A General Cooperative Intrusion Detection Architecture for MANETs

D. Sterne[1], P. Balasubramanyam[2], D. Carman[1], B. Wilson[1], R. Talpade[3], C. Ko[1], R. Balupari[1], C-Y. Tseng[2], T. Bowen[3], K. Levitt[2] and J. Rowe[2]

[1] *McAfee Research* *Dan_Sterne@McAfee.com*

[2] *UCDavis* *poornima@cs.ucdavis.edu*

[3] *Telcordia Technologies* *rrt@research.telcordia.com*

Abstract[1]

Intrusion detection in MANETs is challenging because these networks change their topologies dynamically; lack concentration points where aggregated traffic can be analyzed; utilize infrastructure protocols that are susceptible to manipulation; and rely on noisy, intermittent wireless communications. We present a cooperative, distributed intrusion detection architecture that addresses these challenges while facilitating accurate detection of MANET-specific and conventional attacks. The architecture is organized as a dynamic hierarchy in which detection data is acquired at the leaves and is incrementally aggregated, reduced, and analyzed as it flows upward toward the root. Security management directives flow downward from nodes at the top. To maintain communications efficiency, the hierarchy is automatically reconfigured as needed using clustering techniques in which clusterheads are selected based on topology and other criteria. The utility of the architecture is illustrated via multiple attack scenarios.

1. Introduction

In recent years, considerable interest has developed in creating new kinds of network applications that fully exploit distributed mobile computing, particularly for military uses. The key underlying technology for such applications is mobile ad hoc network (MANET) technology.

Flexibility and adaptability, which are the strengths of MANETs, are unfortunately accompanied in MANETs by increased security risks. This is because radio-based mobile communications among the components of distributed applications, and the infrastructure protocols that enable these communications, are exposed new threats, yet must remain available continuously, even in harsh environments. Intrusion detection technology will undoubtedly be a crucial ingredient in any comprehensive security solution to address these threats.

Intrusion detection in MANETs, however, is challenging for a number of reasons [16][17][18]. These networks change their topologies dynamically due to node mobility; lack concentration points where traffic can be analyzed for intrusions; utilize self-configuring multi-party infrastructure protocols that are susceptible to malicious manipulation; and rely on wireless communications channels that provide limited bandwidth and are subject to noise and intermittent connectivity.

To overcome these constraints, researchers have proposed a number of decentralized intrusion detection approaches tailored specifically for MANETs. These approaches, however, have focused almost exclusively on detecting malicious behavior with respect to MANET routing protocols (see Section 5) and have provided little evidence that they are applicable to a broader range threats, including attacks on conventional protocols, which also pose new problems in MANETs.

This paper describes a generalized, cooperative intrusion detection architecture proposed as the *foundation* for *all* intrusion detection and supporting activities in mobile ad hoc wireless networks. The architecture was designed with military applications in mind. However, we believe it is likely to be useful in the civil sector as well, for example, in multinational disaster relief efforts in regions having little remaining telecommunications infrastructure.

The remainder of this paper is organized as follows. Section 2 describes the problem domain of interest including unique challenges, a threat model, and general architectural requirements. Section 3 presents the proposed cooperative intrusion detection architecture including an overview of its structure and operation. Section 4 illustrates use of the architecture in three intrusion detection scenarios: intentional data packet dropping, attacks on MANET routing protocols, and attacks on network and higher-layer protocols. Section 5 discusses related work. We summarize our results and future work in Section 6.

[1]Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

2. Problem domain

Network nodes in the problem domain of interest encompass a heterogeneous mixture of manned and unmanned mobile systems including autonomous vehicles and sensors. Platform types include PDAs, processors embedded in special purpose devices, laptop-class systems, and server-class systems, which may be positioned in various kinds of vehicles.

A network in this problem domain can be characterized as a collection of interconnected islands, each containing up to a few hundred mobile nodes and corresponding to a single routing domain. Relationships between these islands may be organized in a way that roughly parallels the hierarchical structure of the human organizations that deploy them. Mobile nodes will communicate with their neighbors over radios, with data rates from tens of kilobits per second to a few megabits per second. Internet-based protocols play a role by binding together the disparate wireless link layers and physical layers in the network, and providing "reach-back" capability to the Internet. All nodes will be IP-addressable, with the IP addressing hierarchy closely coupled with the domain hierarchy. Specific nodes in each domain may be connected to nodes in other domains with higher-data-rate links of a few Mbps. All links are dynamic since nodes may rapidly establish or lose connectivity with their neighbors.

2.1. Key operational and technical challenges

Key operational and technical challenges of this problem domain include the following:

Mobility and dynamism: MANETs in this problem domain consist of highly mobile nodes and mobile network infrastructure. Mobility causes the size and topology of each network domain and relationships among domains to change continually. Consequently, infrastructure services must be decentralized and must adapt specifically to support this dynamism. Such infrastructure services include routing, autoconfiguration, mobility management, quality of service, and security.

Lack of fixed traffic concentration points: The conventional approach of positioning a relatively small number of intrusion detection systems (IDS) and firewalls at traffic concentration or "choke" points to inspect all network traffic will not work because traffic is normally dispersed over many routes that change dynamically, i.e., persistent choke points will not exist.

Limitations of host-resident network intrusion detection: An alternative strategy is to perform network intrusion detection only at the communications endpoints rather than intermediate points. This approach, however, has a number of key drawbacks. If network intrusion detection functionality is placed only at traffic endpoints, no intrusion protection will be present at intermediate nodes that must route the traffic to its intended destinations. Placing network intrusion detection only at endpoints also means that each node is completely responsible for protecting itself. This may result in having multiple, single-points-of-failure. Furthermore, detectors at endpoints are *collocated* on the same platform as the targets they are responsible for defending. As a result, attacks on a target platform may disable its detector. By contrast, placing detectors at some set of intermediate nodes enables a *layered defense* in which the attacker must evade or otherwise foil multiple sets of overlapping defensive functions. This is the approach we advocate below.

Wireless communications: MANETs rely on the RF medium and protocols that are susceptible to eavesdropping, jamming, interference, noise, collisions, and many other physical and MAC layer effects. These effects may lead to packet loss and intermittent connectivity.

Resource constraints: Cooperative intrusion detection systems must overcome (i) limited communications capabilities, (ii) varying and possibly limited energy, power, processing, and storage resources available in different parts of the heterogeneous networks, and (iii) greatly varying throughput due to dynamic network topology, propagation, interference, and data traffic patterns.

Use of end-to-end cryptography: Use of network or higher layer cryptography (e.g., IPSEC, SSL, etc.) to protect end-to-end communications has the side effect of making packet payloads and header information at various protocol layers *opaque* to network intrusion detection systems on intermediary nodes. Without additional measures, such cryptography renders these intrusion detection systems ineffective.

2.2. Threat model

We assume that adversaries may have a wide range of objectives including impersonating personnel and systems, obtaining sensitive information, corrupting critical information, interfering with progress of applications by various forms of denial of service, and penetrating nodes for subsequent use.

We assume that these networks will be protected by (i) cryptography at the link layer, providing strong protection for the confidentiality, integrity, and authentication of communications – at least to the granularity of distinguishing between insiders and outsiders and protecting communications among the former from the latter, and (ii) communications hardware that will provide resistance to jamming.

With these secure communications capabilities in place, the primary attack sources of concern are the following: (i) friendly nodes that are now in the possession of an adversary; (ii) friendly nodes that have been penetrated by viruses, worms, or other malware including infection from other network nodes, nodes on the reach-back network, or malware planted during the software development or maintenance activities; (iii) adversaries' nodes that possess cryptographic keying material stolen from a compromised friendly node and are now able to impersonate that node; and (iv) nodes on a reach-back network that provide services to the MANET, but have been compromised in some manner.

Attacks from these sources are generally considered "insider" attacks. As mentioned above, nearly all traffic in such wireless networks will require some level of authentication. However, since most link-layer, routing, and autoconfiguration protocols used in MANETS rely on broadcast addressing to various degrees, we assume that cryptographic protection will typically be based on group keying (e.g., one group key per link-layer neighborhood) and will not provide individual source authentication for every packet. Consequently, we do not assume that such authentication facilities will necessarily allow all attack packets to be cryptographically traced to their sources.

2.3. General requirements

As a backdrop for discussion below, we list the general requirements that should be met and services that should be provided by an *ideal* intrusion detection architecture for this domain. While the proposed architecture was developed with these in mind, some are not explicitly addressed.

The architecture should:

- *Address the broad spectrum of attacks* that may target the MANET, including both MANET-specific and conventional attacks, especially those having distributed sources or distributed targets;
- *Provide intrusion detection coverage for all traffic, all of the time,* regardless of changes in topology and routing that occur because of node mobility and other dynamic environmental factors.
- *Support layered defense* by imposing independent, overlapping intrusion detection mechanisms across potential attack paths.
- *Support a broad spectrum of detection techniques*, including signature-based, statistical anomaly, specification-based detection techniques, techniques that utilize promiscuous eavesdropping of wireless transmissions, and cooperative detection techniques involving exchange of data among detectors.
- *Provide access to intrusion detection data* from multiple protocol layers, operating system logs, and application logs, since some attacks and attack patterns may be detectable only via multi-source sensing.
- *Minimize consumption of bandwidth by communications among intrusion detection components,* e.g., avoid unnecessary flooding.
- *Adapt its behavior* in the event of failure or compromise of nodes and communications links, to *degrade gracefully.*
- *Provide autonomy of intrusion detection* capabilities when the MANET is partitioned or disconnected from the reach-back network or other fixed infrastructure.

Required Services: The architecture should *provide efficient services for transferring data* from widely distributed sources so that data can be collected, interpreted, and correlated locally, regionally, and "globally", as appropriate, and exchanged among pairs or groups of peer nodes for correlation or traceback. It should *provide services for querying data sources* for additional related data as needed. It should *provide services to support data fusion/integration and data reduction* including support for correlating distributed events to a single attack, reconciling conflicting data and compensating for possibly bogus data, and avoiding or compensating for overlapping reports. The architecture should *provide services for relaying intrusion detection management and intrusion response directives.* It should *provide a tailored interface to key-sharing services* provided by underlying cryptographic components to enable designated nodes to decrypt and inspect packet headers and payloads.

These services should be integrated with policy and configuration mechanisms that *dynamically assign and reassign intrusion detection, correlation, response, and security management responsibilities to nodes* based on their topological placement, capabilities, trustworthiness, and other factors, including desired tradeoffs among detection coverage and accuracy, bandwidth utilization, session key exposure, redundancy, survivability, and other factors.

3. A cooperative intrusion detection architecture

This section proposes a cooperative intrusion detection architecture for the MANET environment described above. Section 3.1, discusses and motivates our organizational model, the dynamic hierarchy. Section 3.2, describes how the dynamic hierarchy facilitates cooperative intrusion detection. Construction of the hierarchy using attribute-based clustering is discussed in

Section 3.3. Two additional topics relating to the utilization of the hierarchy are discussed in Sections 3.4 and 3.5. Section 3.6 describes the responsibilities of nodes according to their placement within the hierarchy. Section 3.7 provides a design overview of the functional components that reside in each node and provide the services that collectively embody the architecture.

3.1. Organizational model: a dynamic hierarchy

The choice of an organizational model is fundamental to the architecture of any distributed system. Common models include static hierarchy, peer-to-peer (P2P) and publish-and-subscribe. The static nature of the static hierarchy model, the potentially huge volume of multi-hop traffic that may be generated as a result of the arbitrary transfer of information in the P2P and publish-and-subscribe models as well as assumptions of uniform trust in P2P models render them inappropriate for our problem domain.

In order to provide incremental aggregation, detection, and correlation, efficient dissemination of intrusion management directives, and scalability, the organizational model we propose is the ***dynamic hierarchy***. The major advantage of a hierarchy is its potential scalability to large networks, since it can provide rapid and communications-efficient detection for local cooperative attack recognition, while still allowing data sharing for more widely-distributed cooperative intrusion detection algorithms. Unlike P2P networks where communications overhead can rise by the square of the number of nodes, a hierarchical approach allows higher-layer nodes to selectively aggregate and reduce intrusion detection data as it is reported upward from the leaf nodes to a root. Moreover, a hierarchy naturally aligns with the authority structure or chain-of-command that is common to many human organizations and governs the control of assets, in this case, network nodes and services. In the proposed architecture, this structure is represented by the flow of data to authoritative nodes at the root of the hierarchy, which dispatch directives down to lower levels.

In this problem domain, mobility and other factors will cause the topology to change continually, such that an initially-defined static hierarchy will soon be inefficient. Since both nodes and links will appear and disappear rapidly and normally, a dynamic, topology-based hierarchy must be formed and constantly maintained. Nodes will communicate intrusion detection information most often with other nodes that are their parents or children in the hierarchy. Efficiency will generally be improved if a significant fraction of children are topologically nearby, such as being link-layer (1-hop) neighbors. Since mobility and other factors will lead to frequent changes in these topological relationships, hierarchical relationships between nodes need to evolve as the topology evolves. We propose to use *clustering* [1,4,5] for establishing and maintaining such a dynamic evolving hierarchy of intrusion detection components.

An example of this infrastructure is shown in Figure 1. Nodes annotated with a "1" are the representatives of first level clusters. Arrows pointing to these nodes originate from the other (leaf) nodes in their cluster that report to them. Similarly, arrows from first level representatives to their second level representative (annotated with a "2"), show the composition of one of the second level clusters. The arrow from the second level representative to the third level representative shows that the former is a member of a third level cluster; other members of that cluster are outside the scope of the figure and are not shown. To avoid having a single representative node at the top of the hierarchy that is a potential single point of failure, one or more members of the highest level cluster should be designated as backup representatives. This infrastructure allows intrusion detection observations to be gathered efficiently from the entire network; provides incremental aggregation, detection, and correlation; and efficient dissemination of intrusion response and management directives (e.g., signature updates).

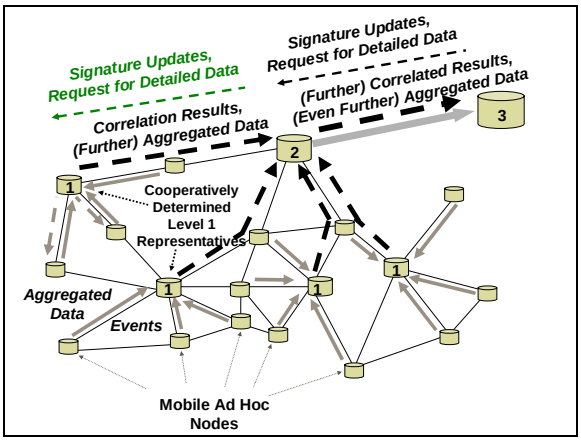

Figure 1. Dynamic intrusion detection hierarchy

3.2. Using the dynamic hierarchy for intrusion detection – an overview

In the proposed architecture, every node is responsible for using its own resident network and host-based intrusion detection mechanisms to protect itself. In addition, nodes are assigned intrusion detection responsibilities to help protect other nodes in the network. These responsibilities include monitoring, logging, analyzing, and reporting network data at various protocol layers, as described in Section 3.6.

The responsibilities of a node depend on its current positions in the topology and the dynamic hierarchy. Nevertheless, data acquisition will generally occur at or near the bottom of the hierarchy where leaf nodes are attached. Intrusion detection data of all forms including alerts will generally flow upward and will be consolidated, correlated, and summarized incrementally as it flows upward. A small collection of nodes at the uppermost levels of the hierarchy will serve as security management nodes that may possess an integrated view of the overall cyber security of the network. These nodes will also provide facilities for sending directives to all the nodes in the network, such as directives to alter all nodes' intrusion detection or intrusion response configurations; these will flow down the hierarchy from top to bottom. In short, data from intrusion detection systems needs to flow from the bottom to the top where it can be utilized in decision-making. Once decisions are made, they are transformed into directives that flow from the top to the bottom.

Different kinds of attacks require different sets of detection data, and this data may aggregate at different levels in the hierarchy. A key principle is that intrusion detection and correlation should occur at the lowest level in the hierarchy at which the aggregated data is sufficient to enable an accurate detection or correlation decision. If the data available at a level is not sufficient, it is pushed upward in the hierarchy where it is further aggregated with other data. One reason for this principle is *detection latency* – in the absence of a suspicious event, data will generally be reported periodically by group members to their clusterhead. If a member possesses sufficient data to make an immediate detection decision, but defers detection processing to its next-level representative and only transfers the data to the parent periodically, this will introduce a delay in detecting an attack. Another reason is that performing intrusion detection is a form of *data reduction* in which concise inferences are drawn from potentially large amounts of data. If a node performs intrusion detection on a set of data, it may free itself from having to transmit the entire data set to its representative; instead, if an attack is detected, the node may only need to send an alert and the associated, relevant evidence.

The dynamic intrusion detection hierarchy provides a scalable and efficient structure for organizing intrusion detection components. Nevertheless, there will be situations in which information may need to flow outside this structure, i.e., it may need to flow directly between components that are neither peers nor hierarchically related. Hence, other styles of communication are also supported.

3.3. Building the hierarchy using attribute-based clustering

Topological clustering, which is typically used in MANETs to construct routes, permits the creation of a logical hierarchy that can adjust to topology changes on the fly. Clusterhead selection occurs at many levels. Peer nodes use clustering to self-organize into local neighborhoods (first level clusters) each of which selects a neighborhood representative i.e., clusterhead. These representatives then use clustering to organize themselves into second level (regional) clusters. These clusters select representatives, which then organize themselves into third level clusters, and so forth until all the nodes in the network are interconnected by a hierarchy of representatives, with a small cluster at the top.

The bandwidth efficiency of such an architecture depends on exploiting topological characteristics to organize nodes into groups. However, a mixture of topological and other criteria are used to select clusterheads. Some of these criteria, which may be used only at particular levels in the hierarchy, include connectivity, proximity, resistance to compromise, accessibility by network security specialists, processing power, storage capacity, energy remaining, bandwidth capabilities, and administratively designated properties.

Connectivity is the measure of how many other nodes a given node can talk to directly. Proximity is particularly important for organizing the lowest level groups; each member should be within one hop of its representative. This restriction provides resilience by ensuring that an initial level of cooperative exchange among neighboring detectors can occur without any reliance on MANET routing, which may be targeted by an adversary and disabled or compromised. In other words, communication within first level groups can function even when routing services are not available. In addition, since single-hop communications are significantly more efficient than multi-hop communications, this approach provides high communications efficiency for a significant fraction of the overall set of communication paths within the cooperative hierarchy.

Resistance to compromise (hardening) is an administratively-designated attribute that describes the probability that the node will not fall into adversarial control. Selection of upper level clusterheads is weighted more heavily to emphasize resistance to compromise. The organization may also allow a roving security management node to take top priority in the hierarchy or allow the hierarchy to tie into a static security management network if available. However, if neither is available, the hierarchy should generally attempt to find alternatives among the nodes that are available and meet

minimum requirements. Since the operation of some MANETs will be overseen by one or more network security specialists, nodes used by such specialists as security management consoles will typically assume positions at the top of the hierarchy.

Processing power and storage capacity are additional attributes describing the ability of the node to perform computation and retain data. Energy remaining is either the measure of battery power left, or indication of an externally powered node (i.e., part of a fuel-powered vehicle). Bandwidth capabilities indicate the node's potential network throughput, and may vary greatly across different hardware platforms. Administratively-designated properties include any additional attributes of the nodes that may be relevant.

3.4. Promiscuous monitoring

Many proposed approaches to intrusion detection in MANETs rely on *promiscuous mode monitoring* of wireless communications [2][4][5][6][12]. As described by Marti, et al [6], this means that "if node A is within range of a node B, it can overhear communications to and from B even if those communications do not directly involve A." For example, suppose nodes A, B, and C are arranged in a straight line geographically such that B is within the communication ranges of both A and C, but A and C are outside each other's range. In other words, A can communicate directly with B, and B with C, but A and C cannot, and must use B as an intermediary. Under optimal conditions, if A is promiscuously eavesdropping and B sends a packet to C, A will also overhear it. Acquiring intrusion detection data in this manner has significant advantages. First, it allows local data collection without consuming any additional communications overhead. Second, it provides first-hand observations (for nearby traffic); this avoids the need to rely on observations from other nodes, which might lie.

On the other hand, data from promiscuous monitoring can be highly unreliable under various conditions, as described by Marti et al [6]. For example, if another neighbor of A attempts to send a packet to A at the same time B sends its packet to C, A will experience a collision and may not hear B's packet. Similarly, if node D, another neighbor of C, sends a packet to C at the same time B does, C may experience a collision. Node A may then erroneously believe that C received B's packet successfully. Hence, data from promiscuous monitoring may be incomplete or misleading.

The alternative to promiscuous monitoring is direct reporting by participants. For example, for a data collection node to find out which packets C received, C would need to explicitly report them to the node. This consumes bandwidth because each such packet must be duplicated and retransmitted, at least once. Moreover, if C is malicious, it could claim to have received packets that it did not, or vice versa. While cryptographic techniques can provide non-repudiation guarantees, these techniques are generally considered too heavyweight to be applied to every packet in the network. On the other hand, direct periodic reporting of packet *counts and statistics* is much more practical than packet *payloads* because such numerical values consume far less bandwidth. More importantly, bandwidth consumed by reporting these values does not rise with the volume of packets they summarize.

Because both promiscuous monitoring and direct reporting have such pronounced advantages and disadvantages as data acquisition modes, the proposed architecture is designed to support intrusion detection algorithms that use either or both.

3.5. Monitoring end-to-end traffic

Given the lack of persistent traffic concentration points, network intrusion detection processing must be distributed throughout the network. In our architecture, all nodes have some responsibilities for certain intrusion detection tasks. For example, many important cyber attacks are end-to-end attacks in which an attacker attempts to penetrate or disable a victim that is several routing hops away. An important issue for detecting such attacks in a MANET is determining which nodes should be responsible for detecting them. This issue also applies to detecting distributed network layer attacks such as DDoS, port scanning, and fingerprinting.

The simplest solution is for every node to monitor (at the network and higher layers) every end-to-end flow that passes through it. An important drawback to this strategy is that it can lead to excessive redundancy and inefficient use of resources. Furthermore, every node in the network will be responsible for full-blown, multi-layer, intrusion detection processing *on every packet* that passes through it. Even worse, if end-to-end encryption is used, then *every node* between the endpoints will need access to the decryption key; this completely undermines the value of encryption in the first place.

Instead, our strategy is to assign end-to-end traffic monitoring responsibilities to the two nodes that are the first hop and last hop routing points between each pair of communicating endpoints. For example, if a flow between nodes A and F was routed through the path ABCDEF, then only nodes B and E would be made responsible for monitoring the flow. If the route changes to AXBCDEF, then X would automatically assume B's monitoring responsibilities.

This strategy, described further in Section 4.3, provides several advantages, including simplicity, modest distributed resource consumption, reduced key

exposure, load balancing, redundancy, and defense in depth.

3.6. Responsibilities of nodes

In this architecture, all nodes are responsible for their own self-defense including performing host-based intrusion detection on their internal events and network-based intrusion detection on network packets they originate and consume as network layer endpoints. In addition, each node is responsible for a broader set of network intrusion detection responsibilities to help protect other nodes. As described below, the responsibilities of a node depend, among other factors, on whether the node is currently acting as a clusterhead and whether the node is currently in a topologically advantaged position to gather relevant observations.

3.6.1. Responsibilities of leaf nodes. All leaf nodes in this architecture (nodes at the bottom of the detection hierarchy) are responsible for certain data acquisition, intrusion detection, and reporting functions. These include the following:

Link-Layer Responsibilities – For each packet received, nodes accumulate link-layer counts and statistics describing source and destination MAC addresses and packet types, i.e., forwardable data packet, consumable data packet, or MANET infrastructure control packet. Note that, since packets may optionally be captured via promiscuous eavesdropping, it is possible to receive packets addressed, at the MAC layer, to other nodes. Similar processing should also be performed on all packets *originated* by the node. Nodes are also responsible for accumulating counts and statistics about cryptographically-detected packet replay attempts and potential indicators of MAC layer "channel hogging" such as unusually small collision backoff intervals or excessive numbers of RTS packets. Each node reports these link-layer counts and statistics to its clusterhead periodically, asynchronously, or when queried.

MANET Infrastructure-Layer Responsibilities – For each MANET infrastructure protocol packet received (e.g., MANET routing control and autoconfiguration packets), nodes log packet headers and payloads. Each node forwards copies or summaries of MANET infrastructure protocol packets to its clusterhead periodically, asynchronously, or when queried.

Network- and Higher-Layer Responsibilities – For specific packets that are received, each node also accumulates network and higher layer counts and statistics. These include 1) packets that are addressed to the node at the network layer, or 2) forwardable data packets belonging to end-to-end flows the node is currently responsible for monitoring. Having every node monitor every flow that passes through it can result in excessive redundancy. Instead, packets belonging to each flow are assigned to one or more nodes for monitoring. For flow assigned to a node, the node is required to perform conventional network layer, transport layer, and application layer intrusion detection processing. These responsibilities include (i) accumulating counts and statistics on network and higher layer attributes such as source and destination IP addresses and ports, protocols, packet lengths, and packet types and reporting such information to its clusterhead periodically, asynchronously, or when queried; (ii) performing attack signature matching or security specification checking on packet headers and payloads; and (iii) if a signature match or specification violation occurs, logging all relevant evidence and sending an alert immediately to the node's clusterhead.

Even though nodes that are not clusterheads collect statistics at multiple protocol layers, they are not generally required to analyze this information for intrusions. This is because any single node in a MANET can observe only a relatively small amount of traffic; typically, this traffic is neither statistically significant nor representative of overall traffic in the network. Consequently, detection computations based on this information are more fruitful when applied to larger collections of statistics aggregated from multiple nodes; these aggregates are available to clusterheads at higher levels in the architecture, as described below.

3.6.2. Responsibilities of Clusterhead Nodes. The responsibilities of clusterhead nodes include all of the responsibilities of leaf nodes. In addition, a clusterhead must:

- Aggregate and consolidate its own (firsthand) intrusion detection data from the link, infrastructure, network, and higher layers with corresponding layer data reported by members of the cluster it represents; this may involve data reduction.
- Perform intrusion detection computations on consolidated data. For example, (i) from consolidated link-layer statistics, detect channel hogging and intentional dropping of forwardable data packets; (ii) from consolidated infrastructure protocol data, detect attempts to distort routes, partition the network, and assign or allocate inappropriate IP addresses; (iii) from consolidated network and higher layer statistics, detect DDoS, intelligence gathering, usurpation, and other kinds of attacks; and (iv) perform alert correlation.

Depending on the correlation algorithms chosen, the node may also be responsible for querying children, peers, and other unrelated nodes for additional data, or responding to queries from other nodes.

Like leaf nodes, each clusterhead reports intrusion detection data (or summaries thereof) upward in the architecture to its own higher level clusterhead. In this case, however, that data has already been aggregated from multiple sources, at least once.

In addition, nodes near the top of the hierarchy will have authority and responsibility for managing the detection and response capabilities of the clusters and clusterheads below them. For example, they will distribute and activate 1) detection rules and signatures, and 2) intrusion response directives and policies.

3.7. Logical components of the architecture

In this subsection, we present an overview of the architecture from the perspective of a single node. We describe the logical components that provide the functionality and services that are part of the architecture and identify key interactions between those components.

Architecture support services in each node are provided by a core agent responsible for communications within the hierarchy, base processing, and loading/unloading of dynamic software components, such as the detection algorithms and associated data acquisition and logging rules, correlation algorithms, and intrusion response modules.

These logical components are depicted in Figure 2. The gray rectangle in the middle of the figure represents the central control component (core agent) on each node, which performs sensor-independent processing and communication. The circles within this component represent relevant logical processes. The rounded boxes outside the central component represent dynamically loadable software components. The cylinders represent local information storage. In the context of this section, we will refer to both alerts and logged audit data as *events*.

Data Sources: The network taps in the upper left of Figure 2 produce streams of packet data from different network layers; these are primary data sources for intrusion detection. Network statistics are collected from interface devices or protocol software layers. The audit logs represent any host-based logging facility. A monitor module reads in this data and performs host-based intrusion monitoring.

The data sources, shown in the lower left, are system dependent. The monitoring modules are responsible for collecting the information and presenting it in a format suitable for the rest of the architecture.

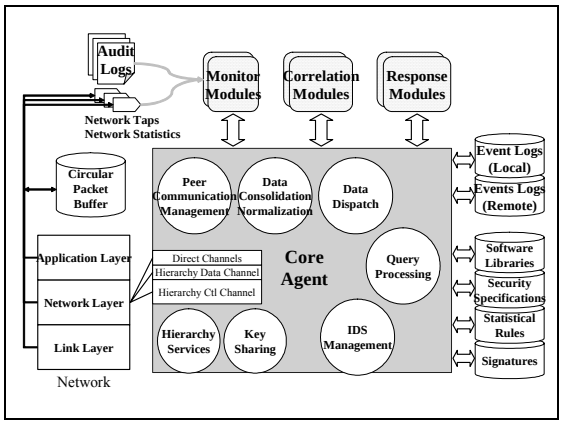

Figure 2. Logical IDS architecture components within each node

Dynamically Loadable Components: The monitor, correlation and response modules are software components designed for specific purposes or operating environments. Monitor modules create events from the inputs they process. Correlation modules attempt to group related events together into a single event, or link events together as part of a single scenario. Response modules perform network and host responses directed by the upper levels to defend and repair. Examples include killing a specific process, disabling a user account, or configuring the routing daemon to avoid a particular node.

Storage Facilities: The local and remote event logs, shown on the right side of the figure, store events generated by the monitor modules (local) and received from other nodes (remote). These logs can be queried by the local correlation modules for their own processing or by upper levels of the hierarchy looking for more detailed information about events that may have been reduced on the way up the hierarchy. There is also a database for storing software libraries and current "rule" files for specification-based detectors, statistical detectors, and signature-based detectors. The third storage facility, on the left side of the figure, is the circular network buffer, which stores a limited timeframe of packet data that was received by the node. This facility allows a node to 'playback' recent activity when a request to take over monitoring duties arrives. As some packets will have already passed through the node, it needs this facility to start processing the new set of traffic from the past.

4. Usage scenarios

In this section we present several short, usage scenarios to explain how the cooperative intrusion detection architecture supports representative intrusion

detection algorithms. The topology and dynamic hierarchy used in these scenarios is depicted in Figure 3.

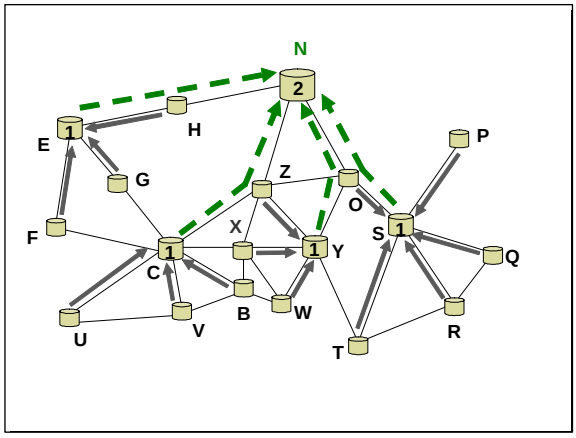

Figure 3. Dynamic hierarchy for usage scenarios

The hierarchy depicted in Figure 3 is based on one-hop leaf clustering, i.e., every leaf node is a one-hop neighbor of its clusterhead. In addition, each leaf node has only one parent. The level-1 clusterheads, marked "1" are nodes E, C, Y, and S. A single level-2 clusterhead, node N, is shown marked "2". All other nodes are leaf nodes. Hierarchical relationships are depicted as arrows. For example, nodes U, V, and B are children of C. Similarly, nodes E, C, Y and S are children of N. In the scenarios that follow, we will use node X as an example of a potential attacker, though the techniques we described are intended to be applied such that attacks from *any node* can be detected.

4.1. Detecting intentional data packet dropping

This example shows the detection of intentional dropping of data packets that should be forwarded by nodes acting as MANET routers. The objective is to detect nodes that intentionally drop a significant number of data packets over time, not to detect intentional dropping of individual packets.

The approach illustrated here utilizes *link-layer monitoring* and accumulation of *packet counts* as described in 3.6.1. The cooperative intrusion detection system as a whole 1) obtains link-layer observations of each node's behavior from *all* of its immediate neighbors; 2) successively aggregates these observations by moving them up the hierarchy; and 3) using the aggregated observations, compares the number of *forwardable* data packets sent to and received by each node with the expected number of *forwarded* data packets sent from the node. If the number of packets forwarded by node is significantly fewer than the number of forwardable packets received by the node, then the node is likely to be intentionally dropping packets.

To make the comparison of inflow and outflow meaningful, other kinds of packets that are sent to or by a node must be discounted. It must be possible to distinguish between forwardable data packets and unforwardable data packets, such as packets that have reached their ultimate destinations and link-layer beaconing packets. (For example, a packet that has reached its ultimate destination should have equivalent network- and link-layer destination addresses. Similarly, an originated packet emanating from a node should have equivalent network and link-layer source addresses.) In the same manner, it must be possible to distinguish between forwarded data packets and other packets originated by the node under observation. Here, we assume both of these conditions hold and, in addition, assume that the link layer supports reliable delivery by means of *ack* (acknowledgement) packets; these provide critical evidence that a monitored node has in fact received packets that were sent to it.

To determine whether node X in Figure 3 is intentionally dropping packets, X's input and output packets are monitored by its immediate neighbors, C, Z, Y, W, and B. These nodes supply observations to their parents in the hierarchy shown in Figure 3. These observations consist of counts of the numbers of different kinds of packets sent to, received by, and sent by X over a specific time period. Observations may be based on promiscuous monitoring or direct participant reporting, but in either case, the acquisition mode must be explicitly denoted. Received packets are categorized as forwardable, unforwardable, or ACK packets; and sent packets are categorized as forwarded, originated, or ACK packets. Counts for each of these packet types are provided for every observed link-layer source and destination address pair. By aggregating and comparing these packet counts, it is possible to determine the total number of *forwardable* packets X has received (acknowledged) and compare it with the total number of *forwarded* packets sent by X.

According to the hierarchy shown in Figure 3, nodes B and V supply their observations to C, while nodes Z and W supply observations to node Y. At the next level in the hierarchy, C and Y provide consolidated summaries of their own and their children's observations to their parent, N. N is the lowest point in the hierarchy at which *all* observations about X are available; consequently, N is optimally positioned to detect whether X is intentionally dropping data packets. More generally, in this approach, N is positioned to detect packet dropping by all nodes whose neighbor sets are entirely contained within the portion of the network covered under N.

Note that for a different topology or different clustering algorithm, all observations about a particular node might aggregate at the first level clusterhead, rather than the second level, as depicted here. In that case, intrusion detection processing for that node would occur there, in keeping with principle that detection should occur at the lowest level in the hierarchy at which the aggregated data is sufficient to enable an accurate detection decision, is discussed in Section 3.2. For yet another topology and clustering algorithm, all observations for a particular node might aggregate at the third level clusterhead.

4.2. Detecting attacks on MANET routing protocols

This example shows the detection of certain kinds of attacks on MANET routing protocols. We illustrate this problem using a man-in-the-middle (MIM) attack on AODV (Ad Hoc On Demand Distance Vector) Routing [14][8] [11].

AODV is a reactive distance vector protocol in which routes are established on demand. When a node needs to establish a route to a new destination, it floods a route request (RREQ) through the network. This is depicted in Figure 4 in which node E floods a route request indicating node B as the intended traffic destination. When the first copy of the RREQ reaches its destination, the destination sends back a route reply packet traversing the path taken by the RREQ in reverse order, thereby establishing this route as the shortest path between the requester and destination. Subsequent RREQs that arrive via longer paths are ignored. Each RREQ contains a sequence number that is propagated from the requester to the destination. However, to distinguish stale copies of old RREQs (which may continue to propagate) from interfering with newer RREQs, the protocol specifies that RREQs with higher sequence numbers override those with older sequence numbers. This creates a vulnerability.

Figure 4 depicts the flow of RREQ packets that results when node E attempts to establish a route to node B. The shortest path from E to B is through nodes G and C, i.e., E→G→C→B. In this scenario, however, node X attempts to insert itself into the route between nodes C and B. It does this by modifying the sequence number of the RREQ it receives from C before forwarding it on to its neighbors, which include node B, the route destination. Copies of the bogus RREQ from X are shown in the figure using dotted lines while copies of the legitimate RREQ are shown using solid lines. When X's bogus RREQ arrives at B, it will override the legitimate RREQ from C that arrived previously that represents the shortest path between E and B. As a result, X will be able to insert itself into the route. This puts X in a position to monitor, delay, delete, or otherwise interfere with traffic traveling from E to B. Similar techniques applied to route replies allow X to insert itself into the reverse path from B to E.

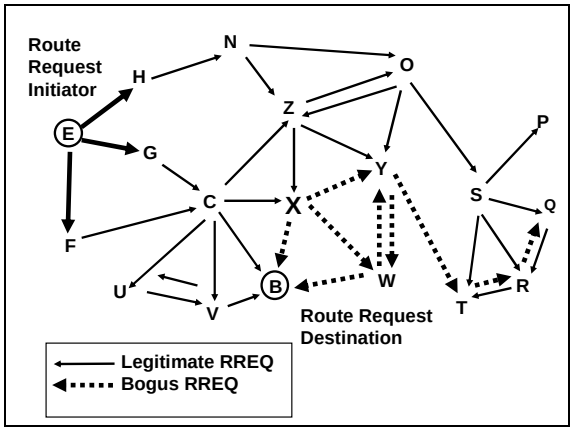

Figure 4. RREQ flow during AODV man-in-the-middle attack

The intrusion detection approach illustrated here utilizes *monitoring and logging of MANET routing protocol packets*, as described in Section 3.6.1. The cooperative intrusion detection system as a whole 1) obtains observations of each node's routing protocol behavior from all of its immediate neighbors; 2) successively aggregates these observations by moving them up the hierarchy; and 3) analyzes the aggregated observations to *detect* RREQ packets with modified sequence numbers and *localize the source(s)* of these packets. Observations may be based on promiscuous monitoring or direct participant reporting.

Each distinct RREQ, as identified by the unique combination of a purported originator and sequence number, is flooded throughout the network. As a result, X's bogus RREQ will likely propagate back to E, which can independently recognize that the RREQ is bogus and detect this as an attack. E will then generate an alert and transmit it upward in the detection hierarchy via its clusterhead. However, no single node can identify X as the source of the attack. Suppose that the alert is subsequently disseminated throughout the network accompanied by a query for related evidence. As shown in Figure 4, node Y, a neighbor of W may have observed W forwarding the bogus RREQ. Does this imply that W is the attacker? No. W simply forwarded the RREQ it received from X, as required by the AODV protocol. Similarly, if B, a neighbor of X, observed X sending the bogus REQ, it would be unable to tell whether X created the bogus RREQ or was simply forwarding an RREQ it received from one of its neighbors. Only by examining the *totality of X's inputs* can the intrusion detection system determine that X did

not receive the bogus RREQ from another node and must therefore be its creator.

To do this, we utilize the dynamic detection hierarchy to aggregate observations of RREQ transmissions from all of X's neighbors using the clustering relationships shown in Figure 3. Thus, node C aggregates RREQ observations from B, V, and itself; Y aggregates RREQ observations from W, Z, and itself; N in turn aggregates observations of X gathered by C and Y. N then possess all available observations of X's inputs and outputs and if these are complete, can in principle determine whether X is the attacker. In the event that these observations are incomplete or ambiguous, N can also analyze the observations aggregated from a larger region of the network (multiple hops from X) to partially localize the source of the attack, i.e., narrow the range of suspects. Note that the gathering and analysis of these observations could be performed periodically, or to reduce potential overhead, it could be done only after some evidence of an attack has been seen, for example, by E as described previously.

This general problem is not unique to AODV and exists in other MANET routing protocols. For example, in OLSR (Optimized Link State Routing Protocol [15]), Topology Control (TC) messages are flooded (forwarded) throughout the network by designated nodes called Multi-point Relays (MPR). A malicious MPR can launch a MIM or other attack by modifying a TC message before forwarding. Determining which MPR in a forwarding chain is the creator of a bogus TC is analogous to the AODV RREQ localization problem described above and is amenable to the same detection and localization techniques.

4.3. Detecting attacks on network and higher layer protocols

This example concerns the detection of conventional attacks on network and higher layer protocols such as:

- buffer overflow attacks directed at HTTP, FTP, SSH, and other protocols;
- port scans, address sweeps, fingerprinting, and other intelligence gathering activities;
- attempts to gain privileges or access sensitive files by exploiting insecurely configured services such as Unix "r commands" and services implemented via CGI scripts; and
- denial of service flooding attacks that attempt to congest network links.

The central intrusion detection challenges for dealing with conventional attacks in a MANET are 1) ensuring coverage of all traffic paths while controlling the degree of redundant detection processing and minimizing coordination overhead; and 2) detecting distributed attacks in the absence of traffic chokepoints.

As described earlier, the proposed architecture supports layered defense in which intrusion detection systems on communicating end points (for self-defense) are augmented by intrusion detection systems on one or more intermediate nodes between the communicating end points (for community defense). The intrusion detection approach illustrated here utilizes the *monitoring of network and higher-layer headers and payloads* and *accumulation of associated counts and statistics*, as described in Section 3.6.1. The cooperative intrusion detection system as a whole assigns the monitoring of each end-to-end flow in the network to individual nodes based on the flow's source and destination addresses and node positions in the topology. Nodes independently apply conventional network intrusion techniques to the flows to which they are assigned and report alerts and flow statistics upward in the hierarchy, which results in successive aggregation. Clusterheads at various levels can then correlate alerts (to identify patterns of attacks) and analyze the aggregated statistics for distributed attacks.

Suppose node X is communicating with nodes W, U, F, and, in particular, Q, as depicted in Figure 5; these paths are shown as wide "flow arrows". All nodes along the routes between X and these end points could potentially be asked to monitor these flows for intrusions, *without relying on promiscuous monitoring*. For traffic between X and Q, however, three nodes (Y, T, and R) would be assigned, leading to excess redundancy and other issues as mentioned earlier. In larger diameter networks, where average path lengths are greater, excess redundancy would be exacerbated. Instead, our strategy is to assign network and higher layer monitoring responsibility to (at most) two intermediary nodes – the first and last ones.

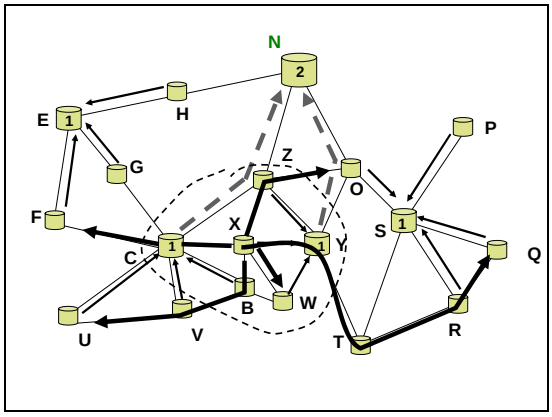

Figure 5. Network and higher layer flows requiring monitoring for intrusions

Consequently, for traffic between X and Q, nodes Y and R would be assigned. This strategy provides complete detection coverage for all end-to-end communication paths in the network. Moreover, the mapping of monitoring responsibilities is immediately and continuously self-evident to all nodes, regardless of topology or mobility by simple examination of the source and destination addresses of each packet they route. No additional communication among nodes is required to coordinate "the handoff" of these assignments, even under the most dynamic conditions.

Detecting all network and higher layer attacks by node X requires examining all packets sent by X. As shown in Figure 5, this is accomplished by having each of X's neighbors examine all of the packets X sends through them. (A neighbor will not examine traffic that X *forwards* to the neighbor, such as traffic sent by F to T through X.) Each neighbor is able to independently detect attacks in this traffic using conventional network intrusion detection technology. If an attack is detected by one of X's neighbors, it reports the attack to its level-1 clusterhead (C or Y), which will in turn report the attack to its own clusterhead (N). In this way, an overall view of X's intrusive behavior is successively aggregated and is then available for correlation, including correlation with reports on the behavior of other nodes.

This strategy also supports the detection of attacks with distributed targets or sources, which are more challenging in a MANET environment. Consider the detection of port scans. The manual for the Snort intrusion detection system [13] defines a port scan as the traffic sent to "more than p ports in t seconds". Clearly X could exceed this threshold while evading detection by spreading its outbound scan traffic across its five neighbors, i.e., using five outbound route points. However, if each neighbor of X collects port usage counts for X, as discussed in Section 3.6.1., and reports these periodically upward through the hierarchy, the port count total for X and other nodes will emerge as a result of successive aggregation. This can prevent an attacker from evading threshold-based detection rules by distributing outbound attack traffic. This strategy is also applicable to detecting distributed attacks *against* X such as DDoS attacks. In this case X's neighbors are the last hop routers for traffic sent to X. If these neighbors report inbound packet statistics upward through the hierarchy, a consolidated statistical view of all the traffic sent to X can be assembled and analyzed, even if X itself is completely saturated and unable to raise an alarm by itself.

5. Related work

The concept of a cooperative intrusion detection architecture for MANETs was first proposed by Zhang and Lee [12]. Their architecture supports and is focused on statistical anomaly detection, particularly detection of abnormal updates to routing tables. Anomaly detection at the MAC and application layers is also discussed briefly, while support for signature-based and other detection technologies is not.

Use of clustering in a cooperative intrusion detection architecture for MANETs was first suggested by Kachirski and Guha [5]. The role of clustering in their architecture, however, is *fundamentally different* than in the architecture we propose here. Kachirski and Guha use clustering only to select a *single layer* of sparsely positioned nodes that partially or completely cover (can hear) all links in the network. These nodes are then utilized as promiscuous monitors and are dynamically tasked by sending them intrusion detection code in the form of mobile agents. The motivation for sparse placement is to reduce the number of nodes used for intrusion detection processing while attempting to observe most, if not all, network traffic. By selecting cluster parameters, the number of monitoring nodes can be decreased, but at the expense of increasing the percentage of traffic that is not observable.

By contrast, we use clustering to organize and maintain a dynamic *hierarchy* of intrusion detection components. We exploit the hierarchy for cooperative exchange of data, in particular: reporting; aggregation, correlation, and reduction of data; and dissemination of directives. The hierarchy also supports graduated levels of authority in that nodes at the top are accorded supervisory privileges. Another significant difference is that the architecture in [5] relies entirely on promiscuous monitoring, which is not sufficiently reliable [6] to serve as the sole source of intrusion detection data. Moreover, promiscuous monitoring is ill-suited for detecting attacks on conventional network- and higher-layer protocols in MANETs. The architecture we propose here allows but does not rely on promiscuous monitoring, and addresses a broad spectrum of attacks.

More recently, Huang and Lee have also proposed use of clustering within a cooperative intrusion detection architecture [4]. Like the approach of Kachirski and Guha, this approach uses clustering to select a single layer of sparsely positioned promiscuous monitors. In [4], these monitors are used to recognize routing misbehavior via statistical anomaly detection. Huang and Lee also propose a clustering algorithm designed to give every node an equal chance to be elected as a clusterhead, to prevent malicious nodes from manipulating cluster formation.

Detecting malicious packet dropping is the primary focus of several intrusion detection approaches for MANETs. A statistical approach is presented by Rao and Kesidis in [9] using estimated congestion at intermediate nodes to make decisions about malicious packet dropping behavior. The work described in [2][3][6][7] uses the mechanism of assigning a value to the "reputation" of a node. Reputations are used to weed out misbehaving nodes and enable interactions with only well-behaved ones. Again, the intrusive activity addressed is that of misbehaving nodes that agree to forward packets to neighbors, but fail to do so. In [2][6], promiscuous monitoring is employed to monitor the nodes. In [3], Buttyan and Hubaux propose a tamper-resistant module at each node that counts packet forwarding events. Data from the counter is used to monitor cooperation and enable penalization of non-cooperative nodes. In [7], Michiardi and Molva present a security mechanism that aims to promote node cooperation through a collaborative monitoring technique that uses game theory to model the interactions between nodes.

In [10], Ramanujan et al. present a system to detect, avoid and recover from malicious attacks on ad hoc networks. They only focus on attacks that target the routing function within these networks. Key ideas include a distributed firewall mechanism to limit the impact of flooding; an algorithm to detect and recover from intruder-induced path failures; and a wireless router extension architecture.

In summary, the architecture proposed here is distinguished from prior research on intrusion detection for MANETs by its breadth of applicability, lack of reliance on promiscuous eavesdropping, and novel features for assigning intrusion detection responsibilities to nodes as a function of topological positioning, protocols, and other factors.

6. Summary and future work[2]

Intrusion detection in MANETs is challenging because these networks change their topologies dynamically due to node mobility; lack concentration points where traffic can be analyzed for intrusions; utilize self-configuring multi-party infrastructure protocols that are susceptible to malicious manipulation; and rely on wireless communications channels that provide limited bandwidth and are subject to noise and intermittent connectivity. We have proposed a cooperative, distributed intrusion detection architecture for MANETs that is intended to address these challenges.

The architecture is organized as a dynamic hierarchy in which data acquisition occurs at the leaves, with intrusion detection data being incrementally aggregated, reduced, analyzed, and correlated as it flows upward toward the root. A key principle is that detection and correlation should occur at the lowest level in the hierarchy at which the aggregated data is sufficient to enable an accurate detection or correlation decision; this strategy can reduce detection latency and bandwidth consumption. Nodes at the top of the hierarchy are responsible for security management functions; they consume alerts and data requiring further analysis. Directives for intrusion management and intrusion response, as well as queries for additional data, emanate from these nodes and flow downward through the hierarchy.

To ensure that the hierarchy provides efficient communications paths and effective regional data aggregation as the network topology evolves, the hierarchy is formed and automatically restructured as needed using topologically-based clustering techniques augmented with other clusterhead selection criteria. These criteria can include proximity to network security specialists, penetration resistance (hardening), and each node's available communications, computation, and energy resources. The potential utility of the architecture has been illustrated via a range of usage scenarios including detecting deliberate packet dropping; attacks on MANET routing protocols; and attacks on conventional network and higher layer protocols, including attacks with distributed targets or sources.

These results go beyond previously published papers on intrusion detection architectures for MANETs in terms of breadth of applicability; lack of reliance on promiscuous eavesdropping (which may produce incomplete and misleading data); and novel features for assigning and seamlessly reassigning monitoring and detection responsibilities to nodes as a function of topological positioning, protocols, and other factors.

Our ongoing research includes quantitative comparison through discrete event simulation of the suitability and communications overhead of existing clustering algorithms and new hybrids when used to build dynamic intrusion detection hierarchies. Our future research plans include the development of Byzantine-resistant techniques for clustering and for distributed intrusion detection and correlation.

7. Bibliography

[1] Stefano Basagni, "Distributed Clustering in Ad Hoc Networks," *Proceedings of the 1999 Intl. Symp. On Parallel Architectures, Algorithms and Networks (I-SPAN '99)*, Freemantle, Australia, 1999.

[2] The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

[2] S. Buchegger and J. Boudec, “Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes - Fairness In Distributed Ad hoc NeTworks,”*Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, Lausanne, CH, June 2002.

[3] L. Buttyán and J.-P. Hubaux, “Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks,” *Technical Report No. DSC/2001/046*, Swiss Federal Institute of Technology, Lausanne, August 2001.

[4] Yi-an Huang and Wenke Lee. “A Cooperative Intrusion Detection System for Ad Hoc Networks.” *Proceedings of the ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN’03),* October 2003.

[5] O. Kachirski, R. Guha, “Effective Intrusion Detection Using Multiple Sensors in Wireless Ad Hoc Networks,” *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03)*, January 06 - 09, 2003.

[6] S. Marti, T. J. Giuli, K. Lai, and M. Baker, “Mitigating Routing Misbehavior in Mobile Ad Hoc Networks,” *Proceedings of the 6th Intl. Conference on Mobile Computing and Networking*, pp 255-265. Boston, MA, August 2000.

[7] P. Michiardi, R. Molva, “Core: A COllaborative REputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks,” *Proceedings of the Communication and Multimedia Security 2002 Conference*.

[8] P. Ning, K. Sun, "How to Misuse AODV: A Case Study of Insider Attacks against Mobile Ad hoc Routing Protocols," *Proceedings of the 4th Annual IEEE Information Assurance Workshop*, pages 60-67, West Point, June 2003.

[9] R. Rao and G. Kesidis, “Detection of malicious packet dropping using statistically regular traffic patterns in multihop wireless networks that are not bandwidth limited,” *Brazilian Journal of Telecommunications*, 2003.

[10] R. Ramanujan, S. Kudige, T. Nguyen, S. Takkella, and F. Adelstein, “Intrusion-Resistant Ad Hoc Wireless Networks”, *Proceedings of MILCOM 2002*, October 2002.

[11] Chin-Yang Tseng, Poornima Balasubramanyam, Calvin Ko, Rattapon Limprasittiporn, Jeff Rowe, and Karl Levitt, “A Specification-Based Intrusion Detection System For AODV,” *Proceedings of the ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN’03)*, October 2003.

[12] Y. Zhang and W. Lee, “Intrusion Detection in Wireless Ad Hoc Networks,” *Proceedings of The Sixth International Conference on Mobile Computing and Networking (MobiCom 2000)*, Boston, MA, August 2000.

[13] Snort™ Users Manual, Version 2.2.0, The Snort Project, 10th August 2004 http://www.snort.org/docs/snort_manual.pdf]

[14] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das. "Ad Hoc On Demand Distance Vector (AODV) Routing." *IETF RFC 3561.*

[15] T. Clausen and P. Jacquet, “Optimized Link State Routing Protocol,” IETF RFC 3626.

[16] P. Brutch and C. Ko, “Challenges in Intrusion Detection for Ad Hoc Networks,” *IEEE Workshop on Security and Assurance in Ad hoc Networks*, Orlando, FL, January 28, 2003.

[17] Konrad Wrona, “Distributed Security: Ad Hoc Networks & Beyond,” *PAMPAS Workshop*, Sept. 16/17 2002, London

[18] Vesa Karpijoki, “Security in Ad Hoc Networks,” http://citeseer.nj.nec.com/karpijoki01security.html