SAWAN: A Survivable Architecture for Wireless LANs

Mohit Virendra, Shambhu Upadhyaya, Vivek Kumar
Dept. of Computer Science and Engineering,
State University of New York at Buffalo, Buffalo, NY
{virendra, shambhu, vkumar2}@cse.buffalo.edu

Vishal Anand
Dept. of Computer Science
SUNY Brockport, Brockport, NY
vanand@brockport.edu

Abstract

This paper[1] describes survivability schemes against Access Point (AP) failures in Wireless LANs. It particularly aims for resiliency and survivability against multistage attacks where the adversary is successful in compromising the AP, and then targets the survived but more vulnerable network. This is true in real life where the adversary knows that survivability is a design consideration built into the network. It then performs a multistage targeted attack that is aimed at compromising the survived network that may have vulnerabilities. We first present a unique Infrastructure for an Ad-hoc Migration Scheme (IAMS) where the nodes under a failed AP form an ad-hoc network and reconnect to the network using available neighboring APs. We then present a scheme for isolating and removing any malicious nodes from the ad-hoc network routes in a transparent manner once the malicious nodes have been identified. This will minimize the chances of further attacks in the survived network, and the removal is done in a distributed fashion without the nodes exchanging any information between them. We report the results of our simulations performed using the network simulation tool GloMoSim.

Keywords: AP Failure, Multistage Attacks, Quality of Service, Robustness, Survivability, WLANs

1. Introduction

Wireless LANs (WLANs) have become increasingly popular in recent years with the number of commercially deployed WLANs running into hundreds of thousands these days. Cities in the Scandinavian countries and more recently major US cities like Philadelphia, PA, and Spokane, WA, have undertaken major efforts towards providing seamless ubiquitous wireless services to users by turning the entire city into a WiFi hotspot. The primary reason for this transition is the "cable free" nature of these networks, whereby the strategic installation of Access Points (APs) enables anyone with an off-the-shelf wireless device and a wireless card to "stay connected anywhere-anytime".

However, since the backbone of such wireless networks are the APs, their failure can disrupt network services by disconnecting the users from the network. Open air nature of the communication channel, limited bandwidth and physical placement limitations make the APs vulnerable to failures resulting from malicious attacks and Byzantine faults. The APs are especially vulnerable to attacks targeted at central authority in networks with centralized control, in which an adversary can bring down an entire network by compromising the central authority. An analogy drawn from the Internet could be that of attacks on the DNS servers.

It is essential that the network survives such failures with minimal penalty on QoS and minimal disruption of services. This paper presents survivability schemes against AP failures in WLANs. Under the proposed schemes, the nodes under a failed AP switch to ad-hoc mode upon not receiving any communication from the AP for a certain period of time. They then form an ad-hoc network and re-connect to the network by relaying their traffic through neighboring APs in the ad-hoc mode. To build in survivability, the ad-hoc routes and the routing schemes are pre-determined when the network is functioning normally in the infrastructure mode. We call this scheme an Infrastructure for an Ad-hoc Migration Scheme (IAMS). The paper further proposes schemes to subsequently isolate any compromised or malicious node(s) (that might have caused the AP failure in the first place) from the survived, but more vulnerable network in a transparent manner. It is important to minimize the chances of further attacks in the survived network, and is done in a

[1] Research supported in part by NSF Grant No. DUE-0417095

Proceedings of the Third IEEE International Workshop on Information Assurance (IWIA'05)
0-7695-2317-X/05 $20.00 © 2005 **IEEE**

distributed fashion without the nodes exchanging any information between them. Robustness in isolating the malicious nodes is particularly relevant in real life where the adversary can find out that survivability is built into the network. It performs a multistage targeted attack that is aimed at compromising the survived network that may have vulnerabilities. The idea is to build maximum possible resiliency specifically against multistage attacks and resulting failures.

1.1. Related Work

This section discusses related work in WLAN survivability and compares it with our approach.

In [16] Chen et al. describe a scheme for enhancing the connection dependability in WLANs by "tolerating" the existence of "shadow regions" through placement of redundant APs. Their work mainly focuses on communication between the primary AP and the redundant AP. They present the details of implementing redundancy by making enhancements to the basic 802.11 channel access protocol and demonstrate improvement in connection dependability. This scheme, though works well for improving dependability through redundancy, it deals with "connection" survivability when a user moves from one AP to the shadow regions. Our scheme is not based on redundancy and does not require shadow APs. It focuses on "network" survivability resulting from AP failures rather than per user connection survivability resulting from user mobility.

Snow et al. [17] describe reliability and survivability in the context of wireless networks. They describe an "outage index", and perform statistical evaluation of impact of outages. However, their work primarily focuses on proposing end to end connectivity schemes for hybrid cellular overlay networks. Our paper addresses AP failures in WLANs and does not consider an underlying cellular infrastructure.

Cisco Systems provide a WLAN security solution called LEAP [18]. It is an authentication type for WLANs that supports strong mutual authentication between the client and a RADIUS server using a logon password as the shared secret. It provides dynamic per-user, per-session encryption keys. While, providing stronger authentication would reduce the chances of a malicious node being able to compromise the network, it is not a comprehensive solution for dealing with AP failures. Survivability schemes are essential incase an AP fails (due to faults or attacks). Our paper also assumes the use of well known encryption schemes [6], [7], for key management and authentication (see Sec. 2.1.), but we deem these schemes as complementary to the additional survivability measures proposed in this paper (Sec. 2). Another important point worth mentioning is that the LEAP protocol is vendor specific and would work with Cisco products only.

1.2. Paper Organization

This paper is organized as follows: Section 2 presents motivation and overview of our approach. Section 3 describes the IAMS scheme. Section 4 describes the Route Discovery Algorithm using Topology Graphs (TGs). Section 5 presents the Bridge Node-Leader Association Algorithm. Section 6 describes a scheme for removing malicious node(s) from the route graphs of the survived network, post-AP failure route re-computation, and an associated Traffic Distribution Protocol (TDP) that is used for distributing traffic to adjacent APs using a load balancing scheme. Section 7 explains the use of existing 802.11 features that are utilized by our model for resiliency. Section 8 details the simulations performed and the results obtained. Finally we conclude the paper in Section 9 with a summary of its contributions, limitations and proposed future work.

2. Overview of the Approach

The primary focus of our paper is resiliency and survivability against AP failures in WLANs. AP failures could occur due to attacks targeted at the central authority and due to Byzantine failures. Under normal circumstances the nodes under a failed AP would lose network connectivity and this would lead to network failure. Assuming that through schemes described later in this paper, the network acquires some degree of resiliency in surviving AP failures, there could then be further attacks or failures in the survived, but more vulnerable network. We also describe schemes to isolate any compromised or malicious nodes in the network after surviving the AP failure, to make the network robust against subsequent disruptions. Thus, the aim of this paper is to provide an admissible degree of survivability and resiliency against such multistage attacks and failures. It is important to note that absolute survivability against all attacks and failures cannot be claimed through any known schemes and this paper also doesn't make any such claims. For example, if an adversary resorts to a crude form of Denial of Service (DOS) attack, by jamming all the channels in the network, then apart from detecting the adversary and physically stopping it, there is no known solution to survive such an attack. However, if the adversary plans to attack covertly without revealing its identity, thus not resorting to such a crude attack, then the schemes described in this paper

would be useful. These schemes would be especially useful in scenarios where the network survivability is of key importance due to the criticality of the operations being performed by the users, and the cost of network failure is prohibitively expensive. Example scenarios include military and corporate networks, and networks utilized for disaster management.

We first describe a new scheme called IAMS in which the nodes of the failed AP switch from infrastructure to ad-hoc mode to regain network connectivity after an AP failure. Such a transition to the ad-hoc mode is supported through built-in features of the 802.11 protocol [1], [2], [3], [19]. In our implementation, we define the following types of nodes to facilitate IAMS:

- ***Bridge Nodes:*** Nodes that are in the radio range of more than one AP. These nodes are associated with one AP but can receive signal from at-least one other AP. These nodes help in restoring network connectivity after an AP failure by relaying the traffic of the nodes under the failed AP to neighboring APs in ad-hoc mode. Bridge node identification through existing 802.11 features is discussed in detail in Sec. 7.
- ***Leader Nodes:*** These are nodes that act as distributed control heads in the network after an AP failure. Leader nodes are responsible for making routing decisions, isolation of any malicious nodes from the route graphs, and acting as control heads in IAMS after an AP failure in the network. The role of leaders is of primary importance in making the network survivable. Leader selection is discussed in Sec. 3 and leader election schemes dependent on the implementation are also given.

The leader nodes play a role in handling the nodes that turn malicious in the survived network. We assume that any malicious node that might have caused the AP failure, or is trying to disrupt services, or snoop for information in the survived network, is identified through some known detection schemes [4]. Malicious node detection is an extensive research area in itself [5], and is not the focus of this paper. Our aim is to now isolate the adversary from the network in a transparent manner so that it can cause no further harm in the survived network. Transparent removal of the malicious node(s) would involve removing it from the routes of the nodes in the ad-hoc network in a distributed fashion, without any information being exchanged between the nodes themselves. This secrecy and non-exchange of information is important to protect the integrity of the network from being compromised. This is important in the scenarios where:

- The malicious node merely wants to be present in the survived network and snoop for information. The idea all-along being to make the network more vulnerable rather than crashing it completely, so that eavesdropping for information becomes easier.
- The malicious node knows that survivability is built into the network, so it performs a multistage targeted attack that is aimed at compromising the survived network that may have vulnerabilities.

Malicious node removal is performed by the leaders (once the identity of the malicious or compromised nodes is established) where they are able to independently and transparently remove the malicious node from the ad-hoc route graphs of all the nodes in a distributed fashion without exchanging information between themselves. Removal of a node results in disconnected routes in the ad-hoc network, and route re-computation from all nodes to the bridge nodes becomes necessary. Route re-computation takes place independently at each leader node, such that all the leaders compute these routes without a potential conflict in the computed routes, and the leaders don't have to exchange any information in this process. This is done through the Leader-Bridge Node Re-association and the Route Re-computation Algorithms (described in Sec. 6). The TDP aids in routing the traffic between the nodes and the distribution system. The entire scheme is shown as a logical diagram in Fig. 1. In Fig. 1, the numbers inside the boxes represent the relevant sections where the schemes are described.

2.1. Assumptions

We assume the use of some well known key establishment and management schemes proposed by Zhu et al. [6], [7] for key establishment, reestablishment and management in the newly formed ad-hoc network. These schemes assume that the nodes are pre-loaded with some keying material which enables them to establish pair-wise keys with one hop neighbors and other nodes in the network on the fly. Thus we assume that nodes have a way of securely communicating with each other even when a malicious node is detected in the network, and compromise of a node does not lead to compromise of the keys between other nodes. New keys can be established and managed as and when old nodes are removed from the network.

For the sake of simplicity, we make the assumption that only one node is assumed to be malicious at any given time. We do not consider the case of collusion-based attacks [14] in this paper. The schemes proposed in this paper have been tested for the scenarios when nodes maybe detected to be malicious in succession. It would be important to test them against well known collusion and replication attacks [15], where a set of nodes collude together to attack the network, or replicate false identities to compromise the network.

Testing against such attacks is an important part of our continuing research and the impact of such attacks on our schemes is further discussed in Sec. 9.

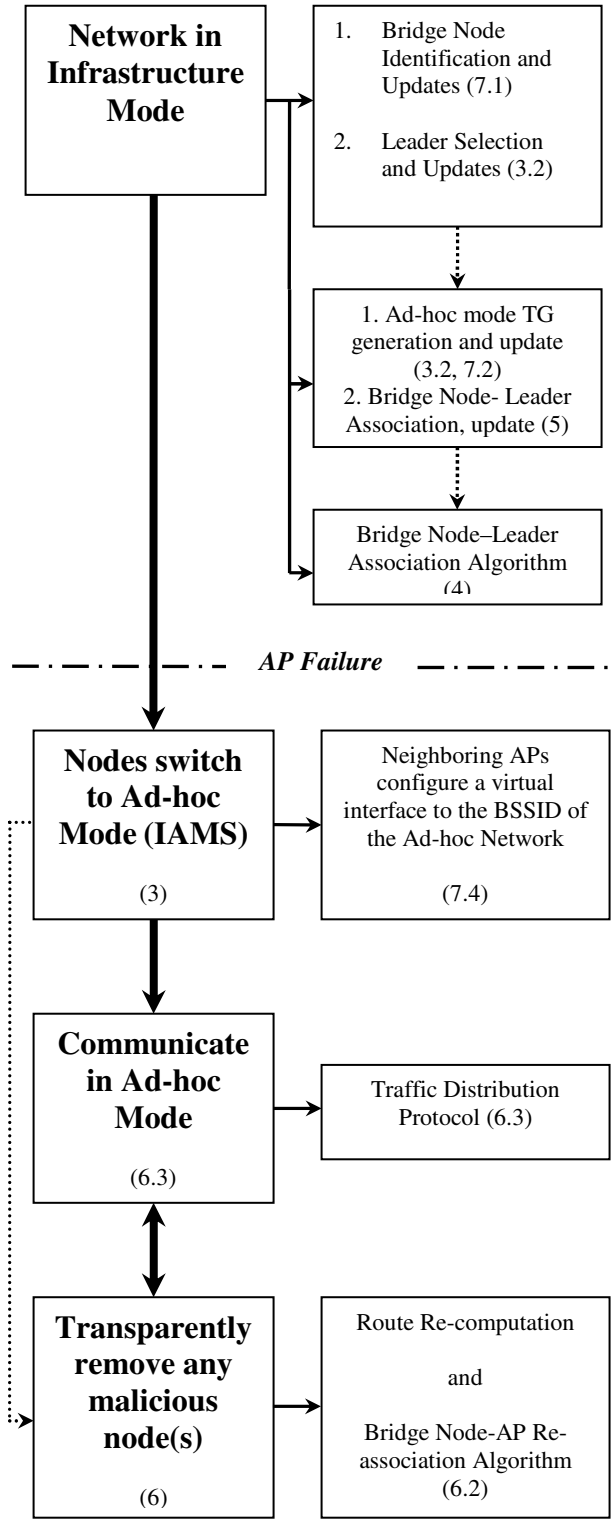

Fig. 1. Logical implementation

Most commercial implementations of wireless networks comprise of multiple APs. This is essential to provide continuous coverage in the area in which the network is deployed. It is very rare to find a wireless area network comprising of a single AP. Our survivability schemes are based on the assumption that there would be some nodes (at least one) in the network which are in the radio range of more than one AP. This is a realistic assumption and should be true for most commercial, educational and military networks since in such networks, the APs are positioned to provide continuous coverage to the users when they move within the network, and the node (user) density is high enough to support this assumption. We also assume that node mobility is low, and no new node is allowed to join the network after the AP has failed.

Another assumption that we make is that the AP is the logical extension of any dedicated monitoring device installed on the Distribution System. Such a device could be a dedicated server that may run any commercial monitoring or intrusion detection software. Thus we assume that any network monitoring and control related decisions are taken at the AP itself.

Our proposed solution is conceptually similar to the Integrated Cellular and Ad-hoc Relay (iCAR) [8] system for increasing the capacity of, and improving hotspots in Cellular Networks, but the solution methodology is unique, and is described below.

3. IAMS Description

The IAMS has three major components (a) Leader Selection Criteria (b) Topology Graphs, and (c) Switching methodology.

3.1. Leader Selection Criteria

Leader Selection criterion can be implementation dependent and application based. We suggest two schemes that would be applicable in different scenarios, and each has its own merits and demerits: (a) Trust based (b) Region based

a) **Trust based leader selection:** If the network is such that certain nodes (users) can be clearly identified as more trusted than others, then these high-trust nodes can be designated as leaders. Trust has often been used as a metric to designate nodes for special operations in networking and systems literature [9], [10], [2], [20], [21]. As an example, in a hierarchical organization, users that are higher in the hierarchy, or are otherwise entrusted with critical operations, could be designated as leaders. This scheme would have the advantage that

the probability of a leader turning malicious would be very low, but it might be difficult to find sufficient number of leaders for optimal network performance. This could potentially load the available leaders in the network severely, thus degrading their performance and creating bottlenecks in the network.

b) **Region based leader selection:** Leaders could be selected based on their proximity to other nodes, and their connectivity to bridge nodes. This could be optimally done to in a manner such that there is a homogeneous distribution of leaders across the network, and the leaders are in proximity to the bridge nodes assigned to them (see Sec. 5). This could lead to good traffic distribution across bridge nodes and good network throughput, but the probability of a leader turning malicious in such a scheme would be the same as the probability of any other node turning malicious. This might be critical from the security point of view, and may lead to increased overhead, if a leader needs to be removed from the network eventually.

Thus, based on the environment in which the network is deployed, and the applications it is being used for, a suitable leader selection scheme can be chosen.

3.2. Topology Graphs

The IAMS utilizes the TGs for determining the routes in the ad-hoc network. These graphs are constructed by the APs (or any dedicated server on the distribution system) when the network is functioning normally in the infrastructure mode. The TG is maintained as an adjacency list, and is a network graph of the nodes under an AP if the nodes form an ad-hoc network. This TG is used in the event of an AP failure for routing the nodes' traffic in ad-hoc mode. The TG is constructed and updated utilizing the existing features of 802.11, and this is explained in detail in Sec. 7.

An AP periodically sends the TG to the leader nodes under it when the network is functioning in the infrastructure mode (pre-failure mode). The TG is sent to only the leader nodes because this helps in:

- Reducing the bandwidth being used for control messages, and
- Keeping the TG secret from a potentially malicious node by minimizing the number of nodes that have this information (The scenario where a leader turns malicious is discussed in Sec. 6).

The AP also sends Topology Update information to the leaders by piggybacking it on the packets sent to the leaders. This information is just one bit: bit value 0 implies that there is no change in the TG since the TG was last sent, and 1 implies that the TG has changed since it was sent the last time. A 1 bit is immediately followed by the TG update packet. This is done to improve robustness while keeping the control overhead low.

In addition the AP periodically sends *leader identification* packets to the wireless nodes. These packets contain a list of all the leaders in the subnet. This would enable the nodes to identify leaders and subsequently, follow routing instructions from them in the ad-hoc mode.

3.3. Switching Methodology

The switching methodology itself is a three stage process:

1) If the nodes are in the infrastructure mode and they stop receiving beacon messages from the AP, they wait for a time period equal to n*T. Here T is the inter-beacon interval [1], and n is an integer. The value of n can be chosen at implementation time based on the conditions in the network. If the nodes do not receive any beacon signals till n*T, then they switch to ad-hoc mode. While switching to ad-hoc mode, they maintain the same Basic Services Set Identifier (BSSID) [1] (This is the BSSID that associated the nodes with the failed AP). Switching to ad-hoc mode from infrastructure mode is not a complex process and has been explained and utilized for congestion control in networks by Chen et al. [3].

2) The non-receipt of beacon messages for n*T time interval is assumed to be an indicator that the AP has stopped functioning. The leaders send out control packets to the nodes giving the details of the ad-hoc mode routing. This information includes routes and the percentage of traffic that each node should send along a particular route (computed using TDP). It is important to mention again that we assume the use of well known encryption schemes for ad-hoc networks [6], [7]. Thus each node can set up pair-wise keys with every other connected node in the network, and the leaders can send the routing information securely to the nodes. The leaders can also update this information through update packets, if there are any subsequent changes due to the removal of a malicious node.

3) On receiving the routing information, the nodes start routing their packets according to the instructions received from the leaders. Here we assume that all non-malicious nodes co-operate in routing each others traffic. This is a valid assumption because ad-hoc routing is based on cooperation between the nodes and any malicious node would be detected and removed

from the route graphs. Thus the remaining nodes would adopt a best-effort approach in routing other's traffic.

4. Route Discovery Algorithm using Topology Graphs

Route computation at the AP employs a modified Prim's Algorithm [11]. This algorithm, called Route Discovery Algorithm, is illustrated in Fig. 2. The set S represents the set of colored nodes. The algorithm computes a minimum spanning tree rooted at the bridge node. However, the tree might not include all vertices in the TG, in case there is a disconnected subgraph. The weight of an edge between u and v, represented by w(u,v) is unity for our model. This results in the shortest path between that bridge node and every other connected node in the graph except other bridge nodes. The algorithm uses a coloring and numbering scheme and assigns a unique color to each bridge node. In the numbering scheme, the tens digit of the nodes numbering label represents the number of hops that the given node is away from the bridge node, and the units digit represents the order in which the node was found. For example, if a node is 3 hops away from a bridge node and it is the 4th node discovered that is 3 hops away from the bridge node, then it is numbered 34. For simplicity, we don't show the unit's digit generation in the algorithm in Fig. 2. In Fig. 2 TensLabel(u) is used to extract the tens digit from the number label of node u. If the algorithm finds a node that was previously colored and assigned a number, but has found a new lower number for it in the same color, it replaces the older higher number with the newer lower number. If the node visited is another bridge node, it is left uncolored and unnumbered, as it's not useful to find paths between two bridge nodes. The algorithm continues in this fashion till it can find no more uncolored nodes. If there is no route from a bridge node to a given node, then that node is not colored by the bridge node's color. This is the working of the algorithm to find shortest paths from *one* bridge node to all other connected non-bridge nodes.

Similarly, the algorithm computes independent spanning trees for each bridge node, and associates a unique color with every bridge node. So, the end result is that nodes end up having multiple colors and a unique number associated with every color. For example, a node might have the color map, say, [Blue 23; Green 44; Red 71], which means that the node is 2, 4 and 7 hops away from the Blue, Green and Red colored bridge nodes respectively.

```
Topology  G=(E,V)
Weight matrix W = ( w_{u,v} , link weight between node u and v)
 B← { Bridge Node}
 S←Φ

procedure ComputeRoute(G,B,S)
{
vertex u, v;
V' = V – {OtherBridgeNodes}
while (B – S != Φ)
{
  for each u ε B – S
{  for each v in Adj(u)  //Adj(u) checks adjacency in V'
          {      if ( TensLabel(v) > TensLabel(u) + w(u ,v)
                   {  route(v)← u
                     TensLabel(v) ← TensLabel(u) + w(u,v)
                   }
         B← B U {v}
          }
          S← S U {u}
}}}
```

Fig. 2. Route discovery algorithm

5. Bridge Node – Leader Association Algorithm

As mentioned earlier, certain nodes are designated as leaders. The AP associates each leader with a unique set of bridge nodes. It is the responsibility of each leader to re-compute paths (if required) and route traffic to its assigned bridge nodes in the ad-hoc mode. The AP computes the ratio of the number of bridge nodes to the number of leaders (B/L). This ratio is useful for determining which leaders are responsible for routing traffic to which bridge nodes. The AP has a set of leaders and a set of colors, both of which are ordered. For example, there are eight bridge nodes in the subnet with assigned colors C = {black, blue, green, indigo, orange, red, violet and yellow}. The AP has identified three leader nodes L = {L1, L2, L3}. In this case

$P = |B| / |L| = 8/3 = 2.666..$, and

$Q = \lceil |B| / |L| \rceil = 3$

P is the number of bridge nodes each leader should be assigned. If P is not an integer, then value Q is computed. If P>1, then the leader with the lowest number (L1) picks the first Q colors from the ordered set of colors. So,

L1 = {black, blue, green}

Starting at the second leader (L2), each leader successively picks up the next (Q-1) colors from the set of unassigned colors. In this case,

L2 = {indigo, orange}
L3 = {red, violet}
If, |B| > [(|L| – 1)*(Q – 1) + Q], i.e., if some colors still remain unassigned at the end of this step, then the algorithm successively assigns one color (from the set of remaining
[|B| – (|L| – 1)*(Q – 1) + Q] colors) to each leader in order until there are no unassigned colors. This process begins at the second leader (L2). Now,
L2 = {indigo, orange, yellow}

If 0<P< 1, then the first B leaders, are each assigned one color in order. If P is an integer greater than 1, then L1 picks up first P colors, L2 picks up the next P colors, and so on. It is required that a leader must itself be colored by all the colors assigned to it. So, the algorithm then makes a pass to check if each leader is itself colored by all colors that have been assigned to it. If it detects an assignment such that the leader itself does not have that color, it removes that color assignment from the leader, and puts it in the unassigned colors set. Note that the unassigned color set would still be ordered. It then sequentially picks up each color from the unassigned set, and assigns it to the first leader having that color in the leader set. This scheme ensures that each bridge node has an assigned leader and no bridge nodes are left out.

So each bridge node is now associated with some leader. The AP sends the bridge node information, leader-bridge nodes association lists and the route discovery tables to the leaders periodically, in addition to the TG through update packets. Thus, all the leaders have consistent and updated information at any given time and this is essential for the correct working of the TDP for routing traffic in the post AP-failure scenario.

6. Post AP Failure Route Re-computation

If an AP has failed because of an attack from one or more than one nodes, or if some nodes are discovered to be malicious when the network has switched to ad-hoc mode (AP has failed due to Byzantine faults), then we assume that the identity of such nodes is established through some well known detection schemes. The malicious node(s) could belong to the following four categories:

1) Outsider Node: This is an outside node that was never a part of the network (never associated with the AP). Hence, this node is not a part of the pre-computed ad-hoc mode route graphs. Thus no new route discovery mechanisms are required in the ad-hoc mode.
2) Insider Node (non-leader and non-bridge node): This node was a part of the network. Hence, it would be in the pre-computed ad-hoc mode route graphs.
3) Insider Node (leader): This node had previously been designated as a leader through schemes mentioned earlier. This node was responsible for routing decisions of its connected nodes and had some bridge nodes assigned to it. Hence, bridge node reassignment and some more corrective actions are required.
4) Insider Node (bridge node): This node was given the responsibility of routing part of the ad-hoc network traffic to adjacent APs. Now, no further traffic can be routed through it. So, the leader responsible for this bridge node simply drops the bridge node's color from its list of colors. No traffic will be routed to this bridge node by its assigned leader.

The second and third scenarios require route re-computation operations to be performed in the ad-hoc mode. Some ad-hoc mode routes would be passing through the malicious or compromised node(s), and in the worst case scenario this node could be a leader responsible for routing traffic to some bridge nodes. So, route re-computation mechanisms are required for these scenarios. Route re-computation schemes are described in Sec. 6.2.

6.1. Impact of Misdiagnosis (False Positives Scenario)

There could also be a scenario where the employed Detection Systems could misdiagnose a node to be malicious, even when the AP is still functioning. For example, the node could be using excessive bandwidth with a non-malicious intent due to certain unforeseen operations. Depending on the severity or impact of the node's behavior deviation from normal, the AP itself could either just take precautionary actions by alerting the leaders, without actually asking the nodes to migrate to ad-hoc mode; or, if the impact of the node's behavior is severe, the AP could ask all the nodes to migrate to ad-hoc mode. However, the impact of misdiagnosis is minimal in both the cases. Only the suspect node is isolated from the network and the rest of the network survives. In the first case, where there is no migration, the rest of the network continues to function normally. In the second case, even when the nodes migrate to ad-hoc mode, there is minimal penalty on the throughput of most of the nodes. It is shown by our simulations in Sec. 8 that for non-bridge nodes, the throughput falls to 82% of its original value, which is a small penalty to ensure network survivability. The misdiagnosed node can later be integrated back into the network.

6.2. Route Re-computation Algorithm

In each of the scenarios (2) and (3), the route re-computation algorithms work on the principle that all the leaders know the identity of the malicious node, and they have the route graphs for all colors that they have been assigned to, as well as route graphs corresponding to all other leaders and their assigned bridge nodes. An important point to note here is that during this re-computation phase, the leaders are assumed to have no means of communicating with the AP or with each other. So, each leader has to independently and uniquely re-compute routes for its assigned colors such that there are no potential routing conflicts with routes computed by other leaders when the actual routing takes place later.

In scenario (2), this is easy to ensure since there is no reassignment of bridge nodes (colors) to leaders. Each leader deletes the malicious node and all edges incident on it in the route graphs of its assigned colors. Let m = maximum number of hops in the route graph of a certain color C[j]. Thus, the node farthest away from the bridge node (number 00) is at level m (its tens digit is m). Let n = level of malicious node. The algorithm is shown in Fig.3.

This algorithm first constructs a set of all nodes at level n-1, removes the malicious node from graph, then re-computes paths from nodes at level n-1 to nodes at level m, using the Route Discovery Algorithm described in Sec. 4. Since our links are uniformly weighted to unity, our algorithm always results in the shortest path from level n-1.It does not need to re-compute routes from the bridge node to the nodes at level n-1. The unit's digit in the number label of nodes affected by the recomputed routes is updated during this re-computation.

```
procedure RecomputeRoutes(Topology G,
MaliciousNode x, level n)
{
V = V – {x}
W ← { v ε V': Level(v) == n-1};
S ← {v ε V': Level(v)  < n-1};
ComputeRoute(G,W,S);
}

procedure Level(vertex v)
{
 return TensLabel(v);
}
```

Fig. 3. Route re-computation algorithm

In scenario (3), the other leaders will know that the malicious node was a leader, and they'll also know which colors it was responsible for. Now these colors need to be re-distributed among other leaders, without the leaders exchanging any information between themselves or the AP. So, using a round robin scheme, each leader in its numerical order, picks up one color at a time from the ordered set of remaining colors, until there are no unassigned colors. Before picking up the color, the leader checks to see if it is colored by that color. If not, then it moves to the next color in the ordered set. Each leader can perform this computation by itself, so each leader is able to uniquely identify the additional bridge nodes that it has to take care of. It then performs the entire route re-computation from that bridge node to all other connected nodes using the Route Discovery Algorithm.

In the event of a graph getting partitioned into disjoint subgraphs, the leaders in each subgraph will only re-compute routes from their remaining assigned bridge nodes to the other nodes in the subgraph. The scenario where a graph gets disjoint such that there are bridge nodes in one subgraph, but no leaders is a focus of our ongoing research.

6.3. The Traffic Distribution Protocol

This protocol is used for the actual routing of traffic using a coarse grain load balancing scheme, when the nodes have switched to ad-hoc mode. After performing route re-discovery, a leader sends the updated routing table corresponding to each color it is assigned, to all nodes having that color. This table is sent along the routes the leader has just re-computed. So, each node receives one route graph corresponding to each color that it has. In addition, the leader sends these tables to all other leaders in the subnet.

Before sending out packets, the nodes check for the destination address. If the destination address is a node within the subnet, then any well known existing scheme for ad-hoc networks like Dynamic Source Routing (DSR) [12] can be used for sending this traffic. If the destination address is a node outside the subnet, then the nodes use the load-balanced TDP. Consider a node that has a color map consisting of n colors i.e., the node has paths to n bridge nodes. Let h_1, h_2, h_3…, $h_{i,\ldots,}$ $h_{n,}$ be the number of hops along routes to bridge nodes 1 to n. The traffic from a source node is distributed in inverse proportion to the number of hops along all its routes: Highest volume of traffic is sent along the route with fewest hops. Fraction of traffic sent along a route i (Ti) is computed as:

$Ti = \{ [(h_1h_2h_{3..} h_{i...}h_n) / (h_2h_{3..} h_{i..} h_n + h_1h_{3..} h_{i...}h_n +...+ h_1h_2h_{3..} h_{i...}h_{n-1})] * (1/ h_i) \}$

For example, consider a node with the color map [Blue 23; Green 34; Red 41]. This node is connected to three bridge nodes and its distance in number of hops from these bridge nodes is 2, 3 and 4, respectively. So, it sends $12/26^{th}$ fraction of its traffic to the Blue bridge node, $8/26^{th}$ fraction to the Green bridge node and $6/26^{th}$ fraction to the Red bridge node.

The traffic is source routed in this scheme. A node knows the complete route to each bridge node it is connected to, and also the proportion of traffic that is to be sent along each link. It includes the entire route in the header of each packet it sends. When a packet reaches a bridge node, it sends it to the AP its communicating with. For an incoming packet from an AP, the AP adds the source route to the packet header and forwards it to the bridge node.

7. Implementation Through Existing 802.11 Features

Each AP with its connected wireless nodes forms an *Infrastructure* Basic Service Set (Infrastructure BSS or wireless subnet) [1]. Our implementation utilizes the existing features of the 802.11 in a three-fold manner to (a) Identify bridge nodes, (b) Help the AP in constructing the Network TG and (c) Facilitate a bridge node to communicate with a neighboring AP in ad-hoc mode.

7.1. Bridge Node Identification

In 802.11, new nodes perform *Scanning* [1] to discover the existing wireless networks in an area. Scanning can be both *active* (transmission of Probe Request frames to identify networks in the area), or *passive* (listening for beacon frames). A *Scan Report* is generated at the conclusion of a scan listing all the BSSs that the scan discovered and their parameters. In our implementation, a new wireless node that is attempting to join a network buffers its Scan Report until after it has associated with an AP. When the node finally associates and authenticates with an AP, it sends its Scan Report to the AP. The AP upon receiving this report is able to identify if this node is in the radio range of other neighboring APs. This is how the AP identifies the bridge nodes in the Infrastructure BSS.

7.2. Network Topology Graph Construction

If a new node performs active scanning and sends out probe frames by using broadcast BSSID [1], these frames are also received by other wireless nodes already present in the network and in the sender's radio range. These frames pass through any filtering at the MAC. In our scheme, the recipient nodes forward these frames to their APs instead of dropping them. When the new node associates with an AP, that AP uses frames forwarded to it by nodes in its BSS, to determine which nodes are in the radio range of the newly admitted node. The AP utilizes this neighbor information to construct the TG of the nodes in its BSS.

If the new node performs passive scan and just listens to periodic beacons, then the AP mandates it to broadcast "neighbor discovery" packets upon joining the network. No response is required for these packets. These neighbor discovery packets are again forwarded to the AP and used for constructing the Network TG. The AP utilizes the TG to compute routes from each node in the BSS to the bridge nodes belonging to that BSS. These routes are used for restoring network connectivity of these nodes through the IAMS in ad-hoc mode in the event of an AP failure.

The advantage of making the neighbor nodes send either the probe frames or the neighbor discovery packets is, that this scheme ensures the credibility of the information conveyed by the existing nodes of the network. These broadcasts (probe frames or neighbor discovery packets) by the new node are received directly by the AP also. When the nodes send these packets to the AP, it compares the packets directly received from the new node with the packets forwarded to it by the other nodes. A malicious node cannot give false information to the AP regarding the nodes in its radio range; hence the AP is able to construct an accurate TG of the subnet. The AP continuously modifies this graph as nodes join or leave the subnet.

7.3. Updating Topology Graphs and Bridge Node Information

We assume that node mobility is low, and no new node is allowed to join the network after the AP has failed. The AP keeps polling the nodes periodically to check if they're still associated with it. In addition, a node's response to these AP poll packets is again broadcast BSSID, so that other neighbor nodes in the radio range again receive and forward these packets to the AP. This helps in keeping the Network TG updated with regards to node mobility. The nodes in the network, especially those that have been designated as bridge nodes, are also required to do a periodic Scanning and generate Scan Reports listing all the available APs in the area from which they can receive

signals. These Scan Reports are sent to the AP and this helps in keeping the Bridge Node information current and up to date at the AP.

7.4. Bridge Node – Neighboring AP Association

The 802.11 specification explicitly mentions that a wireless node can associate with only one network at a time, i.e., a node can have only one BSSID at any given time [1]. Our implementation requires the bridge nodes to be able to communicate with both, the nodes in the ad-hoc network as well as with the neighboring APs.

Most enterprise grade APs offer some support for Virtual LANs (VLANs). These VLANs are implemented through multiple virtual interfaces in the AP [1]. For example, the Cisco Aironet 350 has 16 virtual wireless interfaces. In our proposed solution, when an AP fails, this is detected through the logic built into the distribution system. Upon detection, the neighboring APs configure one of their virtual wireless interfaces to the BSSID of the newly formed ad-hoc network. The BSSID of the ad-hoc network is the same as that of the failed AP: the nodes have just switched to ad-hoc mode, keeping the same BSSID. Thus, these APs would now be able to communicate with the bridge nodes in the ad-hoc mode on that interface, and the bridge nodes see these APs as any other nodes in the ad-hoc network.

8. Simulation and Results

In this section we present the results of our simulations, which we performed in GloMoSim [13] to evaluate the efficiency of our IAMS. We consider a terrain to be a square region of size 300 meters by 300 meters, with a grid unit size of 100 meters. Further, we consider that each node has a transmission range of 100 meters. Below we report the results of one such simulation scenario having a total of 17 nodes, numbered from 0 to 16. As shown in Fig. 4, the nodes numbered 0, 1, 2 and 3 make up the 4 APs, nodes 4 and 5 are the leaders, and nodes 6, 7 and 8 are the bridge nodes. The client's nodes are clustered around the AP numbered 1. We assume that when the network is up and running (in the no attack state), each node may send traffic at a constant bit rate in the range 80Kbps to any of the 10 other nodes in the network. We now assume that AP 1 (i.e., the node numbered 1) comes under attack from the malicious node numbered 10 at 50 unit time. Our simulation results show that during 50 to 400 time units the throughput of the network fell to 37.3%. The rate of decline is observed to be the maximum just after the AP goes down (before vertical line in Fig. 5 at 100 unit time). This is the period before the alarms reach the leaders and the migration can be initiated. The continued decline is explained by the fact that it takes some time for the network (and associated routes to be recomputed after removing all links to the malicious node) to stabilize after the switch to the ad-hoc mode. Also the control messages sent by the respective leaders take time to propagate to all the nodes through the network. Inconsistent states of intermediate nodes before the control messages are received might result in temporary disconnected topology sub-graphs, which in turn cause packet drops while the node states are in flux. Once the nodes have reconfigured they start forwarding packets. This can be seen in Fig. 5 as the gradual increase in goodput, which then stabilizes at about 70% of its original observed value. In Fig. 6 we plot the ratio of actual data generated by a bridge node and the total traffic the bridge node transmits (including forwarded packets). This illustrates the congestion experienced at the bridge node due to the necessity of forwarding packets for other nodes. The bridge node's own traffic falls to a small 28% of the total traffic transmitted by itself, this though a large drain on its resources is required to maintain network connectivity and is key to ensure survivability.

In Fig. 7 we plot the ratio of actual data generated by normal client node and the total traffic transmitted by such a node. We observe that after the switch to ad-hoc mode, the node's self traffic falls to nearly 82% of the total traffic transmitted by it. This illustrates the fact that even under attack our protocol incurs a small penalty on majority of the participating nodes but still achieves a graceful migration. We do not report results from other simulations with different number of nodes, however, we note that similar results were obtained in all cases, thus demonstrating the feasibility of our IAMS.

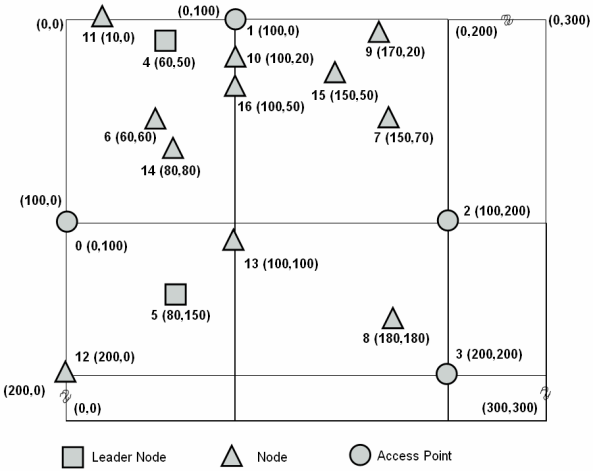

Fig. 4. Simulation topology

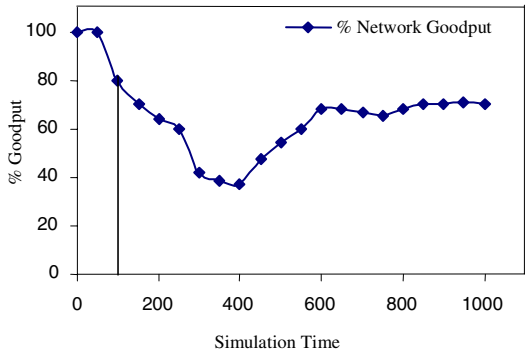

Fig. 5. Network goodput

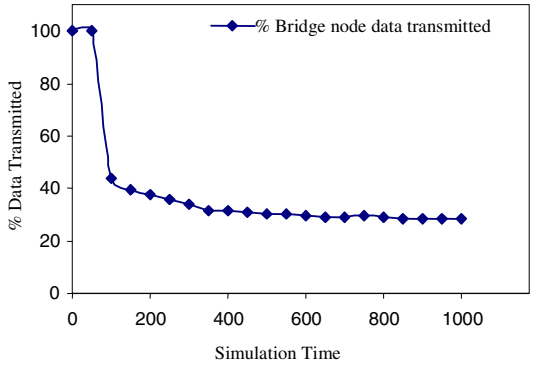

Fig. 6. Bridge node data transmitted

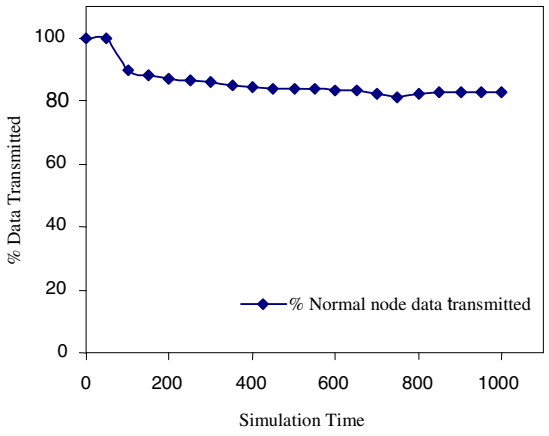

Fig. 7. Non-bridge node data ratio

9. Discussion and Conclusion

The primary focus of this paper is (a) to design and test a scheme that would build in certain degree of survivability into WLANs in the event of an AP failure, and (b) to develop a transparent scheme to handle any subsequent malicious nodes in the network, specially keeping in mind Multistage attacks by adversaries. The IAMS, the associated Bridge node-Leader Association Algorithm and the TDP are an attempt towards describing new paradigms and concepts in this direction. Even though the bridge nodes suffer a severe degradation in throughput, the IAMS has demonstrated that the network can survive an AP failure. We assume that there is at least one node in the network which is in the radio range of more than one Access Point. However, it is important to note that this scheme will fail to deal with AP failures if the above condition is not true.

Another assumption is that the WLAN comprises of more than one AP. This is a reasonable and realistic assumption since in real life scenarios like defense installations, corporate offices, airports, universities, etc., the WLANs always comprise of several APs. It is very rare to find a commercial WLAN consisting of just one Access Point.

For the sake of simplicity, in this paper we assumed that only one node may turn be malicious at any given time. As mentioned earlier, we did not consider the case of collusion-based attacks. The IAMS and associated schemes worked fine when nodes were detected to be malicious in succession, until the removal of malicious nodes did not partition the TG. If the TG got partitioned such that there were bridge nodes in one subgraph, but no leaders present, then the schemes failed to route the traffic for the subgraph with no bridge nodes. Thus, the IAMS is able to withstand the removal of up to a certain number of nodes, such that their removal does not partition the TG as mentioned above. Even for collusion-based attacks with an upper bound on the number of colluding nodes (bounded by above mentioned condition), the IAMS and associated schemes should work fine with a minor modification: the re-computation and re-association algorithms would have to make multiple passes. The number of passes would be equal to the number of colluding nodes to be removed. Thus testing the schemes against well known collusion and replication attacks forms an important part of our continuing research.

Other related work and ongoing research focuses on the following issues:

(i) Improve Load Balancing: In the current scheme, if the paths from a node to different bridge nodes run concurrently on certain hops, then nodes along those hops would get more traffic to route. We're working on improving our current scheme to take this into account.

(ii) Load Balancing utilizing the traffic conditions in the network: In the present scheme the load balancing for traffic distribution is only dependent on the number of hops in a given route. We are developing a load

balancing scheme based on fairness vs. throughput. This could accommodate localized congestion scenarios.

(iii) Lowering of throughput through degradation estimation: A scheme in which the APs are able to estimate and pre-compute the degradation in network throughput in case of a potential failure. The nodes could be requested to decrease their traffic by a certain amount when the failure occurs.

(iv) Requesting the nodes to move: Certain nodes could be requested to physically move closer to adjacent APs in the event of an AP failure, so that number of bridge nodes could increase.

10. References

[1] M.S. Gast, *802.11 Wireless Networks, The Definitive Guide*, O'Reilly & Associates, Inc., First Edition, Apr 2002.

[2] M. Virendra and S. Upadhyaya, "SWAN: A Secure Wireless LAN Architecture", *Proceedings of the* 29th Annual IEEE International Conference on Local Computer Networks (LCN), Tampa, FL, Nov 2004, pp. 216-223.

[3] J. Chen, S.G. Chan, J. He, and S. Liew, "Mixed-Mode WLAN: The Integration of Ad Hoc Mode with Wireless LAN Infrastructure", *Proceedings of* GLOBECOM 2003, San Francisco, CA, Dec 2003, pp. 231-236.

[4] Y. Huang, W. Fan, W. Lee, and P. S. Yu, "Cross-Feature Analysis for Detecting Ad-Hoc Routing Anomalies", *Proceedings of the* 23rd International Conference on Distributed Computing Systems (ICDCS), Providence, RI, May 2003.

[5] L. Zhou and Z. J. Haas, "Securing ad hoc networks", *IEEE Network*, 13(6):24–30, Nov/Dec 1999.

[6] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing Pair-wise Keys For Secure Communication in Ad Hoc Networks: A Probabilistic Approach", *Proceedings of the* IEEE International Conference on Network Protocols (ICNP). Atlanta, GA, Nov 2003.

[7] S. Zhu, S. Setia, S. Xu, and S. Jajodia, "GKMPAN: An Efficient Group Rekeying Scheme for Secure Multicast in Ad-Hoc Networks", *Proceedings of the* 1st International Conference on Mobile and Ubiquitous Systems (Mobiquitous), Boston, MA, Aug 2004.

[8] H. Wu, C. Qiao, S. De, and O. Tonguz, " An integrated Cellular and Ad hoc Relaying system: iCAR", in *IEEE Journal on Selected Areas in Communications (JSAC)*, Vol. 19, No. 10, pp. 2105-2115, Oct. 2001.

[9] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized Trust Management", *Proceedings of the* 1996 IEEE Symposium on Security and Privacy", Oakland, CA, May 1996.

[10] D. Balfanz, D. K. Smetters, P. Stewart, and H. Chi Wong, "Talking to Strangers: Authentication in Ad-Hoc Wireless Networks", *Proceedings of the* Network and Distributed System Security Symposium (NDSS), San Diego, CA, 2002.

[11] R.C. Prim, "Shortest Connection Networks and Some Generalizations", *Bell System Tech. Journal*, Vol. 36, Nov. 1957, pp. 1389-1401.

[12] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad hoc Wireless Networks", *Mobile Computing*, Kluwer Academic Publishers, 1996, pp.153-181.

[13] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks", *Proceedings of the* Twelfth Workshop on Parallel and Distributed Simulation, Alberta, Canada, May 1998, ISSN: 0163-6103, pp. 154-161.

[14] W. Trappe, M. Wu, and Z.J. Wang, K.J.R. Liu, "Anti-Collusion Fingerprinting for Multimedia," *IEEE Transactions on Signal Processing*, pp. 1069-1087, 2003.

[15] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil Attack in Sensor Networks: Analysis and Defenses", *Proceedings of the* 3rd International Symposium on Information Processing in Sensor Networks (IPSN), Berkeley, CA, Apr 2004.

[16] D. Chen, S. Garg, C. Kintala, K. Trivedi, "Dependability Enhancement for IEEE 802.11 Wireless LAN with Redundancy Techniques", *Proceedings of the* 2003 International Conference on Dependable Systems and Networks (DSN'03), San Francisco, CA, Jun 2003, ISBN: 0-7695-1952-0, pp. 521-529.

[17] A. Snow, U. Varshney, A. Malloy, "Reliability and Survivability of Wireless and Mobile Networks", *IEEE Computer*, Vol. 33, Issue 7, pp. 49-55, 2000, ISSN: 0018-9162.

[18] http://www.cisco.com/en/US/netsol/ns395/ns176/ns178/netqa0900aecd801764f1.html

[19] Q. Xue, A. Ganz, "QoS Routing in Mesh-based Wireless Networks", *International Journal of Wireless Information Networks*, Vol. 9, Issue 3, pp. 179-190, Jul 2002, Kluwer Academic Publishers.

[20] B. Bhargava, Y. Zhong, "Authorization Based on Evidence and Trust", *Proceedings of the* Data Warehouse and Knowledge Management Conference (DaWak-2002), Aix en Provence, France, Sep 2002.

[21] M. Virendra, M. Jadliwala, M. Sudhanan, S. Upadhyaya, "Quantifying Trust in Mobile Ad-Hoc Networks", IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS'05), Waltham, MA, Apr 2005, to appear.