© 2015 IJIRT | Volume 1 Issue 12 | ISSN: 2349-6002

Anomalous Behavior Based Intrusion Detection in Wireless Sensor Network

M.Viswanathan
Assistant Professor, Department of Computer Science School of Engineering and Technology
Madawalabu University, Bale Robe, Ethiopia

***Abstract*— Wireless Sensor Network is easily vulnerable to intrusion due to its features of open medium, dynamic changing topology, cooperative algorithms, lack of centralized monitoring and management point, and lack of a clear line of defense. An intruder may drop, change or misroute the packet passing through the malicious node. Detection of abnormal behavior in the use of network services can be done using anomaly based intrusion detection. In this paper, we consider the problem of detecting the abnormal behavior of malicious node which drops packets destined to particular node. It is quite challenging to attribute a missing packet to a malicious action because normal network congestion can produce the same effect. Previous detection protocols have tried to address this problem with a user-defined threshold. However, this heuristic is fundamentally unsound and will certainly create unnecessary false positives or mask highly focused attacks. We have implemented a compromised node detection protocol that dynamically infers, based on measured traffic rates and buffer sizes, the number of congestive packet losses that will occur. Once the ambiguity from congestion is removed, subsequent packet losses can be attributed to malicious actions.**

***Index Terms*— distributed systems, intrusion detection, malicious nodes, reliable networks and tolerance.**

I. INTRODUCTION

Wireless sensor networks are a trend of the past few years, and they involve deploying a large number of small nodes. The nodes then sense environmental changes and report them to other nodes over flexible network architecture. Sensor nodes are great for deployment in hostile environments or over large geographical areas. Since the nodes are deployed in sparse area it is easily susceptible to external attacks. An intruder may drop, change or misroute the packet passing through the malicious node. Hence intrusion detection plays a major role in wireless sensor network. There are two basic approaches to intrusion detection: misuse intrusion detection and anomaly intrusion detection. In misuse intrusion detection, known patterns of intrusion are used to try to indentify intrusions when they happen. In anomaly intrusion detection, it is assumed that the nature of the intrusion is unknown, but that the intrusion will result in behaviour different from that normally seen in the system. In our work, anomaly intrusion detection approach is used.

II. BACKGROUND

There are inherently two sets of threats posed by an adversary: First, an attacker may subvert the network control plane and attack by means of the routing protocol. For example, by issuing false routing advertisements, a compromised node may manipulate how other nodes view the network topology, and thereby disrupt service globally. Second, an attacker may subvert the network data plane and attack by means of the packet forwarding process. By causing the node to violate the forwarding decisions that it should make based on its routing tables, a compromised node may disrupt communication in the network. Once a node has been compromised, an attacker may interpose on the traffic stream and manipulate it maliciously to attack others by selectively dropping, modifying, or re-routing packets.

By issuing false routing advertisements in control plane, a compromised node may manipulate other nodes' views of the network topology, and thereby disrupt service globally. For example, if a node claims that it is directly connected to all possible destinations, it may become a "black hole" for most traffic in the network. Perlman described robust flooding algorithms for delivering the key state across any connected network, and described a means for explicitly signing route advertisements. There have subsequently been a variety of efforts to impart similar guarantees to existing routing protocols with varying levels of cost and protection. Generally, these techniques break down into two categories: approaches based on ensuring the authenticity of route updates and those based on detecting inconsistency between route updates.

By contrast, the threat posed by subverting the forwarding process has received comparatively little attention until very recent years. This is surprising since, in many ways, this kind of attack presents a wider set of opportunities to the attacker not only denial-of-service, but also packet sniffing, modification and insertion and is both trivial to implement and difficult to detect.

III. SYSTEM MODEL

Our work proceeds from an informed, yet abstracted, model of how the network is constructed, the capabilities of the attacker, and the complexities of the traffic validation problem. In this section, we briefly describe the assumptions underlying our model.

A. Network Model

The bandwidth, the delay of each link, and the queue limit for each interface are all known publicly. Within a network, we presume that packets are forwarded in a hop-by-hop fashion, based on a local forwarding table. These forwarding tables are updated via DSDV routing protocol. This is critical, as we depend on the routing protocol to provide each node with a global view of the current network topology. Finally, we assume the administrative ability to assign and distribute cryptographic keys to sets of nearby nodes.

Every protocol assumes a synchronous network model of coarsely synchronized clocks and/or bounded message delays. This assumption is required by the protocols in order to decide whether a packet has been delivered within the expected time interval which is determined via timeout mechanism.

We define a path to be a finite sequence $\langle n_1; n_2; \ldots n_n \rangle$ of adjacent nodes. Operationally, a path defines a sequence of nodes a packet can follow. We call the first node of the path as source and the last node its sink; together, these are called terminal nodes. A path might consist of only one node, in which case the source and sink are the same. Terminal nodes are leaf nodes: they are never in the middle of any path.

An x−path-segment is a sequence of x consecutive nodes that is a subsequence of a path. A path-segment is an x−path-segment for some value of x > 0. For example, if a network consists of the single path <a, b, c, d> then <c, d> and <b, c> are both 2-path-segments, but <a, c> is not because a and c are not adjacent.

B. Threat Model

A node can be traffic faulty by maliciously dropping packets and protocol faulty by not following the rules of the detection protocol. Specifically, a node n is traffic faulty with respect to a path-segment π during τ if π contains r and, during the period of time τ, r exhibits anomalous behavior with respect to forwarding data that traverses π . For example, node n can selectively alter, misroute, drop, reorder, or delay the data that flows through π , and it can fabricate new data to send along π such that the packets, if they were valid, would have been routed through π . A node can drop packets without being faulty, as long as the packets are dropped because the corresponding output interface is congested.

A compromised node n can also behave in an arbitrarily malicious way in terms of executing the protocol we present, in which case we indicate n as protocol faulty. A protocol faulty node can send control messages with arbitrarily faulty information, or it can simply not send some or all of them. A faulty node is one that is traffic faulty, protocol faulty, or both. Attackers can compromise one or more nodes in a network. However, for simplicity, we assume in this paper that adjacent nodes cannot be faulty.

IV. POTOCOL VIEW

The problem of detecting and removing compromised nodes can be thought of as an instance of anomalous behaviour-based intrusion detection. That is, a compromised node can be identified by correct nodes when it deviates from exhibiting expected behaviour. This problem can be broken into three sub problems.

A. Traffic Validation

Traffic information is the basis of detecting anomalous behaviour: given traffic entering a part of the network, and an expected behaviour of the nodes in the network, anomalous behaviour is detected when the monitored traffic leaving that part of the network differs significantly from what is expected. Implementing such validation involves tradeoffs between the overhead of monitoring, communication and accuracy.

A compromised node can make arbitrary alterations to the forwarding behaviour of that node, but given the distributed nature of packet forwarding it is not possible in general for an adversary to perfectly conceal such behaviour. As long as the packets traverse some uncompromised node, there is enough data redundancy to detect the alteration. Hence, implementing a traffic validation mechanism is an engineering problem.

The most precise way to validate traffic is store, at each node, a complete copy of the packets sent and the time at which each was forwarded. However, the storage requirements to buffer these packets and the bandwidth consumed by resending them make this approach impractical. In practice, designing a traffic validation function is a trade-off between accuracy and overhead. In addition, real networks occasionally lose packets due to congestion, reorder packets due to internal multiplexing, and corrupt packets due to interface errors. Traffic validation needs to accommodate this abnormal but non-malicious behaviour. That is, one must address an inherent trade-off between an acceptable number of false positives and false negatives.

Consider the queue Q in a node n associated with the output interface of link $\langle n, n_d \rangle$ (see Fig. 1). The neighbor nodes $ns_1; ns_2 ; \ldots ; n_{sn}$ feed data into Q.

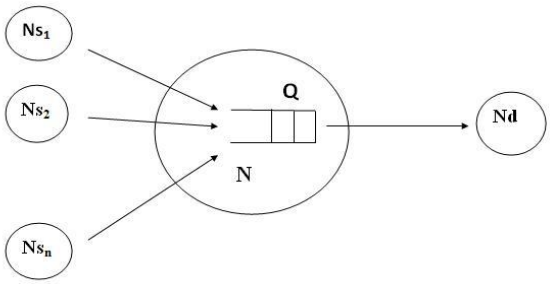

Fig.1 Validating the queue of an output interface

Traffic Validation TV can be implemented by simulating the behaviour of Q. Let P be a priority queue, sorted by increasing time stamp. All the traffic information source S and destination D are inserted into P along with the identity of the set (S or D) from which the information came. Then, P is enumerated. For each packet in P with a fingerprint fp, size ps, and a time stamp ts, q_{pred} is updated as follows. Assume t is the time stamp of the packet evaluated prior to the current one.

- If fp came from D, then the packet is leaving Q: $q_{pred}(ts) := q_{pred}(t) - ps$.
- If fp came from S and (fp Є D), then the packet fp is entering and will exit: $q_{pred}(ts) := q_{pred}(t) + ps$.
- If fp came from S and fp does not belongs to D, then the packet fp is entering into Q and the packet fp will not be transmitted in the future: $q_{pred}(ts)$ is unchanged, and the packet is dropped.
- If $q_{limit} < q_{pred}(t) + ps$, where q_{limit} is the buffer limit of Q, then the packet is dropped due to congestion. Otherwise, the packet is dropped due to malicious attack. Detect failure.

B. Distributed detection

The detection of a compromised node requires synchronizing the collection of traffic information and distributing the results for detection purposes. Since the behaviour of the queue is deterministic, the traffic validation mechanisms detect traffic faulty nodes whenever the actual behaviour of the queue deviates from the predicted behaviour. However, a faulty node can also be protocol faulty: it can behave arbitrarily with respect to the protocol, by dropping or altering the control messages of detection protocol X. We mask the effect of protocol faulty nodes using distributed detection.

Given TV, we need to distribute the necessary traffic information among the nodes and implement a distributed detection protocol. Every outbound interface queue Q in the network is monitored by the neighboring nodes and validated by a node n_d such that Q is associated with the link $<n; n_d>$.

There are two stages in distributed detection

1. Traffic information collection.
2. Information Dissemination and Detection.

Step 1. Traffic Information Collection

Each node collects the following traffic information during a time interval τ:

n_{s*}: Collect Tinfo ($n_{s*;}$ Qin; $<n_{s*;}$ n; $n_d>$; τ).
n: Collect Tinfo (n; Qin; $n_{s*;}$ $<n_{s*;}$ n; $n_d>$; τ). This information is used to check the transit traffic information sent by the n_{s*} nodes.
n_d: Collect Tinfo (n_d; Qout; h<n; $n_d>$; τ).

Step2. Information Dissemination and Detection

- n_{s*}: At the end of each time interval τ, node n_{s*}sends [Tinfo ($n_{s*;}$ Qin; $<n_{s*;}$ n; $n_d>$; τ)] n_{s*} that it has collected. $[M]_*$ is a message M digitally signed by x. Digital signatures are required for integrity and authenticity against message tampering.

1. Detection -I. n: Let ▲ be the upper bound on the time to forward traffic information.

   a. If n does not receive traffic information from n_{s*} within ▲, then n detects $<n_{s*,}$ n>.

   b. Upon receiving sends [Tinfo ($n_{s*;}$ Qin; $<n_{s*;}$ n; $n_d>$; τ)] n_{s*}, node n verifies the signature and checks to see if this information is equal to its own copy Tinfo (n; Qin; $n_{s*;}$ $<n_{s*;}$ n; $n_d>$; τ). If so, then r forwards it to n_d. If not, then n detects $<n_{s*,}$ n>.

   At this point, if n has detected a failure $<n_{s*,}$ n> then it forwards its own copy of traffic information Tinfo (n; Qin; $n_{s*;}$ $<n_{s*;}$ n; $n_d>$; τ). This is required by n_d to simulate Q's behavior and keep the state q up to date.

2. Detection -II. n_d :

   a. If n_d does not receive traffic information Tinfo (n; Qin; $n_{s*;}$ $<n_{s*;}$ n; $n_d>$; τ) originated by n_{s*} within 2▲, then it expects n to have detected n_{s*} as faulty and to announce this detection through the response mechanism. If n does not do this, then n_d detects $<n, n_d>$.

   b. After receiving the traffic information forwarded from n, n_d checks the integrity and authenticity of the message. If the

digital signature verification fails, then n_d detects <n, n_d>.

c. Collecting all traffic information, node nd evaluates the TV predicate for queue Q. If TV evaluates to false, then nd detects <n, n_d>.

C. Response

Once a node n detects node n' as faulty, n announces the link <n', n> as being suspected. This suspicion is disseminated via the distributed link state flooding mechanism of the routing protocol. As a consequence, the suspected link is removed from the routing fabric. A protocol faulty node r can announce a link <n', n> as being faulty, but it can do this for any routing protocol. And, in doing so, it only stops traffic from being routed through itself. Node n could even do this by simply crashing itself. To protect against such attack, the routing fabric needs to have sufficient path redundancy.

V. EVALUATION

A. Network with no model

We had investigated how accurately the protocol predicts the queue lengths of the monitored output interfaces. We considered the results for the output interface Q of the compromised node n associated with the link <n, s>. Background traffic was created to make <n,s> a bottleneck.

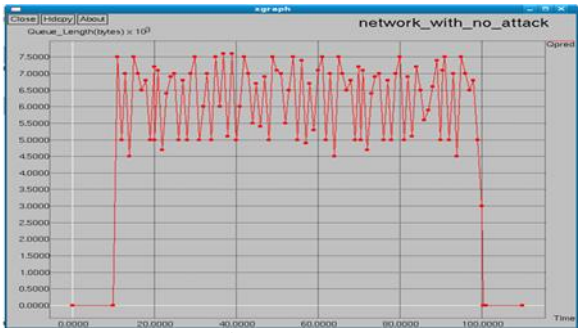

Fig. 2 Q pred of network with no attack

The result of no attack run is shown in Figure 2. q_{pred} is the predicted queue length of Q computed by source node s executing the protocol . q_{act}, which is the actual queue length of Q recorded by compromised node n, is not shown in the graph because it is so close to q_{pred}.

B. Detecting Attacks

We then experimented with the ability of protocol to detect attacks. In these experiments, the compromised node n is compromised to attack the traffic selectively in various ways, targeting the chosen two ftp flows.

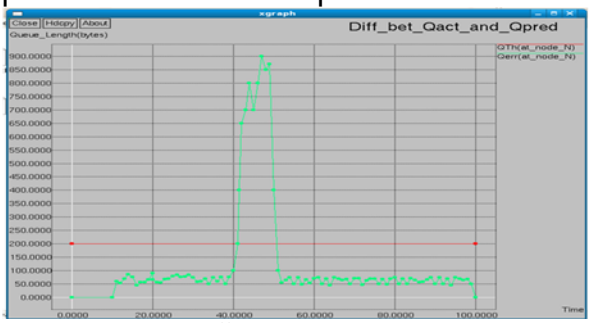

Fig. 3 Difference between q_{pred} and q_{act}

VI. CONCLUSION

Anomaly intrusion detection helps in the detection of abnormal behaviour in the use of network services and computing resources. The problem of detecting whether a compromised node is maliciously manipulating its stream of packets is focused here. It is quite challenging to attribute a missing packet to a malicious action because normal network congestion can produce the same effect. Compromised node detection protocol that dynamically infers, based on measured traffic rates and buffer sizes, the number of congestive packet losses that will occur. If we made clear that the packet loss is not due to congestion, subsequent packet losses can be attributed to malicious actions. Compromised node detection protocol does not suffer from the limitations of static thresholds.

REFERENCES

[1] A.T. Mizrak, Y.-C. Cheng, K. Marzullo, and S. Savage, "Detecting and Isolating Malicious Routers", IEEE Trans. on Parallel and Distributed systems, vol. 20, no.2, February 2009.

[2] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. Katz, "Listen and Whisper: Security Mechanisms for BGP", Proc. First Symp. Networked Systems Design and Implementation (NSDI '04), Mar. 2004.

[3] B.R. Smith and J. Garcia-Luna-Aceves, "Securing the Border Gateway Routing Protocol", Proc. IEEE Global Internet, Nov. 1996.

[4] Yih Chun Hu, A. Perrig, and D.B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks", Proc. ACM MobiCom '02, Sept. 2002.

[5] S. Cheung, "An Efficient Message Authentication Scheme for Link State Routing", Proc. 13th Ann. Computer Security Applications Conf. (ACSAC '97), pp. 90-98, 1997.

[6] S. Cheung and K.N. Levitt, "Protecting Routing Infrastructures from Denial of Service Using Cooperative Intrusion Detection", Proc. Workshop on New Security Paradigms (NSPW '97), pp. 94-106, 1997.

[7] K.A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R.A. Olsson, "Detecting Disruptive Routers: A Distributed Network Monitoring Approach", Proc. IEEE Symp. Security and Privacy (S&P '98), pp. 115-124, May 1998.

[8] J.R. Hughes, T. Aura, and M. Bishop, "Using Conservation of Flow as a Security Mechanism in Network Protocols", Proc. IEEE Symp. Security and Privacy (S&P '00), pp. 131-132, 2000

[9] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy." Highly secure and efficient routing", In Proceedings of INFOCOM 2004 Conference, March 2004.

[10] A. Herzberg and S. Kutten. "Early detection of message forwarding faults". SIAM J. Comput., 30(4):1169–1196, 2000.

[11] K. Argyraki, P. Maniatis, D. Cheriton, and S. Shenkerm, "Providing packet obituaries", In Proceedings of ACM SIGCOMM HotNets-III, 2004.

[12] C. N.-R. Baruch Awerbuch, David Holmer and H. Rubens. "An on-demand secure routing protocol resilient to byzantine failures", In ACM Workshop on Wireless Security (WiSe), September 2002.

[13] V. N. Padmanabhan and D. R. Simon. "Secure traceroute to detect faulty or malicious routing", SIGCOMM Computer Communications Review, 33(1):77–82, 2003.

[14] I. Avramopoulos and J. Rexford. "Stealth Probing: Efficient Data-Plane Security for IP Routing", In Proc. USENIX Annual Technical Conference, May-Jun 2006.

[15] R. Perlman. "Network Layer Protocols with Byzantine Robustness", PhD thesis, MIT LCS TR-429, Oct. 1988.

[16] R. Perlman. "Interconnections: Bridges and Routers", Addison Wesley Longman Publishing Co. Inc., 1992.

[17] N. G. Duffield and M. Grossglauser. "Trajectory sampling for direct traffic observation", IEEE/ACM Transactions on Networking, 9(3):280–292, 2001.