International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 3, Issue 8, August 2014 ISSN 2319 - 4847

Bloom Filter based efficient key storage with RSA based scheme for node replication detection in mobile sensor networks

Author Ms Sidhhi Raut , Prof Vrunda Bhusari
Dept of Computer ,BSITOR,Pune

ABSTRACT

Due to the benefits of low cost, rapid deployment, self-organization capability and cooperative data processing, wireless sensor networks (WSNs) have been proposed as a practical solution for a wide range of applications such as habitat monitoring, intelligent agriculture, home automation where an adversary can physically capture some of the nodes. when a node is captured, the attacker can replicate it and replicate the node in a large number of replicas, transfer over the network. The detection of node replication attacks in a wireless sensor network is a fundamental problem. Compared to replication attacks in static networks, only a few implementative solutions in mobile networks have been proposed. most of the existing schemes they are widely used in static networks rely on the witness finding strategy which cannot be applied to mobile networks the velocity exceeding strategy used in existing schemes in mobile networks efficiency and security problems. In this paper we focus on Bloom filter based key storage and RSA based Scheme to detect node replication in mobile sensor network.

Index Terms— BS,BF,NODE REPLICATION ,RSA etc.

1. INTRODUCTION

A new set of security challenges arises in sensor networks due to the fact that current sensor nodes lack hardware support for tamper-resistance and are often deployed in unattended environments where they are vulnerable to capture and compromise by an adversary. A sequence of node compromise is that once an adversary has obtained the credentials of a sensor node, it can be insert replicas of that node at strategic locations within the network. These replicas can be used to a variety of application and hard to detect attacks on the sensor application and the networking protocols. In a centralized approach for detecting node replication, when a new node joins the network, it broadcasts a signed message (referred to as a location claim) containing its location and identity to its neighbors. One or more of its neighbors then forward this location claim to a central trusted party (e.g., the base station). With location information for all the nodes in the network, the central party can easily detect any pair of nodes with the same identity but at different locations. Hence, a distributed solution is desirable. Distributed approaches for detecting node replications are based on location information for a node being stored at one or more witness nodes in the network. When a new node joins the network, its location claim is forwarded to the corresponding witness nodes. If any witness receives two different location claims for the same node identity (ID), it will have detected the existence of replica and can take appropriate actions to revoke the node's credentials. The basic challenge for any distributed protocol detecting node replicas is to minimize communication and per node memory costs while ensuring that the adversary cannot defeat the protocol. and reduce storage capacity.

2. LITERATURE SURVEY

A Bloom filter is a space-efficient probabilistic data structure. That is used to test whether an element is a member of a set. Bloom proposed the technique for applications where the amount of source data would require an impracticably large hash area in memory. An empty Bloom filter is a bit array of *m* bits, all set to 0. There must also be *k* different hash functions defined, each of which maps or hashes some set element to one of the *m* array positions with a uniform random distribution. To add an element, feed it to each of the *k* hash functions to get *k* array positions. Set the bits at all these positions to 1. A sensor network with a large number of low-cost nodes distributed over a wide area. In our approach, the existence of a trusted base station, and the sensor network is considered to be a geographic grid, each unit of which is called a cell. Sensors are distributed uniformly in the network. New sensors may be added into the network regularly to replace old ones. Each node is assigned a unique identity and a pair of identity-based public and private keys1 by an offline Trust Authority (TA). In identity-based signature schemes like the private key is generated by signing its public key (usually a hash on its unique identity) with a master secret held only by the TA. In other words, to generate a new identity-based key pair, cooperation from the TA is a must. Therefore, we assume that adversaries cannot easily create sensors with new identities in the sense that they cannot generate the private keys corresponding to the identities claimed and thus fail to prove themselves to the neighbours during the authentication of the location claims. We require that, when a node joins into the network, it needs to generate a signed location claim and broadcast the claim to its neighbours. Only when the location claim is successfully verified, it will then be accepted as a valid network member.

3. RELATED WORK

Node Replication attack

A Sensor networks are a collection of a number of sensor nodes with limited resources have been demonstrated to be useful in various applications, such as environment monitoring and object tracking. Sensor networks could be deployment in critical missions, the sensor networks are unattended and the sensor nodes are not equipped with tamper resistant hardware. This allows a situation where the one sensor node, fabricate many replicas having the same identity (ID) from the captured node, and place these replicas back into strategic positions in the network for further malicious activities. Hence it is a called as a node replication attack. Detection of replication node is very difficult. From the security point of view the node replication attack is extremely harmful to networks because replicas, having keys, can easily launch insider attacks, without easily being detected.

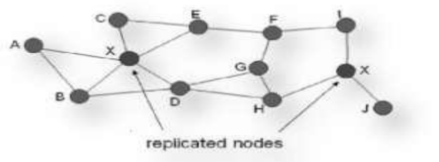


Fig 1 Node Replication Attack

EXISTING SYSTEM

A Sensor network, we would like to detect a **node replication** attack, i.e., an attempt by the adversary to add one or more nodes to the network that use the same ID as another node in the network. we would like to detect this behaviour without centralized monitoring, since centralized solutions suffer from several inherent drawbacks .

OBJECTIVE

- Localized detection
- Efficiency & Effectiveness
- Network wide synchronization avoidance
- Network wide revocation avoidance

PROBLEM STATEMENT

The detection of node replication attacks in a wireless sensor network is a fundamental problem.

Compared to replication attacks in static networks, only a few solutions in mobile networks have been proposed. Most of the existing schemes they are widely used in static networks rely on the witness finding strategy which cannot be applied to mobile networks the velocity exceeding strategy used in existing schemes in mobile networks efficiency and security problems.

PROPOSED WORK

In this proposed work each protocol's security by examining the probability of detecting an attack given that the adversary Inserting L replicas of a subverted node. The protocol must provide robust detection even if the adversary captures additional nodes. We evaluate the efficiency of each protocol. In a sensor network, communication(both sending and receiving) requires at least an order of magnitude more power than any other operation so our first priority must be minimizing communication, both for the network as a whole and for the individual nodes (since hotspots will quickly exhaust a node's power supply). sensor nodes typically have a limited amount of memory, often on the order of a few kilobytes .

PROPOSED SYSTEM MODEL

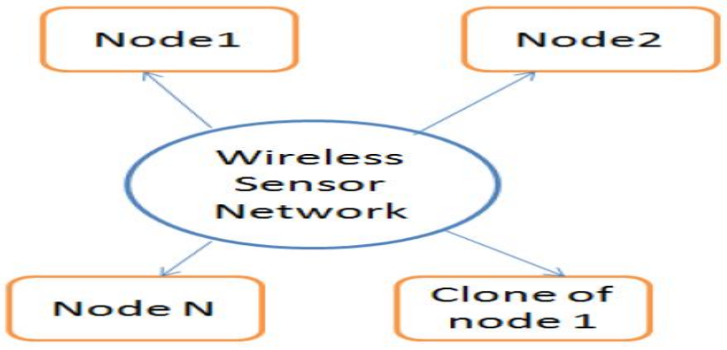


Fig 2 Wireless sensor node

Consider a wireless sensor network Fig 2 consisting of set of nodes. These nodes may or may not be static. These nodes communicate with each other by sending claim message. Adversary captures the credential of compromised node and injects cloned node into network. This clone node can take part in communicating with other node as this node is going to be considered as original node as clone node is holding required credential. At least one clone node in network can fulfils adversary intention. Wireless sensor network contains static or movable nodes. These nodes communicate with each other or with base station for performing certain operation. Base station or access point is considered to be used for storing distributed node information table and used for broadcasting random value to other node. This distributed node information table holds the node information in terms of key value pair. Key is ID of the node and value is location of node. Adversary attacks this network and captures credentials and details of compromised node resulting cloned node. When any new or cloned node enters into network, iteration of algorithm runs which checks the newly entered node's id against node information table. If the ID is already present into table with incoherent location then revocation procedure is getting invoked for that node ID else algorithm updates node information table with newly entered node.

PROPOSED SYSTEM STEPS

Step 1: We are creating a mobile sensor network with one node considering as KDC (Key Distribution Center). The role of KDC is to create Public and Private Keys using RSA algorithm.

Step 2: KDC will generate and distribute the keys to all the mobile nodes in the network such that public key is known to all the nodes in network and private key is known to only that node.

Step 3: When nodes want to interact with each other they will encrypt the message using its own private key and message will be decrypted with its public key.

Step 4: When the node is replicated the replicated node will encrypt the message with its own private key but pretending as original node it will send that message to destination while decrypting the message destination node will come to know that node is replicated as it is node decrypted with source public key.

Step 5: Once the destination node will find the replica it will try to decrypt the message using all the available public keys once it will find the public key with which message is decrypted it will be consider d as a replica and broadcast message of replication to all the nodes including KDC.

Step 6: The replicated node is added in to the black list and it will not be used for the further communication.

PERFORMANCE MATRIX

Five performance metrics are used in our evaluation:

- **Detection Accuracy**——Detection accuracy is used to represent the false positive ratio and false negative ratio of the underlying detection algorithm, which are the ratios of falsely considering a genuine node as a replica and falsely considering a replica a genuine node, respectively.
- **Detection Time**——Detection time is evaluated according to the average time (or, equivalently, the number of moves) required for a genuine sensor node to add the replica's ID .
- **Storage Overhead**—:-Storage overhead is counted in terms of the number of records required to be stored in each node the storing records differ in different algorithms.
- **Computation Overhead**.:-Computation overhead accounts
- for the number of operations required for each node to be executed per move.
- **Communication Overhead:-**Communication overhead accounts for the number of records required for each node to be transmitted . Similarly, it can be considered in terms of the number of bits, but we do not use such a kind of estimation.

MATHEMATIC MODEL

In the mathematical model, three steps are included which are key generation, encrypt and decrypt. We will see all this steps in details.

1. Key Generation

As we know that RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the help of public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

a. Choose two distinct prime numbers p and q.

- For security purposes, the integer's p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primarily test.

b. Compute n = pq.

- n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.

c. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1) = n - (p + q - 1)$, where φ is Euler's to tient function.

d. Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; i.e., e and $\varphi(n)$ are co prime.

- e is released as the public key exponent.

- *e* having a short bit-length and small Hamming weight results in more efficient encryption – most commonly $2^{16} + 1 = 65,537$. However, much smaller values of *e* (such as 3) have been shown to be less secure in some settings.

e. Determine *d* as $d \equiv e^{-1} \pmod{\varphi(n)}$; i.e., *d* is the multiplicative inverse of *e* (modulo $\varphi(n)$).

- This is more clearly stated as: solve for *d* given $d{\cdot}e \equiv 1 \pmod{\varphi(n)}$

- This is often computed using the extended Euclidean algorithm. Using the pseudo code in the *Modular integers* section, inputs *a* and *n* correspond to *e* and *φ(*n*)*, respectively.

- *d* is kept as the private key exponent.

The public key consists of the modulus *n* and the public (or encryption) exponent *e*. The private key consists of the modulus *n* and the private (or decryption) exponent *d*, which must be kept secret. *p*, *q*, and φ(*n*) must also be kept secret because they can be used to calculate d.

2.Encryption

Alice transmits her public key *(n, e)* to Bob and keeps the private key secret. Bob then wishes to send message *M* to Alice.He first turns *M* into an integer *m*, such that $0 \le m < n$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext *c* corresponding to

$$c \equiv m^e \pmod{n}$$

This can be done quickly using the method of exponentiation by squaring. Bob then transmits *c* to Alice.

3. Decryption

Alice can recover *m* from *c* by using her private key exponent *d* via computing

$$m \equiv c^d \pmod{n}$$

Given *m*, she can recover the original message *M* by reversing the padding scheme.

Set Theory Analysis

A] Identify the Sensor networks
S= {s1, s2, s3….}
Where 'S' is main set of Sensor networks like s1, s2, s3…
B] Identify the Sensor nodes
SN= {sn1, sn2, sn3….}
Where 'SN' is main set of sensor nodes in the network like sn1, sn2, sn3…
C] Identify the primary integer number generated for Key Generation
PIN= {pin1, pin2, pin3}
Where 'PIN' is main set of primary integer numbers generated for Key Generation pin1, pin2, pin3
D] Identify the Key distribution.
KD= {kd1, kd2, kd3….}
Where 'KD' is main set of key distribution to all nodes kd1, kd2, kd3…
E] Identify the public key generated for each node.
PK= {pk1, pk2, pk3….}
Where 'PK' is main set of public key generated for each node pk1, pk2, pk3…
F] Identify the private key generated for each node
PRK= {prk1, prk2, prk3….}
Where 'PRK' is main set of private key generated for each node prk1, prk2, prk3…
G] Identify the replica node
RN= {rn1, rn2, rn3….}
Where 'RN' is the main set of replica node.
I] Identify the processes as P.
P= {Set of processes}
P= {P1, P2, P3, P4……}
P1 = {e1, e2, e3}
Where
{e1= creating the network}
{e2= key generation}
{e3= distribution of keys}
J] Identify failure cases as FL

Failure occurs when –
FL= {F1, F2, F3…}
a) F1= = {f|'f' if the replicated node is not detected }

K] Identify success case SS:-
Success is defined as-

SS= {S1, S2, S3, S4}
a) S1={s| 's' if fast the replicated node is detected}
b) S2={s| 's' if data is send successfully}

L] Initial conditions as I_0

a) User want to set network connection

b) User want to create replicas

ALGORITHM

Step 1:
The KDC generates and for every node using RSA algorithm**.**

Step 2: The KDC forms a Bloom Filter structure

Step 3: The KDC then takes 16bit array and Hash functions H1, H2, H3 and all the array positions are initialized to zero**.**

Step 4: When any node start communicating with node ; node sends query to KDC for getting its public key with respect to network layer address.

Step 5: As node is having someone else's network address the key obtained by node will not be able to authenticate message sent by Hence node will be detected**.**

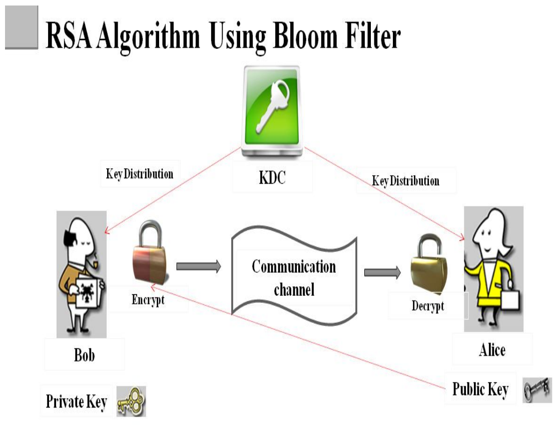


Fig 3 RSA Algorithm using Bloom Filter

Bloom Filter

Bloom filters are compact an optimize data structures is used for probabilistic representation of a set in order to support membership queries (i.e. queries that ask: "Is element X in set Y?"). This compact representation is the payoff for allowing a small rate of false positives in membership queries; that is, queries might incorrectly recognize an element as member of the set.

1 Constructing Bloom Filters

Consider a set $A=\{a_1,a_2,...,a_n\}$ of n elements. Bloom filters describe membership information of *A* using a bit vector *V* of length *m*. For this, *k* hash functions, $h_1,h_2,...,h_k$ with $h_i : X \rightarrow \{1..m\}$, are used as described below:

The following procedure builds an *m* bits Bloom filter, corresponding to a set A and using $h_1,h_2,...,h_k$ hash functions Therefore, if a_i is member of a set *A*, in the resulting Bloom filter *V* all bits obtained corresponding to the hashed values of a_i are set to 1. Testing for membership of an element *elm* is equivalent to testing that all corresponding bits of *V* are set:

Bloom Filters

One prominent feature of Bloom filters is that there is a clear tradeoff between the size of the filter and the rate of false positives. Observe that after inserting *n* keys into a filter of size *m* using *k* hash functions, the probability that a particular bit is still 0 is:

$$p_0 = \left(1 - \frac{1}{m}\right)^{kn} \approx 1 - e^{-\frac{kn}{m}}. \tag{1}$$

(Note that we assume perfect hash functions that spread the elements of *A* evenly throughout the space {1..*m*}. In practice, good results have been achieved using MD5 and other hash functions [10].)
Hence, the probability of a false positive (the probability that all *k* bits have been previously set) is:

$$p_{err} = (1 - p_0)^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-\frac{kn}{m}}\right)^k \tag{2}$$

In (2) p_{err} is minimized for $k = \frac{m}{n}\ln 2$ hash functions. In practice however, only a small number of hash functions are used. The reason is that the computational overhead of each hash additional function is constant while the incremental benefit of adding a new hash function decreases after a certain threshold (see Fig 4).

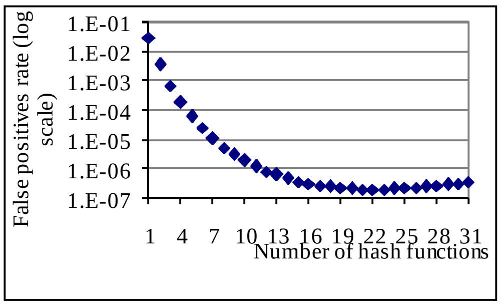


Figure 1: False positive rate as a function of the number of hash functions used. The size of the Bloom filter is 32 bits per entry (m/n=32). In this case using 22 hash functions minimizes the false positive rate. Note however that adding a hash function does not significantly decrease the error rate when more than 10 hashes are already used.

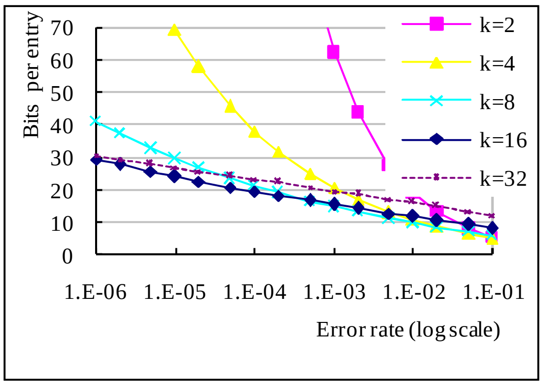


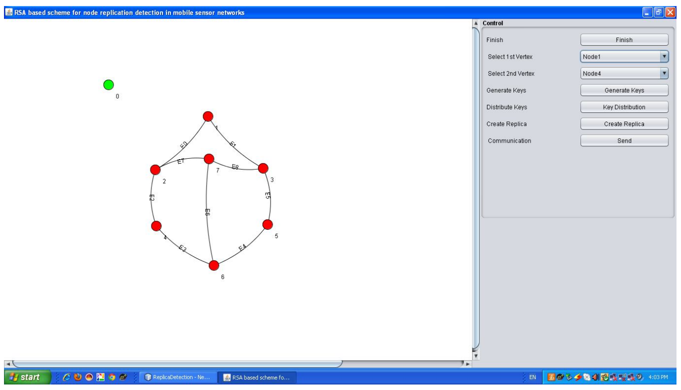
Figure 4: Size of Bloom filter (bits/entry) as a function of the error rate desired. Different lines represent different numbers of hash keys used. Note that, for the error rates considered, using 32 keys does not bring significant benefits over using only 8 keys.

(2) is the base formula for engineering Bloom filters. It allows, for example, computing minimal memory requirements (filter size) and number of hash functions given the maximum acceptable false positives rate and number of elements in the set (as we detail in Figure 5).

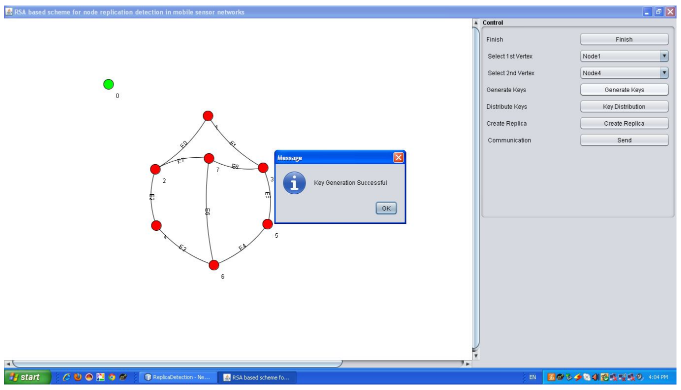
$$\frac{m}{n} = \frac{-k}{\ln\left(1 - e^{\frac{\ln p_{err}}{k}}\right)} \text{ (bits per entry)} \tag{3}$$

Bloom filters are compact data structures for probabilistic representation of a set in order to support membership queries. The main design tradeoffs are the number of hash functions used (driving the computational overhead), the size of the filter and the error (collision) rate. Formula (2) is the main formula to tune parameters according to application requirements.

4. RESULTS

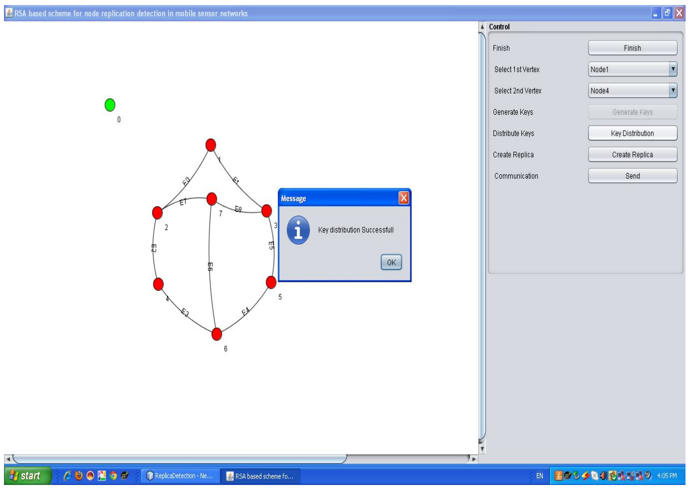

RSA based scheme for node replication detection in mobile sensor networks

Control
Select 1st Vertex: Node1
Select 2nd Vertex: Node4


RSA based scheme for node replication detection in mobile sensor networks

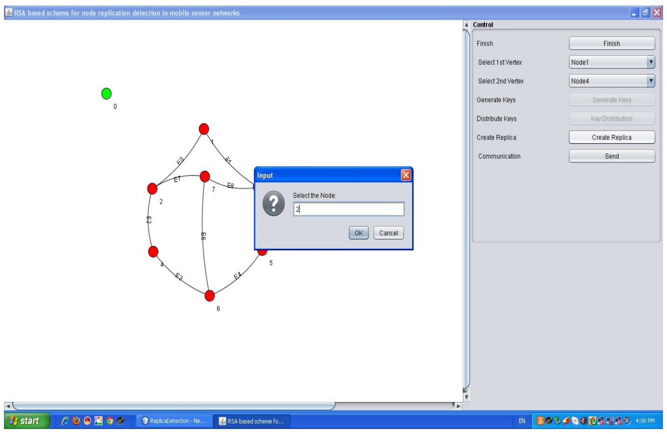
Message
Key Generation Successful

Select 1st Vertex: Node1
Select 2nd Vertex: Node4


RSA based scheme for node replication detection in mobile sensor networks

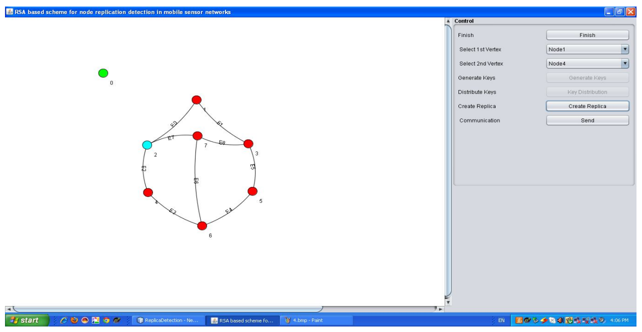
Message
Key distribution Successfull

Select 1st Vertex: Node1
Select 2nd Vertex: Node4


RSA based scheme for node replication detection in mobile sensor networks

Input
Select the Node: 2

Select 1st Vertex: Node1
Select 2nd Vertex: Node4


RSA based scheme for node replication detection in mobile sensor networks

Control

Finish: Finish
Select 1st Vertex: Node1
Select 2nd Vertex: Node4
Generate Keys: Generate Keys
Distribute Keys: Key Distribution
Create Replica: Create Replica
Communication: Send

5. CONCLUSION

In this paper, focus on replication detection RSA algorithms for mobile sensor networks, In proposed solution consist of Bloom filter Based key storage and RSA based Scheme to detect node replication in mobile sensor network. proposed solution is efficient and effective to store a data within less memory space and in encrypted form .hence propose solution is feasible for detecting node replication attack.

REFERENCES

[1] Florian Hess. Efficient identity based signature schemes based on pairings. In Proceedings of the 9th Annual International Workshop on Selected Areas in Cryptography (SAC'02), pages 310–324, 2002.
[2] David J. Malan, Matt Welsh, and Michael D. Smith. A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography. In Proceedings of IEEE Conference on Sensor and Ad Hoc .
[3] Bryan Parno, Adrian Perrig, and Virgil Gligor. Distributed
[4] detection of node replication attacks in sensor networks. In Proceedings of The 2005 IEEE Symposim on Security and Privacy (S&P'05), pages 49– 63, 2005.
[5] M. Conti, R. D. Pietro, and A. Spognardi, .“Wireless sensor replica detection in mobile environment,.” in Proc. Int. Conf. Distributed Computing and Networking (ICDCN), Hong Kong, China, 2012, pp. 249.–264.