DEMO: Demonstrating a Lightweight Data Provenance for Sensor Networks

Bilal Shebaro, Salmin Sultana, Shakthidhar Reddy Gopavaram, Elisa Bertino
Cyber Center and CERIAS, Purdue University, West Lafayette, IN 47907, USA
{bshebaro, ssultana, sgopavar, bertino}@purdue.edu

ABSTRACT

The popularity of sensor networks and their many uses in critical domains such as military and healthcare make them more vulnerable to malicious attacks. In such contexts, trustworthiness of sensor data and their provenance is critical for decision-making. In this demonstration, we present an efficient and secure approach for transmitting provenance information about sensor data. Our provenance approach uses light-weight in-packet Bloom filters that are encoded as sensor data travels through intermediate sensor nodes, and are decoded and verified at the base station. Our provenance technique is also able to defend against malicious attacks such as packet dropping and allows one to detect the responsible node for packet drops. As such it makes possible to modify the transmission route to avoid nodes that could be compromised or malfunctioning. Our technique is designed to create a trustworthy environment for sensor nodes where only trusted data is processed.

Categories and Subject Descriptors

E.0 [**Data**]: General

Keywords

Provenance, Sensor networks, Bloom filters, Malicious attacks, Data trustworthiness

1. INTRODUCTION

The goal of this work is to demonstrate a secure and light-weight provenance scheme for wireless sensor networks that can guarantee secure and efficient data transmission.

Motivations. Sensor networks support the real-time collection and transmission of large amounts of data from many different sources. Once acquired by the source sensors, this data is transferred through intermediate nodes on their way to the base station. In many application domains, such data is used for real-time decision making and other critical actions and thus the data must be trusted. Data provenance represents an important factor that can used, together with other factors, to assess whether data can be trusted as it conveys information about data origin, ownership, and usage. The importance of data provenance is highlighted by Lim *et al.* [6] in their approach for assessing of data trustworthiness in sensor networks, and by showing how untrusted data could lead to severe failures. Even though several researchers [3, 4] have applied data provenance in many applications such as databases and workflows, it is more challenging to build data provenance in sensor networks due to their limited power budget and processing capabilities, as well as their dynamic topology [1]. We thus propose a light-weight provenance encoding and decoding scheme for sensor data based on Bloom filters to trace the source and the path of every individual data packet. Our experimental results show that our scheme is efficient, light-weight and scalable.

Related Work. Conventional provenance security solutions use cryptography and digital signatures [5] that require encryption, checksum, and incremental chained signature mechanisms. Syalim *et al.* [9] also uses digital signatures for a DAG model of provenance. Such solutions cannot be applied on sensor networks due to their specific constrained resources. Vijaykumar *et al.* [10] propose a near real-time provenance for application specific systems that trace the source of data streams after the process has completed. However, the real time operations in sensor networks require immediate responses before processing the data to prevent malicious activities that could cause catastrophic failures [11]. Other approaches capture provenance of network packets in the form of per packet tags [7] that store the history of all nodes. However, such approaches have high memory requirements especially in large scale sensor networks. Chong *at al.* [2] proposes a scheme that embeds the provenance of a data source within a dataset. However, such approach is not intended as a security mechanism and does not deal with malicious attacks. Our approach has been designed to specifically protect from malicious attacks while at the same time assuring good performance.

Contributions. In this demonstration, we show how provenance of sensor data is encoded as it travels from the source node towards the base station at every intermediate node. With a low power consumption and light-weight processing, the provenance of every packet is encoded using Bloom filters, which use their cumulative property to add information of every nodes on its path. The encoded provenance is finally decoded at the base station node that verifies its path and the trustworthiness of its data. In the case of verification failure, the base station performs some further analysis to determine the cause of the failure. Our approach is able to differentiate between normal network misbehavior and malicious attacks, such as packet dropping. Our approach also allows one to determine and localize the responsible node for such behavior, so that it is possible to dynamically switch to an alternative trusted route without affecting the overall network behavior.

Our approach requires a single channel for both data and provenance as opposed to other approaches that require separate transmission channels for each [8]. Moreover, it is based on fixed size Bloom filters in contrast to traditional provenance security solutions based on cryptography and digital signatures which may overload the limited sensor resources.

CCS'12, October 16–18, 2012, Raleigh, North Carolina, USA.
ACM 978-1-4503-1651-4/12/10.

This demonstration paper is organized as follows: Section 2 describes our data provenance mechanism. Then Section 3 introduces the scenarios that we plan to show in order to demonstrate the various aspects of our approach. Finally some conclusions are outlined in Section 4.

2. BASIC PROVENANCE SCHEME

Our approach encodes the provenance within the data packet in a distributed manner, and decodes it at the base station. Each data packet consists of a sequence number, its own data, and an in-Bloom filter iBF field containing the provenance. Every sensor node stores the location of its neighbor nodes that can connect to directly as well as the packet sequence number of the last seen packet for every source. This will serve to detect if any packet has been dropped during the next round of packet transmission, and to localize the responsible node. In what follows, we will discuss how provenance is encoded and decoded, as well how dropped packets are detected and the responsible nodes located.

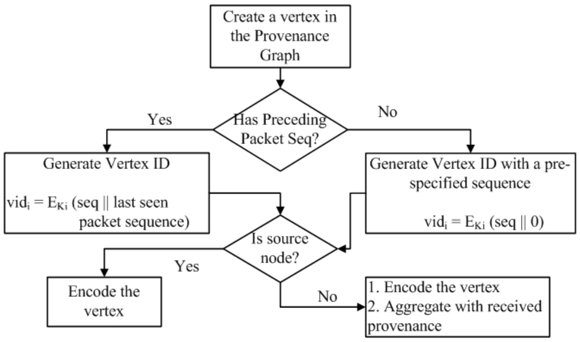

Create a vertex in the Provenance Graph
Has Preceding Packet Seq?
Yes
No
Generate Vertex ID
vid$_i$ = E$_{Ki}$ (seq || last seen packet sequence)
Generate Vertex ID with a pre-specified sequence
vid$_i$ = E$_{Ki}$ (seq || 0)
Is source node?
Yes
No
Encode the vertex
1. Encode the vertex
2. Aggregate with received provenance

Figure 1: Provenance encoding at a sensor node.

2.1 Provenance Encoding

Figure 1 shows the encoding steps of our provenance mechanism that are performed at every node. Initially when a source node generates a data packet, it creates a corresponding Bloom filter *(ibf*$_0$*)* initialized to all $0's$. The node then generates a vertex according to equation 1 below and inserts the vertex ID (VID) in to *ibf*$_0$, which then transmits the Bloom filter (BF) as part of the data packet. At every intermediate node, the VID is generated dynamically based on the node ID *(*n_i*)*, the current packet sequence number *(seq)*, the previous packet sequence number from the same source *(pSeq)*, and the node secret K$_i$. The provenance is aggregated at every node using the cumulative nature of Bloom filters until it reaches the base station with the full encoded provenance *ibf*.

$$
\begin{aligned}
vid_i &= generateVID(\, n_i,\, seq,\, pSeq_i\,) \\
&= E_{K_i}(\, seq \,||\, pSeq_i\,)
\end{aligned} \tag{1}
$$

2.2 Provenance Decoding

Figure 2 shows the provenance mechanism decoding steps. When the base station receives a data packet, it checks for all possible safe paths of packets from the same source node of this packet. These paths have been previously saved by the base station. It then computes the provenance of these paths by using the information in the packet, and compares it to the provenance encoded by Bloom filter enclosed with the data packet. If there is a match, the data is considered for further processing and analysis. If there is a mismatch, then the arrived data packet has either taken a new route that could be safe but had not been previously saved at the base station node, or a previous packet(s) of the same source node has (have) been dropped while on its way to the base station. In the former case, the base station traces back the path and computes the provenance at each node until it reaches the source node. If the provenance matches, then the base station adds this new route to its set of safe paths. If there is still a mismatch, then the data packet has been tampered either by inserting a malicious node in to the network or by compromising an existing node. In this case, we check for compromised nodes and change the route dynamically to avoid such nodes. In the latter case where a mismatch is caused by a dropped packet, this dropping may be due to non-malicious network errors, or to malicious attacks. In the latter case, the base station traces back the node responsible for dropping packets and analyzes the frequency of dropped packets by this node to determine whether it is a normal behavior or an attack. This procedure is executed by tracing back every intermediate node starting from the base station and comparing the components used for encoding the provenance, and finally comparing the node's last sequence number of the data packet *(pSeq)* with the current sequence number of the passing data packet *(Seq)* from the corresponding source node.

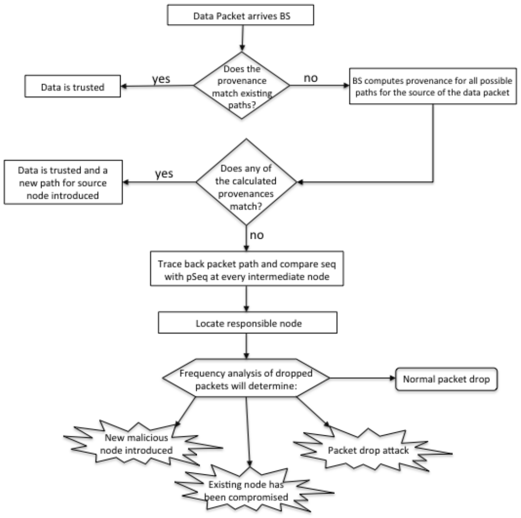

Data Packet arrives BS
Does the provenance match existing paths?
yes
no
Data is trusted
BS computes provenance for all possible paths for the source of the data packet
Does any of the calculated provenances match?
yes
Data is trusted and a new path for source node introduced
no
Trace back packet path and compare seq with pSeq at every intermediate node
Locate responsible node
Frequency analysis of dropped packets will determine:
Normal packet drop
New malicious node introduced
Existing node has been compromised
Packet drop attack

Figure 2: Provenance decoding at the Base Station (BS).

2.3 Detecting Packet Dropping Attacks

Every intermediate node stores the last sequence number of the data packet *(pSeq)* that passed through it for every source node. When the same source node sends another packet through the same intermediate node, it uses the *(pSeq)* together with the current sequence number of the passing data packet *(Seq)* to encode the provenance. This approach helps the base station locate the node that dropped certain data packets when there is a mismatch in the provenance by checking the last processed packet coming from the same source. The base station will be checking the variables at every intermediate node to find which node dropped the packet that could have caused the mismatch in the computed provenance. By comparing its own *(pSeq)* and *(Seq)* for the corresponding source node with every intermediate node in the path, the base station can determine the responsible node for packet dropping, and it measures the frequency of dropped packets at that particular node to determine whether a packet dropped attack exists or not.

3. SCENARIOS

For demonstration purposes, we have simulated the entire sensor network on a single computer. Each node in the network is running its own Java program on a separate port acting as a sensor node, where the network map identifies the nodes by their corresponding port number. Our goal is to show different scenarios that reflect the main contribution of our work allowing the audience to interact with our system and test its efficiency.

Scenario 1: Multiple source nodes will send packets towards the base station. We will show how the provenance is encoded at every intermediate node and how it is decoded once it arrives the base station. The goal is to see the provenance decoding code running at the base station and to see a matching provenance.

Scenario 2: While source nodes are sending packets to the base station, some malicious nodes are introduced into the network. The purpose of this scenario is to show how the base station raises the flag at untrusted data received by the network.

Scenario 3: We will show how the base station detects packet dropping attacks and locates the responsible node.

We believe the above scenarios will demonstrate the importance of our approach by showing its effectiveness and efficiency on sensor data for secure data transmission. Figure 3 shows how our audience will visualize the path that every data packet takes and the Bloom filter provenance encoding at every node as in figure 4.

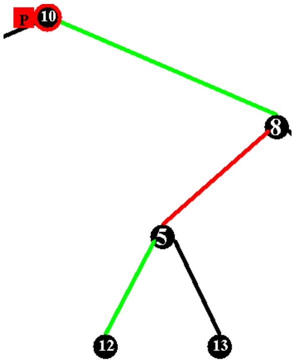


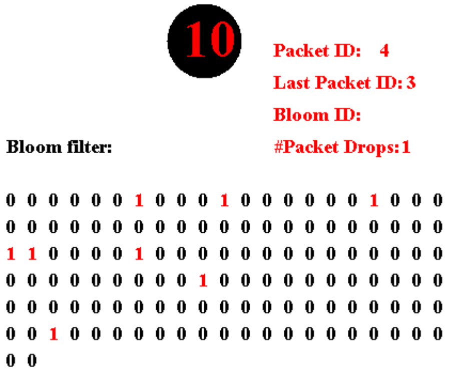
Figure 3: Complete path of a data packet from source node 12 to base station node 10.

4. CONCLUSION

In this demonstration, we develop a light weight scheme for securely transmitting provenance for sensor networks. Our scheme uses Bloom filters to encode and decode the provenance, and is capable of detecting packet dropping attacks and localizing malicious sensor nodes as well as to dynamically change the route of data packets to avoid using such nodes.

5. ACKNOWLEDGMENTS

The work reported in this paper has been partially supported by the Northrop Grumman Cybersecurity Research Consortium, and NSF under grants NSF-CNS 0964294 and NSF CNS-1111512.



Packet ID: 4
Last Packet ID: 3
Bloom ID:
#Packet Drops: 1

Bloom filter:

```
0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0
```

Figure 4: Final Bloom filter based provenance at the base station node 10.

6. REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, Aug 2002.

[2] S. Chong, C. Skalka, and J. A. Vaughan. Self-identifying sensor data. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '10, pages 82–93, New York, NY, USA, 2010. ACM.

[3] I. Foster, J. VÃűckler, M. Wilde, and Y. Zhao. Chimera: A virtual data system for representing, querying, and automating data derivation. In *14th Conference on Scientific and Statistical Database Management*, pages 37–46, 2002.

[4] A. Ghani and P. Nikander. Secure in-packet bloom filter forwarding on the netfpga. In *Proceedings of the European NetFPGA Developers Workshop*, 2010.

[5] R. Hasan, R. Sion, and M. Winslett. The case of the fake picasso: Preventing history forgery with secure provenance.

[6] H.-S. Lim, Y.-S. Moon, and E. Bertino. Provenance-based trustworthiness assessment in sensor networks. In *Proceedings of the Seventh International Workshop on Data Management for Sensor Networks*, DMSN '10, pages 2–7, New York, NY, USA, 2010. ACM.

[7] T. M. . F. N. Ramachandran A., Bhandankar K. Packets with provenance. Technical report, Georgia Institute of Technology, 2008.

[8] Y. L. Simmhan, B. Plale, and D. Gannon. A survey of data provenance in e-science. *SIGMOD RECORD*, 34:31–36, 2005.

[9] A. Syalim, T. Nishide, and K. Sakurai. Preserving integrity and confidentiality of a directed acyclic graph model of provenance. In *Proceedings of the 24th annual IFIP WG 11.3 working conference on Data and applications security and privacy*, DBSec'10, pages 311–318, Berlin, Heidelberg, 2010. Springer-Verlag.

[10] N. N. Vijayakumar and B. Plale. Towards low overhead provenance tracking in near real-time stream filtering. In *Proceedings of the 2006 international conference on Provenance and Annotation of Data*, Berlin, 2006.

[11] M. N. Wybourne. National cyber security, research and development challenges related to economics, physical infrastructure and human behavior. Technical report, Institute for Information Infrastructure Protection (I3P), Dartmouth College, 2009.