A Lightweight Protocol for Data Integrity in Sensor Networks

[#] Arjan Durresi, Vamsi Paruchuri, Rajgopal Kannan, S.S. Iyengar
Department of Computer Science, Louisiana State University
Baton Rouge, LA 70803, USA,
{*durresi, paruchuri, rkannan, iyengar* }*@csc.lsu.edu*

Abstract

In this paper, we present a novel lightweight protocol for data integrity in wireless sensor networks. Our protocol is based on leapfrog strategy in which each cluster head verifies if its previous node has preserved the integrity of the packet using the secret key it shares with two hop uptree node. The proposed protocol is simple. The analysis and simulation results show that the protocol needs very few header bits, as low as three bits, thus resulting in negligible bandwidth overhead; the protocol poses very low computational overhead, it needs to compute just a hash as compared to multiple complex operations required by any cryptographic implementation for verifying authenticity.

1. INTODUCTION

Wireless sensor networks have recently emerged as a critically important disruptive technology resulting from the fusion of wireless communications and embedded computing technologies [1], [2], [3], [4], [5], [6]. In the future we envision thousands to millions of small sensors form self-organizing wireless networks. Potential applications include monitoring remote on inhospitable locations, target tracking in battlefields, disaster relief networks, early fire detection in forests, and environmental monitoring.

Security is a crucial part of the architectures for sensornet. Sensor networks are vulnerable to a vast number of security threats [7], [8], [9], [10] with variable application-specific attack mechanisms and variable impact on the network. Due to their nature and operational resource constraints sensor networks are vulnerable to various types of attacks. While designing the new network architecture for future sensors networks, the research community has a unique chance to integrate security and privacy since the beginning as a fundamental part of the architecture. As shown by the Internet example, security cannot be implemented properly as patches to an existing network architecture, rather security mechanisms must developed as part of an integral security framework.

Wireless networks, in general, are more vulnerable to security attacks than wired networks, due to the broadcast nature of the transmission medium. Furthermore, wireless sensor networks have an additional vulnerability because nodes are often placed in a hostile or dangerous environment where they are not physically protected. Note that security issues in ad-hoc networks are similar to those in sensor networks and have been well enumerated in the literature [11], but the defense mechanisms developed for ad-hoc networks are not directly applicable to sensor networks. For example, some ad-hoc network security mechanisms for authentication and secure routing are based on public key cryptography [8], [12], [13], [14], [15], [16], [17], [18] which is too expensive for sensor nodes. Similarly, security solutions for ad-hoc networks based on symmetric key cryptography have been proposed [19], [20], [21], [22]. They are too expensive in terms of node state overhead and are designed to find and establish routes between any pair of nodes-a mode of communication not prevalent in sensor networks. The authors in [23], [24] consider the problem of minimizing the effect of misbehaving or selfish nodes through punishment, reporting, and holding grudges. The application of these techniques to sensor networks is promising, but these protocols are vulnerable to blackmailers.

There are several recent research efforts exploring different aspects of sensornet security, for example key management, secure multicast communication, authentication and anonymous routing [25]. Among the original sensornet security solutions, SPINS [26] presents two building block security mechanisms for use in sensor networks, SNEP and μ-TESLA. SNEP provides confidentiality and authentication between nodes and the sink, and μ-TESLA provides authenticated broadcast.

Sensor networks are expected to consist of hundreds to thousand of nodes dispersed in hostile environments. It is clearly impractical to monitor and protect each individual node from physical or logical attack. An enemy can easily alter existing data or even inject spurious data in the sensornet by capturing or insert new malicious nodes into the network. A key technical challenge is to detect such activity by distinguishing fake/altered data from the correct one and identifying the malicious nodes. In data-centric sensor networks, data is typically aggregated for energy-efficiency [27]. Since sensor networks are highly unstructured, both routing and aggregation of data occurs in an ad-hoc manner depending on current resource distributions and current (localized) sensing activity. It is therefore extremely difficult to identify vulnerable nodes/network zones apriori. Therefore there is a need to develop a broad spectrum of dynamic defense mechanisms for detecting such malicious behavior.

In this paper, we propose a new lightweight security proto-

0-7803-8894-1/04/$20.00 © 2004 IEEE

ISSNIP 2004

col to provide data integrity. Data integrity is the assurance that the data received by the destination is the same as generated by the source. Data Integrity ensures that data is unchanged from its source and has not been accidentally or maliciously altered. Integrity attacks modify content without the knowledge or permission of the owner. The key advantages of the protocol are: 1) The protocol is simple; 2) it needs very few bits in the header (as low as three bits). This results in negligible bandwidth overhead; 3) the protocol poses very less computational overhead (it needs to compute just a hash as compared to multiple complex operations required by any cryptographic implementation for verifying authenticity).

The rest of the paper is organized as follows: Section 2 describes current network security trends specially for data integrity. Section 3 describes our protocol for providing integrity. In Section 4 we present the communication and computational overhead imposed by our protocol and analyze the performance. Finally, we conclude in Section 5.

2. Related Work

Data integrity is the assurance that the data received by the destination is the same as that generated by the source and has not been accidentally or maliciously altered enroute. Integrity attacks modify content without the knowledge or permission of the owner.

The security community has paid vast attention to confidentiality issues, which are solved through encryption of data transmissions such as email or encrypting files in storage. While encryption has been possible for decades, this security technique lags in implementation in sensor networks due to complex key management and low processing and memory capabilities of sensor networks. Asymmetric cryptographic techniques might not be possible at all in sensor networks [26]. Symmetric cryptographic techniques though implementable, still consume lot of energy. The issue of denial of service attacks began to be solved through better intrusion detection, high-speed reaction mechanisms, redundancy, fault tolerance, better disaster planning and system reconstitution.

Integrity mechanisms have been part of the computer security professional's arsenal in many forms. The simplest method is called CRC or a Cyclic Redundancy Check. The contents of the file are XORed with another set of (random) data and the results create an integrity key. When the reverse CRC process is run, and if the integrity key doesn't match the original, the file has been corrupted in some form and cannot be trusted.

A stronger integrity method is called Message Authentication Code (MAC) [28], a cryptographic technique that is based on the Data Encryption Standard. Again, a key is generated when the file is 'sealed'. Upon decoding, the key must match if the files are to be trusted. MAC is based on a secret key shared between the communicating parties, i.e., source and destination. Key distribution in sensor networks is an on going research issue [29], [30], [31].

Though, cryptographic techniques can ensure complete security, they are very computational intensive and consume lot of energy. Moreover, in many scenarios, all security issues need not be addressed. For instance, consider a sensor network deployed for intrusion detection. Once a sensor *S* detects an intruder, it sends an alert message to the base station. Encrypting the alert message need not essentially prevent someone from realizing the contents of the message itself. For another scenario, consider a sensor network deployed to detect fires by monitoring the temperature. Once the base station gets a packet from a sensor that has a temperature reading greater than some threshold, a warning might be issued. In these cases, it is more important that message integrity is ensured than message secrecy.

3. Data Integrity-Lightweight Network Layer Security

We present a lightweight algorithm to preserve the integrity of messages in a sensor network even in presence of compromised nodes. Our protocol prevents compromised nodes from changing the contents of a packet. Our mechanism can be used upon any other security protocol with slight modifications. Our mechanism can be modified to work even with as low as three bits. Even with just three bits in header, a compromised node could send only a few packets (less than 10 packets in 99.9% of cases) before being detected.

Our schemes add little or no overhead to the node's critical forwarding path. In fact, the only invariant that we can depend on is that a packet from the attacker must traverse all of the nodes between it and the victim.

A. Assumptions

We assume a sensor network that is logically represented as a set of clusters. Several protocols have been proposed to efficiently divide the network into clusters and elect cluster heads [32], [33], [34]. The cluster heads form a d-hop dominating set. A node either becomes a cluster head, or is at most d hops from a cluster head. Cluster heads form a virtual backbone and may be used to route packets for nodes in their cluster. The value of d is a parameter of the network. We assume that multiple malicious nodes might be present in the network and the nodes would not collaborate with each other.

B. The protocol

We propose an effective mechanism to prevent compromised router nodes from modifying the contents of a packet. Our mechanism can work even with as low as three bits as we illustrate later in this section. With just three bits, a compromised router could send few packets (less than 10 packets in 99.9% of cases) before being detected. We first present a generalized version of our mechanism in which we assume the available header space to be $2t + 1$ bits. Later in the section, we examine the different choices of t.

We divide the $(2t + 1)$-bit header space into three fields a t-bit One-hop Neighbor Authentication Field (ONAF), a t-bit Two-hop Neighbor Authentication field (TNAF) and a 1-bit flag field as shown in Table 1.

Our scheme is based on a lightweight strategy. We define for each cluster head x the set $N(x)$, which contains the nodes

TABLE 1: HEADER SPACE ALLOCATION FOR DIFFERENT FIELDS FOR AUTHENTICATION PURPOSE

ONAF	TNAF	Flag
4 bits	4 bits	1 bit

in G that are neighboring cluster heads of x (which does not include the x itself). That is,

$$N(x) = y : (x, y) \in E \; and \; y \neq x$$

The security of our scheme is derived from a secret key $k(x)$ that is shared by all the cluster heads in $N(x)$, but not by x itself. This key is created in a setup phase and distributed securely to all the members of $N(x)$. Note, in addition, that $y \in N(x)$ if and only if $x \in N(y)$.

When a cluster head s wishes to send a packet P to be forwarded by a neighboring cluster head, x, it sets the above fields as follows:

$$ONAF = h[P, k(x)]$$

$$Flag = 0$$

where h is a cryptographic keyed-hash function that is collision resistant using a key. s and d are the source and destination addresses present in the packet P and the T is the marking in the packet used for traceback and set by the underlying traceback mechanism being used. When the cluster head x receives a packet P from one of its neighbors y, it verifies the authenticity of the packet as follows:

Node x computes $h[P, k(y)]$ and compares it with TNAF.

- If the are same, then x does the following operations:

$$TNAF \leftarrow ONAF$$

$$ONAF \leftarrow h[P, k(z)]$$

z being the cluster head to which the packet is being forwarded by x

$$Flag = 1$$

to indicate that it is not the originator of the packet.

- If the values are different, and if the Flag is not set to 0 then immediately x can decide that y has been compromised.
- If Flag is set to 0, then x definitely marks the packet.

The protocol follows a leap frog approach. Each cluster head verifies if the packet was modified by previous node by checking the hash value of the packet generated by the *up tree* node that is two hops far away from it. If the verification fails, the previous node has either originated the packet (which is indicated by the flag) or has modified the packet.

4. ANALYSIS AND RESULTS

In this section, we analyze the overhead (bandwidth and computational) and the performance of our protocol.

A. Bandwidth Overhead

We present some simulation results on number of header bits the protocol needs. A node malicious y can successfully escape from being detected with a probability of $1/2t$. When $t = 4$, this probability is $1/16$, and the node will be discovered with a probability of $15/16$. The probability of y passing this test for more than three packets is less than 0.00025. That is, in more than 99.97% of cases, y will be discovered even before it could modify three packets. To generalize, the probability that a node can change and send p packets without being detected is $(1/2^t)^p$. Figure 1 illustrates this for different values of t and p. It should be noted that even with $t = 1$, 99.6% of times, a malicious node will be detected even before it can change 8 packets. In Figure 2 it is shown the required size of the header field (t) so as to detect a malicious node before modifying a given number of packets with a given probability. It could be noted that even with a modest total header space of 5 bits, a node would be detected even before it is able to modify four packets.

Once a cluster head discovers one of its neighbors as compromised, it can report it to the base station for further action and also broadcast the entire network alerting all nodes about this node.

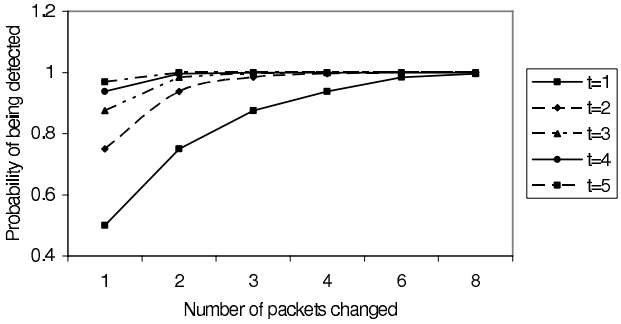

Fig. 1: Probability a node can change p

B. Computational Overhead

As explained earlier, our mechanism is quite effective even with just three bits. The hash function generates a keyed hash value on source and destination addresses and all other fields being marked by the underlying mechanism. Thus, the hash function operates over an input of 70 bits apart from the key itself and at each node at most two such hash values will be generated for each packet. Hence, computing the hash values hardly poses any processing or computational overhead on the node.

C. Future work

The protocol we presented is a lightweight protocol that ensures end-to-end data integrity of a packet. When data aggregation protocols are being implemented in the network, then a node that is receiving an aggregated packet might no longer be able to verify the integrity. We will investigate this case and enhance our protocol. A simple technique might be to forward the aggregated packet along with the packets from

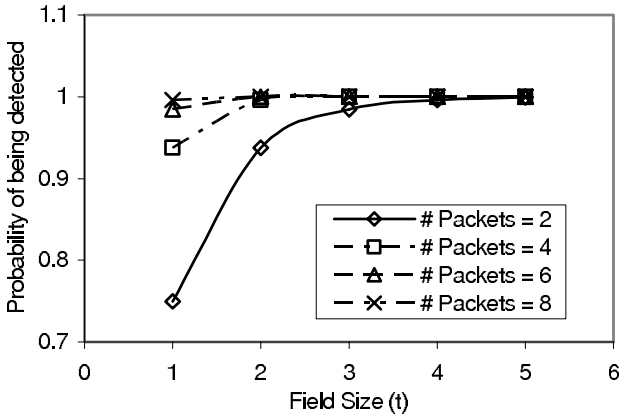

Fig. 2: Probability of detecting a malicious node for a given number packets

which the aggregated packet was generated to the next base station, which verifies that the aggregation has properly taken place. Then, the original packets can be discarded and only the aggregated packet can be forwarded on.

5. CONCLUSIONS

In this paper, we presented a novel lightweight strategy to ensure data integrity. Our protocol is based on leapfrog strategy in which each cluster head verifies if its previous node has preserved the integrity of the packet using the secret key it shares with two hop uptree node. The key advantages of the protocol include: the protocol is simple; it needs very few header bits, as low as three bits, thus resulting in negligible bandwidth overhead; the protocol poses very low computational overhead, it needs to compute just a hash as compared to multiple complex operations required by any cryptographic implementation for verifying authenticity. We also discussed the performance of the protocol.

ACKNOWLEDGMENTS

This work was supported in part by the NSF under Grants CNS # 0413187.

REFERENCES

[1] I.F. Akyldiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, Vol. 38, No. 4, pp. 393-422, March 2002.

[2] D. Estrin and R. Govindan, "Next Century Challenges: Scalable Coordination in Sensor Networks," in *Proc. ACM/IEEE Conf. Mobicom'99*, pp. 263–270, Aug. 1999.

[3] H. Abelson et. al., "Embedding the Internet: Amorphous Computing," in *Communications of ACM*, Vol. 43, No. 5, pp. 74–82, May 2000.

[4] G. Borriello, R. Want, "Embedding the Internet: Embedded Computation Meets the World Wide Web," in *Communications of ACM*, Vol. 43, No. 5, pp. 59–66, May 2000.

[5] G. J. Pottie, W. J. Kaiser, "Embedding the Internet:Wireless Integrated Network Sensors ," in *Communications of ACM*, Vol. 43, No. 5, pp. 51–58, May 2000.

[6] G. S. Sukhatme, M. J. Mataric, "Embedding the Internet: Embedding Robots into the Internet ," in *Communications of ACM*, Vol. 43, No. 5, pp. 67–73, May 2000.

[7] A. Wood and J. Stankovic, "Denial of Service in Sensor Networks," *IEEE Computer*, pp. 54–62, October 2002.

[8] L. Zhou and Z. Haas, "Securing Ad-Hoc Networks, *IEEE Network Magazine*, Vol. 13, No. 6, pp. 24-30, 1999.

[9] NAI Lab , www.nai.com/nai_labs/asp_set/crypto/ crypt_senseit.asp

[10] A. Perrig, R. Szewczyk, V. Wen, A. Woo,, "Security for SmartDust Sensor Network" http://www.cs.berkeley.edu/ vwen/classes /f2000/cs261/project/sensor_security.html .

[11] F. Stajano and R. J. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless networks", in *Seventh International Security Protocols Workshop*, 1999, pp. 172–194.

[12] J. Hubaux, L. Buttyan, S. Capkun, "The quest for security in mobile ad hoc networks", in *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC 2001)*, 2001.

[13] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad-hoc networks, in *ICNP*, 2001, pp. 251-260.

[14] M. G. Zapata, "Secure ad-hoc on-demand distance vector (SAODV) routing", *IETF MANET Mailing List*, Message-ID: 3BC17B40.BBF52E09@nokia.com, Available at ftp://manet.itd.nrl.navy.mil/pub/manet/2001-10.mail, October 8, 2001.

[15] H. Luo, P. Zefros, J. Kong, S. Lu, and L. Zhang, "Self-securing ad hoc wireless networks", in *Seventh IEEE Symposium on Computers and Communications (ISCC 02)*, 2002.

[16] J. Binkley and W. Trost, "Authenticated ad hoc routing at the link layer for mobile systems", *Wireless Networks*, vol. 7, no. 2, pp. 139-145, 2001.

[17] B. Dahill, B. N. Levine, E. Royer, and C. Shields, "A secure routing protocol for ad-hoc networks", Electrical Engineering and Computer Science, University of Michigan, Tech. Rep. UM-CS-2001-037, August 2001.

[18] J. Kong, H. Luo, K. Xu, D. L. Gu, M. Gerla, and S. Lu, "Adaptive security for multi-layer ad-hoc networks", *Special Issue of Wireless Communications and Mobile Computing*, Wiley Interscience Press, 2002.

[19] Y.-C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks", in *Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002)*, June 2002, pp. 313.

[20] Y.-C. Hu, A. Perrig, and D. B. Johnson, Ariadne: "A secure on-demand routing protocol for ad hoc networks", Department of Computer Science, Rice University, Tech. Rep. TR01-383, December 2001.

[21] S. Basagni, K. Herrin, E. Rosti, and D. Bruschi, "Secure pebblenets, in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001)*, October 2001, pp. 156-163.

[22] P. Papadimitratos and Z. Haas, "Secure routing for mobile ad hoc networks", in *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, January 2002.

[23] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks, in *Sixth annual ACM/IEEE Internation Conference on Mobile Computing and Networking*, 2000, pp. 255-265.

[24] S. Buchegger and J.-Y. L. Boudec, "Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks", in *Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing*. Canary Islands, Spain: IEEE Computer Society, January 2002, pp. 403-410.

[25] J. Kong, Xiaoyan Hong, Mario Gerla, " An Anonymous On Demand Routing Protocol with Untraceable Routes for Mobile Ad-hoc Networks", UCLA Computer Science Department Technical Report 030020

[26] A.Perrig, R.Szewczyk, V.Wen, D.Culler and J.D. Tygar, "Spins: Security Protocols for Sensor Networks", *Proc. of 7th Int'l Conf. on Mobile Computing and Networks*, July 2001.

[27] B. Krishnamachari, D. Estrin, S. Wicker, "Modeling Data-Centric Routing in Wireless Sensor Networks," in *Wireless Sensor Network Applications WSNA 2002*, June 2002.

[28] RFC 2104, "HMAC: Keyed-Hashing for Message Authentication," http://www.ietf.org/rfc/rfc2104.tx

[29] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," *10th ACM Conference on Computer and Communications Security (CCS '03)*, Washington D.C., October, 2003.

[30] Wenliang Du , Jing Deng , Yunghsiang S. Han , Pramod K. Varshney,"A pairwise key pre-distribution scheme for wireless sensor networks,"*Proceedings of the 10th ACM conference on Computer and communication security*, October 27-30, 2003, Washington D.C., USA

[31] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. *Proc. of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, Washington D.C., October, 2003

[32] S. Bandyopadhyay and E. Coyle , "An Energy-Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks, in Proceedings", of *IEEE INFOCOM 2003*, April 2003.
[33] A. D. Amis, R. Prakash, T. H. P. Vuong and D. T. Huynh, "Max-Min D-Cluster Formation in Wireless Ad Hoc Networks", in Proceedings of *IEEE INFOCOM 2000*, March 2000.
[34] Hongwei Zhang, Anish Arora, "GS3: Scalable Self-configuration and Self-healing in Wireless Sensor Networks", *Computer Networks (Elsevier)*, 43(4):459-480, November 15, 2003.