corresponding to U with the value dsn and, therefore, future route discoveries performed by D to obtain a route to U will fail (because U's destination sequence number will be much smaller than the one stored in D's routing table).

vii. According to the current AODV draft, the originator of a RREQ can put a much bigger destination sequence number than the real one. In addition, sequence numbers wraparound when they reach the maximum value allowed by the field size. This allows a very easy attack in where an attacker is able to set the sequence number of a node to any desired value by just sending two RREQ messages to the node.

5.10 Broadcast Authentication WSN Protocol - TESLA Protocol, Biba Protocol

5.10.1 Broadcast Authentication WSN Protocol

1. Achieving broadcast security is a must for wireless sensor networks. Hence it is necessary for the base station to broadcast commands and data to sensor nodes.
2. Without secure communication, sensors may be involved in incorrect operations and can't meet the network requirements. The current security solutions for wired and wireless networks cannot be utilized for a wireless sensor network because of the energy, memory and computation restrictions of the latter. These limitations make the design and operation completely dissimilar to those of regular wireless networks.
3. Broadcast authentication based on asymmetric key cryptography cannot deal with the limited resource constrains. Symmetric key cryptography and hash functions are cheaper in their computational requirements and are more widely utilized in sensor networks.
4. WSNs' broadcast authentication was first covered by TESLA, and μTESLA that provides the asymmetric cryptographic property of authenticated broadcast through delayed disclosing (time-varying) of symmetric keys. The base-station installs a key chain by repeatedly applying a one way hash function (OWHF) to an initial random value, called seed. The chain construction allows nodes to verify the authenticity of the disclosed keys. Loosely time synchronized and MAC (Message Authentication Code) generations are required. Revelation of session keys by the base-station is delayed, thus

allowing nodes to verify the key validity.

5. Multilevel μTESLA is designed to reduce the need to reinitialize the network by implementing multiple levels of key chains, in which high-level keys are used to communicate root-keys (or commitments) for low-level chains, which are used in turn for broadcast authentication as in standard μTESLA. Network lifetime is extended. Significant computation and storage are required. Receivers can't deal with the received messages instantly and have to store them within one or several time intervals. Considering the broadcasting of urgent messages like alerts and alarms; the TESLA family has great shortcomings in dealing with such matters. Furthermore, the delayed authentication can be subject to Denial-of- Services (DoS) attacks.
6. Merkle tree utilization was introduced to overcome this shortage in bandwidth and storage resources utilization. TIK was proposed to achieve immediate authentication based on sensitive time synchronization between the sink and the receiving nodes. However, this technique is not suitable for WSNs. Sensor nodes have a limited battery life, which can make using asymmetric key techniques impractical as they use much more energy for their mathematical calculations.

5.10.2 Desirable Security Attributes for Authenticated Broadcast

1. Data Integrity

Data integrity ensures that data has not been altered by unauthorized entities.

2. Data Origin Authentication

Data Origin Authentication guarantees the origin of data. It is a fundamental step in achieving entity authentication in protocols as well as establishing keys.

It can be said that data origin authentication implies data integrity. So it is not possible to achieve data integrity without data origin authentication.

3. Freshness

Packets that have been captured and replayed at a later time should be ignored by the sensor nodes.

4. Delay Tolerance

No time synchronization should be required in the system for data verification. Each packet must be verifiable without having to wait for additional data.

5. Confidentiality

Confidentiality ensures that data is only available to those authorized to obtain it.

6. Denial-of-Service Attack

The denial of service attack is an attempt to make a node resource unavailable to its intended users.

7. Small Challenge Attack

This attack challenges the backward hashing with small values to respond with the chain initial values.

8. Limitation for N times Authentications

Process re-initialization after N of authentications is necessary.

5.10.3 TESLA Protocol

1. TESLA (Timed Efficient Stream Loss-tolerant Authentication) broadcast authentication protocol, an efficient protocol with low communication and computation overhead, which scales to large numbers of receivers, and tolerates packet loss. TESLA is based on loose time synchronization between the sender and the receivers.
2. Despite using purely symmetric cryptographic functions (MAC functions), TESLA achieves asymmetric properties.
3. The main idea of TESLA is that the sender attaches to each packet a MAC computed with a key k known only to itself. The receiver buffers the received packet without being able to authenticate it. A short while later, the sender discloses k and the receiver is able to authenticate the packet. Consequently, a single MAC per packet suffices to provide broadcast authentication, provided that the receiver has synchronized its clock with the sender ahead of time.
4. TESLA requires that the receivers are loosely time synchronized with the sender.
5. TESLA also needs an efficient mechanism to authenticate keys at the receiver
6. In TESLA, the elements of the one-way chain are keys, so the chain is called as a one-way key chain. Furthermore, any key of the one-way key chain commits to all following keys, so such a key is called ad a one-way key chain commitment, or simply key chain commitment.
7. **Time Synchronization in TESLA -** Time Synchronization in TESLA does not need the strong time synchronization properties that sophisticated time synchronization protocols provide, but only requires loose time synchronization, and that the receiver knows an upper bound

on the sender's local time.

8. **A viable broadcast authentication protocol has the following requirements,**

- Low computation overhead for generation and verification of authentication information.
- Low communication overhead.
- Limited buffering required for the sender and the receiver, hence timely authentication for each individual packet.
- Robustness to packet loss.
- Scales to a large number of receivers.
- The TESLA protocol meets all these requirements with low cost and it has the following special requirements,
- The sender and the receivers must be at least loosely time-synchronized.
- Either the receiver or the sender must buffer some messages.
- Despite the buffering, TESLA has a low authentication delay. In typical configurations, the authentication delay is on the order of one round-trip delay between the sender and receiver.

9. TESALA Protocol Process

(A) The Sender

- The sender splits up the time into time intervals of uniform duration. Next, the sender forms a one-way chain of self-authenticating values, and assigns the values sequentially to the time intervals (one key per time interval). The one-way chain is used in the re-verse order of generation, so any value of a time interval can be used to derive values of previous time intervals. The

sender defines a disclosure time for one-way chain values, usually on the order of a few time intervals. The sender publishes the value after the disclosure time.

- The sender attaches a MAC to each packet. The MAC is computed over the contents of the packet. For each packet, the sender determines the time interval and uses the corresponding value from the one-way chain as a cryptographic key to compute the MAC. Along with the packet, the sender also sends the most recent one-way chain value that it can disclose.
- Each receiver that receives the packet per-forms the following operation. It knows the schedule for disclosing keys and, since the clocks are loosely synchronized, can check that the key used to compute the MAC is still secret by determining that the sender could not have yet reached the time interval for disclosing it. If the MAC key is still secret, then the receiver buffers the packet.
- Each receiver also checks that the disclosed key is correct (using self-authentication and previously released keys) and then checks the correctness of the MAC of buffered packets that were sent in the time interval of the disclosed key. If the MAC is correct, the receiver accepts the packet.
- One-way chains have the property that if inter-mediate values of the one-way chain are lost, they can be recomputed using later values. So, even if some disclosed keys are lost, a receiver can recover the key chain and check the correctness of packets. The sender distributes a stream of messages {Mi}, and the sender sends each message M_i in a network packet Pi along with authentication information. The broadcast channel may be lossy, but the sender does not retransmit lost packets. Despite packet loss, each receiver needs to authenticate all the messages it

receives.

Sender Setup

(i) TESLA uses self-authenticating one-way chains. The sender divides the time into uniform intervals of duration Tint. Time interval 0 will start at time T_0, time interval 1 at time $T_1 = T_0 + Tint$, etc. The sender assigns one key from the one-way chain to each time interval in sequence. The one-way chain is used in the reverse order of generation, so any value of a time interval can be used to derive values of previous time intervals. The sender determines the length N of the one-way chain $K_0, K_1, ..., K_N$, and this length limits the maximum transmission duration before a new one-way chain must be created.

(ii) The sender picks a random value for KN. Using a pseudo-random function f, the sender constructs the one-way function F: F(k) = fk(0). The remainder of the chain is computed recursively using Ki = F(Ki+1).

(iii) Note that this gives Ki = FN−i(KN), so one can compute any value in the key chain from KN even if it does not have intermediate values. Each key Ki will be active in time interval i.

(iv) The time synchronization property that TESLA requires is that each receiver can place an upper bound of the sender's local time.

(v) The sender sends the key disclosure schedule by transmitting the following information to the receivers over an authenticated channel (either via a digitally signed broadcast message, or over unicast with each receiver),

- **Time interval schedule -** Interval duration Tint, start time Ti and index of interval i, length of one-way key chain. There is Key disclosure delay d (number of intervals). A key commitment to the key chain Ki(i< j−d where j is the current interval index).

(B) Broadcasting Authenticated Messages - Each key in the one-way key chain corresponds to a time interval. Every time a sender broadcasts a message, it appends a MAC to the message, using the key corresponding to the current time interval. The key remains secret for the next d−1 intervals, so messages sent in interval j effectively disclose key K_{j-d}.

(C) Authentication at Receiver - When a sender discloses a key, all parties potentially have access to that key. An adversary can create a bogus message and forge a MAC using the disclosed key. So as packets arrive, the receiver must verify that their MACs are based on safe keys: a safe key is one that is only known by the sender, and safe packets or safe messages have MACs computed with safe keys. Receivers must discard any packet that is not safe, because it may have been forged.

10. TESLA Security Considerations

The security of TESLA relies on the following assumptions,

- The receiver's clock is time synchronized up to a maximum error of Δ. (because of clock drift, the receiver periodically resynchronizes its clock with the sender.)
- The functions F, F′are secure PRFs, and the function F furthermore provides weak collision resistance.

5.10.4 TESLA Family Broadcast Authentication

Timed Efficient Stream Loss-tolerant Authentication (TESLA) is a multicast stream authentication protocol. Keys used to authenticate the i-th message is disclosed along with (i + 1) th message. μTESLA provides authentication for data broadcasts, and requires that base station and sensor nodes be loosely time synchronized.

According to Lamport's scheme, a base station (BS) randomly selects the last key k_n, the chain seed, and applies a one-way public function $h(\cdot)$ to generate the rest of keys: $k_0, k_1, ..., k_{n-1}$ as $k_i = h(k_{i+1})$. Given k_i, every sensor node can generate the sequence $k_0, k_1, ..., k_{n-1}$.

However, given k_i, no one can generate k_{i+1}. At i-th time slot, BS sends an authenticated message MAC_{ki} (message). Sensor nodes store the message till the verification key in the (i + 1)-th time slot is disclosed. Sensor nodes verify disclosed key k_{i+1} by using key k_i as k_i = $h(k_{i+1})$. In μTESLA, nodes are required to store a message until the authentication key is disclosed. This operation may create storage problems, and encourages DoS types of attacks.

μTESLA has been expanded to Multi-level μTESLA by simplifying the key distribution phase and introducing a new concept of a multi-level key chain generation using pseudo-random functions that improves the protocol efficiency. Multi-level μTESLA reduces the need to reinitialize the network (although re-initialization is still required) by implementing multiple levels of key chains, in which high-level keys are used to communicate root-keys (or commitments) for low-level chains which are used in turn for broadcast authentication as in standard μTESLA.

The chains are further connected in that each root-key is derived from the corresponding high-level chain using another pseudo-random function. Network lifetime is extended many times over, but it is still limited. A problem would result if a receiver dropped a related commitment distribution message initializing a new low-level chain; it would be unable to verify any broadcast data received during this entire lifetime of the chain itself. The data would still be verifiable eventually as the receiver could use any later commitment distribution message to reconstruct all the lost high-level keys

and the corresponding chains. This would require significant computation and storage.

5.10.5 BiBa Signature Protocol

1. BiBa stands for Bins and Balls signature in which a collision of balls under a hash function in bins forms the signature. BiBa exploits the birthday paradox such that the signer has many balls to throw into the bins which results in a high probability to find a signature, but an adversary has few balls so it has a low probability to forge a signature.
2. The BiBa protocol is a general solution that can be applied to sign broadcast data based on one-way functions without trapdoors. The BiBa signature scheme is efficient, robust to packet loss and scales well to a large number of receivers.
3. However, the public keys used in the BiBa protocol are large and the time to generate the signatures is long.
4. The small signature sizes make the BiBa protocol a good candidate for the system which is to be deployed over a bandwidth constrained network. Moreover, small signature verification overhead allows the end devices to be simple and cheap.
5. The sender divides time into periods of equal duration. The sender then creates t chains of SElf Authenticating vaLues (SEALs), S<1;i>; :::S<t;i>, and a Salt chain, K_i, associated with time interval i. The SEAL and Salt chains are of length 'l',

   hence they last 'l' time intervals. The Salt key is used by the sender to create the SEALs and is required for authentication of SEALs at the receiver. The SEALs are generated recursively by applying a pseudo-random function F as follows : S<i;j> = FS<i;j+1>(Kj+1); for (1 <= i <= t) and (1 <= j <= l). The use of the Salt key forces an attacker to obtain the pre-image of the Salt

chain as a pre-requisite to finding the pre-images of the SEAL chains. Therefore an attacker cannot precompute the SEALs for subsequent time periods without knowledge of the Salt key.

6. At the beginning of each active interval, the sender broadcasts the value of the active Salt (K_i) to the receivers. The dotted box in above Fig. 5.10.1 shows an active time interval with the associated Salt key and SEALs. To sign a message m during the active interval i, the sender creates a hash of the message h = H(mjc), which is used to seed a hash function Gh () used to produce a signature; where c is a counter that is incremented when a signature could not be obtained.

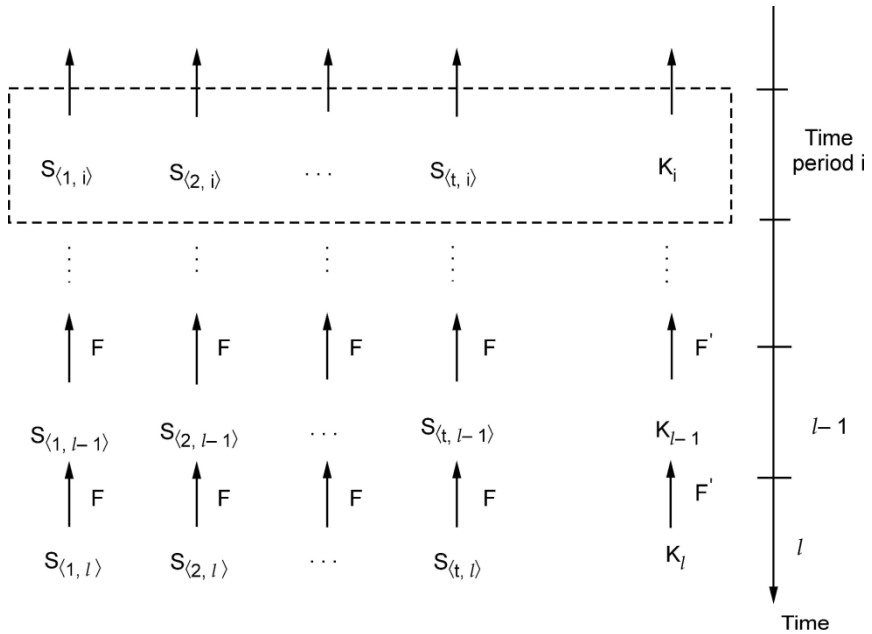


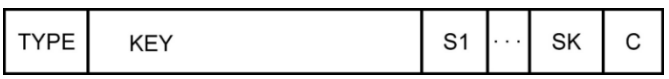
Fig. 5.10.1 The BiBa broadcast protocol dynamics

7. The sender uses the hash function Gh() on the t SEALs and observes any k-way collisions from distinct SEALs. That is, S<1;i> != S<2;i> != .. != S<k;i> such that Gh(S<1;i>) = Gh(S<2;i>).. = Gh(S<k;i>). The k SEALs that result in a collision form the signature and are then sent together with the message as (< S1; .. ; Sk > || m).
8. The receiver then authenticates the message if Gh(S1) = .. = Gh(Sk) and S1 != .. != (Sk). During signature generation, it is possible that Gh() applied on all t SEALs fails to produce at least k collisions, in which case a signature cannot be formed. The counter c serves to get a different hash value h in the event that Gh() fails to produce at least k collisions from all t SEALs. The receiver is assumed to know the value k, the hash function H and hash function family G.

9. **The security of the BiBa protocol relies** on the fact that a potential attacker knows fewer SEALs than the sender with which to forge a signature. Therefore, the sender only reveals the SEALs that are used in creating a signature. The receiver is able to verify that an adversary has a smaller number of SEALs with which to forge a false signature by relying on time synchronization. The BiBa protocol requires loose synchronization between the sender and receiver. When a receiver receives a signed message, it verifies that the sender has not yet revealed r SEALs based on synchronization. If the sender and receivers have a maximum synchronization error of ∂, the sender can only send at most (r=k) messages within ∂ time without compromising the security, where r is the maximum number of active SEALs an attacker is allowed to know and k is the number of SEALs revealed in one message.

10. **Protocol Messages**

Figure below shows the structure of the messages sent by the BiBa protocol. A description of the structure of the protocol messages sent to facilitate authentication using BiBa instances is given below. Reference



is made to Fig. 12.10.2 to describe the different fields of the messages.

Fig. 5.10.2 Structure of BiBa messages

TYPE - Describes the type of data that is carried in the message.

0 - Application messages are carried in the KEY field

1 - Short-term BiBa instance commitment key is carried in the KEY field 2 - A Salt key is carried inthe KEY field

3 - Long-term BiBa commitment key. This option is not used if the long-

term BiBa instance commitment is done off-line.

KEY - The Data that is being sent in the message which is signed. Depending on the value of the TYPE field it can either be a message sent by the application or Salt key to be signed by the short-term BiBa instance, or short-term BiBa commitment key.

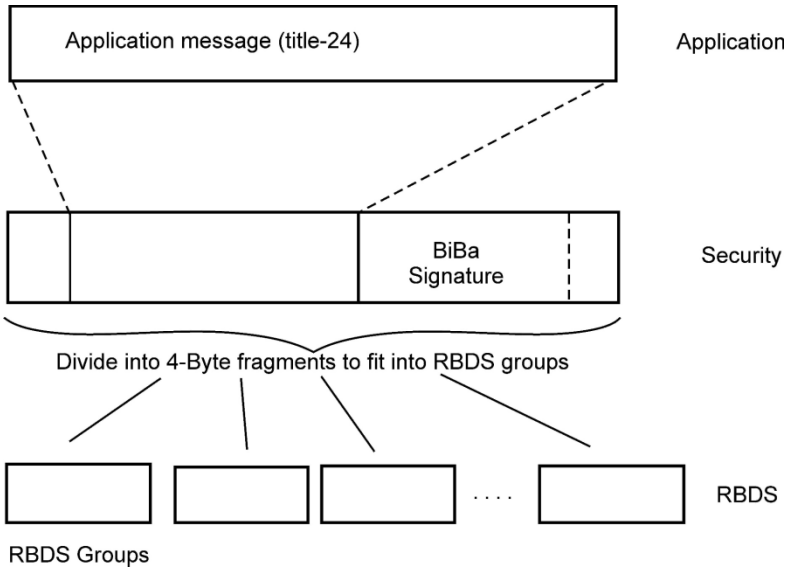
S1...SK - Part of the signature formed by the k SEALs that resulted in a collision. The size depends on the value ofk.

C - The counter that is incremented when a signature is not obtained, which is part of the signature.

11. Fig. 5.10.3 below shows the actions applied to application messages as they traverse through the different layers. The reverse operation is performed at the receiver. The application generates messages as described in the Title-24 specification. The messages are then delivered to the System operator who encrypts them for authentication purposes and sends them over the RDS network. The messages are sent over the RDS network as type-11A groups.

    Each RDS group can only carry 4 bytes of data, so the message is fragmented into multiple RDS groups and sent over the network. The receiver reconstructs the messages from the multiple RDS groups and

sends it up the protocol stack to the security layer. The security layer then authenticates the messages and present them to the application



layer if the authentication is successful.

Fig. 5.10.3 Operations on the application messages

12. Setting the BiBa Parameters

The parameters of the security layer are based on approximated application data rate. A BiBa instance with 1024 SEAL chains (t = 1024), using 4-way collisions (k = 4) can be used to sign 25 messages ($v = t_y /k$), with y = 0:10 ; where y is the fraction of SEALs that can be revealed to an adversary without compromising the security of the protocol (typically y = 10 %). An adversary is only allowed to learn r SEALs from one active period; where $r = t_y$. Each signature reveals k SEALs to the adversary, hence only ($v = t_y /k$) messages can be signed within a single time interval. An adversary who knows r SEALs needs to make 2^{35} computations to forge a valid signature of a BiBa instance with the above parameters. If it is assumed that the application

sends an average of 20 event messages every day, a single time interval for the short-term BiBa instance is sufficient to authenticate an entire day's messages. Consequently, a short-term BiBa instance with SEAL chain lengths of 50 (l = 50), can be used for 50 days before it expires. If the long-term BiBa instance is designed with the same parameters as the short-term instances (that is t = 1024, k = 4, y = 10%), then it can be used to commit 25 short-term BiBa instances in a single time interval. A single time interval for the long-term BiBa instance can then be made to last up to 1250 days (3.4 years). The entire long-term BiBa instance will then last 171 years.