

DYNAMIC KEY EXCHANGE PROTOCOL FOR SECURITY IN WIRELESS NETWORKS

*Dissertation report submitted in partial fulfilment of the requirement for the
degree*

BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING

By

NIMIT BHARDWAJ (131082)

AMAN GUPTA (131097)

SUDHANSHU SINGH (131098)

UNDER THE GUIDANCE OF
Dr. SHWETA PANDIT



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

TABLE OF CONTENTS

	Page Numbers
LIST OF FIGURES	iii
DECLARATION BY THE SCHOLAR	iv
SUPERVISOR'S CERTIFICATE	v
ACKNOWLEDGEMENT	vi
ABSTRACT	vii
CHAPTER-1	
INTRODUCTION	02
1.1 SYMMETRIC KEY ENCRYPTION	02
1.2 ASYMMETRIC KEY ENCRYPTION	02
CHAPTER -2	
CRYPTOGRAPHIC HASH FUNCTIONS	05
2.1 MD4	07
2.1.1 MD4 HASHES	07
2.2 MD5	08
2.2.1. MD5 HASHES	08
2.3 SHA-1	09
2.4 SHA-3	10
CHAPTER-3	
WIRELESS SENSOR NETWORKS	12

3.1	CHARACTERSTICS	12
3.2	COMMUNICATION ARCHITECTURE OF WSN	13
3.3	CONSTRAINTS IN WSN	15
3.4	POTENTIAL APPLICATION	16
3.5	SECURITY ISSUES IN WSN	17
3.6	FUTURE WORK IN WSN	17

CHAPTER-4

MOTIVATION AND RELATED WORK	20
------------------------------------	-----------

CHAPTER-5

PROPOSED LIGHT WEIGHT CRYPTOGRAPHIC HASH ALGORITHM FOR WSN	22
---	-----------

5.1	REQUIREMENTS OF AN IDEAL HASH FUNCTION	22
5.2	PROPOSED HASH FUNCTION	22
5.3	PERFORMANCE ANALYSIS	26
5.4	RESULTS AND OUTPUTS	28

CHAPTER-6

CONCLUSION AND FUTURE ASPECTS	32
--------------------------------------	-----------

REFERENCES

LIST OF FIGURES

FIGURE NUMBER	CAPTION	PAGE NUMBER
1.1	Symmetric Key Encryption	02
1.2	Asymmetric Key Encryption	03
2.1	Signature of Long Message with a Hash Function	05
2.2	Checking Integrity at Bob's End	06
3.1	Wireless Sensor	14
5.1	96-bit hash digest for a input message	28
5.2	Plot for hash digest of a input message	28
5.3	96-bit hash digest for a input message	29
5.4	Plot for hash digest of a input message	29
5.5	Comparison of the two hash digests	30

DECLARATION BY THE SCHOLAR

I hereby declare that the work reported in the B-Tech project entitled “**Dynamic Key Exchange protocol for Security in Wireless Networks using Hybrid Hash Functions**” submitted at Jaypee University of Information Technology, Waknaghat India, is an authentic record of my work carried out under the supervision of Dr. Shweta Pandit. I have not submitted this work elsewhere for any other degree or diploma.

NIMIT BHARDWAJ (131082)

AMAN GUPTA (131097)

SUDHANSHU SINGH (131098)

Department of Electronics and Communication

Jaypee University of Information Technology, Waknaghat, India

Date:

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the B-Tech. project entitled “**Dynamic Key Exchange protocol for Security in Wireless Networks using Hybrid Hash Functions**”, submitted by Nimit Bhardwaj (131082), Aman Gupta (131097) and Sudhanshu Singh (131098) at Jaypee University of Information Technology, Waknaghat, India, is a bonafide record of his / her original work carried out under my supervision. This work has not been submitted elsewhere for any other degree or diploma.

(Dr. SHWETA PANDIT)

Assistant Professor

Department of Electronics and Communication

Jaypee University of Information Technology, Waknaghat, India

Date:

ACKNOWLEDGEMENT

I would like to express my gratitude to my supervisor Dr. Shweta Pandit and co-supervisor Dr. Pradeep Chauhan of Electronics and Communications Department of Jaypee University, Waknaghat for the useful comments, remarks and engagement through the learning process of this bachelor project. Dr. Shweta Pandit consistently allowed this project to be my own work, but steered me in the right direction whenever she thought I needed it.

Finally, I must express my very profound gratitude to my family for providing me with unfailing support and continuous encouragement throughout my study and through the process of researching and writing this project report. This accomplishment would not have been possible without them. Thank you.

ABSTRACT

In today`s world of advanced wireless communication, authentication of a message is a huge research challenge. An appropriate solution of this problem is Cryptographic hash function. Some of the most popular Cryptographic hash functions are MD5 and SHA1.

However, these commonly used Cryptographic hash algorithms require huge computational energy which is not affordable by energy starved networks e.g. Wireless Sensor Networks(WSN). In these application of wireless communication, major constraints like communication and computation overhead require high energy which is generally not affordable. So, keeping this fact in mind in this project work a light weight, one way hash algorithm is designed producing a small length hash digest especially for these energy starved networks. The main aim of the work is to make the algorithm light weight so that when networks like WSN uses this then each node can successfully run the algorithm with low energy. This proposed algorithm is developed using MATLAB simulation and the results were compared with conventional algorithms.

CHAPTER 1

INTRODUCTION

Symmetric key & Asymmetric key cryptography are two different techniques available to use keys or secrets for encryption. Both types of algorithms are used for data encryption and decryption in cryptography and network security.

1.1 SYMMETRIC KEY ENCRYPTION

Symmetric key and Asymmetric key cryptography are two unique procedures accessible to utilize keys or insider facts for encryption. Both sorts of calculations are utilized for information encryption and unscrambling in cryptography and system security. Symmetric key cryptography is a traditional encryption strategy which is utilized to encode and unscramble the information. The procedure used to create figure message in Symmetric key calculations are less convoluted because of which these calculations execute substantially quicker than Asymmetric key calculations. The quantity of bits (i.e. length) used to characterize the key decides the quality of the security. A key can be 160-512 bits in length. The significant test in executing Symmetric key cryptography is that 2 parties must share the mystery securely.

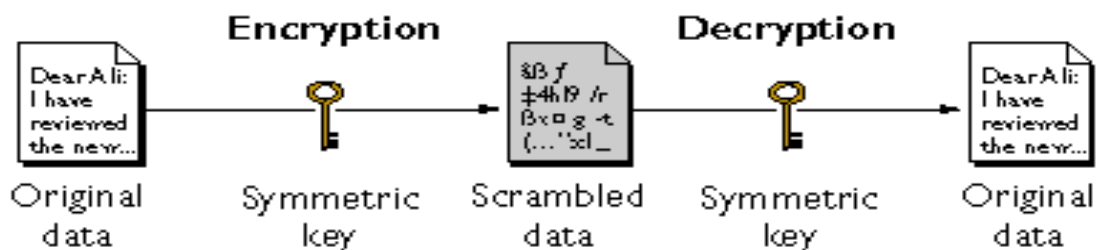


Figure 1.1 Symmetric Key Encryption

1.2 ASYMMETRIC KEY ENCRYPTION

Asymmetric key cryptography utilizes a couple of scientifically related keys. The key pair consist of a public key and a private key. Public key is known to everybody and the private key is dependably possessing its proprietor as it were. The working instrument of Asymmetric key cryptography goes for broke required in key sharing between 2 parties. The

private key is never uncovered in this procedure. The message is encoded by applying general society key before sending. The scrambled message can be decoded by utilizing the comparing private key. In another utilization of Asymmetric key cryptography, a message encoded with the private key is decoded by the relating public key. It is essentially unrealistic to compute the private key regardless of the possibility that the comparing public key is known.

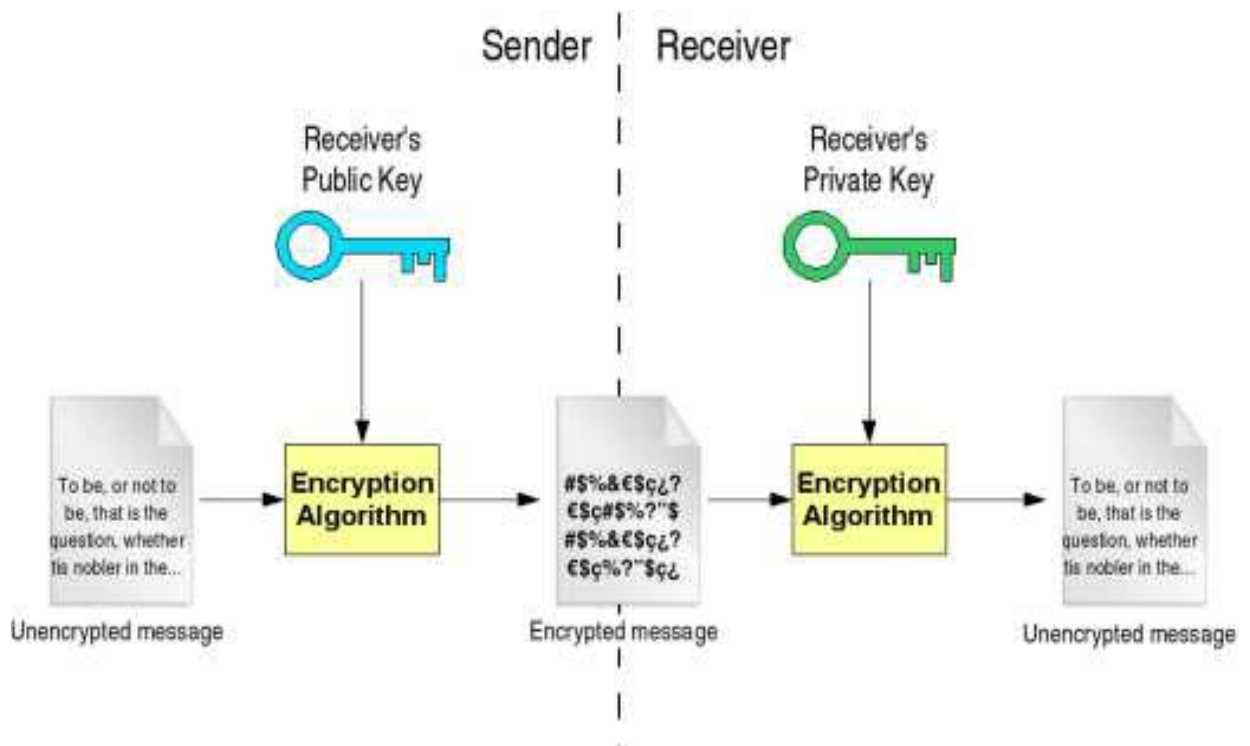


Figure 1.2 Asymmetric Key Encryption

CHAPTER 2

CRYPTOGRAPHIC HASH FUNCTIONS

An alternate arrangement of problems is still there -

At the point when a message is received by Bob sent from Alice, 2 things are to be checked

- Is the message original (Uprightness of the message has not been traded off)?
- Has the message come from Alice herself?

The solution to these problems is the Cryptographic Hash Functions.

A hash (also known as message digest or signature) is a one-way function that by a few means registers a unique fingerprint of the message. It is more commonly known as the hash value of the message.

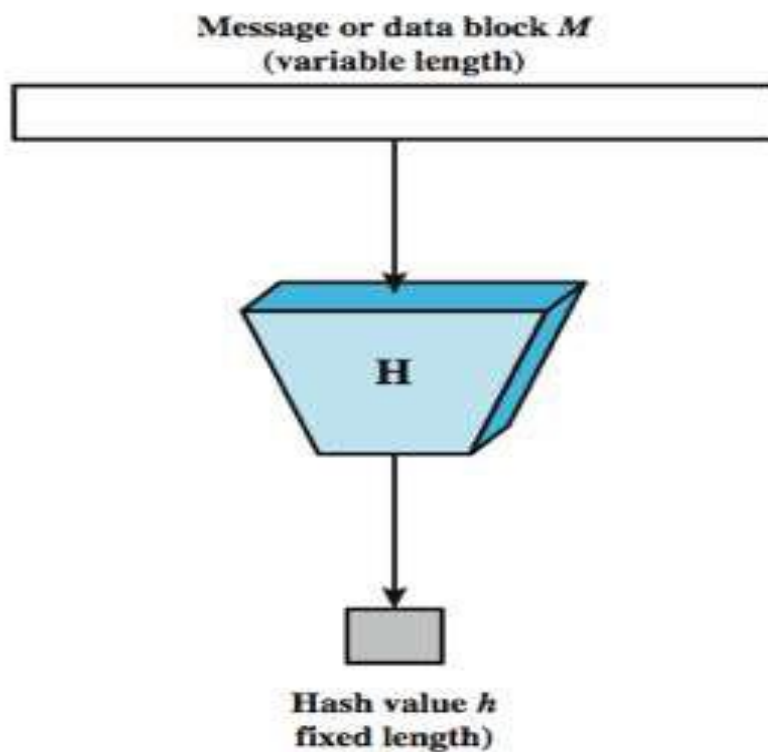


Fig 2.1 Signature of Long Message with a Hash Function

So, if Alice wants to guarantee the integrity of her message, she can attach the fingerprint of the message at the end of the document. Bob knows the scheme used by Alice to produce the hash at his end. If it matches with the hash value from Alice, Bob knows the message is original.

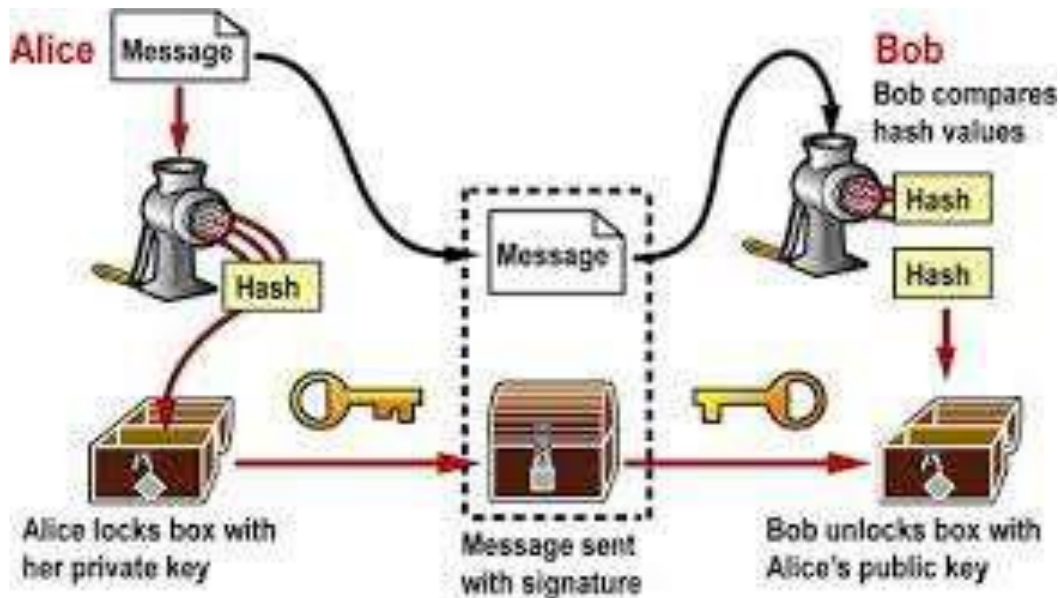


Fig 2.2 Checking Integrity at Bob's End

A good way of producing a hash function is to amalgamate lots of vicious operations into a good digest function, and then play with it. If specific patterns are identified over and over in the yield, the function perhaps requires a change or it is permanently rejected.

Ideally, a good hash function should be very easy to produce. However, there is no minimal function which is fully secure. It is better for a hash function to be overload and do a lot of shuffling, even more than what is needed. The function should use all the input data. The hash function must uniformly arrange the hash values across the complete set of available values. The hash digest is to be calculated in many rounds. The programmers find the minimum number of rounds required to produce a hash output which satisfies different randomness tests, and then do a few more tests just to make it more and more robust and safe. Some of the popularly known hash functions are MD4, MD5, SHA-1, and SHA-512.

2.1 MD4

The **MD4**, also known as Message Digest - 4 is a cryptographic hash function made by Ronald Rivest in the year 1990. The length of hash digest is 128 bits. This algorithm influenced later designs, like the MD5, SHA-1 and RIPEMD hash algorithms. The acronym "MD" stands for "Message Digest."

The security of MD4 algorithm has been severely broken. The first full collision attack on MD4 was published in 1995 and since then several newer attacks have been published. As of 2007, an attack can produce collisions easily in less than 2 MD4 hash operations. A theoretical preimage attack is also in existence.

Weaknesses and problems in MD4 were shown by Den Boer and Bosselaers in a research paper published in 1991. The first full MD4 collision attack was done by Hans Dobbertin in 1995, which took him only seconds to carry out even at that time. In August of 2004, Wang et al. demonstrated a highly efficient collision attack, along with attacks on the later hash function designs in the MD4/MD5/SHA-1/RIPEMD algorithm family. This result was however improved later by Sasaki et al., and producing a collision is now very cheap, as cheap as verifying it (a few micro-seconds).

The pre-image resistance of MD4 was successfully broken by Gaëtan Leurent in 2008, with a 2 attack. In 2011, RFC 6150 said that RFC 1320 (MD4) is **historic** (obsolete).

2.1.1 MD4 Hashes

The 128-bit (16-byte) MD4 hashes (also termed message digests) are typically represented as 32-digit hexadecimal numbers. The following demonstrates a 43-byte ASCII input and the corresponding MD4 hash:

```
MD4("The quick brown fox jumps over the lazy dog")  
= 1bee69a46ba811185c194762abaeae90
```

Even a small change in the message will (with overwhelming probability) result in a completely different hash, e.g. changing d to c:

```
MD4("The quick brown fox jumps over the lazy cog")  
= b86e130ce7028da59e672d56ad0113df
```

The hash of the zero-length string is:

```
MD4("") = 31d6cfe0d16ae931b73c59d7e0c089c0
```

2.1.2 MD4 collision example

Let:

```
k1=
839c7a4d7a92cb5678a5d5b9eea5a7573c8a74deb366c3dc20a083b69f5d2a3bb3719dc6989
1e9f95e809fd7e8b23ba6318edd45e51fe39708bf9427e9c3e8b9
k2=
839c7a4d7a92cbd678a5d529eea5a7573c8a74deb366c3dc20a083b69f5d2a3bb3719dc6989
1e9f95e809fd7e8b23ba6318edc45e51fe39708bf9427e9c3e8b9
```

$k1 \neq k2$, but $MD4(k1) = MD4(k2) = 4d7e6a1defa93d2dde05b45d864c429b$

Note that two hex-digits of $k1$ and $k2$ define one byte of the input string, whose length is 64 bytes.

2.2 MD5

The **MD5 algorithm** is a commonly used hash function generating a 128-bit hash digest value. While MD5 was designed to be used as a cryptographic hash function in the beginning, it was being found to suffer from much problematic vulnerability. It is still used as a checksum and also to verify data integrity, but against unintentional corruption only.

Like many hash algorithms, MD5 uses neither encryption nor it uses encoding. It can be broken down by brute-force approach and it also suffers from many other vulnerabilities.

MD5(Message Digest 5) was developed by Ronald Rivest in 1991 to replace its predecessor, hash function MD4. The source code published in RFC 1321 contains a "by attribution" RSA license. We know that, abbreviation "MD" stands for "Message Digest."

The security of the MD5 has also been compromised like the MD4, with its short comings having been exploited in the field extensively, by the Flame malware most infamously in 2012. The CMU Software Engineering Institute considers the MD5 algorithm "cryptographically broken and unsuitable for further use".

2.2.1 MD5 hashes

The 128-bit (16-byte) MD5 hashes (also termed *message digests*) are typically represented as a sequence of 32 hexadecimal digits. The following demonstrates a 43-byte ASCII input and the corresponding MD5 hash:


```
MD5("The quick brown fox jumps over the lazy dog.") =  
9e107d9d372bb6826bd81d3542a419d6
```

Each little change in the message will (with overpowering likelihood) result in a for the most part different hash, because of the torrential slide impact. For example, adding a period to the end of the sentence:

```
MD5("The quick brown fox jumps over the lazy dog") =  
e4d909c290d0fb1ca068ffaddf22cbd0
```

The hash of the zero-length string is:

```
MD5("") =  
d41d8cd98f00b204e9800998ecf8427e
```

The MD5 algorithm is mainly made for information consisting of any random number of bits; it is not limited to multiples of eight bit (octets, bytes). Some MD5 implementations such as `md5sum` might be limited to octets, or they might not support *streaming* for messages of an initially undetermined length.

2.3 SHA 1

SHA-1, also known as **Secure Hash Algorithm 1** is a cryptographic hash function developed by the NSA and is published by the United States of America NIST. SHA-1 generates a 160-bit hash digest value also known as a message digest. A SHA-1 hash digest value is rendered as a hexa-decimal number, which is 40 digits long.

However, SHA-1 is no longer called secure against its well-funded opponents. In 2005, cryptanalysts found various attacks on SHA-1 and they suggested that the algorithm is now not secure enough for further use, and therefore, since 2010 many companies have considered its replacement by SHA-2 or SHA-3. Microsoft, Google, Apple and Mozilla, all have said that their browsers will not accept SHA-1 SSL certificates in 2017.

On February 23, 2017 CWI Amsterdam and Google announced they successfully performed a collision attack against SHA-1 algorithm. They published two dissimilar PDF files which generated the same SHA-1 hash as proof of their concept.

2.4 SHA 3

SHA-3, also known as the **Secure Hash Algorithm 3** is a subset of the cryptographic primitive family **Keccak** and is a cryptographic hash function developed by Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche, building upon RadioGatún. The hash function SHA-3 is a member of the Secure Hash Algorithm family. The SHA-3 standard was actually released by NIST on August 5, 2015.

CHAPTER 3

WIRELESS SENSOR NETWORKS

Wireless sensor systems (WSN), in some cases called **wireless sensor and actuator networks** (WSAN), are spatially circulated independent sensors to screen physical or natural conditions, for example, temperature, sound, weight, and so forth and to agreeably go their information through the system to a fundamental area. The more present day systems are bi-directional, additionally empowering control of sensor action. The advancement of wireless sensor systems was persuaded by military applications, for example, war zone reconnaissance; today such systems are utilized as a part of numerous mechanical and customer applications, for example, modern process observing and control, machine wellbeing checking, and so on.

The WSN is worked of "nodes"- from a couple to a few hundreds or even thousands, where every node is associated with one (or once in a while a few) sensors. Each such sensor network hub has normally a few sections: a radio handset with an inward reception apparatus or association with an outer receiving wire, a microcontroller, an electronic circuit for interfacing with the sensors and a vitality source, ordinarily a battery or an implanted type of vitality collecting. A sensor node may shift in size from that of a shoebox down to the extent of a grain of tidy, albeit working "bits" of real tiny measurements still can't seem to be made. The cost of sensor hubs is likewise factor, running from a couple to several dollars, depending upon the multifaceted nature of the individual sensor hubs. Size and cost requirements on sensor hubs bring about comparing limitations on assets, for example, vitality, memory, computational speed and correspondences transmission capacity. The topology of the WSNs can shift from a basic star system to a progressed multi-hop wireless work organize. The proliferation procedure between the jumps of the system can defeat or flooding.

.

3.1 CHARACTERSTICS

The fundamental attributes of a WSN include:

- Power utilization imperatives for hubs utilizing batteries or vitality reaping
- Ability to adapt to hub disappointments (strength)
- Some versatility of hubs (for very portable hubs see MWSNs)
- Heterogeneity of hubs
- Scalability to vast size of sending
- Ability to withstand unforgiving natural conditions
- Ease of utilization
- Cross-layer outline

Cross-layer is turning into a vital considering zone for remote interchanges. What's more, the conventional layered approach presents three principle issues:

1. Conventional layered approach can't share diverse data among various layers , which prompts each layer not having complete data. The customary layered approach can't ensure the advancement of the whole system.
2. The conventional layered approach does not be able to adjust to the environmental change.
3. In view of the impedance between the distinctive clients, get to clashes, blurring, and the change of condition in the remote sensor systems, customary layered approach for wired systems is not appropriate to remote systems.

In this way, the cross-layer can be utilized to make the ideal modulation to enhance the transmission execution, for example, information rate, vitality proficiency, QoS (Quality of Service), and so on. Sensor hubs can be envisioned as little PCs which are to a great degree fundamental as far as their interfaces and their parts. They for the most part comprise of a processing unit with constrained computational power and restricted memory, sensors or MEMS (counting particular moulding hardware), a specialized gadget (typically radio handsets or on the other hand optical), and a power source for the most part as a battery. Other conceivable incorporations are vitality reaping modules, auxiliary ASICs, and perhaps optional correspondence interface (e.g. RS-232 or USB).

The base stations are at least one segments of the WSN with significantly more computational, vitality and correspondence assets. They behave as a passage between sensor hubs and the end client as they regularly forward information from the WSN on to a server. Other unique parts in steering based systems are switches, intended to compute, ascertain and appropriate the directing tables.

3.2 COMMUNICATION ARCHITECTURE OF WSN

A Wireless Sensor Network is generally made out of couple of hundred to a few large number of small sensor nodes. Such networks are utilized to screen environmental conditions. These sensor hubs are thickly conveyed to make a communication network in a sensor field. A sensor node comprises of 4 essential parts: a sensing unit, a processing unit, a power unit and a handset unit. In some cases, it might have a location tracking framework

which monitors the particular location. It might likewise have power generator which give longer power reinforcement. In addition to these, a hub may likewise have mobilizer. Sensing unit consist of two subunits: Sensors and analog-to-digital converters (ADCs). A processing unit is for the most part comprising of microcontroller or chip and a little storage unit.

Its principle job is to prepare the assembled information and execute the correspondence protocol for communication. The power unit of a sensor is generally limited (e.g., a single battery). It is sometimes bolstered by power rummaging gadgets (e.g., solar cells). For large networks, battery substitution is extremely troublesome or even impossible. A transceiver unit gives an association of the hub to the system. A large portion of the sensing assignments and sensor network directing procedures require information of area, which is given by a location tracking framework. On the basis of the application, a mobilizer is at times used to give movement to the sensor hub.

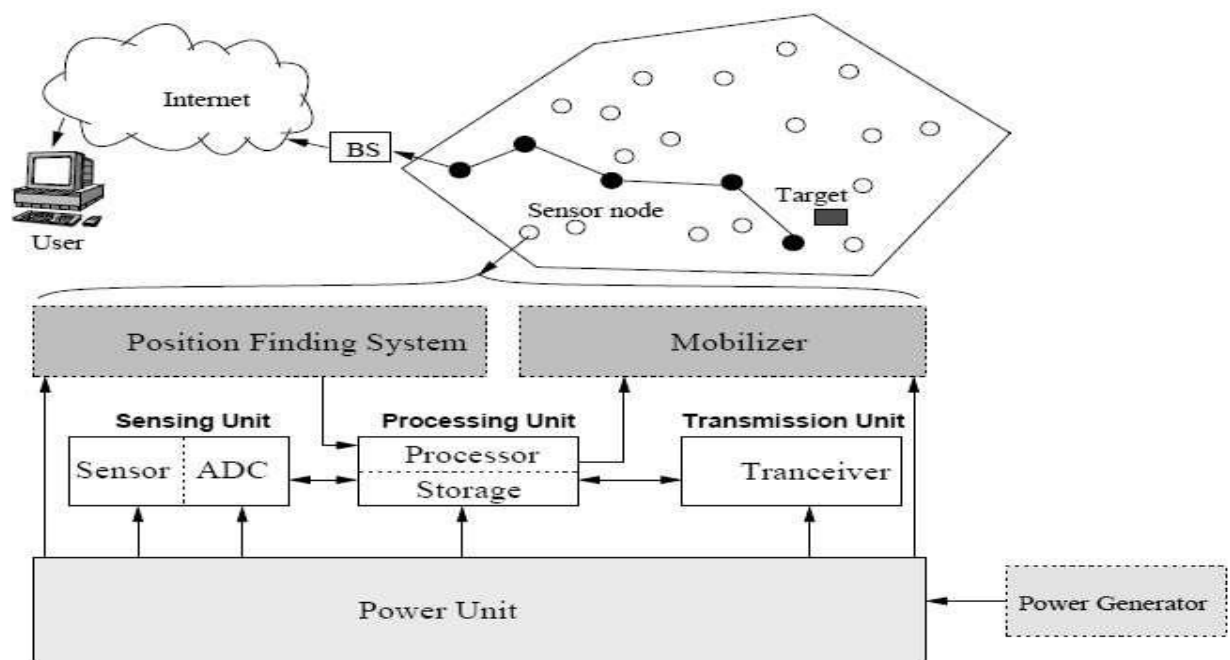


Fig 3.1 Wireless Sensor

The protocol stack used in sensor nodes contains the following layers

- **Physical layer:** responsible for transmission and reception of data, generation and selection of carrier frequency, signal deflection, modulation, and data encryption & decryption.
- **Data link layer:** responsible for error detection and correction, the multiplexing of data streams, detection of data frames, medium access, reliable point-to-point and point-to-multipoint connections.
- **Network layer:** responsible for assignment addresses and specify the process of packet forwarding to other nodes.
- **Transport layer:** responsible for the reliable transport of packets from one node to other node.
- **Application layer:** responsible for the interactions with the end users. It specifies how data provided to individual sensor nodes upon request.

3.3 CONSTRAINTS IN WSN

Energy consumption is the most important factor to determine the life of a sensor network because usually sensor nodes are driven by battery. Sometimes energy optimization is more complicated in sensor networks because it involved not only reduction of energy consumption but also prolonging the life of the network as much as possible. The optimization can be done by having energy awareness in every aspect of design and operation. This ensures that energy awareness is also incorporated into groups of communicating sensor nodes and the entire network and not only in the individual nodes. Nodes of a Wireless Sensor Network are resource constrained as they have restricted computing capability, communication bandwidth and storage capacity. The small size of sensor node and limited computing power are 2 greatest constraints. So the security services in a sensor node must be implemented keeping in view of following hardware constraints of the sensor node:

- **Energy:**
 - Energy is required for the sensor transducer which converts one form of energy into another

- Energy is required to establish communication among sensor nodes
- Energy is also required for microcontroller & microprocessor computation

- **Computation:**

Conventional complex cryptographic algorithms would require a lot of computing power for which the processors of sensor nodes is not feasible. Lightweight cryptographic algorithms are required to reduce the computing burden on sensor node

.

- **Memory:**

Flash memory and RAM are usually included for the storage purpose in a sensor node. Flash memory is used to keep downloaded application code and RAM is used to keep sensor data, storing application programs, and intermediate computations. Generally, after loading OS and application code there is not sufficient space left to run complex algorithms. Due to the memory constraint it is not viable to use the majority of traditional cryptographic algorithms.

- **Transmission range:**

Limited operating power imposes restrictions on the communication range of sensor nodes. The actual transmission range of a signal depends on various environmental factors such as weather and terrain.

3.4 POTENTIAL APPLICATION

Engineers have made WSN applications for regions including social insurance, utilities, and remote observing. In health care, remote gadgets make less intrusive patient observing and human services conceivable. For utilities, for example, the power matrix, streetlights, and water municipals, remote sensors offer a lower-cost strategy for gathering framework wellbeing information to diminish vitality use and better oversee assets. Remote observing spreads an extensive variety of utilizations where remote frameworks can supplement wired frameworks by lessening wiring costs and permitting new sorts of estimation applications. Remote monitoring applications consist of:

- Structural observing for structures and extensions

- Industrial machine observing
- Process observing
- Asset following
- Environmental observing of air, water, and soil

3.5 SECURITY ISSUES IN WSN

Security issues in sensor systems rely on upon the need to realize what we will ensure. The creators characterized four security objectives in sensor systems which are Confidentiality, Integrity, Authentication and Availability. Privacy is the capacity to hide message from a uninvolved assailant, where the message conveyed on sensor systems stay secret. Honesty alludes to the capacity to affirm the message has not been altered, modified or changed while it was on the system. Validation Need to know whether the messages are from the hub it cases to be from, deciding the unwavering quality of message's inception. Accessibility is to decide whether a hub can utilize the assets and the system is accessible for the messages to proceed onward. Freshness suggests that collector gets the current and crisp information and guarantees that no foe can replay the old information. This prerequisite is particularly essential when the WSN hubs utilize shared-keys for message correspondence, where a potential foe can dispatch a replay assault utilizing the old key as the new key is being revived and spread to every one of the hubs in the WSN. To accomplish the freshness the system like nonce or time stamp ought to add to every information bundle.

3.6 FUTURE WORK IN WSN

The advances of wireless networking and sensor innovation open up a fascinating chance to oversee human exercises in a keen home condition. Genuine exercises are regularly more intricate than the contextual investigations for both single and multi-client. Examining such complex cases can be exceptionally testing while we consider both single-and multi-client exercises in the meantime. Future work will concentrate on the basic issue of perceiving exercises of numerous clients utilizing a remote body sensor organize. Wireless Sensor Networks hold the guarantee of conveying a brilliant correspondence worldview which empowers setting up a clever system fit for dealing with applications that advance from client necessities. We trust that in near future, WSN research will put an incredible effect on our day by day life. For instance, it will make a framework for ceaseless observation of

physiological signs while the patients are at their homes. It will bring down the cost required with checking patients and increase the efficient misuse of physiological information and the patients will have entry to the most astounding quality medicinal care in their own homes. In this way, it will keep away from the misery and disturbance brought on by a protracted inpatient stay.

CHAPTER 4

MOTIVATION AND RELATED WORK

The most generally utilized hash functions are one-way functions for which finding an input which hashes to a pre-indicated hash-esteem is exceptionally troublesome. Two usually utilized hash functions are MD5 and SHA-1. Both MD5 and SHA-1 are gotten from MD4 in which shortcomings have been distinguished. MD5 utilizes a hash algorithm with 128-piece long yield hash was outlined in 1991 and in 2005 it was demonstrated how quickly arbitrary collisions for MD5 can be figured. MD5 is in like manner not suited or applications like SSL certificates or digital signatures. Previously authors have uncovered that how two or three X.509 certificates can deliver that outcome in the same MD5 hash digest. This disclosure led the cryptographers suggesting the utilization of different algorithms like SHA-1 and other hash algorithms of SHA family. SHA-1 has been observed to be powerless too and generally U.S. government applications now utilize the SHA-2 and SHA-3 group of hash functions. However, the greater part of the discussed hash functions are utilized as a part of extensive conventional networks. In spite of conventional networks, in a brief and energy constrained networks like Wireless Sensor Networks, various sensor nodes are sent in outside condition without human intervention. Unwavering quality and data authenticity turns into the primary stress to manage such kind of networks. WSN experiences number of limitations like low processing power, low battery life, and little memory. Due to these imperatives, it is not ready to manage ordinary cryptographic algorithms. Along these lines, it winds up plainly mandatory to outline a lightweight security hash function for WSNs. Numerous comparative works are accounted for towards hash-based security arrangements and some of them are said here. Here endorse answers for WSNs though is not for the most part implied for WSNs.

In previous algorithms authors have exhibited a hash-based signature strategy which can be utilized to check the messages for unicast and broadcast communication. It has guaranteed that both the signature generation and verification are faster than the other existing plans e.g. ECDSA (elliptic curve digital signature algorithm). In security examination of the functions, creators have guaranteed that the signature plan is preimage resistant and second preimage resistant. Nonetheless, the creators have not guaranteed that the plan is collision resistance, which is additionally another vital property.

Earlier authors have outlined a solid and proficient plan against node catch assault utilizing hash chain in WSN. The essential thought of the plan is to utilize preloaded keys to compute the hash value. The hashed key is utilized as a communication key. This plan moreover demonstrates a change in examination of other existing plans.

CHAPTER 5

**PROPOSED LIGHT WEIGHT CRYPTOGRAPHIC HASH
ALGORITHM FOR WSN**

Any hash function which has to be evolved is required to satisfy a number of desirable characteristics.

5.1 REQUIREMENTS OF AN IDEAL HASH FUNCTION

Following are the essential characteristics which a hash calculating functions need to follow

- i. It should accept the arbitrary length inputs.
- ii. Its output length must be fixed size.
- iii. It should be efficient and its computation must be fast.
- iv. **Pre-image resistant:** It is —one-wayness property of the hash function (i.e. it should not be possible to calculate the input from the hash output). A hash function for which preimage/input cannot be efficiently solved is said to be preimage resistant. So, a preimage resistant function must ensure that given $h(X)$, it should not be possible to calculate X .
- v. **Second Preimage Resistant (Weak Collision Resistant):** Attacker should not be able to compute X' which has the Same hash as X has. If it is computationally feasible $h(X)$ cannot be considered as a fingerprint unique to X .
- vi. **Strong Collision Resistant:** The difference between weak and strong collision resistance is very subtle. This can be clarified using Birthday Paradox.
 - Given a person (& his birthday), identifying the second person with the same birthday corresponds to weak collision problem.(There is a fix X_1 , finding another element X_2 where $h(X_1) = h(X_2)$, this instance corresponds to weak collision)
 - Identification of any 2 people in the group having the same birthday corresponds to the strong collision problem.(Finding any 2 elements in a group of n elements s.t. $h(X_1) = h(X_2)$, this instance corresponds to strong collision)

5.2 PROPOSED HASH FUNCTION

The proposed algorithm takes a message of arbitrary length as input. The output is a 12-byte ($12 \times 8 = 96$ bits) hash digest. The code generated using MATLAB is as follows-

1. Initialize the substitution table S_{Table_1} consisting of prime numbers picked up randomly. This table is utilized in the first substitution.

```
function Hash_Digest = Hash_Main (str)
%treating it as a function with input a string message
```

```
STable_1 = [521, 997, 983, 733, 11, 13, 17, 19, 23, 29,...
            31, 37, 41, 43, 47, 53, 59, 61, 67, 71,...
            7, 79, 83, 89, 97, 101, 103, 107, 109, 113,...
            127, 131, 137, 139, 149, 151, 157, 163, 167, 173,...
            179, 181, 191, 193, 197, 199, 211, 223, 227, 229,...
            233, 239, 241, 251, 257, 263, 269, 271, 277, 281,...
            809, 293, 307, 311, 313, 317, 331, 337, 347, 349,...
            353, 359, 367, 373, 379, 383, 389, 397, 401, 409,...
            863, 421, 431, 433, 439, 443, 449, 457, 461, 463,...
            467, 479, 487, 491, 499, 503, 509];
```

2. Initialize the substitution table STable_2 consisting of 67 prime numbers picked up randomly. This helps in reducing overhead and uniform transformation. This table is utilized in the first transformation.

```
STable_2 = [2911, 2899, 2893, 2887, 2871, 2857, 2851, 2837, 2833, 2827,...
            2809, 2797, 2791, 2779, 2773, 2767, 2749, 2743, 2737, 2731,...
            2723, 2719, 2713, 2711, 2701, 2687, 2683, 2677, 2659, 2653,...
            2647, 2641, 2629, 2623, 2617, 2611, 2587, 2563, 2539, 2503,...
            2497, 2447, 2431, 2419, 2413, 2407, 2401, 2389, 2383, 2377,...
            2359, 2347, 3193, 3173, 3157, 3139, 3121, 2257, 2251, 2239,...
            2227, 2221, 2197, 2191, 2179, 2173, 2167];
```

3. Initialize these variables at starting

```
first_conv = 1;
second_conv_p3 = 7;
state = {'01234567', '89ABCDEF', 'FEDCBA10'}; %hexadecimal numbers
```

in this state[0], state[1] and state[2] are 3 variables which help in finding the final hash digest.

4. Preprocessing the random input message by transforming each one of the input character into 8 bit binary number

```
len = length(str); %the length of the input message string
str_1 = reshape(dec2bin(str,8),len,8); %converting message to decimal
str_1 = reshape(str_1',1,[]); %converting to a row vector
```

5. Padding is applied at most significant position to make it divisible by 512

```
len_bin = length(str_1); %binary message length
blocks = ceil(len_bin / 512); %the required number of blocks
```

```

still = blocks*512 - len_bin; %remaining number of bits to be padded
if still == 0
    %do nothing
else
    for i = 1:still
        str_1 = [str_1 '0']; %concatenate a zero
    end
end
len_bin_final = length(str_1); %checking length of padded message

```

6. Splitting the message further 3 times using loops as follows

```

%starting the splitting procedures:
t = len_bin_final / 512; %number of top level blocks

```

7. Firstly, divide the input message (arbitrary length) in a block of 512 bit

```

for i = 1:t %for the available number of blocks
    start = ((i-1)*512) + 1;
    last = (i*512);
    block = str_1(1,start:last); %the investigated block

```

8. After this, splitting each block into 8 blocks of 64 bits

```

for j = 1:8 %second splitting approach
    start_j = ((j-1)*64 + 1);
    last_j = (j*64);
    block_j = block(1,start_j:last_j); %the result of the second split
p = zeros(1,8); %the values of p for each subblock.

```

9. At third step, each of 64 bits block is divided into 8 blocks of 8 bits each.

```

for k = 1:8 %third split level
    start_k = ((k-1)*8 + 1);
    last_k = (k*8);
    subblock = block_j(1,start_k:last_k);

```

10. Finding subblock[i][j][k] // as a result of 3-level split.

```

subblock_dec = bin2dec(subblock);
if subblock_dec == 0
    p_k = 1;
    p(1,k) = p_k;
else
    p_k = abs(subblock_dec - 31);
    p(1,k) = p_k;
end

```


11. Inner most block (8 bit) is substituted using substitution table Stable_1 as per follows and make sure that subblock[i][j][k] consist of at least one 1, update subblock[i][j][k] as follows

```
if subblock_dec == 0
    p_k = 1;
    p(1,k) = p_k;
else
    p_k = abs(subblock_dec - 31);
    p(1,k) = p_k;
end
```

12. Value of first_conv variable is calculated

```
first_conv = subblock_ijk * first_conv; %calculate first conv
if first_conv > 65535
    first_conv = rem(first_conv,65536);
end
end
```

13. After first conversation second conversation takes place

```
second_conv_p1 = rem(first_conv, 67);
%acquiring block number 7:
k = 8;
start_k = ((k-1)*8 + 1);
last_k = (k*8);
subblock_7 = block_j(1,start_k:last_k);
subblock_7_dec = bin2dec(subblock_7); %decimal version
if subblock_7_dec == 0
    second_conv_p2 = second_conv_p1;
else
    second_conv_p2 = 67 - second_conv_p1;
end
second_conv_p3 = second_conv_p3 + rem(second_conv_p1 + first_conv, 256);
```

14. Now third conversation occurs followed by swapping of values

```
%Third conversion start here:
after_second_conv = (rem(first_conv, second_conv_p3)) + first_conv +
STable_2(1,second_conv_p2);
after_third_conv = rem(after_second_conv, 256) + p(3) + rem(p(1),127);
after_third_conv = dec2hex(after_third_conv);
after_third_conv = strcat(state{1}(1,1:6), after_third_conv);
```

15. Inter-hex number swapping is applied on hash output.

```

%Apply intra-hexnumber hexdigit swapping on each hexnumber:
    state{1} = state{2};
    state{2} = state{3};
    state{3} = after_third_conv;
end
end

```

16. Finally calculating the final hash digest.

```

Hash_Digest = strcat(state{1}, state{2}, state{3});
End

```

The final hash digest calculated is always 96 bits long irrespective of the length of input message.

5.3 PERFORMANCE ANALYSIS

As we see in section 5.1, every hash algorithm must always approve six essential properties. By satisfying these basic properties of a cryptographic hash function we can find the strength of any proposed algorithm.

- i. The proposed algorithm takes after the main property of a cryptographic hash work as the information message can be self-assertive long.
- ii. It fulfills the second property as it produces settled length output.

The efficiency of the proposed calculation can be seen from the usage part as takes after:

iii. **Preimage Resistance** – Having a hash digest H with respect to any arbitrary input, it is not feasible to find an input message m such that $h(m) = H$ where $h(m)$ is the message digest of m . It symbolizes the one-way property of a hash function. The final hash digest is the sum of t times of 12 byte hexadecimal numbers where, t is number of 512 bit blocks. Let H be the final hash digest.

$H = \sum H_i$, where i varies from 0 to $t-1$ and H_i is hash digest of each of t number of 512 bit blocks. There will be $H + t - 1C_{t-1}$ ways to calculate H_i . If $t=1$, the number of solutions would become one.

In a serious and powerful attack, for hacking the digest H , it is attempted to find a message m such that $h(m) = H(\text{given})$. The attacker needs to perform of the request of 296 operations which will set aside high measure of opportunity to perform beast compel assault. Thus the calculation is preimage resistant.

iv. **Collision Resistance** – To find out $h(m_1) = h(m_2)$ the followings need to be true

i) $hexnumber_{ij}(m1) = hexnumber_{ij}(m2)$ for all i, j where $i \in \{0,1,\dots,t-1\}$ and $j \in \{0,1,\dots,7\}$ and

ii) $subblock_{ij}(m1) \neq subblock_{ij}(m2)$ for all i, j where $i \in \{0,1,\dots,t-1\}$ and $j \in \{0,1,\dots,7\}$

For these conditions to be correct at least one of the following conditions need to be satisfied:

first_conv (m1) = first_conv (m2) and sub_blockij (m1) \neq sub_blockij (m2)

after_second_conv (m1)=after_second_conv (m2) and sub_blockij (m1) \neq sub_blockij (m2)

after_third_conv (m1) = after_third_conv (m2) and sub_blockij (m1) \neq sub_blockij (m2)

So the main attainable strategy to fulfill no less than one of the above conditions is beast compel assault. Thus our calculation is crash safe. 248 operations are to be performed by the beast constrain strategy to figure two messages having a similar message process which would be tedious for a sensor node.

v. **Second Preimage Resistance** – For a given input message $m1$, if it is impossible to find another input message $m2$ where the hash output of first message is equal to the hash output of second input message $m2$ i.e., $h(m1) = h(m2)$.

Second preimage resistance is a simpler or weaker form of strong collision resistance. Along these lines, a solid crash safe capacity would likewise take after the property of second preimage safe.

5.4 RESULTS AND OUTPUT

1.

Following figure shows the 96 bit hash digest of input message information “Good Morning Everybody, This is a Test Message number one, and I am very glad to be working on it” and graphical representation of the same hash digest.

```
Current Folder: C:\Users\TIN-TIN\De
Shortcuts How to Add What's New
Hash Digest:
89ABCD1DFEDCBA1501234563

Hash Digest in Binary format for 12 bytes:

Hash_output =

11011000
00010000
01011000
01101000
11111000
01110000
00000000
10110000
11110000
01110000
10100000
01100000
```

Fig 5.1: 96 bit hash digest for first input message

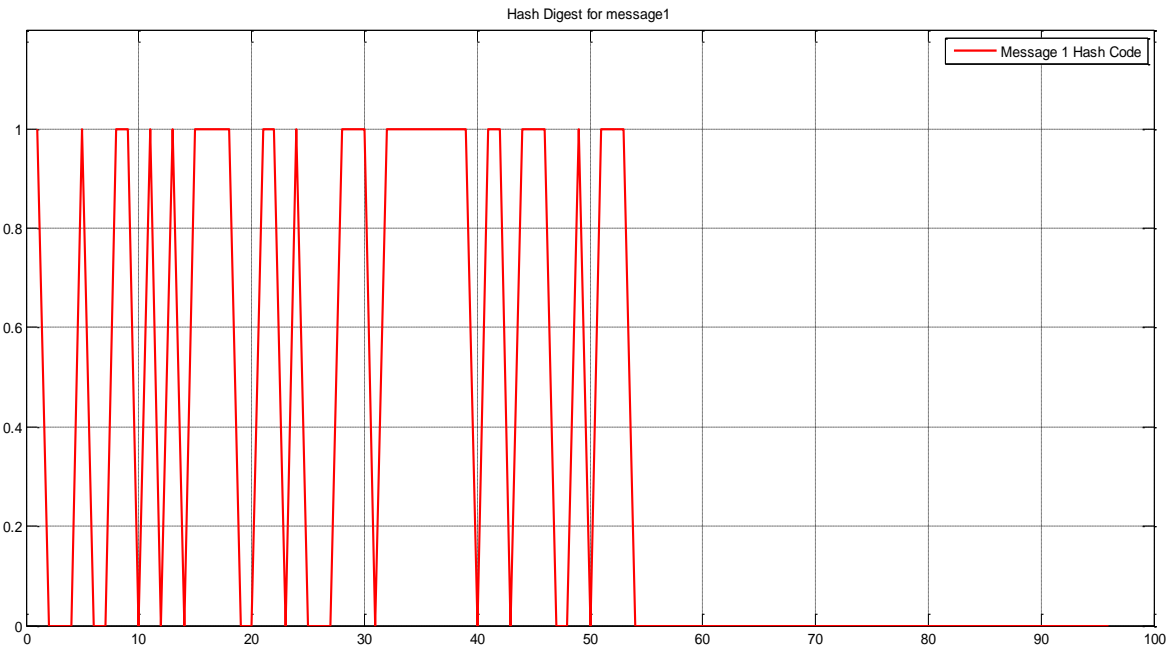


Fig 5.2: Plot for first hash digest

2.

This figure shows the 96 bit hash digest of input message information “Hello World” followed by the graphical representation of the same hash digest in the form of bits.

```
Hash Digest:
FEDCBA94012345E389ABCD15

Hash Digest in Binary format for 12 bytes:

Hash_output =

11100000
11001000
10000000
10110000
11001000
10100000
11010000
01000000
11000000
10000000
01010000
10010000
```

fx >> Start OneNote 2016

Fig 5.3: 96 bit hash digest for second input message

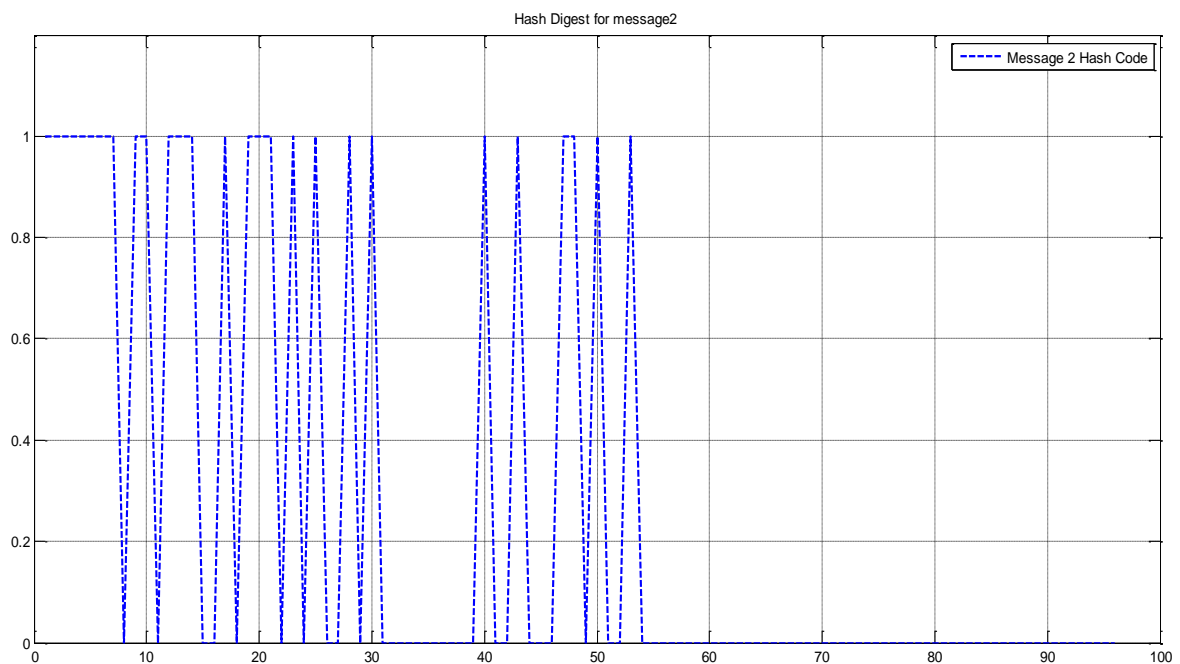


Fig 5.4: Plot for second hash digest

3.

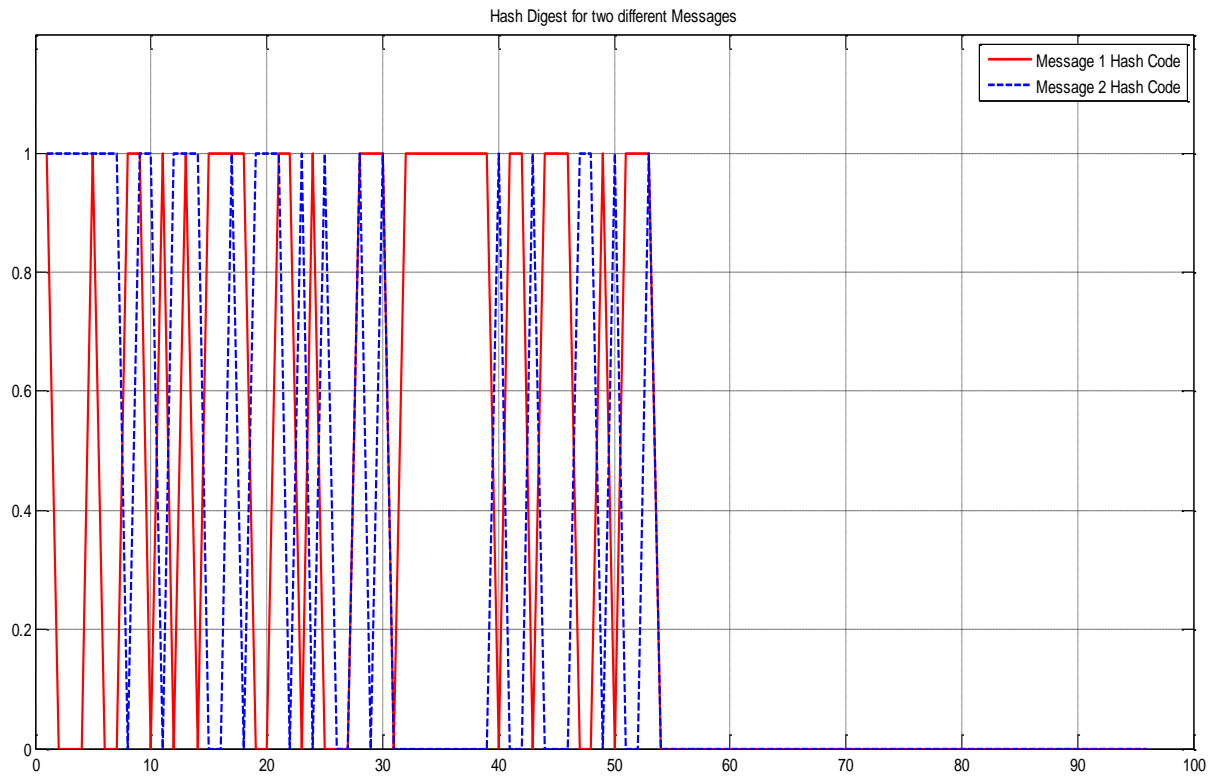


Fig 5.5: Comparison of the two hash digests

In Fig 5.5 the comparison of the graphical representation of the two hash digests of input messages is shown. It is shown in this clearly that the hash digest of these two message inputs are completely different. Hence even if we calculate the hash function of two very close input messages the hash function is very different. With this we have shown how packets can be transmitted securely in Wireless sensor networks without any computational overhead.

CHAPTER 6

CONCLUSION AND FUTURE ASPECTS

In this work the fundamental necessities to outline a lightweight, one-way hash function which creates a fixed and moderately short length hash process which is used for securing energy starved remote system e.g. WSN are talked about. Operations like MOD and SWAP are utilized broadly to make the proposed hash algorithm lightweight. All the essential properties, for example, preimage resistance, solid Collision resistance and feeble or second preimage resistance are satisfied in the proposed algorithm. The proposed algorithm is actualized utilizing MATLAB software. It is contrasted with MD5 and SHA-1 and has demonstrated critical change as far as communication overhead and calculation speed.

Since energy starved networks consume huge energy for calculations and deciding routing protocol. Hence this 96 bit light weight, one way hash digest provides the best algorithm for such networks. It also helps in removing all computational overheads involved with such networks. Also conventional algorithms produced a hash digest consisting of larger number of bits and hence require higher computational energy was required which is completely removed by using this algorithm.

One more very important property of this algorithm is Second Preimage Resistance which is also implemented. With the help of this property it is impossible to find two messages having same hash function. Making it more difficult for the hackers to intercept the information message even if they have knowledge of a hash digest of a message very close to original message.

As per Mica2 specification, energy consumption for transmitting one byte of data for ATmega 128 processor is $16.25\mu\text{J}$ and energy required per clock cycle is 3.2nJ or $0.0032\mu\text{J}$. For the given determination, the energy requirements for all the contending plans are processed underneath:

Communication overhead:

MD5 -- $16.25 \times 128 \text{ (bit)} = 16.25 \times 16 \text{ (byte)} = 260 \mu\text{J}$

SHA1-- $16.25 \times 160 \text{ (bit)} = 16.25 \times 20 \text{ (byte)} = 325 \mu\text{J}$

Proposed Algorithm— $16.25 \times 96 \text{ (bit)} = 16.25 \times 12 \text{ (byte)} = 195 \mu\text{J}$

REFERENCES

1. I. F. Akyildiz *et al.*, —A Survey on Sensor Networks, *IEEE Commun. Mag.*, vol. 40, no. 8, Aug. 2002.
2. Handbook of Cryptography by A. Menezes, P. van Oorschot, and S. Vanstone, CRC Press, 1996.
3. H. Dobbertin: Cryptanalysis of MD4, Journal of Cryptology.
4. X. Wang and H. Yu: How to Break MD5 and Other Hash Functions, EuroCrypt 2005, Springer LNCS 3494, pp. 19–35, 2005.
5. M. Stevens, A. Lenstra and B. Weger.: Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different
6. Rijmen, V.Oswald: Update on SHA-1, RSA 2005, LNCS 3376, pp. 58-71.
7. R. P. McEvoy, F. M. Crowe, C. C. Murphy and W. P. Marnane: Optimisation of the SHA-2 Family of Hash Functions on FPGAs,
8. IEEE Computer Society Annual Symposium on VLSI: Emerging VLSI Technologies and Architectures (ISVLSI 06), IEEE Computer Society, Washington DC.
9. Y. Jararweh, L. Tawalbeh, H. Tawalbeh and A. Moh'd: Hardware Performance Evaluation of SHA-3 Candidate Algorithms, Journal of Information Security.
10. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci: Wireless sensor network: a survey, Computer Networks.
11. Erik Dahmen and Christoph Kraus: Short Hash-based Signatures for Wireless Sensor Networks, 8th International Conference on Cryptology & Network Security (CANS), Kanazawa, Ishikawa, Japan, December, 2009.
12. Tao Qen, Hanli Chen: An Enhanced Scheme against Node Capture Attack using Hash Chain for Wireless Sensor Network, Information Technology Journal 11.

13. Paulo Barreto, Ventzislav Nikov, Svetla Nikova, Vincent Rijmen, Elmar Tischhauser: Whirlwind: a new cryptographic hash function..

14. https://en.wikipedia.org/wiki/Cryptographic_hash_function accessed on 18-07-2015
Atmel AVR8-bit Microcontroller ATmega 128 processor datasheet
(<http://tools.ietf.org/html/rfc4270>).

B.T-34
28/4/17

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT
LEARNING RESOURCE CENTER

PLAGIARISM VERIFICATION REPORT

Date: 28/04/2017

Type of Document (Tick): ☒ Thesis ☐ M.Tech Dissertation/ Report ☒ B.Tech Project Report ☐ Paper

Name: Nimit Bhardwaj, Aman Gupta, Sudhanshu Singh Department: ECE


Enrolment No. 131082, 131097, 131098 Registration No.

Phone No. 9816927557, 9736122448 Email ID. singshiv22@gmail.com

Name of the Supervisor: Dr. Shweta Pandit

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): DYNAMIC
KEY EXCHANGE PROTOCOL FOR SECURITY IN
WIRELESS SENSER NETWORKS

Kindly allow me to avail Turnitin software report for the document mentioned above.


(Signature)

FOR ACCOUNTS DEPARTMENT:

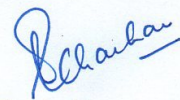
Amount deposited: Rs. 900/- Dated: 28/4/17 Receipt No. CRV 1704/369
(Enclosed payment slip)



(Account Officer)

FOR LRC USE:

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Report delivered on	Similarity Index in %	Submission Details	
			Word Counts	6946
28-04-17	01-05-17	26 %	Character Counts	37501
			Page counts	42
			File Size	1.19 M


Checked by
Name & Signature


LIBRARIAN
LEARNING RESOURCE CENTER
Jaypee University of Information Technology
Waknaghat, Distt, Solan (Himachal Pradesh)
Pin Code: 173234

Submission Info

SUBMISSION ID	807579315
SUBMISSION DATE	01-May-2017 10:11
SUBMISSION COUNT	1
FILE NAME	Project_Report_Nimit_A...
FILE SIZE	1.19M
CHARACTER COUNT	37501
WORD COUNT	6946
PAGE COUNT	42

ORIGINALITY

OVERALL	26%
INTERNET	19%
PUBLICATIONS	9%
STUDENT PAPERS	21%

GRADEMARK

LAST GRADED	N/A
COMMENTS	0

QUICKMARKS



LIBRARIAN
LEARNING RESOURCE CENTER
Jaypee University of Information Technology
Waknaghat, Distt, Solan (Himachal Pradesh)
Pin Code: 173234