Original Research Paper

Strengthening the Integrity of Forwarding First Communication Using Forward Key Chain and Bloom Filter in the Wireless Sensor Networks

[1]Muhammad Taqqyyudin Ilham Ramadhan, [1]Farah Afianti, [1]Bambang Ari Wahyudi, [1]Prasti Eko Yunanto and [2]Dedy Rahman Wijaya

[1]*School of Computing, Telkom University Bandung, 40257, Indonesia*
[2]*School of Applied Science, Telkom University Bandung, 40257, Indonesia*

Article history
Received: 21-04-2022
Revised: 26-09-2022
Accepted: 05-12-2022

Corresponding Author:
Farah Afianti
School of Computing, Telkom University Bandung, 40257, Indonesia
Email: farahafi@telkomuniversity.ac.id

Abstract: Wireless Sensor Networks (WSNs) are collections of sensor nodes that can help people monitor things wirelessly, replacing the need for human presence in hard-to-reach areas. While forwarding first communication reduces communication delays among the sensor nodes, it lacks a security mechanism. Since sensor nodes have limited resources, using complex security measures can shorten their lifetime. This study proposes a protocol modification for forwarding first communication using a forward keychain and Bloom Filter Vector (BFV). The sender is required to send a BFV in every message exchange. The process of sending the BFV is separated to increase the size of the message payload in each communication. The results show that while the communication overhead and verification time increase to an acceptable level, the message payload is larger and message integrity is guaranteed.

Keywords: Bloom Filter Vector, Forward Key Chain, Forwarding First, Wireless Sensor Network

Introduction

Wireless Sensor Network (WSN) is a group of nodes capable of communicating wirelessly and collecting data between nodes. WSN can solve the cabling problems that traditional networks once faced. In addition, WSN has many other advantages such as small size, ease of deployment, and ease of configuration (Yu *et al.*, 2020). Because of these advantages, many studies have been conducted to apply WSN in daily activities, such as (Abdullah *et al.*, 2017) who developed a system to detect early symptoms of forest fires based on ambient air temperature, humidity, oxygen, and carbon dioxide levels, as well as wind speed and direction (Abdullah *et al.*, 2017). In addition, (Xu and Liu, 2017) in their research, applied WSN to monitor water quality with parameters such as water temperature, PH value, and oxygen solubility. Moreover, WSNs can be used in monitoring systems (Lalar *et al.*, 2019), transportation (Aguirre *et al.*, 2016), healthcare (Yao *et al.*, 2019), etc. This shows that the scope of WSN is quite broad and covers all aspects of life.

However, these nodes only have limited hardware usage capacity, such as memory capacity, battery life, and processor speed (Zhao *et al.*, 2018). Limits on memory capacity determine the amount of data that can be stored, while battery life determines the age of nodes, and slow processors limit node performance, so these nodes cannot be paired with security systems that have a high level of complexity. These sensor nodes will be accessed by many other devices to communicate. Therefore, a low-computation authentication process is needed to verify the connected devices. The forwarding first method is developed to reduce communication delay, where the node will forward the message to other nodes before the sender of the message is verified. This method was developed because the Authentication first method is vulnerable to Denial of Service (DoS) attacks where attackers will send fake packets in large numbers to sensor nodes, thus consuming resources on those nodes (Chuchaisri and Newman, 2012). An illustration of the use of these two methods is shown in Fig. 1.

Due to these conditions, the application of forward Fig. 1. Comparison of forwarding first and authentication first keychain and bloom filters are appropriate to overcome the encountered problems. The forward keychain only requires less memory than the backward keychain because it will only store one commitment key, besides that the use of the key can be done repeatedly without being limited and this scheme will be more secure because the sender does not have to always send keys and messages in every communication process (Chuchaisri and Newman, 2012) and the bloom filter only requires very low computation. Also, bloom filters required efficient storage, which is especially important for nodes with limited resources.

In addition, all the notations used are summarized in Table 1.

Several studies that discuss the use of cyber schemes in wireless sensor networks have been carried out. For example, the research conducted by Du *et al*. (2008) introduces a scheme to secure sensor nodes from DoS attacks when authenticating received messages. They use a sender specific one-way keychain to authenticate incoming messages. Their research shows that the proposed technique is more secure and efficient, but this method will result in high storage overhead because it must store all user identities and the commitment key of each user (Du *et al*., 2008).

Ning *et al*. (2008) have also conducted a study to secure sensor nodes by using a backward keychain. The results of the research result indicate that the use of backward keychain schemes is quite good in overcoming the threat of DoS attacks.

Huang *et al*. (2011) introduced a method with dynamic keys in the Heterogeneous Sensor Network. The proposed scheme implements hash functions on the base station, cluster head, and sensor nodes. Cluster heads and sensor nodes will create their own keychain to prevent attacks. This study showed satisfactory results to prevent several attacks. However, this scheme still has a challenge regarding the number of keys.

Chuchaisri and Newman (2012) used a forward keychain scheme and bloom filter to distribute the secret key to every existing node. It creates security to ward off DoS attacks on the WSN. However, the confidentiality of the message was not discussed.

Dong *et al*. (2013) introduces ways to mitigate DoS attacks by using digital signatures to authenticate messages on sensor networks. They proposed 2 filtering techniques, namely, group-based, and keychain-based filtering. Both Techniques can reduce unnecessary authentication when a DoS attack occurs.

Hu *et al*. (2018) focus on improving keychain distribution by using blockchain. They avoid a centralized system because of a single point of failure. Furthermore, the keychain integrity is guaranteed. The proposed system increases the key distribution performance both in time and storage space.

A backward keychain is also used as a session key to produce a puzzle scheme (Afianti and Suryani, 2018). This scheme defends signature-based DoS attacks in the IoT environment. The experimental results show that the decreased performance because of the additional security mechanism is still tolerable.

Keychain techniques are used as supporting methods in the authentication process. However, the main method needs high computation. Therefore, forwarding first is the option that can be chosen. The trade-off between the security aspect and time consumption becomes a challenge in its implementation.

The authentication schemes that are well-known in IoT implementation such as zero knowledge and HMAC (Thungon and Hussain, 2021). Zero knowledge is chosen because of low memory consumption and computation rather than symmetric cryptography. Fiat Shamir is a protocol that uses zero knowledge proof to authenticate the identity of the entity. Therefore, this study compares both schemes in the forwarding first implementation.

Wireless Sensor Networks

Wireless Sensor Network (WSN) is a group of nodes capable of communicating wirelessly and collecting data between nodes. The size of each sensor node is usually very small, it can even reach the size of a match. A wireless sensor network consists of thousands of nodes that are connected either directly or indirectly.

Some of the characteristics of a wireless sensor network are (Kandris *et al*., 2020):

a. The network topology can be made according to the needs, depending on the condition of the existing nodes
b. Low production costs and easy node deployment
c. Each sensor node must have adequate security to prevent attacks
d. Due to the large number of nodes used to measure the existing data, the energy use must be managed as low as possible

Apart from the above characteristics, wireless sensor networks also have several other advantages compared to wired sensors, (Dargie and Poellabauer, 2010):

a. Simple, practical, or concise because there is no need for complicated cable installation (wireless) and in certain geographical conditions it is very advantageous
b. It is mobile, or nodes can be moved easily to another location for more accurate measurements without the need to rearrange the layout of the room

But behind the advantages, wireless sensor networks also have disadvantages when compared to Wired Sensors, including Dargie and Poellabauer (2010):

a. Higher load compared to Wired Sensor makes WSN data rate lower
b. The more sensors there are, the bigger the collision domains, making the WSN security aspect weaker and the data rate even smaller
c. Periodic maintenance is required to replace the existing battery on the sensor when it runs out

Table 1: Nomenclature

Notation	Description
N	Number of keychains
c	Index
s	Sender number
tt	Timestamp
hashed num	Hashed keychains
digital sign	Private key digital signature
M	Message + timestamp
M2	Message + timestamp received
Hexa bloom	Hex value of bloom filter
H_{bfv}	HMAC value of bloom filter
Verif H_{bfv}	HMAC verification value of bloom filter
Time now	Time now
c'	Last index received
hashed num2	Keychains constructed by the receiver
Comm key2	Commitment key

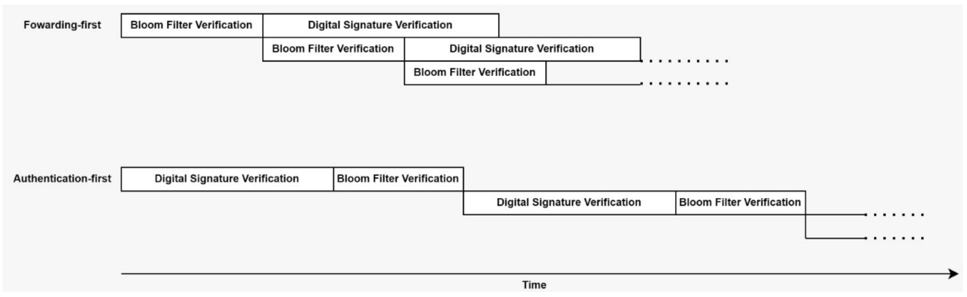

Fig. 1: Comparison of forwarding first and authentication first

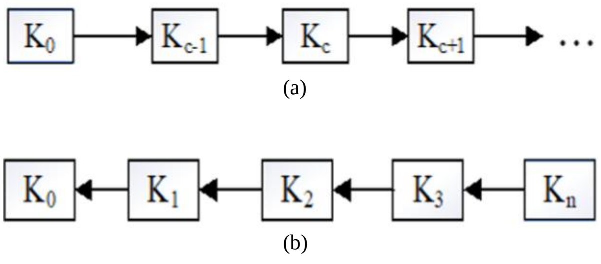

Fig. 2: Keychain model: (a) Forward keychain and (b) backward keychain

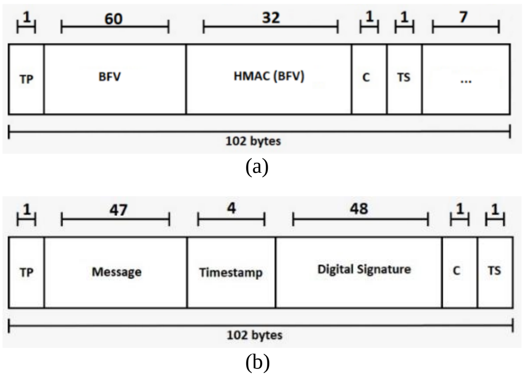

Fig. 4: (a) Packet A payload and (b) packet B payload

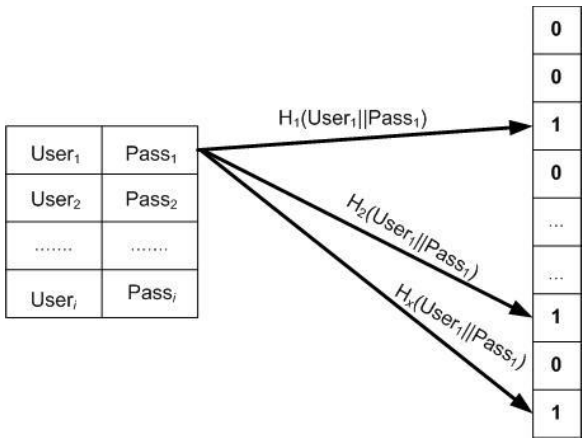

Fig. 3: Bloom filter mechanism

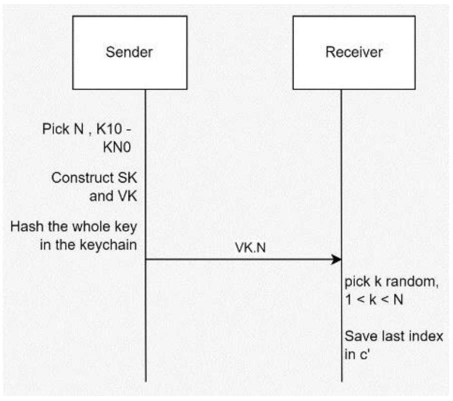

Fig. 5: Pre-deployment phase

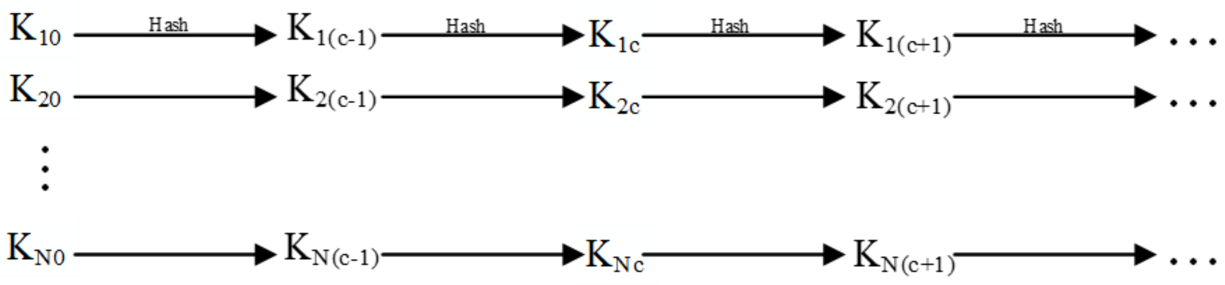
$$\begin{tikzcd}
K_{10} \arrow[r, "Hash"] & K_{1(c-1)} \arrow[r, "Hash"] & K_{1c} \arrow[r, "Hash"] & K_{1(c+1)} \arrow[r, "Hash"] & \cdots \\
K_{20} \arrow[r] & K_{2(c-1)} \arrow[r] & K_{2c} \arrow[r] & K_{2(c+1)} \arrow[r] & \cdots \\
\vdots & & & & \\
K_{N0} \arrow[r] & K_{N(c-1)} \arrow[r] & K_{Nc} \arrow[r] & K_{N(c+1)} \arrow[r] & \cdots
\end{tikzcd}$$

Fig. 6: Key sequence hashing process

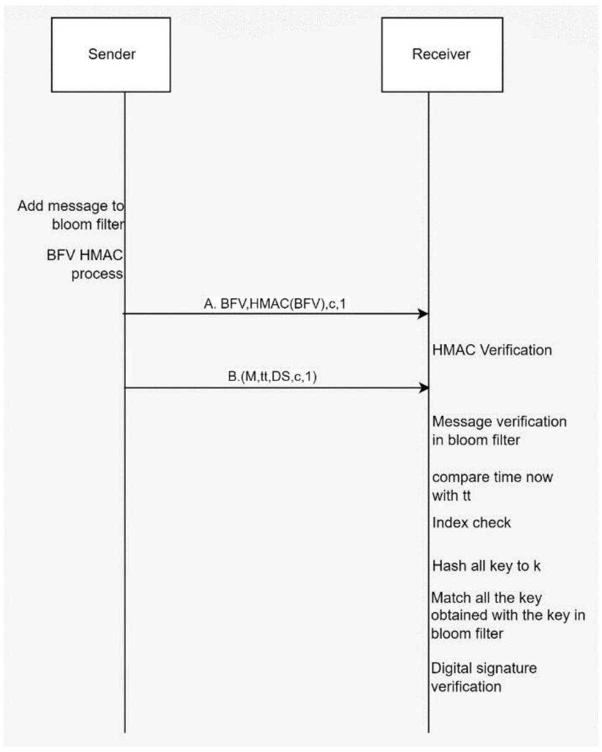
Sender
Receiver
Add message to bloom filter
BFV HMAC process
A. BFV,HMAC(BFV),c,1
HMAC Verification
B.(M,tt,DS,c,1)
Message verification in bloom filter
compare time now with tt
Index check
Hash all key to k
Match all the key obtained with the key in bloom filter
Digital signature verification

Fig. 7: Signature generation, message verification, and forwarding

Keychain

A keychain can be called a one-way keychain, has been widely used in various scenarios to provide efficient authentication. There are two main steps to generating a keychain. First, the sender is required to choose a random value K_n which will be used as the last key in the backward keychain or as the first key in the forward keychain. Second, the sender must perform the hash function repeatedly to generate the next keys. The illustration of both keychains can be seen in Fig. 2.

The forward keychain has several advantages compared to backward, such as the forward keychain does not need to send a key in every communication between nodes; if the key is not known by the attacker, then the number of available keys is unlimited. However, the forward keychain has a weakness, such as: If the latest indexed key is known by the attacker, then the attacker will be able to find out the key which will be used in the next session. Therefore, the user is required to generate a new key, called a rekeying process, and send the new commitment key to all nodes. The commitment key distribution is a sign that a new keychain has been built.

Compared to the forward keychain, the backward type has a better security level. Even if the attacker already has the previous key, they will not be able to guess what key will be used next due to the nature of the hash function which is irreversible. However, the backward keychain has several drawbacks, including the limited number of keys and the high communication overhead. The number of keys is limited because the user must define it at the beginning of the keychain generation, and high communication occurs due to the space required to send the key used for verification on each message distribution (Ning *et al*., 2008).

The keychain scheme aims to eliminate the computation overhead in the key generation, by using several hash functions. Each random hash value prevents the next key to be easily guessed. At the same time, it also eliminates the need to transmit each session key in the broadcast messages. The hash function is denoted by $H(\cdot)$ and the initial key as K_0. The key with position or index i can be calculated by Eq. 1:

$$K_i = H\left(K_{i-1}\right) \tag{1}$$

Hash

A hash function is a mathematical function that produces output in a fixed size even with different input sizes. In addition, the hash function has an irreversible property. It means that the resulting value cannot be reversed back to its previous form. Therefore, hash functions are widely used for cryptographic purposes. The hash value h is generated by the hash function H based on Eq. 2:

$$h = H(M) \tag{2}$$

M is the message to be generated, H is the hash function and h is the hash value with a fixed size based on the hash function algorithm (Stallings and Tahiliani, 2014).

Elliptic Curve Digital Signature Algorithm (ECDSA)

ECDSA is one of the algorithms producing digital signatures using the analogy of an ellipse curve. This algorithm is widely used because it has the same security level as the RSA digital signature algorithm but has a smaller key. Therefore, ECDSA is suitable to be used in devices that have limited computation (Hyun *et al*., 2008).

A digital signature is a mathematical scheme to prove the authenticity of the entity either digital messages or user or documents. This scheme guarantees the information comes from the right source. The public key of the algorithm is needed to verify the digital signature (Husni *et al*., 2015).

Bloom Filter

Bloom filter is a simple and space-saving and probabilistic data structure to represent groups in member search (Afianti *et al*., 2021). Bloom filter was first introduced by Burton Bloom in 1970 and immediately gained popularity in database applications.

This technique will build a bit vector called Bloom Filter Vector (BFV), to represent all data and will later be used to distinguish whether an element is a member of a group or not. If the element is categorized as a member of the group, then it will be represented by bit 1. If the element is categorized as a non-member of the group, then it will be represented by bit 0 (Mbarek *et al*., 2018).

Bloom filters can be used to store usernames and passwords. The number of i data consisting of username and password will be used as an input in the x hash functions. The output of the hash operation will be mapped to the index (position of b-bit in BFV). The bit value of the position obtained after the mapping is changed to 1 (Shuai *et al*., 2018). This process is carried out until all usernames and passwords are mapped in the BFV. An illustration of the bloom filter mechanism can be seen in Fig. 3. The black arrows show the process of mapping the pair of User1 and Pass1 on the BFV.

Materials

The simulation is conducted using python language programming. In addition, the hardware specifications are summarized in the Table 2.

Table 2: Hardware specification

Parameter	Value
Processor	11th Gen Intel® Core ™ i3-1115G4 @3.00Ghz (4 CPUs), ~3.0GHz
Memory	8192MB RAM
Storage	256GB

Proposed Methods

There are two main discussions in the proposed Fig. 5 pre-deployment phase method. First, the entire communication protocol is explained. Second, the detailed steps of each phase in the sender and receiver communication are discussed.

Communication Protocol

In the proposed communication protocol, improvement is made based on the previous research by splitting the message payload into two parts, packet A and packet B, Fig. 4. That process is chosen because the maximum packet length is 102 bytes (refers to the IEEE 802.15.4 standard). By splitting the message into two packets, the required communication costs will increase compared to the previous one. However, the size of the user's message can increase. In addition, the HMAC processing to the BFV and the current key is carried out using the SHA-256 hash. It aims to ensure the integrity of BFV. The last contribution is compressing the value of the timestamp sent to the receiver into 4 bytes. The illustrations of the payload from packets A and B are depicted in (Fig. 4a-b).

Packet A consists of 1 byte of packet mark (TP), 60 bytes of BFV, 32 bytes of the HMAC (BFV), 1 byte of the index (c), and a sender sign (TS). Once combined, packet a still has 7 bytes left. Meanwhile, packet B consists of 1 byte of packet mark (TP), 4 bytes of timestamp, 48 bytes of digital signature, 1 byte of the index (c), 1 byte of sender sign (TS), and the message that can be included up to 47 bytes.

Message Transmission

The proposed communication protocol consists of three phases, namely: Pre-deployment; signature generation phase; message verification, and forwarding phase.

Pre-Deployment Phase

Pre-deployment phase is the initialization phase before message exchange between nodes occurred. The details of this process are shown in Fig. 5. There are 4 processes in this phase.

Random Number Generation

In this process, the sending node creates N bytes of random value that is used as a commitment key or the first value in the keychain.

Hashing Process

The commitment key will be hashed using SHA-256 and marked as the next key by increasing the index value. This process is repeated until reaches the desired keychain size.

Signature Key and Verifying Key Generation

The sender node creates the signing key (*sk*) and the verifying key (*vk*) using Elliptic Curve Digital Signature Algorithm. The verifying key is transmitted to the receiver to verify the digital signature of the sender node in the next communication.

The k Random Keys Generation

The illustration of generating *an N* set of key chains can be seen in Fig. 6. Each keychain set contains *c* keys. The entire value of the keychain set is only known by the sender. The receiving node only knows *k* sets of keychains, where $k < N$. The k keychain sets are selected randomly for each receiving node. The sender node sends *k* random numbers, which will be used as a commitment key by the receiving node.

The pre-deployment phase only occurs once in the communication process. The output obtained in this phase is the transmission of *k* commitment keys and a verifying key from the sender to the receiving node through a safe path.

Signature Generation Phase

After the keychain set is established, the message-sending process can be carried out. Details of this process will be shown in Fig. 7. In this phase there are four processes before the message is sent to the other nodes, such as.

Timestamp Calculation

A timestamp is a series of information that describes when an event occurred. In this process, the sender will mark the time when the message was created. It aims to prevent replay attacks. So that, the receiving node will drop the repeated packet.

Digital Signature Generation

The digital signature will be produced using ECDSA and curves from the NIST-P192 standard. The digital signature authenticates the message *M* and timestamp *tt* using Eq. 3:

$$DS = sk\left(H\left(M \parallel tt\right)\right) \tag{3}$$

Mapping the Key into the BFV

The pre-built keychain set, timestamp, and digital signature will be concatenated and hashed using SHA-256 as shown in the Eq. 4:

$$H\left(K_{NC} \parallel tt \parallel DS\right) \tag{4}$$

The hash value of the key, timestamp, and digital signature will be mapped into the BFV. The size of the BFV can be calculated using Eq. 5 (Mbarek *et al*., 2018):

$$m = -\frac{kn}{\ln\left(1 - FPP^{\frac{1}{k}}\right)} \tag{5}$$

m is the size of BFV, *k* is the number of hash functions, *n* is the number of members, and *FPP* is the number of false positive probabilities.

HMAC of BFV Processing

The *HMAC* process will be carried out using the SHA-256 hash function and *c*th key from the keychain *ts* after the BFV is filled and ready to be sent. The detailed process is described in Eq. 6:

$$H_{bfv} = HMAC\left(K_{ts(c)}, BFV\right) \tag{6}$$

The final message that will be sent to the receiving node can be seen in Eq. 7:

$$\left\{tp, BFV, H_{bfv}, c, ts\right\} \& \left\{tp, M, tt, DS, c, ts\right\} \tag{7}$$

tp is the packet marker, *c* is the current index, and *ts* is the timestamp of the message. The detailed steps of the signature generation phase can be seen in Algorithm 1.

Algorithm 1: Signature generation

```
    Input c, s, message, tt, hashedNum
    Output message_A, message_B
1:  (ds, M) ← Digital Signature(message) addBloom(M)
2:  For i = 0 to N
3:      addBloom (hashedNum,digitalSign,c,tt)
4:      i ++
5:  End For
6:  (hexaBloom) ← to Hex
7:  (H_bfv) ← HMACBFV(index,sender,hexaBloom)
8:  message_A ← (bloom + hashBFV + c + s)
9:  message_B ← (message + tt + digitalSign + index +
    sender)
```

Message Verification and Forwarding Phase

After the message is sent by the sender, the recipient will verify the incoming message before forwarding it. There will be several processes carried out by the recipient to assess the validity of the message, such as.

a. HMAC Verification

The receiving node will get message A which consists of the BFV and HMAC. The receiver node will build the HMAC from the previously sent BFV and compare it with the HMAC it received. The message will be rejected if the value of the HMAC, which is received by the recipient node, is different from the HMAC it built.

b. Message Checking

After verifying the HMAC value, the receiver node will calculate whether the message is in the bloom filter or not and compare the timestamp of the received message. The message will be rejected if the message received is not in the BFV or the timestamp difference is higher than the accepted threshold. Afterward, the received index will be compared with the latest stored index in the receiver node.

c. BFV Checking

After making sure the message is valid, the receiver node calculates the hash of the key (k) several times until reaches the current index. Afterward, the receiver node checks whether the hashed key is in the bloom filter.

d. Digital Signature Verification

In this process, the receiver node will verify the digital signature of the message using ECDSA. If the digital signature is not valid then the process will be terminated.

The detailed steps of the message generation and verification phase can be seen in Algorithm 2.

Algorithm 2: Verification phase

```
     Input message_A, message_B
     Output True (verified) or False (not verified)
1:   (timenow) ← Time ()
2:   verifH_bfv ← HMACBFV (index,sender,hexaBloom)
3:   If verifH_bfv == H_bfv
4:      If timenow > tt
5:         If c' <= c
6:         hashedNum ← checkKeyChain
           (CommKey2,c,N)
7:            While x < 5
8:         dataCheck ← (hashedNum2[x],tt,digitalSign)
9:                bloomCheck (dataCheck)
10:               x ++
11:           End
12:        End if
13:  End if
14: End if
15: vk.verify(ds,M2)
```

Results and Discussion

This section discusses the testing scenarios and analyses for each scenario implementation result.

Testing Scenario

There are two scenarios to test the proposed method's performance.

First Test Scenario

The idea of the first scenario is based on previous research (Chuchaisri and Newman, 2012). The packet, that is sent by the sender node to the receiver node, is only transmitted 1 time in each communication. This packet consists of a message, timestamp, digital signature, index, and BFV with a total size of 102 bytes. Based on the sender's point of view, there are 4 activities to encapsulate the message such as generating a timestamp, keychain, BFV, and digital signature. Based on the receiver's point of view, 2 different methods can be implemented in this scenario, such as Authentication first and Forwarding-First. Authentication first means the receiver must verify the timestamp, keychain, BFV, and digital signature first, before forwarding it to the other sensor nodes. Forward-first means the receiver only needs to verify the timestamp, keychain, and BFV before forwarding to the other sensor nodes.

Second Test Scenario

The idea of the second scenario is based on the proposed method. The message in the sender node will be split into 2 packets that have a size of 102 bytes for each packet. Based on the sender's point of view, there are 5 activities to encapsulate the message such as generating timestamp, keychain, BFV, HMAC, and digital signature. Based on the receiver's point of view, 2 different methods can be implemented. Authentication first in the second scenario means the receiver node must verify both packet *A* and packet *B* which consists of HMAC, timestamp, keychain, BFV, and the digital signature. Forwarding first in the second scenario means the receiver must verify both packet *A* and packet *B* which only consists of HMAC, timestamp, keychain, and BFV. The packet can be forwarded to the other sensor nodes before digital signature verification.

Analysis

Based on the sender's point of view, there is one additional activity in the second test scenario. However, this activity increases the integrity of the transmitted BFV using HMAC. The running time for the message construction can be seen in Fig. 8. This test was carried out 100 times to measure the time required by the sender node to encapsulate the message.

The time needed by the sender node to construct 100 messages for the first and second scenarios is 133.76 and 387.99 milliseconds, respectively. The average time needed to construct the message in the first and second scenarios is 1.366521 and 3.926965 milliseconds, respectively. It differs by about 2.56 milliseconds. The time needed for scenario 2 is higher than for scenario 1. It is because the second scenario guarantees the integrity of BFV using HMAC-SHA256 with a length of 32 bytes. Besides HMAC, Fiat Shamir is implemented to strengthen the authentication scheme and to be used as a comparison method. The running time of Fiat-Shamir authentication schemes is higher than HMAC both in scenarios 1 and 2 by about 95%. In addition, the probability of the attacker guessing the HMAC digest using brute force methods is about $\frac{1}{2^{356}}$.

Furthermore, testing is carried out to analyze the time required by the receiver node to verify the received message. Two different message verification methods were analyzed in the first scenario testing. As can be seen in Fig. 9. The time required by the receiver node to verify the 100 messages using the authentication first method is 435.86 milliseconds, while the time required by the receiver node to verify the 100 messages using the forwarding first method is 350.08 milliseconds. The average time verification between the authentication first and forwarding first are 4.273 and 3.407 milliseconds, respectively. The difference between both methods is about 0.86 milliseconds for each message transmission.

As can be seen in Fig. 10, the time required by the receiver node to verify the received message until it can be sent to the other node using the proposed method is 337.06 milliseconds, whereas the time required by the receiver node using the authentication first method is 598.35 milliseconds. The average time needed for the receiver node to verify the packet using authentication first and forwarding first is 5.9084 and 3.1921 milliseconds, respectively. The difference between both is about 2.716 milliseconds for each message transmission.

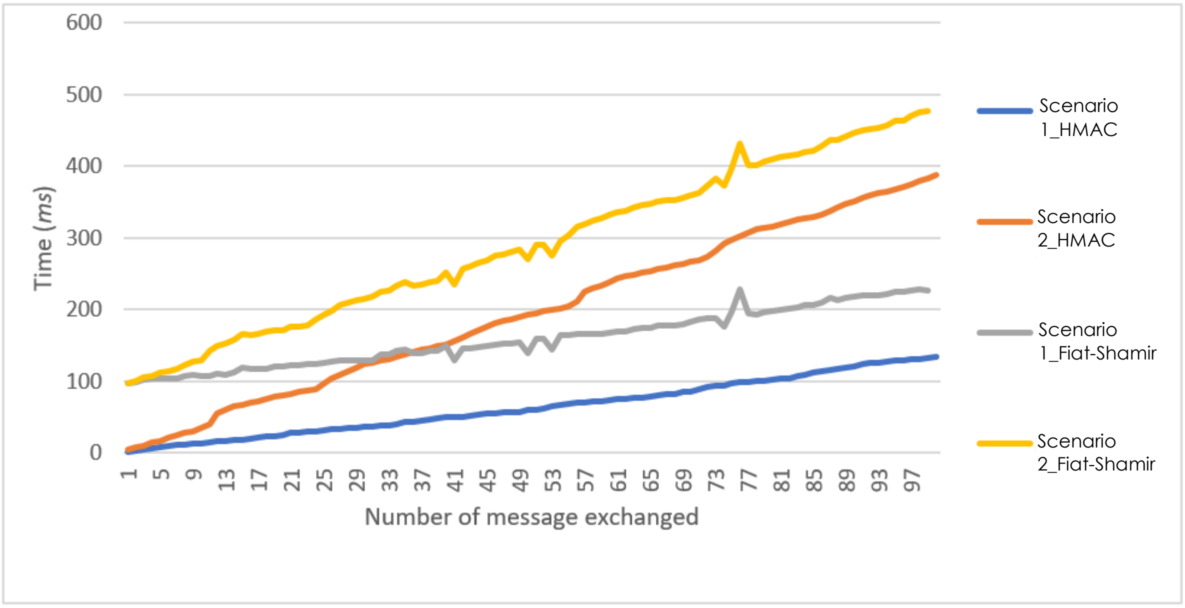

Fig. 8: Time elapsed to construct a message

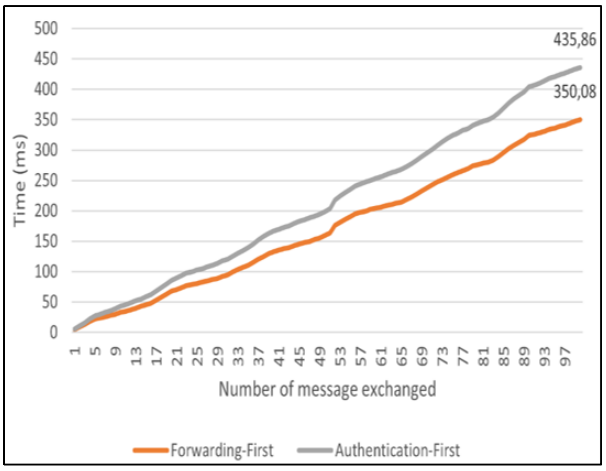

Fig. 9: Time comparison in the first test scenario

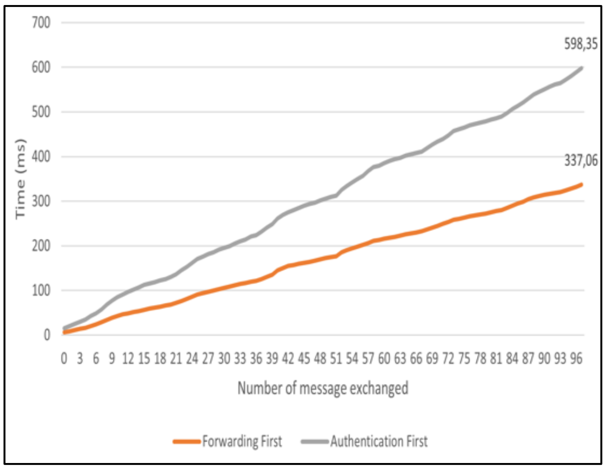

Fig. 10: Time comparison in the second test scenario

Based on the test results illustrated in Fig. 9 and 10. In the first and second scenarios, the average time required by the receiver node to forward messages using the authentication first method is 4.273 and 5.9084 milliseconds, respectively. The time needed for the second scenario is longer than the time required to forward messages using the first scenario. It differs by about 1.634 milliseconds for each message transmission. Afterward, the average time required by the receiver node to forward messages using the forwarding first method is 3.407 and 3.19 milliseconds, respectively. It differs by about 0.21 milliseconds for each message transmission. Based on those results, the running time of the authentication first method in the second scenario is higher than in the first scenario. It is because the additional communication overhead and HMAC computation affect the whole process. However, the running time of the forwarding first in the first method is higher than in the second scenario. The difference is only 0.21 milliseconds. It shows that additional communication overhead and HMAC computation only slightly affects the whole process. There is an increase in the verification time required by the receiver node is still acceptable with the integrity guarantee of the BFV transmission.

Conclusion

Forwarding first communication is proposed to reduce the delay between the receiver to the other sensor nodes. However, its implementation still lacks security. Sensor nodes as resource-constrained devices have limitations to perform high-end security. Therefore, modification of the forwarding key chain and bloom filter is proposed. That mechanism has low computation and less storage requirement. The experiment result shows that the time needed to construct a packet on the sender side increased by about 2.56 milliseconds. The communication overhead is increased to give space for HMAC-SHA256 to guarantee BFV's integrity. However, the verification time increased by only 0.21 milliseconds. It is still acceptable and the payload for messages is increased. In future work, the other security aspect such as confidentiality and authentication will be used as another baseline.

Acknowledgment

This study was supported by Telkom University, intelligent systems research group, and computing laboratory.

Funding Information

This research was funded by Telkom University, Indonesia.

Author's Contributions

Muhammad Taqqyyudin Ilham Ramadhan: Conceptualization, methodology, software, validation, resources, data curation, written original draft preparation, visualization.

Farah Afianti: Conceptualization, methodology, formal analysis, investigation, resources, written review, edited and draft finalization, supervision.

Bambang Ari Wahyudi: Validation, investigation, supervision.

Prasti Eko Yunanto: Software, validation, supervision.

Dedy Rahman Wijaya: Validation, written review, edited and draft finalization, supervision.

Ethics

This manuscript is original and contains unpublished material that has not been previously published as a whole or a substantial part elsewhere. The corresponding author confirms that the other authors have read and approved the manuscript and no ethical issues are involved.

References

Abdullah, S., Bertalan, S., Masar, S., Coskun, A., & Kale, I. (2017, July). A wireless sensor network for early forest fire detection and monitoring as a deciding factor in the context of a complex integrated emergency response system. In *2017 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)* (pp. 1-5). IEEE. https://doi.org/10.1109/EESMS.2017.8052688

Afianti, F., & Suryani, T. (2018). Dynamic cipher puzzle for efficient broadcast authentication in wireless sensor networks. *Sensors*, *18*(11), 4021. https://doi.org/10.3390/s18114021.

Afianti, F., Asrini, C., & Wijaya, D. R. (2021). "Scalable Two-Dimensional Bloom Filter Membership Scheme on Massive Scale Wireless Sensor Networks. *International Journal of Intelligent Engineering and Systems*, 14(1), 474-481. https://doi.org/10.22266/IJIES2021.0228.44

Aguirre, E., Lopez-Iturri, P., Azpilicueta, L., Redondo, A., Astrain, J. J., Villadangos, J., ... & Falcone, F. (2016). Design and implementation of context-aware applications with wireless sensor network support in urban train transportation environments. *IEEE Sensors Journal*, *17*(1), 169-178. https://doi.org/10.1109/JSEN.2016.2624739.

Chuchaisri, P., & Newman, R. (2012). Fast response PKC-based broadcast authentication in wireless sensor networks. *Mobile Networks and Applications*, *17*(4), 508-525. https://doi.org/10.1007/s11036-011-0349-8.

Dargie, W., & Poellabauer, C. (2010). *Fundamentals of wireless sensor networks: Theory and practice*. John Wiley & Sons.

Dong, Q., Liu, D., & Ning, P. (2013). Providing DoS resistance for signature-based broadcast authentication in sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)*, *12*(3), 1-26. https://doi.org/ 10.1145/2442116.2442123.

Du, X., Guizani, M., Xiao, Y., & Chen, H. H. (2008, May). Defending DoS attacks on broadcast authentication in wireless sensor networks. In *2008 IEEE International Conference on Communications* (pp: 1653-1657). IEEE. https://doi.org/10.1109/ICC.2008.319

Hu, Y., Xiong, Y., Huang, W., & Bao, X. (2018, August). KeyChain: Blockchain-based key distribution. In *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)* (pp: 126-131). IEEE. https://doi.org/ 10.1109/BIGCOM.2018.00027

Huang, J. Y., Liao, I. E., & Tang, H. W. (2011). A forward authentication key management scheme for heterogeneous sensor networks. *EURASIP Journal on Wireless Communications and Networking, 2011*, 1-10. https://doi.org/10.1155/2011/296704

Husni, E., Leksono, B., & Rosa, M. R. (2015, September). Digital signature for contract signing in service commerce. In *2015 International Conference on Technology, Informatics, Management, Engineering & Environment (TIME-E)* (pp. 111-116). IEEE. https://doi.org/10.1109/TIME-E.2015.7389757

Hyun, S., Ning, P., Liu, A., & Du, W. (2008, April). Seluge: Secure and dos-resistant code dissemination in wireless sensor networks. In *2008 International Conference on Information Processing in Sensor Networks (ipsn 2008)* (pp: 445-456). IEEE. https://doi.org/10.1109/IPSN.2008.12

Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of wireless sensor networks: An up-to-date survey. *Applied System Innovation*, *3*(1), 14. https://doi.org/10.3390/asi3010014

Lalar, S., Bhushan, S., & Surender. (2019). An efficient tree-based clone detection scheme in a wireless sensor network. *Journal of Information and Optimization Sciences*, *40*(5), 1003-1023. https://doi.org/10.1080/02522667.2019.1637998

Mbarek, B., Sahli, N., & Jabeur, N. (2018, June). BFAN: A Bloom Filter-Based Authentication in Wireless Sensor Networks. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)* (pp: 304-309). IEEE. https://doi.org/10.1109/IWCMC.2018.8450292

Ning, P., Liu, A., & Du, W. (2008). Mitigating DoS attacks against broadcast authentication in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, *4*(1), 1-35. https://doi.org/10.1145/1325651.1325652.

Shuai, C., Yang, H., Ouyang, X., & Gong, Z. (2018). A Bloom Filter for High-Dimensional Vectors. *Information*, *9*(7), 159. https://doi.org/ 10.3390/info9070159

Stallings, W., & Tahiliani, M. P. (2014). Cryptography and network security: Principles and practice, vol. 6. *Editor: Pearson London*.

Thungon, L. C., & Hussain, M. (2021). Performance Evaluation of Zero Knowledge and HMAC-Based Authentication in Fog-Based Internet of Things. In *Proceedings of the International Conference on Computing and Communication Systems* (pp. 633-641). Springer, Singapore. https://doi.org/10.1007/978-981-33-4084-8_61

Xu, Y., & Liu, F. (2017, July). Application of wireless sensor network in water quality monitoring. In *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)* (Vol. 2, pp. 368-371). IEEE. https://doi.org/10.1109/CSE-EUC.2017.254

Yao, X., Liao, W., Du, X., Cheng, X., & Guizani, M. (2019). Using bloom filter to generate a physiological signal-based key for wireless body area networks. *IEEE Internet of Things Journal*, *6*(6), 10396-10407. https://doi.org/ 10.1109/JIOT.2019.2939144.

Yu, J. Y., Lee, E., Oh, S. R., Seo, Y. D., & Kim, Y. G. (2020). A survey on security requirements for WSNs: Focusing on the characteristics related to security. *IEEE Access*, *8*, 45304-45324. https://doi.org/10.1109/ACCESS.2020.2977778

Zhao, W., Liu, J., Guo, H., & Hara, T. (2018). ETC-IoT: Edge-node-assisted transmitting for the cloud-centric internet of things. *IEEE Network*, *32*(3), 101-107. https://doi.org/10.1109/MNET.2018.1700164