communication Gray Zones in IEEE 802.11b based Ad Hoc Networks. In *ACM International Workshop on Wireless Mobile Multimedia*, Atlanta, GA, September 2002.

[7] C. Hsin and M. Liu. A Distributed Monitoring Mechanism for Wireless Sensor Networks. In *ACM Workshop on Wireless Security*, Atlanta, GA, September 2002.

[8] J. Zhao, R. Govindan, and D. Estrin. Computing Aggregates for Monitoring Wireless Sensor Networks. In *International Workshop on Sensor Net Protocols and Applications*, San Diego, CA, April 2003.

[9] M. K. Marina and S. R. Das. Routing Performance in the presence of Unidirectional Links in Multihop Wireless Networks. In *ACM International Symposium on Mobile Ad hoc networks*, Lausanne, Switzerland, June 2002.

[10] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *ACM Sigcomm*, Pittsburgh, PA, August 2002.

[11] P. Rajvaidya, K. Almeroth, and K. Claffy. A Scalable Architecture for Monitoring and Visualizing Multicast Statistics. In *IFIP/IEEE International Workshop on Distributed Systems: Operations & Management*, Austin, TX, December 2000.

[12] N. Spring, D. Wetherall, and T. Anderson. ScriptRoute: A Facility for Distributed Internet Measurement. In *USENIX Symposia on Internet Technologies and Systems*, Seattle, WA, March 2003.

[13] P. Gummadi, S. Saroiu, and S. Gribble. King: Estimating Latency between Arbitrary Internet End Hosts. In *ACM Internet Measurement Workshop*, Marseille, France, November 2002.

[14] N. Spring, R. Mahajan and T. Anderson. Quantifying the Causes of Path Inflation. In *ACM Sigcomm*, Karlsruhe, Germany, August 2003.

[15] P. Rajvaidya and K. Almeroth. Analysis of Routing Characteristics in the Multicast Infrastructure. In *IEEE Infocom*, San Fransisco, CA, April 2003.

[16] V. Paxson. End-to-end Routing Behavior in the Internet. In *ACM Sigcomm*, Palo Alto, CA, August 1996.

[17] A. Balachandran, G. Voelker, P. Bahl, and P. Rangan. Characterizing User Behavior and Network Performance in a Public Wireless LAN. In *ACM Sigmetrics*, Marina Del Ray, CA, June 2002.

[18] D. Kotz and K. Essien. Analysis of a Campus-wide Wireless Network. In *ACM International Conference on Mobile Computing and Networking*, Atlanta, GA, September 2002.

[19] D. Tang and M. Baker. Analysis of a Metropolitan-Area Wireless Network. In *ACM International Conference on Mobile Computing and Networking*, Seattle, WA, August 1999.

[20] D. Tang and M. Baker. Analysis of a Local-area Wireless Network. In *ACM International Conference on Mobile Computing and Networking*, Boston, MA, August 2000.