Securing WSN update from Intrusion using Time Signature of Over the Air Update Protocol

A S M Ashraful Alam **David Eyers**

Department of Computer Science, University of Otago, New Zealand
Email: {aalam, dme}@cs.otago.ac.nz

Abstract

While over the air (OTA) software updates makes certain WSN management tasks easy, they are vulnerable to intrusion attempts. An adversary can take advantage of the resource-constrained nature of the sensor motes to break through the limited protection the sensors have. To deal with this problem, we have suggested a passive security mechanism—an IDS that works on a timing analysis principle.

Keywords: Sensor, Wireless Sensor Network, WSN, Intrusion Detection System, IDS, Security

1 Introduction

Wireless sensors or motes are deployed in large numbers in uncontrolled environments, which makes them difficult to manage. Tasks such as bug-fixing are often managed using over the air (OTA) software update protocols such as Deluge (Hui & Culler 2004). There are risks and vulnerabilities associated with Wireless Sensor Network (WSN) OTA software updates that can enable an attacker to steal sensitive information, log network events, or cause denial of service (Dutta et al. 2006). An adversary can take advantage of the resource-constrained nature of the sensor motes to break through the limited protection that they have.

Security threats to WSNs are different in nature from threats to the Internet or even other wireless technologies. Implementation of security techniques in WSNs are characterised by constrained resources. As a result, well developed cryptographic security mechanisms such as WPS and PKM are inappropriate. Wireless channels offer little physical protection. Moreover, the capability of computing devices that can computationally compromise the cryptographic protection in sensors is growing rapidly. As a result, WSNs are more vulnerable to intrusion attempts and security threats from attackers. An Intrusion Detection System (IDS) can detect break-in events and raise alerts, possibly also estimating the location and extent of the problem.

Proposed IDS techniques vary greatly in their approaches. Misuse-based detection techniques aim to detect known attacks by identifying undesired activities based on usage signatures. However, they are ineffective against previously-unseen attacks and may wrongly identify legitimate activities as an intrusion because of similarities to existing signatures. Conversely, anomaly-based IDSs build statistical signatures based on a definition of normal activity. Unfortunately, a system may display previously unknown behaviour e.g., rare critical events like earthquakes or tsunamis may raise false positive intrusion alerts within environmental monitoring sensors. On the other hand, an intrusion that masks its pattern under the hood of normal behaviour, for example: increased packet traffic during OTA update, would go undetected. Specification-based and reputation-based techniques rely on manual input for defining normal and anomalous behaviour with lesser reliance on automation. However, these techniques have not been proved to be very effective because human interaction are more prone to errors and require more learning time (Wang & Zhang 2009, Roman et al. 2006, Doumit & Agrawal 2003, Chen et al. 2009).

Onat and Miri present an IDS design in which each node builds up statistics about received packets from its neighbours. If packets received from any neighbour display anomalous patterns after a predefined number of consecutive packets, intrusion alarms may be raised (Onat & Miri 2005). Unfortunately, packet count statistics are not suitable in OTA software update scenarios due to the significant increase in packet traffic from the update itself. However, this statistical treatment motivates us to employ timing analysis within OTA update protocols.

In this paper, we present an IDS that can detect intrusion in case of WSN software update. The algorithm requires update protocol to send back a datagram containing the update time to work out a 'Surprise Score' or 'Intrusion Warning Score' (IWS) (Alam et al. 2015). Our main contributions are: the technique identifies anomalies in software update patterns and scores them quantitatively; we demonstrate that simulation can indicate the positions of WSN nodes that will best support the ability of similar IDSs to detect anomalies. The rest of this paper is organised as follows. In Section 2, we give a brief overview of the system design and the experiment methodology. Then we analyse the experimental results and present our evaluation in Section 3. Finally, we conclude our arguments in Section 4.

2 Methodology

System Design

The IDS has three kinds of components—i) motes in radio network; ii) an IDS transport client (ITC) application in the sink; and iii) an IDS application that runs on an system with higher resources, coordinates with ITC, remains physically attached to the sink,

and houses a network database (NDB).

Following activities are necessary for the intrusion detection technique to work—i) Deluge protocol initiates update; ii) Deluge disseminates the new software image using a multi-hop relay mechanism; iii) nodes deliver a datagram to the sink containing their timing information; iv) packets from the nodes rely on other nodes *en route* to reach to the sink; and v) the ITC application in the sink delivers relevant information to the IDS for processing

After checking for correctness, the ITC then inserts the update time and mote ID into the NDB in the IDS. The IDS maintains an identifier to differentiate among different update times from different runs for the same mote. The IDS uses timing measurements from authenticated initial software update runs to build signatures. The time analyser module in the IDS inspects the NDB once an update activity has been detected. It determines the changes from the signature and calculates IWS based on following utility function:

$$IWS = \sum_{i=0}^{n} \frac{|\mu_i - t_i|}{\sigma_i + 1} \quad (1)$$

where, i—mote index; t_i—image update time at mote i; μ_i—mean update time at mote i; and σ_i—standard deviation of update time at mote i.

The system has been described elaborately in (Alam et al. 2015).

Threat Model

The IDS is expected to handle malicious node update attempts, node compromise, and node failure. We assume that an attacker can take over one or more nodes deployed in hostile environment, access the code in a compromised node, and use cryptanalysis to represent a false identity without physically compromising a node. However, he has no special access to the network and is unable to modify the underlying protocols running in the uncompromised nodes. We assume a static WSN with a single sink that cannot be compromised. All motes are assumed to have time synchronisation accurate to 1 μs, e.g., using TSMP.

Experiment Settings

We used 'Cooja'—a network simulator in Contiki v2.7 running on Ubuntu 12.04, to simulate the software update pattern associated with Deluge protocol. The simulations evaluated the IDS in different WSN topologies and explored the effect of parameters such as power level. Evaluating the IDS on each of the topologies consisted of several test runs that were of two kinds: aimed at building signatures consisting of 20 individual simulations initiated from the sink; and simulations that replicated intrusion from nodes other than the sink. We explored different topologies such as: linear, circular, elliptical, double rings, wide line, grid, tree and other random topologies, but do not have space to discuss them in detail here. The motes were placed at a distance of 30–35 meters to allow multiple transmission paths to all motes. For most experiments, the radio transmission power level was set at 100% which has an approximate maximum effective transmission range of 50 meters and is able to interfere with other motes up to 100 meters away. The 'Owheo WSN'[1] topology was additionally tested at 50% power level.

[1]WSN deployed in the Owheo Building at the University of Otago. It is intended to be used for research purposes.

Experimental Procedure

Step 1 All motes in the WSN were booted up at same time running one version of a diagnostic application that reports its time, version number and mote ID every second on its serial terminal.

Step 2 The sink dynamically began the update process by propagating another version of the software image using Deluge protocol. Whenever the transfer to a mote was complete, the new image would reboot to begin executing. In a real system, the modified Deluge protocol that runs with the OS would send a report datagram to the sink. A similar technique has been employed for WSN failure detection in real system by (Kamal et al. 2014). The simulator continued until all the motes in the WSN were updated and the times of their updates were logged.

Step 3 Processed data were statistically treated to build the signature consisting of mean and standard deviation of mote update time. Changes to the topology or configuration of the WSN would necessitate recalibrating the signature.

Step 4 Intrusion attacks at different nodes were replicated in simulations. The individual simulation runs were treated in a fashion similar to Step 1 and Step 2. The update time recorded at each of the motes was then analysed using Equation 1 to produce IWS. The resulting values from this function presented an approximation of the abnormality of the new update pattern. The higher the IWS, the more unusual the pattern was.

Step 5 We classified the IWS values into Intrusion Warning Zones (IWZ) based on a scale built from maximum expected IWS for a topology to imply certain warning levels. The heuristics follows from the feasibility of finding an easily identifiable boundary. The IDS marks an IWS up to 10% from the scale as GREEN to indicate safe zone, 10%–30% as GREY to indicate a situation where rules are not known and above 30% to be possible intrusion and is marked as RED zone. Forming IWZs requires the IDS to have knowledge of IWS for intrusions at each individual node in the network. Such knowledge can be built by replicating intrusion activity from the furthest few nodes.

3 Results and Evaluation

While evaluating all experimented topologies is useful, the results from Double Ring topology only is presented. The topology has 46 nodes in two equicentric circles (Figure 1). The circles are within each others' transmission range. Table 1 shows the signature data: all times are in seconds. It is observed that, standard deviation increases gently with distance. In contrast to this, motes closer to the sink have lower mean which follows a steep rise with the increase in distance.

Figure 1 shows the placement of the motes with mote ID written at the centre and IWS beneath the tiny circle. The motes represented an intrusion point in the process of measuring IWS using Equation 1. The IWS increases linearly with the increase in distance from the sink. If the IWS in a network is organised in a specific order, e.g., chronologically along some shape, it would represent the original topology in some fashion. For example, in the double ring

Node ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
μ	0	19	79	143	197	268	340	404	466	539	594	547	476	408	326	268	210
σ	0	2	11	12	13	25	27	31	30	38	37	49	37	35	25	24	23

Table 1: Statistics of mote update times in Double Ring topology (partial data presented)

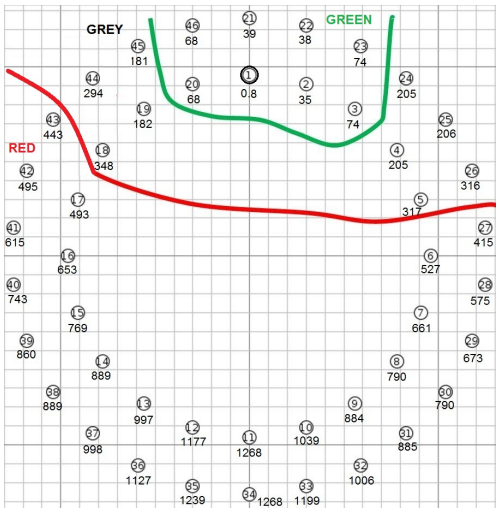

Figure 1: IWS of intrusion initiated at each node and placement of nodes in Double Ring Topology

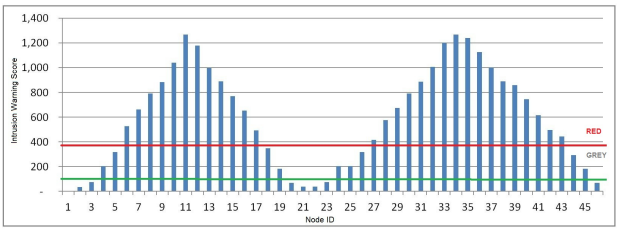

Figure 2: Comparison of IWS of intrusion initiated at different motes

topology, the circles are represented by the bimodal curve formed from the IWS represented by columns in Figure 2.

The main contribution of this research comes from the ability to report anomalies in terms of the quantitative IWS measure. For example: a regular legitimate update, initiated from node ID 1, resulted in a IWS of 0.8. This result matches the theoretical expectation of very low IWS. In addition to this, for intrusion scenario simulated from the nodes near the sink, the IWS was still fairly low (e.g. at mote ID 2, 3, 20–23, and 46). Because the IWS of the motes close to the sink is likely to be very low, it is quite difficult to conclude if such an IWS indicates an intrusion. This area is marked as being within the GREEN zone, and needs to be adequately secured with additional physical security as intrusion in these motes is not easily detectable by the IDS. For other motes, such as mote ID 4, 5, 24–26, 18, 19, 44 and 45, the IWS is neither too low, nor too high to be considered an intrusion. The area is marked as GREY to indicate the requirement of additional monitoring to reduce false positive cases. However, for the distant motes–marked in the RED zone, the IWS is quite high and the IDS can conclude such cases are intrusions, such as the IWS of 1268 for mote ID 34, and 885 for mote ID 31. Accordingly, Figure 1 and 2 shows the region boundaries and thresholds of GREEN, GREY and RED zones.

The IDS can also detect the source of intrusion by analysing update time of all motes. For example, Table 2 shows the update time recorded at different motes when intrusion was initiated from note 14. The timing data shows that lowest timing was recorded at node 13, 15, 37, 38, and 39 which was ≈16 seconds. Node 14 did not report any time because of three reasons: 1. node 14 initiated the update, so it was updated at 0 seconds; 2. it acted as a sink, the protocol did not require it to notify update time; and 3. it is the compromised node whose protocol is expected to be altered. From the timing data, the IDS can conclude that node 14 was the intrusion point. The extent of intrusion, as specified earlier, is found out from reported version information.

The idea of zoning associated with the IWS is useful in designing secure WSN deployments. A network is more secure when it has a smaller GREEN and GREY zone, which implies that fewer resources will be needed to strengthen the security of these motes. Obtaining a real world scenario by varying different sets of design parameters such as power levels or position of nodes is an impossible task even for a small WSN. The simulation environment used to design the IDS can be effectively utilised to obtain the expected IWS score range for different sets of parameters and then rank the sets based on how effectively they produce IWS values. However, any change in topology or design parameters, like power level, will require recalibrating the IDS. For example, we have simulated the set of IWS outcomes within the 'Owheo Sensor Network' at two different power levels to investigate improving the security of our WSN using the IDS.

Table 3 shows the IWS at different motes within the Owheo WSN as intrusion points at 100% and 50% power levels. At 100% power level, only 16 nodes are in the RED zone. When the same deployment is simulated at 50% power level, 27 motes are in the RED zone. The deployment and the zoning boundaries at 100% and 50% power level have been shown in Figure 3 and 4 respectively. Comparing the scenarios, it can clearly be established that deployment at 50% power level is more secure than the other for two reasons: 1. it has a GREEN and GREY zone of smaller size; and 2. there are fewer nodes in these zones.

4 Conclusion

In this paper, we modelled an IDS solution to detect intrusions in WSNs during OTA software update. We discussed the potential dangers associated with such updates. The IDS is hosted on a server that remains physically connected to the sink. The system watches over the timing of software update patterns using a modified version of the Deluge protocol.

The first contribution of the research is in quantifying intrusion through an IWS that indicates the location and extent of intrusions. The other important contribution is related to designing secure WSNs using the IWZ metric as a guide. The IDS also can indicate some other kinds of anomalies such as node relocation, node repudiation or node compromise.

The proposed technique has several limitations. It assumes some special situations, such as a static WSN with a single sink. It also assumes that network-wide information will not be forged. These assumptions

Node ID	1	4	10	12	13	15	16	18	27	31	34	36	37	38	39	40	41
Update Time	404	588	170	66	16	16	65	204	526	329	113	66	15	16	16	65	125

Table 2: Mote update time for intrusion at node 14 in Double Ring topology (partial data presented)

Node ID	1	2	3	4	5	6	7	8	9	10	11	33	34	35	36	37	38
Power 100%	512	529	512	512	512	512	476	478	476	459	458	53	48	49	51	47	29
Power 50%	31	59	254	31	31	283	32	32	254	31	253	31	30	31	31	30	0

Table 3: IWS from intrusion at motes in Owheo WSN (partial data presented). Further explained in Figure 3 and Figure 4.

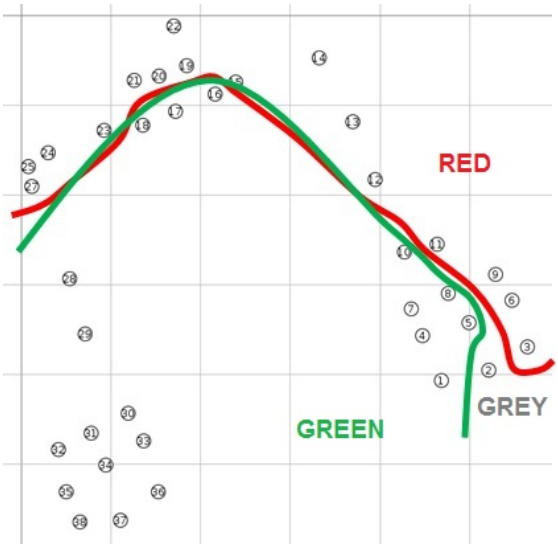

Figure 3: Zone boundary at full power level

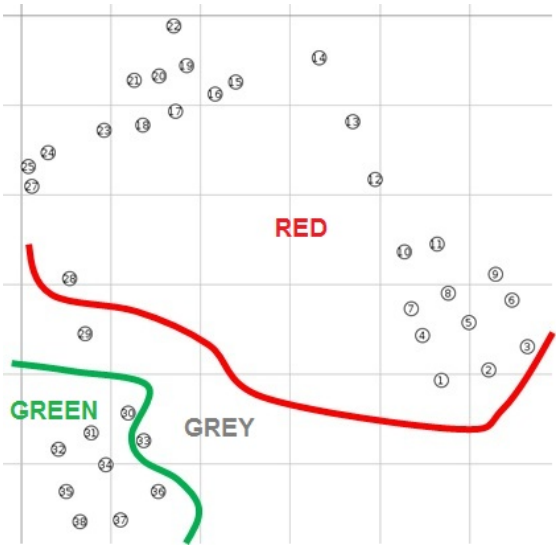

Figure 4: Zone boundary at half power level

may not hold in a real WSN and can be addressed in a more realistic way using a decentralised technique, which we plan to experiment in future. In addition to this, assumptions about the attacker are: the attacker can employ enough resources to break into the cryptographic protections within the sensors; however, he is not able to modify the protocols in an uncompromised sensor. The IDS has an obvious functional limitation. It cannot identify an intrusion in certain scenarios such as in the close vicinity of the sink.The other limitation stems from the fact that the results are based on simulated data and need support from real world implementation. In future, the IDS can be implemented to examine real world performance including cases of multiple sinks with mobile sensors.

References

Alam, A. S. M. A., Eyers, D. & Huang, Z. (2015), Securing robots in WSN environment through intrusion detection of WSN software update, *in* 'International Conference on Automation, Robotics and Applications (ICARA-2015)', IEEE. Under review.

Chen, R.-C., Hsieh, C.-F. & Huang, Y.-F. (2009), A new method for intrusion detection on hierarchical wireless sensor networks, *in* 'Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication', ICUIMC '09, ACM, New York, NY, USA, pp. 238–245.

Doumit, S. & Agrawal, D. (2003), Self-organized criticality and stochastic learning based intrusion detection system for wireless sensor networks, *in* 'Military Communications Conference, 2003. MILCOM '03. 2003 IEEE', Vol. 1, pp. 609–614 Vol.1.

Dutta, P. K., Hui, J. W., Chu, D. C. & Culler, D. E. (2006), Securing the deluge network programming system, *in* 'IPSN '06: Proceedings of the fifth international conference on Information processing in sensor networks', ACM, New York, NY, USA, pp. 326–333.

Hui, J. W. & Culler, D. (2004), The dynamic behavior of a data dissemination protocol for network programming at scale, *in* 'SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems', ACM, New York, NY, USA, pp. 81–94.

Kamal, A. R. M., Bleakley, C. J. & Dobson, S. (2014), 'Failure detection in wireless sensor networks: A sequence-based dynamic approach', *ACM Trans. Sen. Netw.* **10**(2), 35:1–35:29.

Onat, I. & Miri, A. (2005), An intrusion detection system for wireless sensor networks, *in* 'Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005), IEEE International Conference on', Vol. 3, pp. 253–259 Vol. 3.

Roman, R., Zhou, J. & Lopez, J. (2006), Applying intrusion detection systems to wireless sensor networks, *in* 'Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE', Vol. 1, pp. 640–644.

Wang, Q. & Zhang, T. (2009), *Security in RFID and Sensor Networks*, Auerbach Publications, chapter A Survey on Security of Wireless Network, pp. 293–320.