ISSN:0975-9646

RSA FOR AIRCRAFT WIRELESS NETWORK USING SESSION BASED KEYS

Dr Peter Reji

Asst Professor,
Electronics and Communication Engg. Dept
Viswajyothi College of Engineering and Technology
Email: {peter@vjcet.org}

***Abstract* -Wired technology has been the preferred means of intra communication between on-board units, in an aircraft. On-board cables apart from increasing the weight also increase the complexity related to maintenance, up-gradation, modifications etc. Wireless technology can address the requirements of modern avionics systems apart from reducing the weight of the wiring and associated hardware significantly. Introducing wireless technology, for aircraft intra communication, can have serious security issues. Therefore, the data flow between the Line Replaceable Units (LRU) needs to be encrypted for preventing an adversary from getting access to the data. The paper proposes a novel approach of using a modified Rivest, Shamir, Adleman (RSA) algorithm for encryption of data between LRUs in an Avionics Wireless Network (AWN). The standard RSA algorithm has been modified for generating the same session based keys, at both time synchronized communicating LRUs, thus eliminating the need for sharing the public key. Time synchronisation is proposed using the aircraft system clock. Session based keys improve the robustness of the algorithm for deployment in avionics wireless environment**.

***Key Words* - Avionics Wireless Network, RSA, LRU, Time Synchronization, AWN**

I INTRODUCTION

Avionics in aircraft has been growing at a fast pace, with an aim to cater for the ever increasing demand for real-time data at a faster rate. The real-time information about on-board system status, warnings or information etc. is provided to the aircrew by a highly reliable, efficient and secured data network. Data networks in all aircraft are presently built upon physical connections using wires like MIL-STD-1553, ARINC 429, ARINC 629 ARINC 664/AFDX (ARINC, 2009) etc. These networks result in large quantity of onboard cables. Length of on-board cables in Boeing 747 to the 777-300 varies from 110 miles to 150 miles [1] and the A380-800 carries cables equal to 290 miles [2]. On-board wires in an aircraft have been a major cause of concern for the aircraft manufacturers and operators primarily due to the additional weight of the wires. This in turn reduces the amount of fuel and payload that can be carried on-board. In addition, sufficiently large amount of maintenance effort involved in rectifying cable faults. These aspects have a direct bearing on the manufacturing and operations cost. Other issues associated with wires are related to aging effect of wires, EMI/EMC issues, routing of wires to moving parts, replacement of wires for up-gradation or modifications etc. Wireless Avionics Intra Communication (WAIC) by Aerospace Vehicle System Institute (AVSI), with participation of leading aircraft manufacturers aims to integrate the on-board LRUs through a wireless network [2].

Implementing an Avionics Wireless Network (AWN) will have serious security risks. A Boeing 757, used by US Department of Homeland Security (DHS) officials, was hacked by a joint team of experts from government agencies, academicians and personnel from industry in two days [3]. The hacking was carried out in a non-laboratory setting and from a remote location. Therefore, AWN should provide sufficient security to ensure that an unwanted attacker is not able to interfere with the normal functioning of aircraft systems. [4] has discussed the possible attacks on an AWN, which may include replay attack, looping attack, ARP spoofing and MAC spoofing attacks. These attacks can be mitigated using asymmetric encryption, while Eavesdropping and Man in the Middle attacks can be countered using session keys. RSA, DSA, Diffie-Hellman etc. are some of the asymmetric encryption algorithms. With an aim of using asymmetric encryption for AWN, the paper proposes a modification to RSA.

The present paper proposes a different approach to the RSA algorithm by introducing the concept of S-box (associated with symmetric encryption algorithm) and by using session keys. The proposed session based keys for RSA eliminates the requirement of sharing the public key between time synchronized LRUs and uses session keys, based on the aircraft system clock. The paper is divided into IV sections. Section II discusses the proposed methodology and time synchronization in AWN, followed by results, and implementation in Section III and conclusion in Section IV.

II PROPOSED SESSION BASE KEYS FOR RSA AND TIME SYNCHRONISATION IN AWN

Rivest, Shamir, Adleman (RSA) algorithm was developed in 1977 [5]. The main operation of RSA involves selection of two random prime numbers (p and q) and the modular exponentiation. For a secure RSA the numbers must be large, of the order of 512-bits (155 decimal digits) or more. From the prime numbers, modulus $n = p*q$ and Euler's Totient Function $\Phi = (p-1) * (q-1)$ are calculated. A number 'e' (public exponent) between 3 and n-1 is then selected. 'e' should be prime to p-1 and q-1, in other words $\gcd(e,\Phi)=1$. A number d, termed as the private exponent between 1 and Φ

(1<d<Φ), is then selected such that e*d mod Φ=1. The public encryption key is then (n,e) and the private key is (n,d). The cipher text c of message m is then c=me mod n and the message is decrypted back from the cipher text by m=cd mod n.

Modifications to the RSA algorithm have been proposed by researchers. In [5], the authors have proposed a loop based key generation algorithm (i-RSA). The modification proposed in [6] uses three prime numbers p, q, r to generate the public and private key. The values of p, q, r, N1 (modulus), E1 (public key index), D1 (private key index) and Φ(N) are then stored in two tables offline before starting the algorithm. This helps in speeding up the encryption and decryption process. Another approach given in [7] proposes to use 'n' prime numbers. The authors of [8] have proposed to randomly generate the value of e (public key exponent). A new encryption function as c=e^m mod n has been proposed and the decrypted value are arrived at by taking the logarithm m_e=log_e (c)mod n. In [9], the authors have proposed a method of eliminating the need to transfer n as part of the public key.

A Proposed Approach using Session Based Keys.

The concept of using session based keys for RSA, using the aircraft clock, and using the concept of S-box, has not been carried out earlier. In order to use RSA for airborne applications, a concept of generating p and q based on aircraft system clock and without the need to share the public key (n, e), is proposed in this paper. Concept of S-box (substitution-box) used in symmetric key algorithms has been used to generate the prime numbers p and q. S-box is used to perform substitution in symmetric encryption. The AES algorithm uses the concept of polynomial multiplication modulo using an irreducible polynomial of degree 8 to generate the S-box. The concept of generating the S-box in the proposed algorithm is similar to the one in AES encryption standard. The proposed modification in this paper are as follows

1) Use of session-based keys.
2) The prime numbers p and q are generated using S-box and inverse S-box. Both S-box and inverse S-box will be varied for each session and therefore p and q are varied for each session, based on aircraft system clock. The session times have been randomly taken for different sessions.
3) In order to have large values for p and q, a new technique is proposed.

The public key is not shared between the sender and recipient LRUs, so, it reduces the chances of any hacker obtaining n and thus, obtaining the value of p and q. As brought out earlier, the key in RSA, needs to be sufficiently large because if n is factorized, the attacker can arrive at the private key to decrypt the message. The keys used in RSA are in the order of 1024/2048/4096 bits. The pair of primes (p and q) in each case is of the order of 512/1024/2048 bits. The proposed methodology to generate p and q is shown in Fig.1.

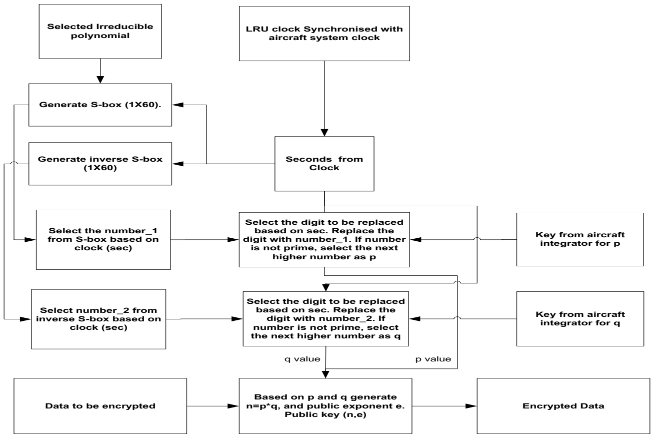
Selected Irreducible polynomial
LRU clock Synchronised with aircraft system clock
Generate S-box (1X60).
Generate inverse S-box (1X60)
Seconds from Clock
Select the number_1 from S-box based on clock (sec)
Select the digit to be replaced based on sec. Replace the digit with number_1. If number is not prime, select the next higher number as p
Key from aircraft integrator for p
Select number_2 from inverse S-box based on clock (sec)
Select the digit to be replaced based on sec. Replace the digit with number_2. If number is not prime, select the next higher number as q
Key from aircraft integrator for q
q value
p value
Data to be encrypted
Based on p and q generate n=p*q, and public exponent e. Public key (n,e)
Encrypted Data

Fig. 1 Generating large values for p and q

The value of p and q can be as large as required depending on the size of the initial key to be shared by the aircraft integrator. This key would subsequently change during each session to generate a new value of p and q. Based on the session based keys, the proposed session based keys RSA approach for AWN is shown in Fig. 2.

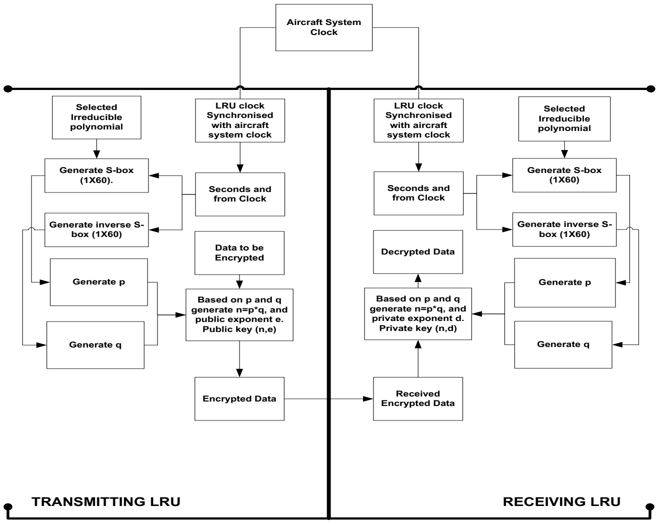
Aircraft System Clock
Selected Irreducible polynomial
LRU clock Synchronised with aircraft system clock
Generate S-box (1X60).
Seconds and from Clock
Generate inverse S-box (1X60)
Data to be Encrypted
Generate p
Based on p and q generate n=p*q, and public exponent e. Public key (n,e)
Generate q
Encrypted Data
LRU clock Synchronised with aircraft system clock
Selected Irreducible polynomial
Seconds and from Clock
Generate S-box (1X60)
Generate inverse S-box (1X60)
Decrypted Data
Generate p
Based on p and q generate n=p*q, and private exponent d. Private key (n,d)
Generate q
Received Encrypted Data
TRANSMITTING LRU
RECEIVING LRU

Fig.2 Proposed approach for airborne application

The steps involved are as follows.

Step 1: Generation of S-box (1x60) and inverse S-box (1x60): Two approaches are proposed for generating the S-box and inverse S-box.

(i) **Approach 1**: An irreducible polynomial of degree 8 specified in AES standard ('100011011') is used to generate the S-box and corresponding inverse S-box. In this case, the S-box remains same for all the sessions, but the prime numbers p and q keep varying based on the time (sec).

(ii) **Approach 2**: To improve the robustness of the modification, the second modification proposes to use 30 different possible irreducible polynomials of degree 8 [10]. Based on {(seconds) mod 30}, one out of the 30 polynomials are selected. In this proposal, the S-box keeps varying for each session, based on the selected irreducible polynomial. The prime numbers p and q are generated using the new S-box for each session.

Step 2: Generation of p and q

1) Based on the seconds from the clock, the value of number_1 (ran_num_p) is selected from the S-box. Similarly, the value of number_2 (ran_num_q) is selected from the inverse S-box, based on the seconds.

2) Based on the seconds the digit to be replaced in the shared p and q keys (from aircraft integrator) is selected.

(i) If number of digits > sec (from the clock), the value of digit in the key is replaced by number_1 (ran_num_p) in the p key and number_2 (ran_num_q) in the q key.

(ii) If number of digits < sec (from the clock), the digit to be replaced is arrived at by calculating {mod (sec, number of digits) +1} and the value of the digit is replaced by number_1 (ran_num_p) in the p key and number_2 (ran_num_q) in the q key.

If the numbers obtained by replacing the digits, is not a prime number the next prime number, greater than the selected value is taken as the values of p and q. The size of p and q is likely to increase with each session, so there may be a requirement to limit the size of the value of p and q beyond a fixed number of digits.

Step 3: The values of p and q are used for the current session. Based on p and q, values of modulus, public key is derived. The remaining encryption follows the standard RSA algorithm.

Step 4: At the receiving LRU, steps 1 and 2 are repeated for generating p and q, based on the system clock. The private key (n, d) at the receiver is derived from the generated p and q. Since p and q values generated at the transmitter and receiver are the same, there is no requirement to share the public key (n, e).

B Time Synchronization in AWN

The proposed modification to RSA, in this paper, is based on time synchronization of LRUs with the aircraft system clock. Time Synchronization between different nodes in a distributed network is crucial for sharing of resources, exchange of information, handling security attacks, error logging, event logging etc. Synchronization in the network can use an external clock synchronization, in which an external reference is used for synchronizing the nodes or through internal synchronization, in which the communicating nodes share their clock times to synchronize. Internal synchronization, in turn can be achieved through centralized clock synchronization or distributed clock synchronization. An aircraft has a heterogeneous distributed system. In such a scenario, time synchronization between multiple on-board systems becomes more critical to ensure coordination of tasks between the LRUs like auto pilot units for navigation, cryptographic schemes in military aircraft, sensor data fusion between multiple sensors like radar, Infrared Sensors etc. also require precise synchronization. An aircraft architecture usually follows a centralized synchronization system, wherein all the units are synchronized to a central server, usually the Mission Computer (MC). The central time server may be an active server wherein the server keeps broadcasting the time at periodic interval as in Berkeley algorithm or a passive server, which responds only when interrogated, as in the case of Cristian algorithm. Three common approaches used for time synchronization in wireless networks are Reference Broadcast Synchronization (RBS) [11], Timing-sync Protocol for Sensor Networks (TPSN) [12], Flooding Time Synchronization Protocol (FTSP) [13]). FTSP is most suitable for AWN because it eliminates all the synchronization errors discussed earlier, except the propagation error.

III Results and Implementation

The results obtained for the proposed approaches are presented in two parts in this section.

1) Generation of S-box and inverse S-box, using the two proposed modifications (Modification 1 and Modification 2).The value of public exponent e is taken to be 65537 for both the modifications [14], [15]

2) Generation of large values of p and q from the S-box and inverse S-box for each session

A Generation of S-box and inverse S-box

1) Approach 1. The irreducible polynomial used for Approach 1 is '100011011' (polynomial of degree 8 specified in AES standard) and the generated S-box and inverse S-box are shown in Table 1 and 2. S-box and inverse S-box remains same for all the sessions. For this session, time is taken as **12** (Seconds).

At **12th** column of S-box (Table 1), value is 183, which is taken as the value of **ran_num_p**. Similarly, at **12th** column of inverse S-box (Table 2), the value is 133, which is taken as value of **ran_num_q**. Similarly, at **12th** column of inverse S-box (Table 2), the value is 133, which is taken as value of **ran_num_q**.

TABLE 1
GENERATED S-BOX FOR RSA (MODIFICATION 1)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
255	224	235	231	110	247	243	89	172	157	251	183	98	75
15	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
55	234	86	30	85	225	102	197	219	108	49	72	62	51
29	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
30	56	238	92	43	97	15	186	170	163	107	80	168	57

43	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
121	109	237	68	173	137	152	91	191	95	132	10	153	6
57	**58**	**59**	**60**										
155	142	28	126										

TABLE 2
GENERATED INVERSE S-BOX FOR RSA (MODIFICATION 1)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
28	117	223	110	226	249	55	232	231	173	53	133	150	172
15	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
116	34	240	180	230	115	151	242	207	206	79	103	220	234
29	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
58	145	17	65	120	205	90	244	154	219	192	254	198	210
43	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
121	32	252	86	62	75	170	24	190	27	111	183	98	14
57	**58**	**59**	**60**										
29	41	197	137										

These numbers are then used to generate the values of p and q based on the methodology described in Step 2 of the proposed algorithm. Similarly, for any session based on time available from aircraft clock, the corresponding values of ran_num_p and ran_num_q are selected from the S-box and inverse S-box to generate p and q. So, p and q keep changing for each session.

2) Approach 2. In this approach, the irreducible polynomial is selected, based on session time, from a list of 30 irreducible polynomials (Forouzan & Mukhopadhyay, 2015). The selected polynomial is used to generate the S-box and corresponding inverse S-box. These are changed for each session. Using the new S-box values of ran_num_p and ran_num_q are selected to generate p and q, which is also based on time. So for each session, S-box and inverse S-box keeps changing and in turn values of ran_num_p and ran_num_q also changes. Two sessions for Approach 2 are discussed here.

(iii) Session 1. For this session, time is taken as **48** (Seconds). Based on this time, the Selected Irreducible Polynomial is '**110100011**'. S-box and inverse S-box generated using this polynomial (for this session) is shown in Table 3 and 4. Based on the time (48 Sec), from **48th** column of S-box (Table 3) 97 is taken as the value of **ran_num_p** to generate p and from **48th** column of inverse S-box (Table 4) 121 is taken as value of **ran_num_q** to generate q.

TABLE 3
GENERATED S-BOX FOR RSA (MODIFICATION 2, SESSION 1)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
99	*124*	*197*	*167*	*25*	*17*	*1*	*23*	*119*	*79*	*115*	*24*	*123*	*243*
15	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
89	*178*	*64*	*44*	*117*	*112*	*107*	*254*	*247*	*19*	*111*	*172*	*2*	*73*
29	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
87	*74*	*162*	*144*	*219*	*50*	*237*	*175*	*65*	*122*	*234*	*181*	*103*	*120*
43	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
173	*217*	*41*	*180*	*114*	*97*	*76*	*182*	*132*	*104*	*250*	*163*	*95*	*141*
57	**58**	**59**	**60**										
121	*130*	*222*	*43*										

TABLE 4
GENERATED INVERSE S-BOX FOR RSA (MODIFICATION 2, SESSION 1)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
235	*6*	*26*	*195*	*84*	*70*	*234*	*165*	*78*	*81*	*219*	*166*	*169*	*153*
15	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
184	*179*	*215*	*5*	*183*	*23*	*227*	*255*	*196*	*7*	*11*	*4*	*130*	*228*
29	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
254	*142*	*87*	*185*	*207*	*79*	*244*	*115*	*68*	*148*	*63*	*200*	*253*	*44*
43	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
106	*59*	*17*	*155*	*170*	*121*	*135*	*223*	*33*	*150*	*248*	*198*	*229*	*132*
57	**58**	**59**	**60**										
225	*67*	*238*	*213*										

(iv) Session 2

For this session, time is taken as **59** (Seconds). Based on this time, the Selected Irreducible Polynomial is '111111001'. S-box and inverse S-box generated for this polynomial is shown in Table 5 and 6. Based on the time, from **59th** column of S-box(Table 5) 109 is taken as the value of **ran_num_p** and from **59th** column of inverse S-box(Table 6), the value is 33, which is taken as value of **ran_num_q**. These values are used to generate p and q.

TABLE 5

GENERATED S-BOX FOR RSA (MODIFICATION 2, SESSION 2)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
99	*124*	*189*	*247*	*12*	*142*	*41*	*92*	*212*	*137*	*196*	*160*	*70*	*105*
15	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
252	*199*	*233*	*97*	*22*	*194*	*225*	*106*	*211*	*154*	*241*	*254*	*102*	*91*
29	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
172	*221*	*96*	*136*	*119*	*58*	*98*	*122*	*217*	*216*	*179*	*9*	*115*	*15*
43	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
231	*118*	*59*	*21*	*159*	*104*	*123*	*143*	*173*	*3*	*176*	*157*	*46*	*64*
57	**58**	**59**	**60**										
132	*203*	*109*	*11*										

TABLE 6

GENERATED INVERSE S-BOX FOR RSA (MODIFICATION 2, SESSION 2)

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
99	*124*	*189*	*247*	*12*	*142*	*41*	*92*	*212*	*137*	*196*	*160*	*70*	*105*
15	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
252	*199*	*233*	*97*	*22*	*194*	*225*	*106*	*211*	*154*	*241*	*254*	*102*	*91*
29	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
172	*221*	*96*	*136*	*119*	*58*	*98*	*122*	*217*	*216*	*179*	*9*	*115*	*15*
43	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
231	*118*	*59*	*21*	*159*	*104*	*123*	*143*	*173*	*3*	*176*	*157*	*46*	*64*
57	**58**	**59**	**60**										
132	*203*	*109*	*11*										

B Generation of large values of p and q for larger keys

The numbers selected for p and q in above discussed sessions are small, due to limitations of the system used for testing the two approaches. In order to meet the requirement of large values of p and q (512/1024/2048 bits), a methodology to generate larger numbers for p and q(for two sessions) is proposed in the subsequent paragraphs. **ran_num_p** and **ran_num_q** from S-box and inverse S-box, discussed earlier in the two approaches, are used to generate large values of p and q. The size of p and q (shared by the aircraft integrator) defines the size of the keys. p and q shared by the integrator is modified and further enhanced by the new technique and keeps varying for each session. For the present two sessions, S-box (Table 1) and inverse S-box (Table 2)) of Approach 1 are used. For both the sessions, it is assumed that keys shared by the aircraft integrator are p_genr=1234567891098, q_genr=1098765432110.

If Session time=**50** (Seconds), the number of digits in p_genr is 13 and in q_genr also it is 13. The numbers selected from S-box (Table 1) and inverse S-box (Table 2), based on t=**50** sec are **ran_num_p**=91 and **ran_num_q**= 24. The digit to be replaced based on{mod (50, No. of digits (13)) +1}is the 12th digit. So, 12th digit in p_genr and q_genr is replaced with 91 and 24. So the new numbers are p_init = 12345678910918, q_init=10987654321240. The numbers p_init and q_init are not prime numbers, so the next higher prime number is selected as p and q. Therefore, p= 12345678910927 and q=10987654321253. The results of the implementation are shown in Table 7.

TABLE 7

GENERATION OF P AND Q FOR T = 50 SEC

Time (Sec)	50
p_genr	1234567891098
q_genr	1098765432110
ran_num_p	*91*
ran_num_q	*24*
p_init	12345678910918
q_init	10987654321240
p	12345678910927
q	10987654321253

When Session time =**5** (Sec), the number of digits are same i.e., in p_genr it is 13 and in q_genr also it is 13. The numbers selected from S-box (Table 1) and inverse S-box (Table 2) based on t=5 sec are **ran_num_p**=110, **ran_num_q**= 226. Since the time 5 sec < number of digits (13), 5th digit is to be replaced. Replacing the 5th digit by 110 and 226 in p_genr and q_genr gives p_init = 123411067891098 and q_init=109822665432110. The new numbers p_init and q_init are not prime numbers, so the next higher prime number is selected as p and q i.e., p= 123411067891117 and q=109822665432167. The results of session 2 are shown in Table 8.

TABLE 8

GENERATION OF P AND Q FOR T = 5 SEC

Time (Sec)	5
p_genr	1234567891098
q_genr	1098765432110
ran_num_p	*110*
ran_num_q	*226*

p_init	123411067891098
q_init	109822665432110
p	23411067891117
q	109822665432167

C Implementation of RSA Algorithm in LRUs

One of the issues in a secured wireless network is implementing the algorithm in the existing LRUs. The proposed modified RSA algorithm in LRUs can be implemented either as software or as hardware. Both have their own advantages and disadvantages. In case of software, the advantages are flexibility, ease of implementation, ease of upgrade or modifications. However, speed and cost are two disadvantages associated with software implementation of security algorithm. In case of hardware, the advantages are speed and security apart from the cost factor. Utilizing the LRUs primary processor (in software implementation), for carrying out encryption and decryption is an inefficient way of implementing the security algorithm because these algorithms involve complicated operations and would result in the processor being busy for a longer duration. Therefore, having an additional hardware would enable faster encryption and decryption. A separate hardware can provide physical security from anyone trying to access the algorithm. Self-destruction of the code on the separate hardware can prevent tampering of the code. Therefore, hardware implementation is usually the preferred mode in critical military applications. In case of aerospace applications also secured data links in military aircraft have resorted to separate hardware for encrypting the data and voice between aircraft. Hardware implementation of RSA is discussed in [17], [18], [19]. The existing LRUs can be modified with the addition of an encryption/decryption module. The encryption/decryption module can be installed as part of the module required for converting the existing LRUs to a wireless enabled LRU. Another issue in the proposed modified RSA algorithm is the execution time. In order to reduce the execution time, the S-boxes and inverse S-boxes can be pre-loaded in FPGA memory or in a separate memory. Based on the time, an index can be passed to the FPGA to fetch the relevant S-box and inverse S-box. This will reduce the time involved in generation of tables and in turn reduce the overall execution time.

IV CONCLUSION

This paper has brought out the important aspect of security in AWN. Modified RSA algorithm based on aircraft clock has been proposed in the paper. Session based asymmetric encryption can prevent replay attacks, looping attack, ARP spoofing and MAC spoofing attacks, Eavesdropping, Man in the Middle attack etc. The mandatory requirement for session based encryption is precise time synchronization between the two communicating LRUs, which can be achieved using FTSP method. The modification will remove the requirement of sharing of public key between the transmitting and receiving LRUs. This in turn will prevent any attacker from gaining access to the keys and the network. The proposed modification uses the concept of S-box and inverse S-box to generate large keys, which is a primary requirement of RSA, for encryption and decryption. Two concepts have been proposed, one using a fixed S-box and inverse S-box and the second using a different S-box and inverse S-box, for each session. The execution time for the modified RSA was found to be slightly more than the standard algorithm, due to the additional computations involved. However, the increase in execution time provides more robustness to the algorithm. A suitable method for reducing the execution time is to store the S-boxes and inverse S-boxes in the LRU memory (FPGA/separate memory).

REFERENCES

[1] H. Slutsken, "When Managing Aircraft Weight Comes Down to the Wire," Apex Experience Apr 2017.

[2] AVSI, "WAIC, Update and Status," ICAO Regional WRC-15 Preparatory Workshop, Cairo, 2015.

[3] C. Biesecker, "Boeing 757 Testing Shows Airplanes Vulnerable to Hacking, DHS Says," *Avionics International,* 8 Nov 2017.

[4] R. N. Akramnd, K. Markantonakis, S. Kariyawasam, S. Ayub, A. Seeam and R. Atkinson, "Challenges of security and trust in Avionics Wireless Networks," in *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, Prague, 2015.

[5] R. L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM,* vol. 21, no. 2, p. 120–126, Feb 1978.

[6] N. Muhammad, J. M. Zain and M. Y. M. Saman, "Loop-based RSA key generation algorithm using string identity," in *13th International Conference on Control, Automation and Systems (ICCAS 2013)*, Gwangju, Korea (South), 2013.

[7] R. Patidar and R. Bhartiya, "Modified RSA Cryptosystem Based on Offline Storage and Prime Number," in *2013 IEEE International Conference on Computational Intelligence and Computing Research*, Enathi, 2013.

[8] M. Islam, M. Islam, N. Islam and B. Shabnam, "A Modified and Secured RSA Public Key Cryptosystem Based on “n” Prime Numbers," *Journal of Computer and Communications,* pp. 78-90, 2018.

[9] M. Frunza and L. Scripcariu, "Improved RSA Encryption Algorithm for Increased Security of Wireless Networks," in *2007 International Symposium on Signals, Circuits and Systems*, Iasi, Romania, 2007.

[10] A. Chhabra and S. Mathur, "Modified RSA Algorithm: A Secure Approach," in *2011 International Conference on Computational Intelligence and Communication Networks*, Gwalior,M.P, 2011.

[11] B. A. Forouzan and D. Mukhopadhyay, Cryptography and Network Security, New Delhi: McGraw Hill Publicationn (India) Pvt Ltd, 2015.

[12] J. Elson, L. Girod and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.,* vol. 36, p. 147–163, Dec 2003.

[13] S. Ganeriwal, R. Kumar and M. B. Srivastava, "Timing-sync Protocol for Sensor Networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, Los Angeles, California, USA, 2003.

[14] M. Maroti, B. Kusy, G. Simon and A. Ledeczi, "The Flooding Time Synchronization Protocol," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, Baltimore, MD, USA, 2004.

[15] J. S. Kraft and L. C. Washington, An Introduction to Number Theory with Cryptography, 2nd ed., CRC Press, 2016.

[16] N. Heninger and H. Shacham, "Reconstructing RSA Private Keys

from Random Key Bits," in *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference*, Santa Barbara, CA, 2009.

[17] S. D. Thabah, M. Sonowal, R. U. Ahmed and P. Saha, "Fast and Area Efficient Implementation of RSA Algorithm," in *Procedia Computer Science*, 2019.

[18] A. P. Fournaris and O. Koufopavlou, "A new RSA encryption architecture and hardware implementation based on optimized Montgomery multiplication," in *2005 IEEE International Symposium on Circuits and Systems*, Kobe, Japan, 2005.

[19] M. Rahman, I. R. Rokon and d. M. Rahman, "Efficient hardware implementation of RSA cryptography," in *International Conference on Anti-counterfeiting, Security, and Identification in Communication*, Hong Kong, China, 2009.