A Lightweight Block Cipher Algorithm for Secure SDN Environment

Jungha Jin
Dept. of Computer Engineering
KONKUK University
Seoul, 05029, Korea
drake75@konkuk.ac.kr

Yewon Oh
Dept. of IT Convergence Information Security
KONKUK University
Seoul, 05029, Korea
iamoyw@konkuk.ac.kr

Keecheon Kim
Dept. of Computer Engineering
KONKUK University
Seoul, 05029, Korea
kckim@konkuk.ac.kr

***Abstract*— Software Defined Network is a next-generation networking technology that transforms a closed network environment based on existing network vendors into a flexible, software-based, centralized management environment that can be simplified by abstracting and programming. Although these advantages can be applied to some security problems rather than existing networks, most of the security problems and vulnerabilities of existing networks are present and various attacks are taking place. In this paper, we propose a structure to enhance the security function of SDN by checking how to implement the network security function using SDN technology and lightening the existing block cipher algorithm for this security problem. Lightweight-AES algorithm, which is a lightweight block cipher algorithm based on the AES-256 algorithm, which can simultaneously satisfy the quality of high level of security. In the case of simply reducing the number of round operations of the AES algorithm, the difference diffusion effect of the KeySchedule function generating the round key is reduced, and the security of the encryption algorithm is degraded due to the related key attack using the related key difference characteristic. The Lightweight-AES algorithm proposed in this paper improves the rate of cancellation and decryption by reducing the number of round operations, and the round internal function is supplemented to increase the differential diffusion effect of the KeySchedule function. In order to evaluate the performance of the Lightweight-AES algorithm proposed in this paper, a comparison simulation is performed with the existing AES algorithm. As a result, we confirmed that the Lightweight-AES algorithm can provide SDN content security equal to the encryption / decryption rate and algorithm security strength of the AES-128 algorithm. Therefore, it is considered that the proposed Lightweight-AES algorithm can provide better security service in SDN environment quality and security than the existing AES-128 algorithm.**

***Index Terms*—SDN, Block cipher algorithm, AES-256, Related Key Attack, Avalanche Effect**

I. INTRODUCTION

Software Defined Network, an issue in the field of networks, is a flexible software-based centralized management system that simplifies networked hardware and software-based, closed network environments based on existing network hardware vendors through network abstraction. It is the next generation networking technology that converts to environment. Network administrators can program the control plane in software using these functions to control the communication functions in the data plane according to the management purpose of the network. Therefore, it is easy to handle general network control, routing and complex operation management functions, Balancing, QoS, and other advanced features can be easily implemented. These flexible technologies and features of the SDN environment can be used for network security in many areas. In addition, it is possible to enhance the security of SDN by expanding its functions more structurally by integrating various existing open source software security solution and software with flexible software controllable SDN environment characteristics. However, there are not many cases where the SDN technology is analyzed / implemented by approaching the method of lightening the encryption algorithm among the viewpoint of network security or strengthening the security by using the existing AES-256 encryption algorithm.

In this paper, we propose a Lightweight Advanced Encryption Standard algorithm, which is a lightweight block encryption algorithm based on the AES-256 algorithm. When the number of round operations of the AES algorithm is reduced, the difference diffusion effect of the KeySchedule function that generates the round key is degraded, and the security of the encryption algorithm is degraded due to the related key attack using the differential characteristics of the round keys. do. However, in order to increase the encryption / decryption rate, we propose a method to increase the differential diffusion effect of KeySchedule function by reducing the number of round operations and implement the algorithm for performance comparison.

II. RELATED WORK

A. SDN (Software Defined Network)

The Software Defined Network (SDN) abstracts the closed network of fixed hardware and software into a three-plane structure of the data plane, control plane, and application plane. The data plane is the layer that receives control through the SDN's specific interface and is responsible for the transmission of the data flow. The control plane is the layer that controls the flow of data, and determines whether to route, forward, or reject the flow of data through application and network

This is an open access article under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/)

services. And transfers the operations of the data plane to the application layer in the form of an API. Finally, the application plane performs various functions of the network using the APIs provided by the control plane. We also define an open interface called a southbound interface and a northbound interface between each plane to support device / resource abstraction, control abstraction, and service abstraction, respectively. Figure 1 is a simplified illustration of the conceptual structure of SDN [1].

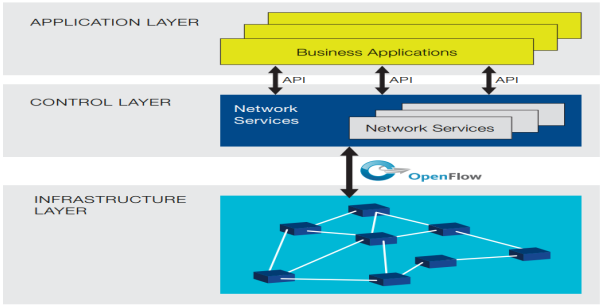


Fig. 1. SDN Architecture

B. Scrambling

Scrambling is a technology that encrypts packets through a block cipher algorithm such as AES, 3DES, etc. so that only legitimate users can receive them normally during data transmission. Encrypts only the data payload part of the transmitted packet. In the case of MPEG-2 TS packet, it encrypts only the data payload part except the TS header and adaptation field as shown in Figure 2.[2][3]



Fig. 2. MPEG-2 TS Format

In the process of generating the scrambled packet, as shown in Figure 3, the data payload part of the media content packet is divided into 16bytes block size, and the encrypted block is generated by applying the block encryption algorithm as the operation mode of the CBC mode having dependency between the blocks.

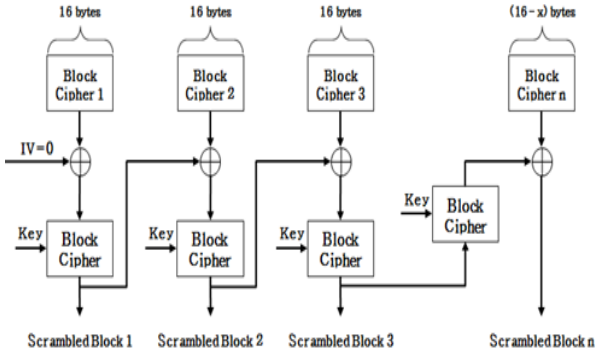


Fig. 3. Scrambling process in CBC mode

The process of scrambling an arbitrary 16-byte block is shown in Equation (1). An Exclusive-OR operation is performed on the n-th block B (n) and the previously generated n-1th scrambled block S (n-1). Then, S (n), which is an n-th block scrambled by using the E block encryption algorithm and the k secret key, is generated.

$$S(n) = E_k[B(n) \oplus S(n-1)] \quad (1)$$

In order to scrambling the first block B (1), S (0) must be used. Since the block does not exist, IV (Initialization Vector) is used. IV value is 16 bytes filled with zero. Therefore, the equation for scrambling the first block can be expressed as (Equation 2).

$$S(1) = E_k[B(1) \oplus IV)] \quad (2)$$

If the last end block is less than 16 bytes, use padding to fill the block size to 16 bytes. The types of padding include zero padding to fill the remaining bytes with zeros, random padding to fill with arbitrary bytes, and PKCS7 to fill padded numbers. PKCS7 is the most commonly used type.

C. AES(Advanced Encryption Standard)

AES algorithm is an algorithm that encrypts a 128-bit block using 128/192 / 256bit secret key. It operates in 10/12/14 round according to the length of the secret key as shown in Table Ⅰ.[4][5]

TABLE Ⅰ. Features of AES Algorithm

	Block Size	Key Size	Number of Round
AES-128	128bit	128bit	10 Round
AES-192	128bit	192bit	12 Round
AES-256	128bit	256bit	14 Round

The functions used in the AES algorithm are composed of five functions as shown in Table II and can be classified into a function that operates outside the round and a function that operates in the round.

TABLE II. AES algorithm function

Function name	Function	Action
AddRoundKey	Function that performs Exclusive-OR operation using 128bit state and 128bit round key	Inside the round
SubBytes	Function to perform nonlinear substitution using 16×16 matrix S-Box	
ShiftRows	A function that performs a Left Circular Shift operation on a State in Row units	
MixColumns	Function to replace State by Column using $GF(2^8)$ arithmetic expression	
KeySchedule	A function that generates a round key for each round using a whitening key	Round Outside

IJARE

This is an open access article under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/)

Figure 4 shows a schematic representation of the encryption and decryption process in the AES algorithm.

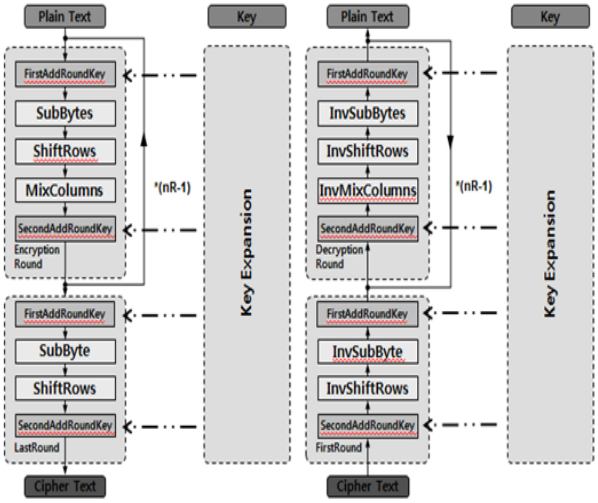


Fig. 4. AES algorithm encryption decryption process

IV. PROPOSED METHOD

The use of cryptographic algorithms in the security of existing network environment is an indispensable factor and is a major factor that causes significant difference in hardware performance. The SDN environment is not much different from this. In this paper, we propose a method to enhance security of SDN environment by using AES-256 encryption algorithm which is used in general security environment by using light weight, and to minimize the lack of security due to light weight.

A. Lightweight AES Algorithm

Existing AES algorithms have no known key decryption technique other than Brute-Force Attack, which is a random assignment attack technique. Each AES algorithm has an attack complexity of AES-128 is $2^{126.1}$, AES-192 is $2^{189.9}$, and AES-256 is $2^{254.4}$ by Brute-Force Attack.[6] However, in the case of the AES algorithm, which reduces the round, the rounding operation of the round key reduces the diffusing effect of the KeySchedule generating the round key, and the related key attack using the two related key difference characteristics, A problem of lowering occurs. The Related Key Attack is known as the best attack technique to decrypt the rounded AES algorithm, although it is the theoretical attack to be carried out with the condition that the plaintext and the ciphertext can be obtained with part of the associated round key.[7][8][9] For the AES-256 algorithm, the vulnerability occurs when the number of round operations is reduced to 9 times. The number of KeySchedule operations is 5 times. Therefore, even if the number of round operations is reduced to reduce the weight, it is possible to solve the safety problem of the lightweight AES algorithm by using the round key so that the KeySchedule function can operate at least six times.

In this paper, we propose a Lightweight AES algorithm, which is a lightweight block cipher algorithm based on AES-256, that secures the security of Related Key Attack even if the number of round operations is reduced by increasing the KeySchedule differential diffusion effect of AES algorithm.

B. Basic Configuration of Lightweight AES Algorithm

The lightweight AES algorithm is a 128-bit block encryption algorithm based on the AES-256 algorithm using a 256-bit secret key as shown in Table III. The lightweight AES algorithm complements the internal function to use 256 - bit round keys in one round, unlike the conventional AES algorithm which used 128 - bit round keys in one round to increase the differential spreading effect of KeySchedule function that generates round keys. Based on the number of operations of the increased KeySchedule function, the number of round operations was reduced by 7, which is half the level of the existing AES-256 algorithm.

TABLE III. Feature comparison between lightweight AES algorithm and AES-256 algorithm

	Block Size	Key Size	Number of Round	Number of KeySchedule
Lightweight AES	128	256	7	7
AES-256	128	256	14	15

C. Round internal function of lightweight AES algorithm

Among the round internal functions of the lightweight AES algorithm proposed in this paper, the functions of SubBytes, ShiftRows, and MixColumns are the same as those used in the existing AES algorithm. However, the AddRoundKey function that exclusively ORs the generated state and the 128-bit round key during encryption / decryption is composed of FirstAddRoundKey and SecondAddRoundKey, which are extended to use 256-bit round keys in one round.

The core elements of the round internal functions of the lightweight AES algorithm proposed in this paper are the FirstAddRoundKey function and the SecondAddRoundKey function. These functions perform the Exclusive-OR operation with the 128-bit state generated during the encryption and decryption of the 128-bit round key generated by the KeySchedule function in the same manner as the existing AddRoundKey function. However, unlike the existing function that simply uses the 128bit round key sequentially, the ZigZag Bit operation method is used to eliminate the continuity of the key usage. The round key of 256 bits is used in one round because of the two AddRoundKey functions. The FirstAddRoundKey function performs Exclusive-OR operation using the 128-bit round key corresponding to the odd column as shown in Figure 5. The SecondAddRoundKey function performs Exclusive-OR operation using the 128-bit round key corresponding to the even column as shown in Figure 6 do. The equations for each function can be defined as (3) and (4).

$$[S'_{xy}, \cdots] = [S_{xy}, \cdots] \oplus [W_{x,(r*2)+8*(nR-1)}, \cdots] \quad (3)$$

This is an open access article under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/)

$$[S'_{x,y},\cdots] = [S_{x,y},\cdots] \oplus [W_{x,((y*2)+1)+8*(nR-1)},\cdots] \quad (4)$$

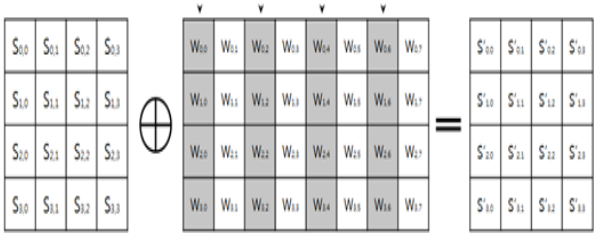


Fig. 5. FirstAddRoundKey Operation Process

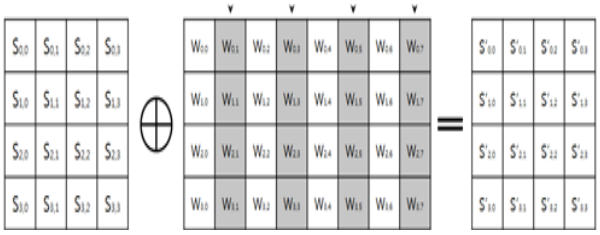


Fig. 6. SecondAddRoundKey Operation Process

D. Encryption process of lightweight AES algorithm

Unlike the existing AES algorithm, the encryption process of the lightweight AES algorithm performs the round function without performing the whitening step through the AddRoundKey function. Round internal functions of the 7-round Lightweight AES encryption algorithm are performed in the order of FirstAddRoundKey → SubBytes → ShiftRows → MixColumns → SecondAddRoundKey. In the last round, only the remaining functions except the MixColumns function operate. Figure 7 shows the encryption process of the lightweight AES algorithm.

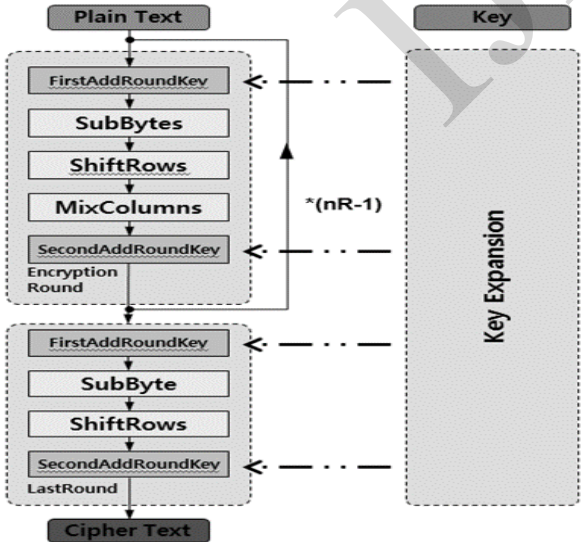


Fig. 7, Cryptographic operation process of lightweight AES algorithm

The code of the lightweight encryption algorithm can be defined as follows. After receiving the plain text and the 256bit secret key, all the round keys needed for each round are generated in advance through the KeySchedule function. Then, the input plain text is generated using the round function composed of FirstAddRoundKey, SubBytes, ShiftRows, MixColumns, and SecondAddRoundKey functions. The round function is repeated seven times in total, and the code is configured so that the MixColumns function does not work in the last round.

E. Decryption process of lightweight AES algorithm

The decryption process of the lightweight AES algorithm proceeds in the reverse order of the encryption process as shown in Figure 8, and decryption is performed using InvSubBytes, InvShiftRows, and InvMixColumns, which are inverse functions of SubBytes, ShiftRows, and MixColumns functions used in the encryption process. The round internal function of the lightweight AES decryption algorithm is repeated seven times in the order of SecondAddRoundKey → InvMixColumns → InvShiftRows → InvSubBytes → FirstAddRoundKey, which is a reverse order of the encryption process, and decrypts ciphertext into plaintext. However, in the first round, only the functions except the InvMixColumns function operate, whereas the MixColumns function is not used in the final round during the encryption process.

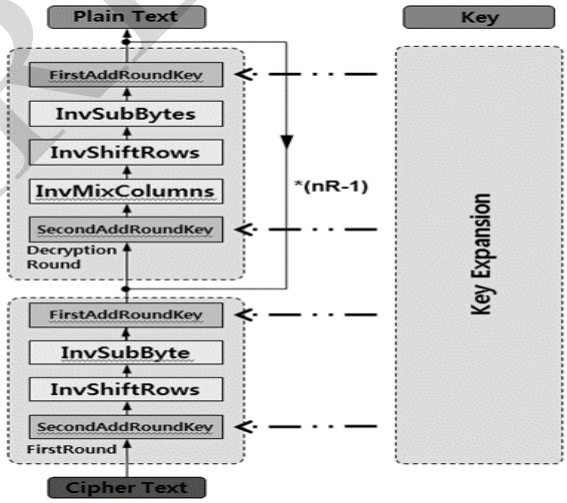


Fig. 8. Decryption process of lightweight AES algorithm

The code of the lightweight AES decryption algorithm can be defined as follows. When a ciphertext and a 256-bit secret key are input in the same way as the encryption process, all the round keys necessary for each round are generated in advance through the KeySchedule function. Then decrypt ciphertext into plaintext using round function consisting of SecondAddRoundKey, invMixColumns, invShiftRows, invSubBytes, FirstAddRoundKey, and function. The round function is repeated 7 times in total, and the code is configured so that the invMixColumns function does not work in the first round.

IJARE

This is an open access article under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/)

V. Implementation and Performance Evaluation of Lightweight AES Algorithm

A. Lightweight AES algorithm implementation

The lightweight AES algorithm proposed in this paper is implemented in an environment using C language in Win10 OS system. Figure 9 shows the result of encrypting / decrypting arbitrary data using the implemented lightweight AES software. When decrypting a cipher text output through an arbitrary 128-bit plaintext value and a 256-bit secret key using the same secret key, it can be confirmed that the 128-bit plaintext value initially input is normally output.



Fig. 9. Lightweight AES encryption / decryption software operation result

B. Evaluation of Cancellation / Decryption of Lightweight AES Algorithm

In order to evaluate the performance of the proposed lightweight AES algorithm, we analyzed the differences in the arm / decryption speed of the existing AES-128 and AES-256 algorithms. Simulation of each algorithm was performed in PC environment based on AMD Phenom II x6 1055T 2.80GHz CPU and 16.00GB memory. For the performance evaluation of each cryptographic algorithm, the sample data composed of 1M, 5M, 10M, 25M, 50M, 75M, 100M, 250M and 500M were encrypted and decrypted.

Table IV shows the results of measuring the encryption time of the Lightweight AES, AES-128, and AES-256 algorithms, and the values specified in the table are average values of 20 samples of each sample data.

TABLE IV. Measure encryption speed

	Lightweight AES (sec)	AES-128(sec)	AES-256(sec)
1MB	0.14	0.17	0.24
5MB	0.7	0.85	1.16
10MB	1.42	1.68	2.32
25MB	3.50	4.14	5.8
50MB	6.99	8.37	11.64
75MB	10.5	12.46	17.67
100MB	13.81	16.64	23.39
250MB	34.58	42.54	58.93
500MB	70.43	84.76	117.84

The encryption speed of the proposed Lightweight AES algorithm is improved by about 17% compared to AES-128 and about 40% compared with AES-256. As a result, it can be confirmed that the encryption process of the Lightweight AES algorithm is lighter than the encryption process of the existing AES algorithm.

Table V shows the result of measuring the decryption time of each algorithm under the same conditions as in Table V. Since the decryption process uses the inverse function, the operation is complicated, and the difference is about twice the encryption speed.

TABLE V. Measure decryption speed

	Lightweight AES (sec)	AES-128(sec)	AES-256(sec)
1MB	0.21	0.28	0.45
5MB	1.24	1.59	2.37
10MB	2.13	2.81	4.43
25MB	5.42	7.14	11.09
50MB	10.73	14.13	22.26
75MB	15.84	21.19	33.57
100MB	20.95	28.18	44.44
250MB	52.43	71.66	112.37
500MB	104.77	143.86	231.68

The decryption speed of the lightweight AES algorithm is improved by about 26% compared to AES-128 and about 53% compared with AES-256. As a result, it can be confirmed that the decryption process is lightened like the encryption process of lightweight AES algorithm.

The security strength of the encryption algorithm is based on Brute-Force Attack, and the attack complexity of Lightweight AES, AES-128, and AES-256 for 10M sample data is measured. As a result, although the lightweight AES algorithm proposed in this paper is lighter than the existing AES algorithm as shown in TABLE VI, it has a higher security strength than AES-128. In addition, there was not much difference in attack complexity between AES-256 using the same size secret key.

TABLE VI. Attack complexity by Brute-Force Attack

	Sample Data (10MB)	
	Encryption	**Decryption**
Lightweight AES	$1.42 * 2^{254.5} = 6.23^{76}$	$2.13 * 2^{254.6} = 9.34^{76}$
AES-128	$1.68 * 2^{126.2} = 1.64^{38}$	$2.81 * 2^{126.2} = 2.74^{38}$
AES-256	$2.32 * 2^{254.5} = 1.01^{77}$	$4.43 * 2^{254.6} = 1.94^{77}$

C. Avalanche Effect Verification of Lightweight AES Algorithm

The Avalanche Effect is a property that causes a significant change in the output value when a slight change is made to the input value of any algorithm. It also corresponds to the diffusion property in Confusion and Diffusion concept introduced by Claude Elwood Shannon. If the size of the change in the key or plaintext due to the weak Avalanche Effect is also present in the cipher text, the probability that the plaintext can be found through the cipher text increases and the security of the algorithm becomes poor.[7] Therefore, the

IJARE

This is an open access article under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/)

Avalanche Effect of the encryption algorithm should be strong enough. In order to verify the Avalanche effect of the lightweight AES algorithm, AES-256 using the same size secret key was compared and analyzed under the same conditions as Figure 10.

	128bit data
Sample data (1)	01 23 45 67 89 AB CD EF 01 23 45 67 89 AB CD EF
Sample data (2)	02 23 45 67 89 AB CD EF 01 23 45 67 89 AB CD EF
Key	2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C A0 FA FE 17 88 54 2C B1 23 A3 39 39 2A 6C 76 05

Fig. 10. Avalanche Effect Simulation Condition

If the 8th bit is used to encrypt two different sample data using the AES-256 algorithm, 5 bits will be different after the first-round operation. At the end of the second round, we can see that more than half of the 76-bit level changes occurred, and we can see that the bits change in similar amounts in the remaining rounds. If encryption is performed using lightweight AES algorithm using plaintext and secret key under the same conditions, we can see that the 73bit, which is more than half the level from the second round, has changed as the existing AES-256 algorithm. It was confirmed through simulations that it changed to similar level in the remaining rounds afterwards.

Simulation results show that the lightweight AES algorithm, like the AES-256 algorithm, produces a large change in data over half the time from the second round, and the variation is also similar. Therefore, we can confirm that the proposed lightweight AES algorithm has a very powerful Avalanche effect even though the round is reduced, and the weight is reduced.

VI. CONCLUSIONS

In this paper, we propose a lightweight AES algorithm based on AES-256 algorithm to provide security in SDN environment. Generally, the number of round operations can be reduced to reduce the weight of the encryption algorithm. However, in the AES algorithm, the difference diffusion effect of the KeySchedule function that generates the round key is reduced, and the related key attack using the difference characteristic of the two related round keys Cryptographic algorithm vulnerability occurs. In this paper, we reduced the number of round operations to reduce the weight but supplemented the round internal function to improve the differential diffusion effect of the KeySchedule function.

The lightweight AES algorithm proposed in this paper is composed of FirstAddRoundKey and SecondAddRoundKey functions to round internal functions to use 256bit round keys in one round, unlike the existing AES algorithm that uses 128bit round keys in one round. By increasing the number of RoundKeys to be generated, security is enhanced. Therefore, even if the number of Round Keys increases by twice the number of KeySchedule functions compared to the existing AES algorithm, and the number of rounds is reduced by half, and it was seen as equivalent level.

The lightweight AES algorithm proposed in this paper solves the weakness of the related key attack in the case of round reduction and proves that it can provide similar level of security and improved encryption / decryption rate compared to the existing AES algorithm through simulation. Therefore, if data is protected in the SDN environment by applying the lightweight AES algorithm proposed in this paper, it can be considered that SDN composed of software rather than hardware can provide an improved level of security with the lightweight AES algorithm.

ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (2014-0-00547, Development of Core Technology for Autonomous Network Control and Management) and Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (No. NRF-2017M3C4A7083678).

REFERENCES

[1] ONF, "Software-Defined Networking: The New Norm for Networks," ONF White Paper(https://www.opennetworking.org/images/stories/dow nloads/sdn-resources/white-papers/wp-sdn-newnor m.pdf

[2] "Common Scrambling Algorithm for IPTV Service Protection", Korea Information Technology Association, 2009.

[3] Digital Content Protection Management Technology Trend, Korea Creative Content Agency 2010. 12

[4] Joan Daemen and Vicent Rijmen, “AES Proposal: Rjindeal”, NIST Document version 2, March, 1999.

[5] Federal Information Processing Standards Publication 197, “Specification for the ADVANCED ENCRYPTION STANDARD (AES)”, November, 2001.

[6] D. J. Bernstein, "Understanding brute force", Proc. ECRYPT STVL Workshop Symmetric Key Encryption (SKEW), 2005.

[7] D. P. N. C. Jayant, P. Bhoge, "Avalanche Effect of AES Algorithm", (IJCSIT) International Journal of Computer Science and Information Technologies, vol. 5, no. 3, pp. 3101-3103, 2014.

[8] A. Biryukov, O. Dunkelman, N. Keller, D. Khovratovich, A. Shamir, "Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds", Proc. EUROCRYPT, vol. LNCS-6110, pp. 299-319, 2010

[9] A. Biryukov, D. Khovratovich, I. Nikolic, "Distinguisher and related-key attack on the full AES-256", Proc. CRYPTO, vol. LNCS-5677, pp. 231-249, 2009.

[10] National Bureau of Standards, Data Encryption Standard, FIPS-Pub. 46. National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January, 1977.

This is an open access article under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/)