An Identity based Routing Path Verification Scheme for Wireless Sensor Networks

Haibat Khan

Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX
United Kingdom
Email : haibat.khan.2016@live.rhul.ac.uk

Abstract. While the ability to express routing policies in Wireless Sensor Networks (WSNs) has been well-studied, unfortunately, the ability to enforce these policies has not been. The core challenge is that if we assume an adversarial, decentralized, and high-speed environment, then how can the receiving node be sure that the path being announced by the incoming packet is the actual path followed by it? In this paper we describe the networking primitive, called Routing Path Verification (RPV), which serves as a tool to enforce routing policies and presents a solution to the defined core challenge. We assess the security of the proposed RPV construction in a formal way. More significantly we augment a suitable key exchange protocol with our proposed RPV construction, to achieve an overall RPV scheme. We also evaluate the computational, communication and storage overhead of our proposed scheme and the experimental results show that the approach is quite scalable.

Keywords. wireless sensor networks; routing policy control; peer to peer protocols; security protocols; routing security; routing path verification; identity based cryptography.

1 Introduction & Background

Designing secure protocols for Wireless Sensor Networks (WSNs) presents unique challenges due to lack of characteristics such as pre-deployed infrastructure (PKI / CA) and centralized policy and control. In addition, if the sensor nodes are not static, then the ever changing network topology due to the mobility of the nodes and the limits on the communication and computational capabilities of individual nodes present new challenges in developing efficient and secure networking protocols. Routing in WSNs enables packet delivery from one node to another by way of intermediate nodes. It is the fundamental issue considered in WSNs, thus secure routing (Shokrzade et al., 2015) is a fundamental issue in WSN security. Taking into consideration both changing topology as well as changing membership, in addition to route establishment or discovery, routing protocols for WSNs need to incorporate ‘‘route maintenance’’, in order to provide for the broken routes in case of member node in the route moving out of the range or otherwise in case of avoidance of malicious nodes. This renders route maintenance quintessential for sensor network paradigm. The wireless medium as well as non-infrastructure nature of the sensor networks makes them increasingly vulnerable to a number of attacks on the underlying routing protocol (Ballav and Rana, 2015). Unlike wired networks where the attacker needs to gain access to the physical medium to launch any kind of attack, in case of wireless networking, an intruder can easily gain access to the on-going traffic. As there is not any centralized infrastructure, it is very difficult to have a key distribution center or a trusted certification authority to provide cryptographic keys and digital certificates to help nodes authenticate themselves. Secure routing aims to ensure correct and successful routing among authentic nodes with adversary nodes existing around or inside the network. One of the tools which supplement this aim of achieving secure routing in WSNs is Routing Path Verification (RPV).

1.1 *Routing Path Verification*

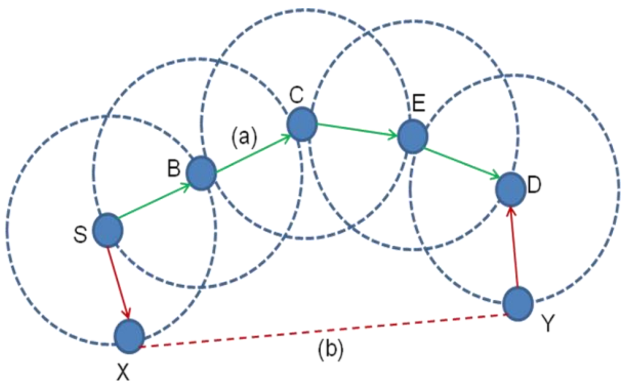
RPV means that the destination node in a putative routing path, agrees on the exact sequence (order) of nodes traversed in that path. For example, if the packet received at the destination announces that it has traversed nodes S, B, C and E (in that order); then the destination node should be able to verify that what is being announced is true.

1.1.1 *Secure RPV*

We say that a RPV scheme is secure if, given an announced path; the destination node can efficiently verify (both) the presence and sequence of each node that appears in the announced path.

Please note that the above definition implies that an honest node cannot appear in the routing path unless it actually took part in routing process of the packet that led to that path. Figure 1 below illustrates the conceptual gist of RPV.

Figure 1 If a packet actually travels from node S to D via path “a” and there exists at least one honest node on path “a”, then node D should be able to detect if the arriving packet announces any other path “b”



To further elaborate the RPV mechanism underway in Figure 1, let’s suppose that the packet in reality traversed from S to D using the path S→B→C→E→D. It is natural to assume that the source and destination nodes i.e. S and D respectively are honest. Now, what we want to emphasize here is that if some arbitrary path (other than the path S→B→C→E→D) is being announced to the node D and if there exists at least one honest node in announced path, using the RPV mechanism the destination node (D in this case) should be able to verify that the path being announced is not the one which the packet in reality traversed.

1.1.2 *Limitations of RPV*

We would like to remark here that there are several functions that RPV cannot handle solely by itself because either they are seemingly infeasible or they fall outside the scope of path verification. For example, a malicious node existing on the path can forward the received packet anywhere; either defying the enforced routing path or acting against the default routing policy. Such kind of misbehavior seems hard to be handled. The packet can also pass via some hidden nodes like wormholes, which do not alter it at all and simply forward it without any change. This also seems hard to be prevented.

1.1.3 *RPV as a Routing Policy Tool*

What we try to emphasize here is that purpose of RPV is not to provide a solution to the problems mentioned in Section 1.1.2, but to provide an assurance to the destination of the avoidance of malicious nodes (wormholes, sinkholes, etc) / un-trusted paths through the use of geographical routing as suggested by Shokrzade et al. (2015). A sufficiently secure routing protocol has some provisions to detect misbehaving nodes working in collusion with each other. Once such misbehaving nodes are detected, their misbehavior should be reported to all other legitimate nodes in the network. So that other nodes, that have routes containing these malicious nodes, can revoke these routes and use alternate routes. RPV would enable a node to ensure the compliance of avoiding these bad nodes. In the overall security architecture, we can think of RPV serving as a routing policy enforcing tool after the trust analysis of the network or alternatively as a routing forensic tool to validate the announced route to a destination.

1.2 *Other RPV Schemes in Literature*

There are a number of schemes in literature which try to achieve the aim of RPV in networks. The first path validation proposal was from Naous et al. (2011) which was later improved upon by Kim et al. (2014). However, these proposals require PKI and are not suitable for WSN scenario. Another scheme was proposed by Jiang et al. (2013). However, this scheme is not suited for resource limited WSNs as it was mainly designed for inter-domain routing path verification using BGP protocol in internet and is dependent upon the underlying network infrastructure. Very recently, another scheme has been proposed by Karumanchi et al. (2016) which attempts at path verification in unstructured peer-to-peer networks (which is the case of WSNs). However, the validation in this scheme is limited only to the resource discovery phase of the paths taken by search queries, while the requirement in our case is that the RPV scheme needs to be spontaneous and independent of the underlying routing protocol and network infrastructure.

To the best of our knowledge, as of now, RPV schemes in WSNs which are instantaneous and independent of underlying network protocols is proposed through the use of ID based sequential aggregate signature schemes (IBSAS). An *aggregate signature* is a digital signature that supports aggregation: given *n* signatures on *n* distinct messages by *n* distinct users using an aggregate signature algorithm, it is possible to aggregate these signatures into a single short signature. This single signature (and the *n* original messages) will convince the verifier that the *n* users did indeed sign the *n* original messages. However, in a *sequential aggregate signature*, aggregation can only be done during the signing process. Each signer in turn sequentially adds her signature to the current aggregate. Thus, there is an explicit order imposed on the aggregate signature and the signers must communicate with each other during the aggregation process.

ID based aggregate signature (IBAS) schemes were introduced by Gentry and Ramzan (2006). The security of their scheme relies on the hardness of Gap Diffe-Hellman problem in a Random Oracle Model (ROM). Based upon their primary work, improved IBSAS schemes (Boldyreva et al., 2007; 2010) were presented later. While these schemes offer the fundamental advantage of universal verification by anyone in the network and avoid the use of PKIs / CAs by employing Identity based Cryptography (IBC), they suffer from the computationally heavy bilinear pairings based operations. This particular aspect obstruct their widespread deployment in the resource constrained WSN routing protocols. On the other hand, Bagherzandi and Jarecki (2010) have proposed an IBAS scheme which avoids the usual bilinear pairings but is unsuitable for RPV accomplishment essentially because of being a "non-sequential" scheme and furthermore because of the requirement of two rounds of communication.

The rest of the paper is organized as follows. Section 2 enlists the details of our RPV scheme. Section 3 presents the security analysis, Section 4 presents the performance analysis of the proposed scheme and Section 5 details the Header Extension to be used in conjunction with the proposed RPV scheme.

2 Routing Path Verification Scheme

In this section we will present the details of our RPV scheme which consists of two major parts. The first part is an on demand Identity based Authenticated Key Exchange (IDAKE) protocol which ultimately shares a symmetric secret key between two nodes separated by multi-hops. The second part of this scheme is an RPV construction based upon Message Authentication Code (MAC).

2.1 On Demand Key Exchange

Whenever a need for having a shared secret key between any node on the path and the destination node arises, an IDAKE protocol is executed between the two nodes. One such IDAKE protocol which happens to be non-interactive was suggested by Sakai et al. (2000). However, this protocol, though non interactive, is not well suited for our scenario as it involves the computation of expensive bilinear pairings (Galbraith, S. D., 2005) and would eventually introduce excessive delay (due to computational limitations) in the delivery of the packet in question. The IDAKE protocol (Yasmin et al., 2014) which we use in conjunction with our RPV construction is a pairing free one pass protocol which is more suited to resource constrained WSNs because of its less communication overhead than other multi pass IDAKE protocols. The important aspect of this protocol is that it avoids the usual pairing operations required in most of the previous published IDAKE schemes (McCullagh and Barreto, 2005; Shim, 2003; Smart, 2002). To the best of our knowledge, it is the only one pass IDAKE protocol which is also pairing free. For the continuity of discussion, we only present a brief overview of the IDAKE protocol in this paper. For further details of the protocol please refer to Yasmin et al. (2014).

Initial Parameters & Key Generation. As this key exchange protocol is essentially ID based, the PKG generates the system parameters as follows:-

(a) Specifies q, p, E/F_p, P and G where q is a large prime number and p is the field size, E/F_p is an elliptic curve E over a finite field F_p, P is a base point of order q on the curve E and G is an additive cyclic group of order q generated by P.

(b) Chooses a random $s \in Z_{q^*}$ as the master secret key and then computes $P_{PKG} = sP$ as the master public key.

(c) Chooses a suitable hash function $H: \{0,1\}^* \times G \rightarrow Z_{q^*}$.

Next the PKG computes the private key of each node corresponding to its ID using the Schnorr signature scheme. For a node Y with identity ID_y, the private key is calculated as follows:

(a) For a randomly chosen $r_y \in Z_{q^*}$, PKG computes $R_y = r_yP$ and $c_y = H(ID_y, R_y)$.

(b) Then it computes the private key as $s_y = c_ys+r_y$.

(c) Finally, outputs (s_y, R_y) where s_y is secret key and R_y is public key.

Note: Before deployment, every node Y stores its identity ID_y, private key s_y, public key R_y and public system parameters $\{q, p, E/F_p, P, G, P_{PKG}, H\}$ in its memory.

The One Pass Protocol. The following steps describe the one pass IDAKE protocol, whenever a node Z wants to communicate with another node Y:

(a) Node Z chooses a random $t \in Z_{q^*}$ as ephemeral key and computes $y = ts_z$ and then the point $L = yP$ on the elliptic curve E. Node Z then signs the ephemeral public key L together with ID_z, ID_y and timestamp TS with any suitable ID based signature scheme and sends $[L, ID_z, ID_y, TS, Sig_{sz}(L, ID_z, ID_y, TS)]$ to the node Y.

(b) The node Y checks the time stamp TS to avoid a replayed message. If the message is a fresh one, Y verifies the signature $Sig_{sz}(L, ID_z, ID_y, TS)$. After successful signature verification node Y computes the shared secret $K_{y,z} = s_yL$ $(= s_yts_zP)$ and deletes L.

(c) Then node Z first computes $S_y = c_yP_{PKG} + R_y$. Then the shared secret $K_{z,y} = yS_y$ $(= ts_zs_yP)$. It then deletes L, t and y.

Both parties then compute the shared session key using a suitable key derivation function.

2.2 The RPV Construction

For our RPV construction, which represents the core of our scheme, we investigate the cryptographic construction of aggregate MACs (Katz and Lindell, 2008). Aggregate MACs have the property that multiple MAC tags, computed (possibly) by various senders on various (possibly different) messages, can be aggregated into a shorter tag that can still be verified by a recipient who shares a distinct key with each sender. Informally, aggregate MACs can be thought of as the symmetric key analogue of aggregate signatures. The complexity of this aggregate MAC scheme is essentially the same as of a regular MAC scheme. However, the construction is not suitable for our application scenario as it does not protect against "remixing" the order of participating entities as explained by Eikemeier et al. (2010). The basic reason for such attacks is that the scheme of Katz and Lindell (2008) supports the aggregation of MACs independent of the order of the participating parties, meaning that the aggregation algorithm is an un-keyed process.

As verifying the order of participating parties (nodes) is very essential in our scenario, in this paper we will present an aggregate MAC based RPV construction which would be able to verify the order of the participating nodes and essentially would offer the same complexity as that of the scheme of Katz and Lindell (2008). As MACs constitutes the core of our construction, we proceed ahead by defining them formally below and then presenting our construction:-

Definition 1 (MAC). We define 'MAC' = (**Mac**, **Vrfy**) over key, message and tag space K, M, T ϵ $[0\ 1]^n$ respectively as a pair of polynomial time algorithms, where:

*Algorithm **Mac**:* The signing algorithm **Mac**(k,m) takes its input a message 'm' in M and a shared key 'k' in K and outputs a tag 't' in T.

*Algorithm **Vrfy**:* The verification algorithm **Vrfy**(k,m,t) outputs a `yes' or `no' by taking inputs as a message 'm', a key 'k' and a tag 't' and always holds the correctness condition **Vrfy**{k,m, **Mac**(k,m)} = 'yes'.

Construction 1 (RPV). Let **Mac** be a deterministic algorithm. We define 'RPV' = (**Mac***, **Agg***, **Vrfy***) as a tuple of following polynomial time algorithms:
*Algorithm **Mac****: Upon input $k \in [0\ 1]^n$ and $m \in [0\ 1]^n$ outputs **Mac**(k,m).
*Algorithm **Agg****: Upon input a tag $t_{i-1} \in [0\ 1]^n$ and a key $k_i \in [0\ 1]^n$ the algorithm **Agg***(t_{i-1}, k_i) outputs a new tag, t_i = **Mac***(k_i, id_{i+1})[1] $\oplus$ **Mac***(k_i, t_{i-1}). (For $t_{i-1} = \emptyset$, simply execute the Mac* algorithm on initial input message m).
*Algorithm **Vrfy****: Upon input an ordered set of keys $k_1; \ldots ; k_x \in [0\ 1]^n$, tag $t \in [0\ 1]^n$ and an input message $m \in [0\ 1]^n$, algorithm **Vrfy***{(k_1;….; k_x),m,t} computes for i = 1,…., x, $t' \leftarrow$ Agg*(t_{i-1}, k_i), with $t_0 = \emptyset$ and outputs 1 if t' = t otherwise 0. The algorithm also returns 0 if any key identifier in the ordered input key set is repeated.

The construction elaborated above has following "chaining" structure:

$$\mathrm{Mac}^*(k_2,\mathrm{Mac}^*(k_1,m) \oplus \mathrm{Mac}^*(k_1,id_2)) \oplus \mathrm{Mac}^*(k_2,id_3) \oplus \ldots\ldots \oplus \mathrm{Mac}^*(k_x,\ldots.(\mathrm{Mac}^*(k_2,\mathrm{Mac}^*(k_1,m) \oplus \mathrm{Mac}^*(k_1,id_2)))) \oplus \mathrm{Mac}^*(k_x,id_{dest})$$

It is easy to verify the correctness of above construction.

2.2.1 MACs from PRFs

The reader maybe wondering that in Definition 1 above, the message space, key space and tag space seems to be same ('n' in our case). The reason for this would become clear as we further explain this section. Due to the design requirements of our RPV scheme, we avoid MAC constructions which are based upon other primitives like collision resistant hash functions, for example HMAC (Krawczyk et al., 1997), primarily because of the fact that in our RPV scheme we require the underlying MAC function to be a Pseudo Random Permutation (PRP) introduced by Luby and Rackoff (1988). This property of being a PRP is required in formal proof of security of the proposed RPV scheme and would become clear once the reader would approach Section 3. The secondary reason of not using keyed hash functions based MACs in our scheme is because of their well-known weakness with respect to generic birthday attack (Joux, 2004) which imposes tight security bounds on their usage with the same secret key 'k'. However, we would like to mention here that this limitation can be easily overcome by using appropriate hash functions like SHA-2 which offer more flexible security bounds. Instead, the MACs to be used in our scheme are the ones build directly from Pseudo Random Functions (PRFs) (Goldreich et al., 1986). In Section 4, performance comparison between the two approaches has been detailed for ease of understanding of the reader. We also take care of messages of arbitrary length by fixing the initial input message size (which actually is a nonce) to 'n' bits. We formally define these types of MACs as follows:

Definition 2 (PRF based MACs). For a PRF F: $K \times X \longrightarrow Y$ where $K, X, Y \in [0\ 1]^n$, we define a MAC I_{PRF} = (Mac,Vrfy) as:

(a) Mac(k,m) := $t \leftarrow F(k,m)$

(b) Vrfy(k,m,t): output \`yes' if t = F(k,m) and \`no' otherwise.

For our case, we use the 128 bit key length version of AES block cipher as a PRF whose domain, range and key space are all same as 128 bits. There are two main reasons for choosing AES. First it fits well into our design criterion because of its smaller key and tag size (i.e. 128 bits) but which is still large enough for brute force security. Secondly, because it is not only a PRF but also a Pseudo Random Permutation (PRP) and this property will help us in the formal proof of security of our RPV construction in Section 3. A PRP is defined as follows:

Definition 3 (PRPs). A PRP E: $K \times X \rightarrow X$ is defined over key and message space (K,X) such that:

(a) There exists an "efficient" algorithm to evaluate E(k,x) for all $k \in K$ and $x \in X$.

(b) The function E(k, .) is one-to-one.

(c) There exists an "efficient" inversion algo-

[1] id_{i+1} here represents the n-bit value of the identifier of the next hop node in the current route and isn't an explicit input to the Agg* algorithm. The value is derived implicitly from the current routing table of the node.

rithm D(k,x) for all k ϵ K and x ϵ X.

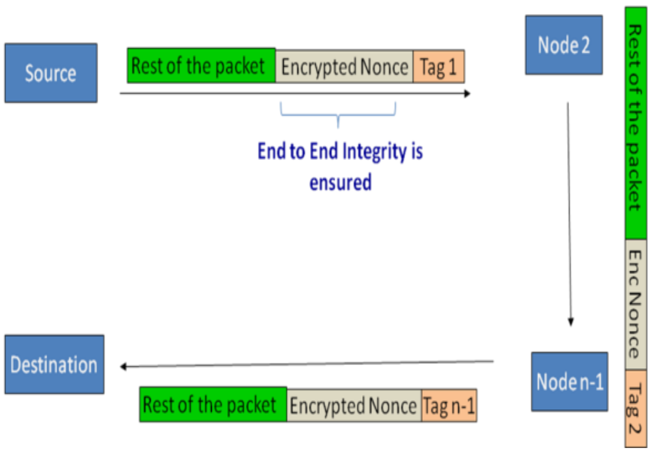
2.2.2 The RPV Scheme Explained

The basis of our path verification construction is "chaining" mechanism. The source node selects a nonce and then generates tag t_1 by using its shared secret key with destination node. After calculation of the tag, the source node encrypts this nonce with its shared key with the destination node and appends this encrypted nonce along with the tag to the rest of the packet. However, here we stress that we assume that the integrity of this encrypted nonce is ensured end-to-end till the destination node along with any other packet content (that also needs to be verified for end-to-end integrity) of the underlying layers. This is usually the case between two communicating end nodes and can be very efficiently realized through MACs. Furthermore, if any authentication / integrity mechanism is already in use between the neighboring nodes on the path, then the appropriate steps are executed; however we do not pre-require any such mechanism for our RPV construction.

Once the packet arrives at the node 2, it checks its memory state for a shared key 'k_2' with the destination node and if cannot find one; initiates the one pass IDAKE protocol. It then calculates the new tag t_2 by first getting Mac (k_2,t_1) and then by taking an XOR of it with Mac (k_2,id_3), where id_3 is the identity of the next node (node 3) on the path. More formally, the aggregation algorithm of Construction 1 above describes this very step. The process is repeated till the packet arrives at the destination node. Basically, the only part of the packet (from RPV viewpoint) that changes its values for every hop is the tag field and the "path field" as they both keep on getting updated for every hop, while the rest of the packet stays unchanged. In this paper for the sake of simplicity and clarity, we assume that the "path field" is already part of the underlying routing protocol and no exclusive measure needs to be taken to cater for this requirement.

After the packet arrives at the destination node and announces the purported path via "path field"; the destination nodes decrypts the nonce and verifies the tag received as in the verification algorithm of Construction 1. For the nodes on the announced path which do not have a shared key with the destination node; new key is established by completing the already initiated one pass IDAKE protocol by these nodes. The contents of the one pass IDAKE protocol(s) (if any) travel along with the packet. Figure 2 below provides an overview of the process.

Figure 2 The RPV "Chaining" Mechanism



3 Security Analysis

3.1 The Attack Model

As usually is the case, the nodes in our system are resource constrained in terms of computation capabilities and available battery power. We assume that both the source and the destination (the end points) are honest nodes. We use the generic terms *source* and *destination* to mean, respectively, the initiator and the target of the path in consideration. We envisage a network where the majority of the bulk traffic between the nodes is secured via symmetric encryption schemes because of their superior computational efficiency over their public key based counterparts. Therefore, it seems very realistic to assume that those nodes which are in the direct radio communication range (one hop nodes) of each other already share a secret symmetric key. This can easily be accomplished during the secure neighbor discovery phase (Khan et al., 2015; Poturalski et al., 2013; Taheri et al., 2016).

After the initial deployment phase, various available routes are established between communicating nodes and routing tables as usual are kept on being updated according to the underlying routing protocol. Whenever, two nodes want to communicate securely with each other, a suitable ID based key agreement protocol is executed between the two nodes and a shared secret key is established. The choice of using a particular protocol depends upon the target application and the required level of security. Some protocols offer superior security but with higher computational and communication requirements while others offer vice versa. To reduce the communication overheads involved during the symmetric key establishment phase, the nodes which are in usual communication with each other cache the shared keys (for a limited time) for future use after the termination of the current session. In this way the key agreement protocol needs only to be executed very occasionally. We believe that this very approach provides an appropriate compromise between computational / communication overheads and storage requirements. Note that the unique private key is the only long term secret that the nodes possess. We also assume that the route to be taken by an individual packet between the two nodes is unknown a priori for every packet. It means that the route can be different for every packet even during the same communication session.

It is assumed that source and destination nodes already share a secret key using a suitable ID based Authenticated Key Exchange (IDAKE) protocol (the protocol could be a two pass or even three pass). However, it is not assumed that every node on the path being verified already has a shared secret key with the destination. An overview of a pairing free one pass IDAKE scheme to be used in conjunction with our RPV construction in these scenarios is already presented in Section 2.1.

We assume an active adversary who has far stronger capabilities than his passive counterpart. It can introduce its own packets as well as delete, delay and modify packets before forwarding them. We focus on protection of our scheme against active adversaries.

3.2 *Security Properties of IDAKE Protocol*

In this section we would very briefly skim through the security properties of the one pass pairing-free IDAKE protocol (Yasmin et al., 2014).

Implicit Key Authentication: The one-pass protocol provides Implicit Key Authentication. The initiating node is authenticated through the verification of the signature scheme while the authentication of the destination node is assured via the calculation of public parameter $S_y = (s_y P)$ in Zq*.

Key Confirmation: The IDAKE protocol does not provide key confirmation. In fact, no one-pass protocol can provide key confirmation.

Known Key Security: Because of the contribution of ephemeral value *t* in calculation of the shared secret, the protocol provides Known Key Security.

Sender's Forward Secrecy: Through the contribution of ephemeral secret *t* in the IDAKE protocol, sender's forward secrecy is also assured.

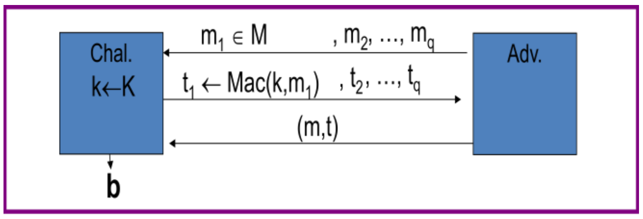
Unknown Key Share: The IDAKE protocol is also protected from Unknown Key Share attacks because of the inclusion of IDs in the calculation of the public parameters of the participating nodes.

Key Control: Being a one-pass, this IDAKE protocol does not protect against Key Control. The selection of ephemeral public parameter is made only by the initiating node and not the destination node.

3.3 *Security of MAC*

We begin by establishing the security of our underlying MAC, as defined in Definition 1, under an adaptive chosen message attack game model (Goldwasser et al., 1988). For a MAC I = (Mac,Vrfy) and adversary A, we define a MAC chosen message attack game as depicted below in Figure 3:

Figure 3 The MAC Attack Game



The attacker can launch a chosen message attack for as many messages $m_1, m_2, \ldots, m_q$ as he wants and is given the corresponding tags $t_i \leftarrow Mac(k, m_i)$ for his chosen messages by the oracle. These we formally call as 'Mac queries'. The attacker A can also submit another type of query which we call as 'Corrupt query', as a result of which he gets to know the corresponding secret key k. The attacker's goal here is existential forgery, i.e. produce some new message/tag pair (m,t), where $(m,t) \notin$

{ (m_1,t_1) , … , (m_q,t_q) } and submits it to the challenger. The challenger returns b, where b=1 if Vrfy(k.m,t) = 'yes' and (m,t) $\notin$ { (m_1,t_1) , … , (m_q,t_q) } and no 'Corrupt query' was issued; otherwise b=0.

Definition 4 (Secure MAC). We say that I = (Mac,Vrfy) is a secure MAC if for all "efficient" adversaries $Adv_{MAC}[A,I]$ = Pr[Chal. outputs 1] is "negligible.", meaning that no "efficient" adversary can win the adaptive chosen message attack game with non-negligible probability.

3.4 Security of PRF based MAC

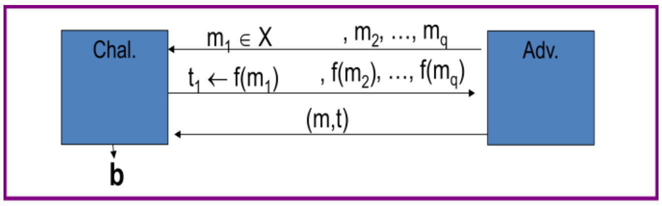
As we use PRF based MACs in our constructions, so here we will very briefly establish the security of our PRF based MAC construction as in Definition 2.

Theorem 1. If F: K×X⟶Y is a secure PRF and 1/|Y| is negligible (i.e. |Y| is large) then I_{PRF} as in Definition 2 is also a secure MAC.

In particular, for every efficient MAC adversary *A* attacking I_{PRF} there exists an efficient PRF adversary *B* attacking F such that $Adv_{MAC}[A, I_{PRF}] \leq Adv_{PRF}[B, F] + 1/|Y|$. This statement means that I_{PRF} is secure as long as |Y| is large, say $|Y| = 2^{80.}$

Proof (Sketch): Suppose f: X ⟶ Y is a truly random function from message space X to tag space Y. Then MAC adversary A must win the following game depicted as in Figure 4 below:

Figure 4 The 'Adaptive Chosen Message Attack' Game


Chal.
Adv.
$m_1 \in X$, m_2, …, m_q
$t_1 \leftarrow f(m_1)$, $f(m_2)$, …, $f(m_q)$
(m,t)
b

Adversary A will win this game if t = f(m) and m $\notin$ {m_1 , … , m_q} i.e. existential forgery. But as f is a truly random function, the previous tags which the adversary got from the challenger don't have any influence on his decision of choosing the tag for the message m as the value of the function f at point m is independent of its value at other points and hence the probability with which A will win this game is 1/|Y|. However, if we replace f with our pseudo random function F the adversary would not be able to differentiate between the two and would behave the same way as if he is interacting with a truly random function. □

3.5 RPV Construction Security

The definition of security for our RPV construction corresponds to existential unforgeability under an adaptive chosen-message attack. To present it formally we prove the following theorem:

Theorem 2. If Mac is existentially unforgeable under an adaptive chosen-message attack (Definition 4) and Mac is deterministic, then RPV = (Mac*, Agg*, Vrfy*) as given in Construction 1 is also secure against existential forgery under an adaptive chosen message attack.

Proof: The outline of this proof generally follows the approach of Katz and Lindell (2008) but as the construction here is essentially different, the reductions used in this proof are also quite different. We prove this theorem using the contra-positive approach of logic theory. More precisely, we will show that if there exists some existentially forgeable (Mac*, Agg*, Vrfy*) RPV construction then from this we can build an existentially forgeable MAC. We start by fixing a polynomial time RPV adversary *A* but here as we would be dealing with numerous numbers of keys, for ease of understanding, we would be using identifiers to link a particular key with a particular node. We formally define adversary *A* as follows:

Definition 5 (RPV Adversary). Let A be a polynomial time adversary involved in the following experiment:
<u>Key Generation:</u> Keys k_1, …. , $k_t \in [0\ 1]^n$ are generated corresponding to IDs 1,….,t respectively.
<u>Queries:</u> Adversary A is allowed following two types of queries:

- Query Mac: Upon input i and m; the oracle will return Mac(k_i,m)
- Query Corrupt: Upon input i; the oracle will return k_i

<u>Outputs:</u> A outputs an ordered set of node IDs 1;….;s ϵ [1,……,t] (representing a particular path), an initial nonce m $\in [0\ 1]^n$ and a tag t $\in [0\ 1]^n$ (Please note that all node IDs in this ordered set

need to be distinct).
Success: We say that A is successful in the experiment if Vrfy*{(k_1;….; k_{s-1}),m,t} =1 and there exists at least one node i ϵ [1,….,s-1] in the announced path such that:

- A never queried Mac(i,t_{i-1})[2]
- A never queried Corrupt(i)

We also fix a MAC adversary *F* which will interact with a challenger (with a secret key k*) and would try to produce a valid existential forgery as illustrated in Section 3.3 by using the adversary *A* in its "belly". *F* proceeds as follows:

(a) It chooses a random ID i* ← {1,…..,t}.

(b) For i = 1 to t:

(i) If i ≠i*, it chooses k_i ← $\{0,1\}^n$.
(ii) If i = i*, *F* does nothing (however, it implicitly sets k_{i*} = k*).

(c) Then *F* "runs" *A*, answering its queries as given below:

Query Mac(I , m): If i ≠ i* then *F* answers the query using the self generated key k_i. If i = i* then *F* queries its own MAC challenger and returns Mac*(k*,m) as the result.
Query Corrupt(i): If i ≠ i* then *F* gives *A* the self generated key k_i. If i = i* then we abort the game.

(d) After completion of its query phase, *A* outputs a forgery to *F* by providing it with an ordered set of node IDs 1;….;s ϵ [1,……,t], initial nonce m ϵ $[0\ 1]^n$ and a tag t ϵ $[0\ 1]^n$. Let j ϵ [1,….,s-1] be an index in the ordered key set such that:

- A never queried Corrupt(j)
- A never queried Mac(j ,t_{j-1})[3]

If j ≠ i*, then we abort the game; otherwise we proceed to step (e).

(e) As j = i* {from step (d)}, *F* calculates the tag t_{j-1} as follows:

For i = 1,…., j-1, t_{j-1} ← Agg*(t_{i-1}, k_i)

Adversary *F* then calculates the tag t_j as follows:

For i = s,…., j+2, t_j ← Dec[k_{i-1},(t_{i-1}⊕ Mac(k_{i-1}, id_i))]

Here 'Dec' means the AES decipher function. As our MAC construction is essentially the AES cipher, we can always apply the corresponding decipher function, as AES is a PRP (see Definition 3). In the above calculations *F* uses those values for keys k_1, …., k_{j-1}, k_{j+1},…, k_{s-1} which it had already chosen in step {b (i)}. Finally, *F* calculates t* = t_j ⊕ Mac(k*, id_{j+1}) by the querying his MAC challenger and outputs (t_{j-1}, t*) as an existential forgery to the Mac challenger.

If we assume that *F* doesn't abort in the above game then the following of the proof becomes evident and unambiguous. In case adversary *A* is successful in outputting a valid existential forgery against the RPV construction then it means *F* also outputs a valid existential forgery against the underlying MAC scheme. To see this note that when adversary *A* succeeds it essentially means that Vrfy*{(k_1;….; k_{s-1}),m,t} =1 because the underlying Mac function is deterministic and the output values are well defined. Hence, the calculations involved in step (e) above would lead to distinct and well defined values of t_{j-1} and t*. Moreover, adversary *F* never queried, directly or indirectly, its own MAC challenger for the input t_{j-1} and still was able to produce a valid output tag. This completes the proof. □

Please note that to provide an additional layer of security in our construction; the initial nonce is encrypted by the shared key between the source and the destination nodes. However, in our security analysis above, we gave adversary the additional capability to choose arbitrary nonce(s) of his own choice to break the scheme.

4 Performance Analysis

In this section we will analyze the performance of our scheme with regards to computation, communication and delay overhead. The tool which we have used for calculation of our results is OPNET

[2] Here t_{i-1} denotes the value of the tag that should be outputted by the previous node on the announced path, if the initial nonce used is 'm'.

[3] The verification of this condition is actually realized in step (e).

Modeler 14.5. C programming language code has been used for implementation of underlying cryptographic primitives. The results have been compiled for networks comprising of 5, 20 and 50 nodes. Figures 5, 6 and 7 below depict the screenshots of the OPNET simulation.

Figure 5 OPNET Simulation of 5 Wireless Nodes

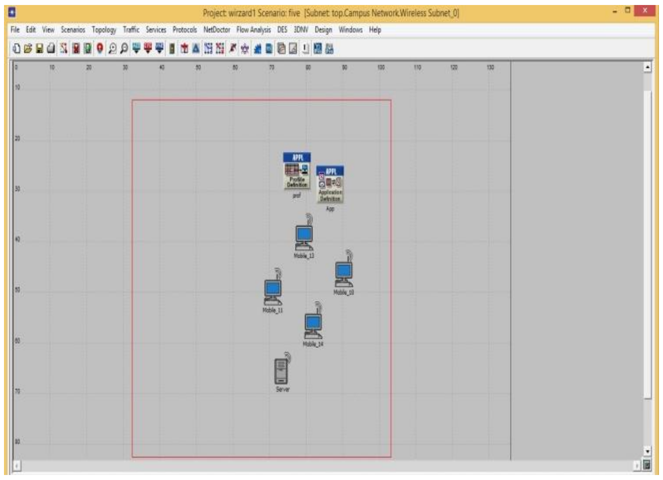


Figure 6 OPNET Simulation of 20 Wireless Nodes

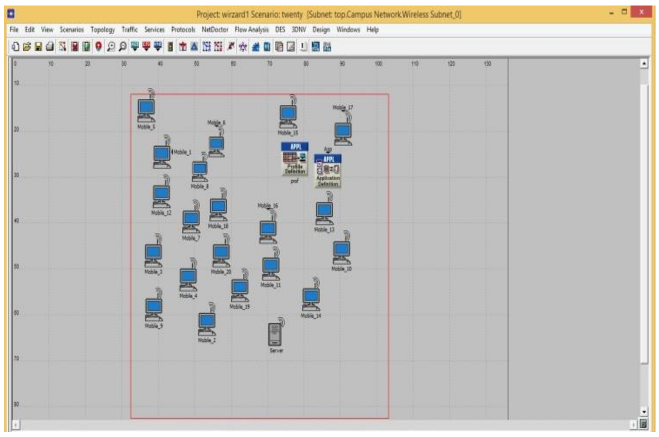
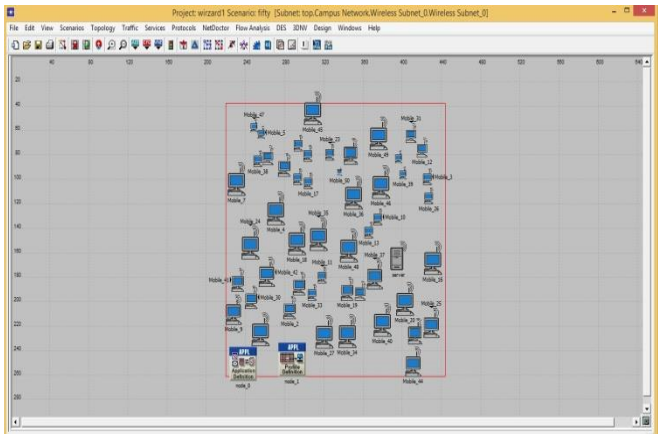


Figure 7 OPNET Simulation of 50 Wireless Nodes



The ensuing sub-sections would detail the performance results of the proposed RPV scheme with respect to computation, communication and delay overhead. Results have been compiled for both symmetric encryption (AES-128) based RPV scheme and keyed hash function (SHA-256) based RPV scheme. From a security viewpoint, both approaches provide same level of security (128 bit). Figure 8 till 16 below presents the details of the experimental results for various configurations.

Figure 8 Simulation Results for 05 Nodes without RPV

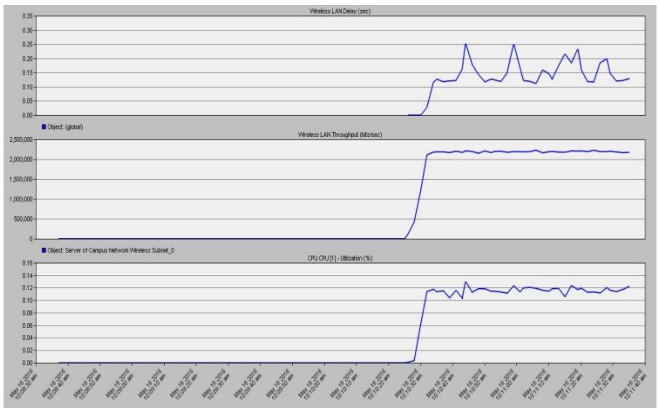


Figure 9 Simulation Results for 05 Nodes with AES based RPV

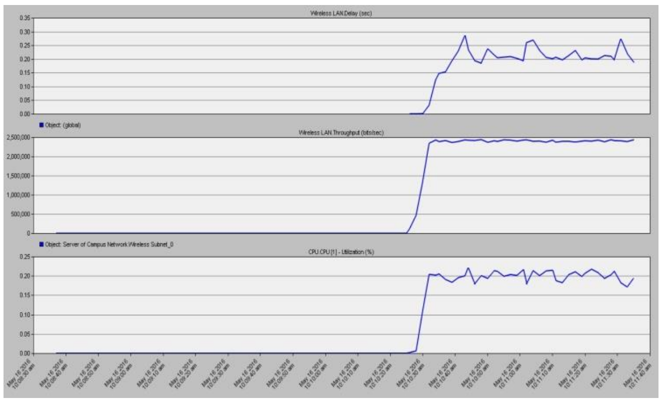


Figure 10 Simulation Results for 05 Nodes with SHA based RPV

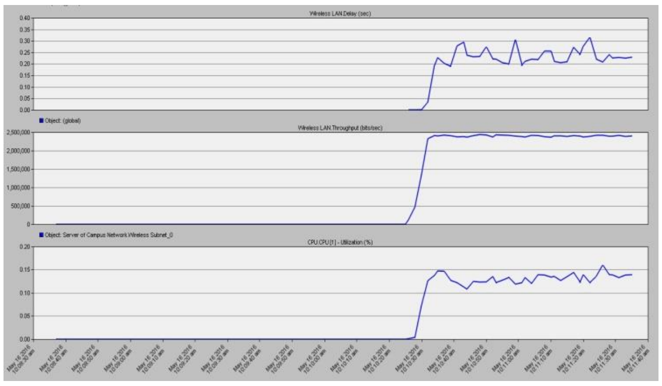


Figure 11 Simulation Results for 20 Nodes without RPV

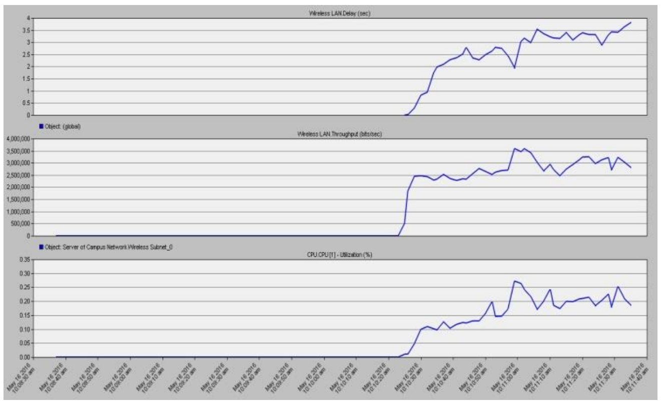


Figure 12 Simulation Results for 20 Nodes with AES based RPV

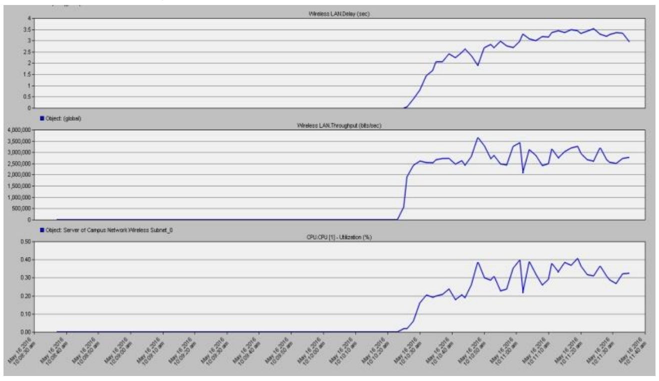


Figure 13 Simulation Results for 20 Nodes with SHA based RPV

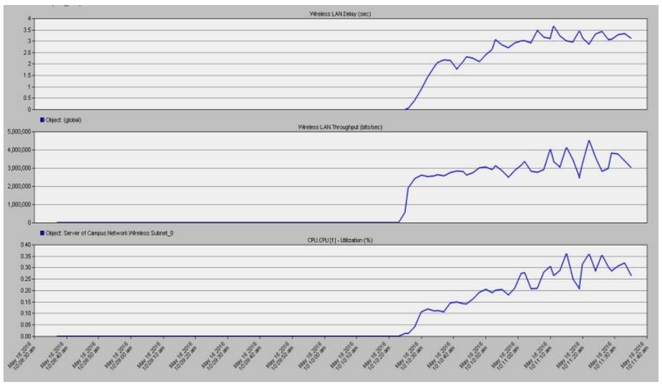


Figure 14 Simulation Results for 50 Nodes without RPV

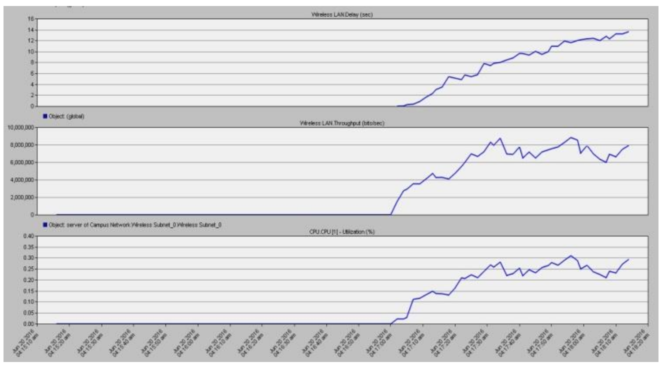


Figure 15 Simulation Results for 50 Nodes with AES based RPV

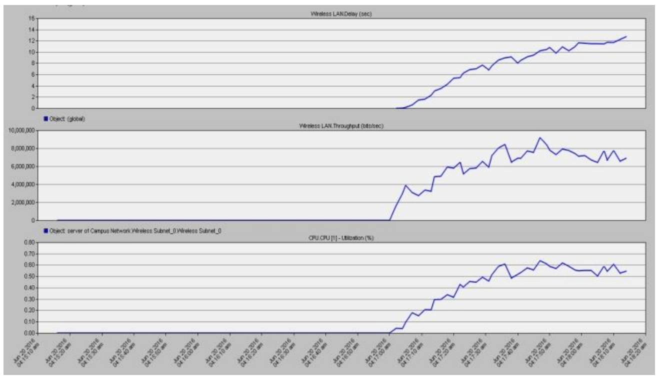
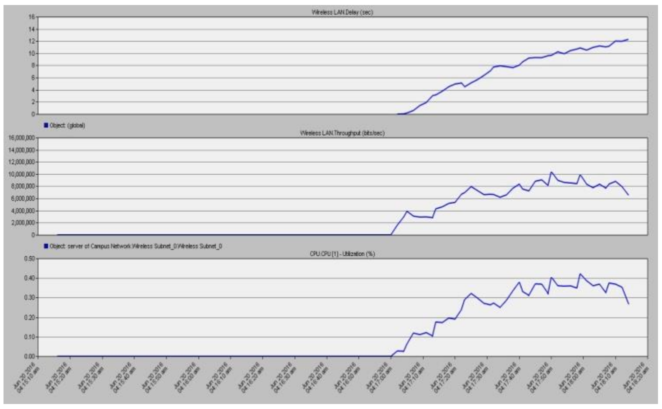


Figure 16 Simulation Results for 50 Nodes with SHA based RPV



4.1 Computational Overhead

In this sub-section, we present the average CPU utilization of all nodes from various OPNET simulation configurations. The results have been compiled for the scenarios when RPV is not active, when AES based RPV is active and when SHA based RPV is active. Table 1 presents the summary of these computational results.

Table 1 Computational Overhead Comparison

No of Nodes	Without RPV	AES based RPV	SHA based RPV
05	0.12 %	0.20 %	0.14%
20	0.20 %	0.28%	0.26%
50	0.24%	0.50%	0.37%

From the experimental results, it is evident that with the RPV scheme in place, CPU utilization increases as expected, however, still remains within acceptable limits. Also evident from the experimental results is that computational overhead between symmetric encryption (AES) based

RPV scheme and keyed hash function (SHA-256) based RPV scheme is better in case of keyed hash functions based RPV with same level of security. The same fact is also supported by (Crypto++ 5.6.0 Benchmarks).

4.2 Communication Overhead

In this sub-section, we will analyze the overall network throughput from various OPNET simulation configurations. The results have been compiled for the scenarios when RPV is not active, when AES based RPV is active and when SHA based RPV is active. Figure 8 till 16 presents the simulation results for these configurations. Table 2 below presents the summary of these communication overhead results.

Table 2 Communication Overhead Comparison

No of Nodes	Without RPV	AES based RPV	SHA based RPV
05	2.20 Mbps	2.35 Mbps	2.40 mbps
20	2.70 Mbps	2.75 Mbps	2.80 Mbps
50	6.90 Mbps	7.05 Mbps	7.5 Mbps

From the experimental results, it is evident that with the RPV scheme in place, the communication overhead is well within acceptable limits. Also evident from the experimental results is that communication overhead between symmetric encryption (AES) based RPV scheme and keyed hash function (SHA-256) based RPV scheme is comparable and slightly in favour of AES based RPV approach. The reason for this is the fact that to achieve same level of security, SHA based RPV scheme has to produce tags of double the size of those of AES based RPV scheme.

4.3 Delay Overhead

In this sub-section, we will analyze the average time delay from various OPNET simulation configurations. The results have been compiled for the scenarios when RPV is not active, when AES based RPV is active and when SHA based RPV is active. Figure 8 till 16 presents the simulation results for these configurations. Table 3 presents the summary of these time delay based results.

Table 3 Time Delay Comparison

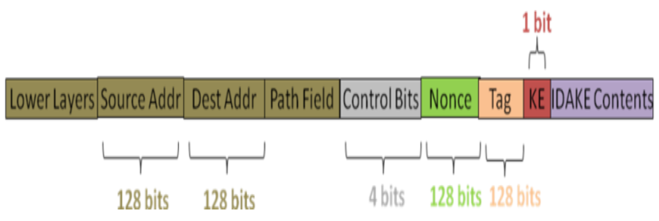
No of Nodes	Without RPV	AES based RPV	SHA based RPV
05	0.16 secs	0.22 secs	0.21 secs
20	3.00 secs	3.10 secs	3.05 secs
50	9.00 secs	9.10 secs	9.05 secs

From the experimental results above, it is evident that with the RPV scheme in place, the effect on end-to-end delay overhead is negligible and is well within acceptable limits. Also evident from the experimental results is that time delay overhead between symmetric encryption (AES) based RPV scheme and keyed hash function (SHA-256) based RPV scheme is comparable through slightly in favour of SHA based RPV.

5 Header Extension & Routing Control

In this section we will present the format of the header extension for our RPV scheme. This extended header has been designed is to be used in conjunction with the headers of the underlying layers' protocols to enable our proposed RPV scheme to accomplish its role as an effective routing policy enforcing tool. We also present some optional routing control features which can be employed with our scheme to give additional functionality. The header extension is detailed in Figure 5 below:

Figure 5 The RPV Header Extension



The fields in brown color indicate that they already form part of lower layer protocols although some of these fields like the *Source Address*, *Destination Address*, *Path Field*, etc (all of them are from the network layer) are also used in our RPV scheme. The RPV header extension consists of five fields which are explained in detail next.

Control Bits: This four bit field is always present whether RPV scheme is active or not. The contents of this field specify the configuration / mode of the RPV scheme and indicate the content type of rest of the fields. The details are listed as in Table 4 below:

Table 4 Details of the "Control Bit" Field

Field Value	Explanation
0000	RPV scheme is not in use and normal routing should be carried out. The other RPV extension header fields are not present.
0001	Indicates a request for RPV initiation by the destination node to the source node without any route specification. The underlying routing protocol would decide the route of each packet. No other additional RPV fields are present.
0010	A request for RPV initiation by the destination node to the source node with route specification. The path field contents indicate the specific route. No other fields are present.
0011	A request for RPV initiation by the destination node to the source node with the authority of route specification delegated to the source node by the destination. No additional fields are present.
0100	An RPV termination message from the source node.
0101	An RPV termination confirmation from the destination node in response to message type of 0100.
0110	An RPV termination request from the source node.
0111	An RPV termination confirmation message from the source node in response to message type of 0110.
1000	An RPV response to RPV request of type 0001.
1001	An RPV response to RPV request of type 0010.
1010	An RPV response to RPV request of type 0011.
1011	A self-initiated RPV by source node with route specification by the source node itself.
1100	A self-initiated RPV by source node without route specification. The underlying routing protocol decides the route on the go.
1101-1111	Future growth.

As obvious from the above table, this control field enables the source and destination nodes with additional capabilities to not only express their routing policies, but with help of our proposed RPV scheme; they are also able to enforce these routing policy directives. Moreover, this field enables our RPV scheme to work in both situations i.e. in a pre-specified path or a priori unknown path. Another important aspect of this field specification is it's reading overhead on the nodes present on the path being followed by the packet. The nodes present between the source and destination only need to act if the RPV scheme is active i.e. type 1000 till 1100; otherwise they just need to forward this packet without taking any action. To accomplish this task efficiently we have designed this field as such that the reading time of the nodes on the path can be reduced significantly. The nodes just need to read the first bit of this field to see whether they have to take some action or not. If the first bit is 0; they just need to forward it without any changes. On the other hand, if it is 1, then they need to update the tag and if required also update the other fields like *Path*, *KE*, *IDAKE Contents*, etc. Obviously, along with the nonce field, the integrity of this control bit field also needs to be ensured.

Nonce: The length of this field is 128 bits and it stores the encrypted value of the initial nonce being selected by the source node. Along with the "Control Bits" field, the integrity of this field also needs to be ensured between the source and the destination nodes.

Tag: This 128 bit field is used to store the updated value of the tag being produced by every node on the path being followed by the packet.

KE: This single bit on-off type field KE (short for Key Exchange) is used to indicate whether the next field "IDAKE Contents" is present or not. If the value of this field is 0 it means that no node on the path has initiated the IDAKE protocol. If it is 1, then at least one node on the path in question has initiated the IDAKE protocol and the "IDAKE Contents" field is present.

IDAKE Contents: This field contains the contents of the one-pass IDAKE protocol. Moreover, it also caters for the total number of nodes and their sequence in the said field.

6 Conclusion

In this paper, we presented a RPV scheme suitable for resource constrained WSNs. The verification provided by our scheme is of all-or-nothing nature meaning that even if just one of the participating nodes provided invalid authentication tag, the whole route is considered invalid. Same is the case with the schemes based upon ID based aggregate signatures. While the piecemeal verification of the route segments can help one identify misbehaving / compromised nodes that produce invalid tags, however, it may be noted that such capability would come at a cost of enhanced communication and computation costs.

Our scheme provides a promising approach towards achieving path verification in resource constrained networks. In a way, we can think of our scheme as one, which tries to distribute the overhead cost to all three aspects of computation, communication and storage instead of overloading just one or at most two out of them. This particular aspect of our scheme along with the feature that our scheme assures path verification in even those scenarios where the route of the packet is not pre-determined enables it to be a strong contender for deployment in WSNs as an RPV tool. Moreover, we also established the security of our scheme in formal way in Random Oracle Model (ROM) with the security dependent upon the underlying Message Authentication Code (MAC) scheme. We also provided an explanation of the associated control message headers and their use as a routing policy enforcing tool.

For the future, the authors suggest that the research community should explore and build upon the suggested 4-bit control field in RPV header extension (Section 5) to express routing policies and their subsequent enforcement through the use of RPV. Specifically, there is a need to standardize these control bit fields in combination with some suitable underlying WSN routing protocol for true utilization of their efficacy and viability. Moreover, there is a need to explore for more routing controls that can be assigned to the control bit field fields left for future growth.

References

Bagherzandi, A. and Jarecki S. (2010) 'Identity-Based Aggregate and Multi-Signature Schemes Based on RSA' in Fischlin, M. et al (Eds.), *Lecture Notes in Computer Science 6056*, Springer-Verlag, pp.480-498.

Ballav, B. and Rana, G. (2015) 'A review of Routing Attacks in MANET and WSN', *International Journal of Engineering Development and Research*, Vol. 3 No. 2, pp. 1401-1406.

Boldyreva, A., Gentry, C., O'Neill, A. and Yum, D. H. (2007) 'Ordered multisignatures and identity-based sequential aggregate signatures with applications to secure routing' in *Proceedings of 14th ACM Conference on Computer & Communication Security*, Alexandria, Virginia, USA , pp. 276-285.

Boldyreva, A., Gentry, C., O'Neill, A. and Yum, D. H. (2010) *Ordered multisignatures and identity-based sequential aggregate signatures with applications to secure routing*. [online] Cryptology ePrint Archive, Report No 2007/438 Revised 21 Feb, 2010. https://eprint.iacr.org/2007/438 (Accessed 05 December 2015)

Crypto++ 5.6.0 Benchmarks. [online] http://www.cryptopp.com/benchmarks.html (Accessed 07 June, 2016)

Eikemeier, O., Fischlin, M., Goetzmann, Jens-Fabian., Lehmann, A., Schroeder, P., Schroeder, D. and Wagner, D. (2010) 'History-Free Aggregate Message Authentication Codes', In Garay, J A. and Prisco, R. D. (Eds.), *Lecture Notes in Computer Science 6280*, Springer-Verlag, pp.309-328.

Galbraith, S. D. (2005) 'Pairings', in Blake, I.F. et al (Eds.), *Advances in Elliptic Curve Cryptography*, Vol. 317, Cambridge University Press, pp. 183–213.

Gentry, C. and Ramzan, Z. (2006) 'Identity-based aggregate signatures' in Yung, M. et al (Eds.), *Lecture Notes in Computer Science 3958*, Springer-Verlag, pp.257-273.

Goldreich, O., Goldwasser, S. and Micali, S. (1986) 'How to construct Random Fuctions', *Journal of ACM*, Vol. 33 No. 4, pp. 792-807.

Goldwasser, S., Micali, S. and Rivest, R. (1988) 'A digital signature scheme secure against adaptive chosen-message attacks', *SIAM Journal on Computing*, Vol. 17 No. 2, pp. 281-308.

Jiang, J., Li, W., Luo, J., and Tan, J. (2013) 'A network accountability based verification mechanism for detecting inter-domain routing path inconsistency', *Journal of Network and Computer Applications*, Vol. 36 No. 6, pp.1671-1683.

Joux, A. (2004) 'Multicollisions in iterated hash functions, application to cascaded constructions', in Franklin, M. (Ed.), *Lecture Notes in Computer Science 3152*, Springer-Verlag, pp. 306-316.

Karumanchi, S., Li, J., and Squicciarini, A. (2016) ‘Efficient Network Path Verification for Policy-routed Queries’, in *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy,* New Orleans, L.A., USA, pp. 319-328.

Katz, J. and Lindell, A. (2008) ‘Aggregate Message Authentication Codes’ in Malkin, T. (Ed.), *Lecture Notes in Computer Science 4964,* Springer-Verlag, pp.155-169.

Khan, A. A., Rehmani, M. H., and Saleem, Y. (2015), ‘Neighbor discovery in traditional wireless networks and cognitive radio networks: Basics, taxonomy, challenges and future research directions’, *Journal of Network and Computer Applications,* Vol. 52, pp. 173-190.

Kim, T. H. J., Basescu, C., Jia, L., Lee, S. B., Hu, Y. C., and Perrig, A. (2014) ‘Lightweight source authentication and path validation’, in *ACM SIGCOMM Computer Communication Review,* Vol. 44 No. 4, pp. 271-282.

Krawczyk, H., Bellare, M. and Cannetti, R. (1997) HMAC: keyed-hashing for message authentication. [online] RFC 2104, February 1997. https://www.ietf.org/rfc/rfc2104.txt (Accessed 05 December 2015)

Luby, M. and Rackoff, C. (1988) ‘How to construct pseudorandom permutations from pseudorandom functions’, *SIAM Journal on Computing,* Vol. 17 No. 2, pp. 373-386.

McCullagh, N. and Barreto, P.S.L.M. (2005) ‘A new two-party identity-based authenticated key agreement’ in Menezes, A. (Ed.), *Lecture Notes in Computer Science 3376,* Springer-Verlag, pp.262-274.

Naous, J., Walfish, M., Nicolosi, A., Mazières, D., Miller, M., and Seehra, A. (2011) ‘Verifying and enforcing network paths with ICING’, in *Proceedings of the Seventh Conference on Emerging Networking Experiments and Technologies,* Tokyo, Japan, pp. 30-41.

Poturalski, M., Papadimitratos, P., and Hubaux, J. P. (2013) ‘Formal analysis of secure neighbor discovery in wireless networks’, *IEEE Transactions on Dependable and Secure Computing,* Vol.10 No 6, pp. 355-367.

Sakai, R., Ohgishi, K. and Kasahara, M. (2000) ‘Cryptosystems based on pairing’ in *Proceedings of Symposium on Cryptography and Information Security*, Okinawa, Japan, pp. 26–28.

Shim, K. (2003) ‘Efficient ID-based authenticated key agreement protocol based on the Weil pairing’, *Electronics Letters*, Vol. 39 No. 8, pp. 653–654.

Shokrzade, H., Majma, M. R., Movassagh, A and Saheb, M. (2015) ‘Routing Security in Wireless sensor Networks’, *Lecture Notes on Software Engineering*, Vol. 3 No. 4, pp.303-307.

Smart, N.P. (2002) ‘An identity based authenticated key agreement protocol based on the Weil pairing’, *Electronics Letters*, Vol. 38 No. 13, pp. 630–632.

Taheri, S., Stoleru, R., & Hogrefe, D. (2016) ‘Secure Neighbor Discovery in Mobile Ad Hoc Networks through Local Topology Visualization’, in *IEEE 30th International Conference on Advanced Information Networking and Applications*, Crans-Montana, Switzerland, pp. 933-940.

Yasmin, R., Ritter, E. and Wang, G. (2014) ‘Provable security of a pairing-free one-pass authenticated key establishment protocol for wireless sensor networks’, *International Journal of Information Security*, Vol.13 No. 5, pp. 453-465.