IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 1, Issue 5, Oct-Nov, 2013
ISSN: 2320 - 8791
www.ijreat.org

PINCODE: Protection in Provenance Conduction Over Data Stream for Sensor Data

Ramya K P[1], Revathi M K[2] and Chithra Devi R[3]

[1]Information Technology, Dr.Sivanthi Aditanar College of Engineering, Tiruchendur, Tamil Nadu 628215, India

[2]Information Technology, Dr.Sivanthi Aditanar College of Engineering, Tiruchendur, Tamil Nadu 628215, India

[3]Information Technology, Dr.Sivanthi Aditanar College of Engineering, Tiruchendur, Tamil Nadu 628215, India

Abstract

Large number of application areas, like location-based services, transaction logs, sensor networks are qualified by uninterrupted data stream from many. Chasing of data provenance in extremely active circumstance is a crucial requirement, because data provenance is a key component in appraising data trustiness which is important for lots of application. Provenance handling of continuous data needs to cover various issues, admitting the storage efficiency, processing throughput, bandwidth conception and secure transmission. This paper addresses the challenges by providing secure and efficient transmission of provenance along with sensor data by embedding it over the inter packet delays(IPDs). The embedding of provenance within a host medium makes this technique reminiscent of watermarking. Spread-spectrum based watermarking technique is proposed, that avoids data degradation due to traditional watermarking. Provenance is extracted effectively based on an optimal threshold mechanism that minimizes the probability of provenance decoding error. The outcome of the observation depicts that this system is scalable and highly resilient in provenance recovery versus several attacks up to specific level.

Keywords: Streaming Data, Water Marking, Provenance Security, Sensor Network, Malicious Attack, Spread Spectrum Watermarking.

1. Introduction

Sensor networks have become more popular as a solution to various large scale networked applications in very diverse areas. It has greatly leaded to broad evaluation of streaming applications, include network backup, location-based services, real-time financial analysis, control of automated systems. Such systems process the data which is created by several origins and treated by multiple intercessor. This variety of data origins speed up the importance of data provenance to ensure secure and foreseeable operation of streaming applications. Data provenance has the history of data product starting from its original source hence it is consider as an efficient tool for assessing data trustworthiness. Past research on streaming data primarily concentrated on simulating, assembling, managing[2], and tracing of origin[5], leaving security undiscovered. Provenance management for streaming data requires to address several challenges, including the assurance of high processing throughput, storage efficiency, low bandwidth conception, and secure transmission. This paper is the first that addresses all of the challenges above. The contribution of this paper include:

- study the problem of secure and efficient transmission of provenance in sensor networks;
- model a spread spectrum watermark framework that transmits provenance along with sensor data by hiding it over the interpacket delays;
- method of regaining the provenance using optimal threshold based mechanism;
- strategy for security analysis.

We address the scalability issue by following spread spectrum based technique that supports multiuser communication[3]. Hence, our proposed method provides scalability and robustness to attacks. be printed directly. The document you are reading is written in the format that should be used in your paper.

2. System Model and Background

2.1. Adversary model

We consider that the origin and the terminus (i.e., the BS) node on the path being supervised are true. Any other arbitrary node may be venomous. An antagonist can listen in and do traffic analysis anyplace on the path. Outside that, the antagonist is capable to spread a few venomous nodes as well as compromise few

legitimate nodes by catching them and physically overwriting their memory. Thus, the attacker might have control of more than one node, and these venomous nodes may conspire to attack the system. The antagonist may drop, or add packets on the connections that are under its control.

Confidentiality. An antagonist can discover the time among consecutive packet transmissions between adjacent nodes and get the IPDs of a specific data flow at related nodes. By utilizing the captured IPDs, an antagonist must not be able to access or recollect the provenance information of legitimate nodes. Thus, we aim to provide the following confidentiality assurances:

- C1: If an antagonist does not know that provenance is being embedded over the IPDs, it cannot find the presence of provenance by discovering the data flow timing features. Even if the antagonist is cognizant of provenance embedding, it cannot recollect the provenance consisting of legitimate nodes.
- C2: Only approved parties can access and check the integrity of the provenance.

Integrity. An inside aggressor might try to alter/demolish the provenance of data packets went through it. The attack is to alter the inter packet timings arbitrarily to build the provenance unworthy. More intelligent attempts include adding legitimate nodes to the provenance of fake data, adding compromised nodes to or removing legitimate nodes from valid provenance.

- C3: An adversary, acting alone or colluding with others, cannot successfully add legitimate nodes to the provenance of fake data.
- C4: An attacker (or a set of colluding attackers) cannot undetectably add or remove nodes from the provenance of data generated by benign nodes.

In addition, we want to prevent provenance forgery attack and to ensure the freshness of provenance.

- C5: (Unforgeability) An adversary cannot claim that a valid provenance for a data packet belongs to a different data packet.
- C6: (Freshness) An adversary cannot replay captured provenance, avoiding detection at the BS.

However, an adversary may increase network jitter in a way that the recorded IPD at the BS is much larger than the desired value. Such an attack is intended to destroy the embedded provenance. As we discuss later, our scheme can recover provenance if the IPD is altered within a certain limit. In any case, the BS can detect such malicious activity and may utilize some auxiliary mechanism to identify the attacker and take necessary actions. Moreover, the attacker can inject or drop data packets which also alters the IPDs and interfere with the embedded provenance. We successfully recover provenance against the insertion attack but survive the deletion attack to a certain extent.

2.2. Network Model

We consider a typical deployment of wireless sensor networks, consisting of a large number of nodes. Sensor nodes are stationary after deployment, though the routing paths may change due to node failure, resource optimization, etc. The network is modeled as a graph G (N,E) where

- N $\{n_i : n_i$ is a network node with identifier i$\}$: a set of network nodes;
- E $\{e_{i,j} : e_{i,j}$ is an edge connecting nodes n_i and $n_j\}$: the set of edges between the nodes in N.

There exists a base station (BS) that acts as sink/root and connects the network to outside infrastructures such as the Internet. All nodes form a tree rooted at the BS and report the tree topology to BS once after the deployment or whenever a change in the topology occurs. Since the network does not change frequently, such a communication will not incur significant overhead. The network is organized into a cluster structure [13]. Sensory data from the children nodes are aggregated at the cluster-head a.k.a. aggregator, and routed to the applications through the routing tree rooted at the BS.

2.3. Data Model

The sensor network supports multiple distinguishable data flows where source nodes generate data periodically. A node may also receive data from other nodes in order to forward them towards the BS. For the rest of the paper, we will use the term data arrival with the meaning of data generation or receipt at a node. While transmitting, a node may send the sensed data or pass an aggregated data value computed from multiple sensors' readings, or act as a routing node. Each data packet contains an attribute value and provenance for this attribute. The packet is also time stamped by the source node with the generation time. As we see later, the packet timestamp is crucial for provenance embedding and decoding processes. Hence we use a message authentication code to maintain its integrity and authenticity.

However in case of aggregation, the cluster head creates a new packet with aggregated data which makes it difficult to preserve the packet timestamps received from all of its children. Hence, we assume that at the beginning of each aggregation round, all of

the cluster nodes synchronize their time and agree upon a timestamp to associate with their data packets for that round. Then the cluster head creates a new packet with the aggregated data and authenticated timestamp from one of its children. Since time synchronization is performed in sensor networks for various purposes [10], it will not add extra overhead to our protocol.

2.4. Digital Watermarking

The key idea of digital watermarking is to hide secret information (watermark) related to a digital content within the content itself thereby ensuring the movement of the watermark along with the content. Thus, digital watermarking involves the selection of a watermark carrier domain and the design of two complementary processes.

1. An embedding process E that utilizes the watermark carrier A, the watermark message w, and, possibly, a key K to generate the watermarked data AW as
   $E(A, w, K) = AW$
2. A detector process that determines the existence of a watermark within the received signal (with the key, if applicable) and extracts it.

Though our proposed scheme resembles a watermarking technique, the detection process in our scheme is more powerful since it can extract individual node identities from the aggregated data watermarked in time domain.

2.5. Spread Spectrum Watermarking

Spread spectrum is a transmission technique by which a narrowband data signal is spread over a much larger bandwidth so that the signal energy present in any single frequency is undetectable [9]. In our context, the sequence of inter packet delays is the communication channel and the provenance is the signal transmitted through it. Provenance is spread over many IPDs such that the information present in one IPD (i.e., container of information) is small. Consequently, an attacker needs to add high amplitude noise to all of the containers in order to destroy the provenance. Thus, the use of the spread spectrum technique for watermarking provides strong security against different attacks. We have adopted the direct sequence spread spectrum (DSSS) technique which is widely used for enabling multiple users to transmit simultaneously on the same frequency range by utilizing distinct pseudo noise sequences [9]. The intended receiver can extract the desired user's signal by regarding the other signals as noise-like interferences. The components of a DSSS system are as follows:

Input:

- The original data signal $d(t)$, as a series of $+1, -1$.
- A PN sequence$px(t)$, encoded like the data signal. Nc is the number of bits per symbol and is called PN length.

Spreading. The transmitter multiplies the data with the PN code to produce spreaded signal as $s(t) = d(t)px(t)$

Despreading. The received signal $r(t)$ is a combination of the transmitted signal and noise in the communication channel. Thus $r(t) = s(t) + n(t)$, where $n(t)$ is a white Gaussian noise. To retrieve the original signal, the correlation between $r(t)$ and the PN sequence $pr(t)$at the receiver is computed as$R(\imath)\frac{1}{N_c}\sum_{t=T}^{T+Nc} r(t)pr(t+\imath)$. $px(t) = pr(t)and\imath = 0$, i.e., $px(t)$ is synchronized with$pr(t)$, then the original signal can be retrieved. Otherwise, the data signal cannot be recovered. So, a receiver without having the PN sequence of the transmitter cannot reproduce the originally transmitted data. This fact is the basis for allowing multiple transmitters to share a channel. In this paper, we refer to $R(O)$ as *cross correlation*.

To retrieve the signal forj[th] user, the cross-correlation between $r(t)$ and $pxj(t)$ is computed. Multi-user communications introduces noise to the signal of interest and interfere with the desired signal in proportion to the number of users. The condition for error free communication in DSSS can be derived from Shannon's channel-capacity theorem

$$C = B\log_2\left(1+\frac{S}{N}\right),$$

where $C$is the amount of information allowed by the communication channel, B is the channel bandwidth, and S/N is the signal-to-noise ratio. As $S/Nisusually \ll 1$ for spread-spectrum applications, the expression becomes

$$\frac{C}{B} \approx \frac{S}{N}$$

3. Overview of our approach

We propose a distributed approach to watermark provenance over the delay between consecutive data packets. Provenance of a data packet includes the nodes in the data flow path. The PN sequence (of Lp bits) of a node is used to uniquely represent its identity in the provenance. Due to the adoption of DSSS based watermarking, all nodes in the provenance use the same medium for transmittingtheir PN sequences. Hence, only Lp bits of digital information are required for watermarking the

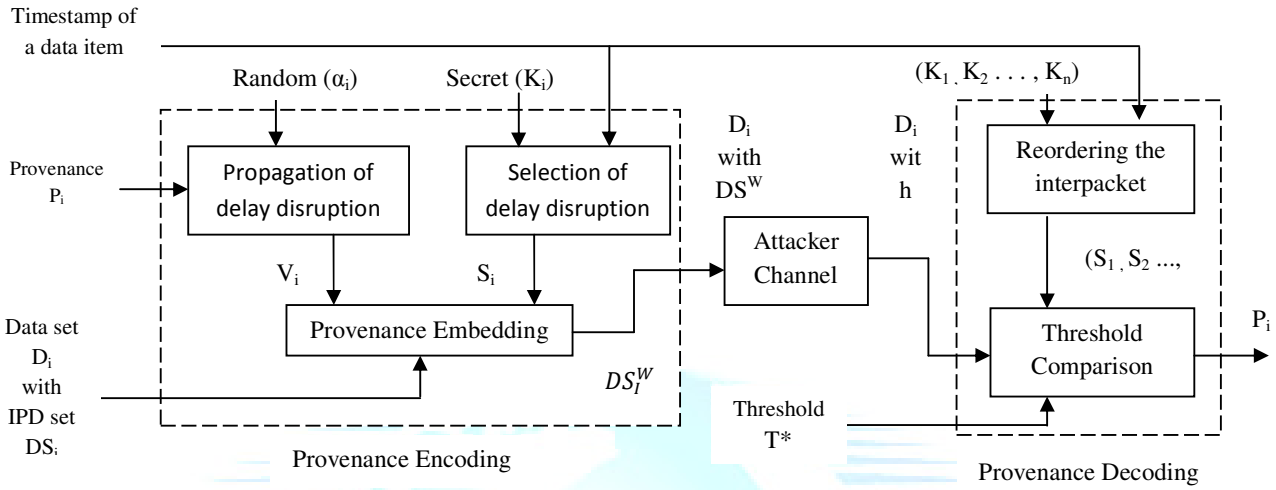

Fig. 1 Overview Of Provenance Encoding and Provenance Decoding

provenance. Since we utilize the IPDs, Lp IPDs (in other words, a sequence of Lp þ 1 packets) are required for embedding and transmitting the provenance of a data packet. Fig. 1represents an overview of our approach for provenance encoding at a sensor node in the data path and decoding at the BS. The process a node ni follows to encode a bit of PN sequence over an IPD is summarized below:

- Step E1 (Generation of Delay Perturbations). n_i generates a set of delay perturbations by using the PN sequence pn_i and impact factor α_i. The set is represented by $V_i = \{v_i(1), v_i(2), \ldots ; v_i(L_p)\}$, where each element $v_i(j)$ is a real number. Note that, $v_i[j]$ corresponds to the provenance bit

  $pn_i(j)$. However, the node may perform the computation offline since it is independent of any packet specific information.
- Step E2 (Selection of a Delay Perturbation). On the arrival of any (j+1)th data packet, n_i records the IPD $\Delta[j]$ and assigns a delay perturbation $v_i[k_j] \in V$ to it. To ensure the robustness of the scheme, the delay perturbations are not assigned sequentially to the IPDs, i.e., $v_i[j]$ is not assigned to $\Delta[j]$. Instead, a delay perturbation $v_i[k_j]$ is selected using the secret K_i and the packet timestamp.
- Step E3 (Provenance Embedding). In this step, n_i delays the packet transmission by $v_i[k_j]$ time unit. As $v_i[k_j]$ corresponds to the
- provenance bit $pn_i[k_j]$, through this step a provenance bit is embedded over an IPD.
- This notion makes our scheme reminiscent of watermarking.

At the end, the BS receives the data set along with watermarked IPDs DS^w, which can be interpreted as the sum of delays imposed by the intermediate nodes, the attackers, and the difference between consecutive propagation delays along the data path. The provenance retrieval process at the BS approximates the provenance from this DSSS signal based on an optimal threshold T*. The retrieval process follows two steps:

- Step R1 (Reordering the IPDs). The IPDs for incoming packets are recorded at the BS. For each node, the IPDs are reordered according to the algorithm used in E2, which produces a node specific permutation of the IPDs. We denote this sequence as CS_i.
- Step R2 (Threshold-Based Decoding). For any node n_i, the BS computes the cross-correlation between CS_i and the PN sequence pn_i. If the correlation result exceeds the threshold T*, the BS decides that pn_i was embedded as a part of the provenance.

As the BS does not know which nodes participated in the data flow, it performs the Bit selection and Threshold Comparison for all nodes. Based on the threshold comparison result, it identifies the nodes in a data flow. In next sections, we discuss these steps in detail.

4. Generation of Delay Perturbations

As the first step to embed provenance, a node ni generates a delay sequence that is used for watermarking. The PN sequence pni and impact factor α_i are used for this purpose. The PN sequence, consisting of a sequence of +1 and -1's, is characterized by a zero mean. The zero mean property is required to ensure successful information decoding at the BS in the context of DSSS-supported multiuser communication. α_i is a random (real) number generated according to a normal distribution $N(\mu, \sigma)$. μ and σ are predetermined and known to the BS and all the nodes. Thus, the BS only knows the distribution of i's, but not their exact values. However, ni generates the set of delay perturbations Vi as a sequence of real numbers as follows:

$$\begin{aligned} Vi &= \alpha_i \times pni \\ &= \alpha_i \times \{pni[1], pni[2], \dots, pni[L_P]\} \\ &= \{(\alpha_i \times pni[1]), \dots., (\alpha_i \times pni[L_P])\} \\ &= \{v_i[1], v_i[2], \dots, v_i[L_P]\}. \end{aligned}$$

5. Selection of A Delay Perturbation

In this section, we present the algorithm that a node applies to select the delay perturbation (from Vi) corresponding to an IPD. If we sequentially assign the delays to the IPDs (which implies that the provenance bits are embedded sequentially), it will be much easier for the attackers to infer information about the provenance or to corrupt the provenance. Hence, we randomize the embedding positions using a different permutation of the elements in Vi. On the arrival of any $(j+1)$th data packet, the j[th] IPD $\Delta[j]$is considered for watermarking and the information to watermark is picked out from Viusing a selection algorithm. Thus, instead of watermarking $vi[j]$ over the IPD$\Delta(j)$, we select a delay $vi[kj]$ for this purpose, where $kj$is an index within $[0, Lp-1]$.The algorithm uses the secret Ki and the packet timestamp, and selects a delay perturbation for the IPD according to the following formula:

$$selection(\Delta[j]) = H(ts[j+1]||Ki) mod Lp.$$

Here, H is a lightweight, secure hash function, k is the concatenation operator, and $ts[j+1]$ represents the packet timestamp. Since secure hash functions generate uniformly distributed message digests, each execution of the selection mechanism will result in a unique integer in the range $[0, Lp-1]$. The resulting integer can be used to index a distinct element in Vi.The indices are used to point the elements in Vi. Thus, the order according to which each node embeds the delays from Viover the IPDs forms a permutation of the elements different from the sequential order. This sequence is denotedas $Si = \{si[1], si[2], \dots si[L_p]\} = \{vi[k1], vi[k2], \dots, vi[kLp]$.

6. Provenance Embedding

The simple provenance is represented as a simple path. Each node in the path watermarks its PN sequence over a set of L_pIPDs, i.e., (L_p+1) packets are utilized. Intuitively, the first packet in a data flow does not experience any delay due to provenance embedding. For any other $(j+1)$th data packet (sent/forwarded), each node in the path hides a provenance bit over the associated IPD $\Delta[j]$. interchangeably, a node $n_i$uses the IPD $\Delta[j]$ to accommodate a delay perturbation$(v_i[k_j] = s_i[j])$. Using $s_i[j]$, the delay to be added to $\Delta[j]$ is computed as$\lambda[j] = s_i[j] \times T$;where T is the value of a time unit. If $s_i[j] > 0$, the resulting $\lambda[j] > 0$ and then we can perform watermarking by simply adding $\lambda[j]$ $to\Delta[j]$. But if $s_i[j] < 0$, the delay to be added to an IPD is negative. To avoid this situation, we introduce a constant offset when calculating $\lambda_i[j]$, which ensures that $\lambda_i[j]$is always positive. The offset may be any constant leading to $\lambda_i[j] > 0$. We use $\mu + const * \sigma$ in our scheme, where $const$ is any constant that makes $\lambda_i[j]$greater than 0, i.e.,

$$const > \frac{-(s_i[j] + \mu)}{\sigma}$$

7. Provenance Retrieval

The provenance retrieval algorithm recovers provenance using the secret parameters including the keys $(K1; K2; \dots; Kn)$, the PN lengthLp, and the optimal threshold T^*. The BS records the watermarked IPDs and executes the retrieval process whenever it collects a number of Lp IPDs denoted by the set DSw. Since the BS does not know which nodes embedded their identities in the provenance, it executes the process for all of the nodes in the network and tries to identify the desired nodes. We denote such a sequence by $CSi = csi[1], csi[2], .., cs[Lp]\}$. Any element (i.e., IPD) in this sequence can be interpreted as the sum of delays added by the nodes in provenance, the difference of propagation delay between two consecutive data packets, and possibly any delay added due to malicious attacks.

The decoding error can be reduced further by embedding the provenance, i.e., each $v[j] \in V$, multiple times. The number of repetitions is called$redundancyfactor$. At the BS, the provenance is extracted multiple times and the decision about the presence of a node in the provenance is taken based on a $majorityvotingtechnique$.

8. Security Analysis

In this section, we discuss possible attacks that can be performed to corrupt the embedded provenance and show how our scheme defeats them. We discuss from

the perspectives of both the outside and inside attackers.

8.1. Outside Attacker

With the capability of capturing data packets and interpacket timing characteristics, an outside attacker may try to disrupt provenance security in different ways.

8.1. Provenance Detection and Retrieval

The attacker tries to infer important watermarking parameters (such as quantization step used to compute watermark delay, proportion of watermarked IPDs, etc.) using packet timestamps at each intermediate host and achieves the attack goals utilizing these parameters.

8.1.2. Replay Attack

An adversary may fraudulently transmit previously heard data packets (transmitted by legitimate nodes) to give a false idea about the sensed environment. For an IPD-based provenance transmission system (like ours), the attacker also observes the timing characteristics in order to maintain them during packet replay.

8.2. Inside Attacker

As discussed earlier, the inside attacker is a more powerful attacker which will try to disrupt the provenance security more intelligently. Obviously, such an attacker can maliciously modify or disable the code of the provenance module on the compromised node.

Deletion attack. A compromised node can destroy the information carried out by the IPDs by dropping data packets routed through it.

Alteration attack. This attack perturbs the IPDs with the goal of moving the cross-correlation values from above the threshold T* to below the threshold T* and vice versa, leading the erroneous retrieval of provenance. As in the deletion attack, embedding provenance multiple times will reduce the impact of this attack.

Insertion attack. A malicious routing node may insert fake data in the data flow generated by a legitimate node.

9. Conclusion

In this paper, we address the novel problem of securely transmitting provenance for data streams. We propose a spread-spectrum watermarking-based solution that embeds provenance over the inter packet delays. The security features of the scheme make it able to survive against various sensor network or flow watermarking attacks. The experimental results show that our scheme is scalable and extremely resilient in provenance retrieval against various attacks. In future, we will investigate the feasibility of this technique for large sized provenance.

References

[1] V. Berk, A. Giani, and G. Cybenko, “Detection of Covert Channel Encoding in Network Packet Delays,” technical report, Dartmouth College, 2005.

[2] S. Cabuk, “IP Covert Timing Channels: Design and Detection,”Proc. ACM Conf. Computer and Comm. Security (CCS), pp. 178-187, 2004.

[3] S. Chong, C. Skalka, and J.A. Vaughan, “Self-Identifying Sensor Data,” Proc. Information Processing in Sensor Networks (IPSN), pp. 82-93, 2010.

[4] I. Cox and M. Miller, “Electronic Watermarking: The First 50Years,” Proc. IEEE Workshop Multimedia Signal Processing pp. 225- 230, 2001.

[5] C. Dai, D. Lin, E. Bertino, and M. Kantarcioglu, “An Approach to Evaluate Data Trustworthiness Based on Data Provenance,” Proc. Fifth VLDB Workshop Secure Data Management (SDM), pp. 82-98, 2008.

[6] R.C. Dixon, Spread Spectrum Systems: With Commercial Applications, third ed. John Wiley and Sons, Inc., 1994.

[7] J. Elson and D. Estrin, “Time Synchronization for Wireless Sensor Networks,” Proc. Int’l Parallel and Distributed Processing Symp. (IPDPS), p. 186, 2001.

[8] I. Foster, J. Vockler, M. Wilde, and Y. Zhao, “Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation,” Proc. Conf. Scientific and Statistical Database Management, pp. 37-46, 2002.

[9] R. Hasan, R. Sion, and M. Winslett, “The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance,” Proc. Conf. File and Storage Technologies (FAST), pp. 1-14, 2009.

[10] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,”Proc. Ann. Hawaii Int’l Conf. System Sciences, pp. 3005-3014,2000.

[11] A. Houmansadr, N. Kiyavash, and N. Borisov, "Multi-FlowAttack Resistant Watermarks for Network Flows," Proc. IEEEInt'l Conf. Acoustics, Speech and Signal Processing, pp. 1497-1500, 2009.

[12] N.B.A. Houmansadr and N. Kiyavash, "Rainbow: A Robust and Invisible Non-Blind Watermark for Network Flows," Proc. Network and Distributed System Security Symp. (NDSS), 2009.

[13] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: A Link LayerSecurity Architecture for Wireless Sensor Networks," Proc. Int'lConf. Embedded Networked Sensor Systems, pp. 162-175, 2004.

[14] X. Wang and D.S. Reeves, "Robust Correlation of Encrypted Attack Traffic Through Stepping Stones by Manipulation of Interpacket Delays," Proc. ACM Conf. Computer and Comm. Security (CCS), pp. 20-29, 2003.

[15] N. Kiyavash, A. Houmansadr, and N. Borisov, "Multi-FlowAttacks against Network Flow Watermarking Schemes," Proc.USENIX Conf. Security Symp., pp. 307-320, 2008.

[16] Y.L. Simmhan, B. Plale, and D. Gannon, "A Survey of DataProvenance in E-Science," SIGMOD Record, vol. 34, pp. 31-36,2005.

[17] A. Syalim, T. Nishide, and K. Sakurai, "Preserving Integrity and Confidentiality of a Directed Acyclic Graph Model of Provenance," Proc. Working Conf. Data and Applications Security andPrivacy, pp. 311-318, 2010.