A Novel Method For Preventing Jamming Attacks Using Random Padding

[1]B.R.S.S.Raju, [2]A.Vasudeva Rao.
[1] M.Tech II Year , DADI INSTITUTE OF ENGINEERING & TECHNOLOGY
2 ASSOCIATE PROFESSOR, DEPT OF CSE, DADI INSTITUTE OF ENGINEERING & TECHNOLOGY

ABSTRACT:

Most of the wireless network face the problem of intentional interference attacks, that is typically referred as jamming. This intentional interference with wireless transmissions can be used as a platform for mounting Denial-of-Service attacks on wireless networks. Most of the jamming attacks addressed external thread model. In this work, we addressed problem of selective jamming attack in wireless network. In these attacks, the adversary is active only for a short period of time, selectively targeting messages of high importance. To mitigate these attacks, we develop Robin signature schemes that prevent real-time packet classification by combining cryptographic primitive with physical-layer attributes. We analyze the security of our methods and evaluate their computational and communication overhead.

1. INTRODUCTION

WIRELESS networks rely on the uninterrupted availability of the wireless medium to interconnect participating nodes. However, the open nature of this medium leaves it vulnerable to multiple security threats. Anyone with a transceiver can eavesdrop on wireless transmissions, inject spurious messages, or jam legitimate ones. While eavesdropping and message injection can be prevented using cryptographic methods, jamming attacks are much harder to counter. They have been shown to actualize severe Denial-of-Service (DoS) attacks against wireless networks. In the simplest form of jamming, the adversary interferes with the reception of messages by transmitting a continuous jamming signal, or several short jamming pulses. Typically, jamming attacks have been considered under an external threat model, in which the jammer is not part of the network. To prevent the jamming attack we are propose the following technique.

2 Related work:

The proposed system is used to prevent selective jamming attack in wire less network. This can be overcome by using Rabin signature algorithm for preventing jamming attacks. Before performing the Rabin signature we are generating random padding using diffie hellman key exchange algoarithm. Because sender and receiver have same type of random padding. So that the generation of random padding has follows..

3 Radom padding Generation:

1. Sender and verifier agree to use a prime number p and base g.
2. sender chooses a secret integer a, then sends verifier $A = g^a \bmod p$.
3. verifier chooses a secret integer b, then sends sender $B = g^b \bmod p$
4. sender computes $s = B^a \bmod p$
5. verifier computes $s = A^b \bmod p$
6. now the sender and the receiver share same random padding .

After generation of random padding the sender generate signature by using Rabin signature algorithm. The generation signature as follows.

Key Generation :

- The signer S chooses primes p,q each of size

approximately *k/2* bits, and computes the product $n = pq$

- *S* then chooses a random *b* in $\{1, \ldots, n\}$.
- The public key is *(n,b)*
- The private key is *(p,q)*

4 Encryption and decryption of Message :

Before sending the message the sender will encrypt the message using cryptography techinique. In this paper the DES algorithm is using for message encryption and decryption purpose. The procedure of Des algorithm as follows.

After generation of secret key to encrypt the message using this key. The encryption and decryption of message can be done by using DES algorithm.

DES (and most of the other major symmetric ciphers) is based on a cipher known as the **Feistel block cipher**. where each round contains bit-shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. Most symmetric encryption schemes today are based on this structure (known as a **feistel network**). As with most encryption schemes, DES expects two inputs - the plaintext to be encrypted and the secret key. The manner in which the plaintext is accepted, and the key arrangement used for encryption and decryption, both determine the type of cipher it is. DES is therefore a symmetric, 64 bit **block cipher** as it uses the same key for both encryption and decryption and only operates on 64 bit blocks of data at a time5 (be they plaintext or ciphertext). The key size used is 56 bits, however a 64 bit (or eight-byte) key is actually input. The least significant bit of each byte is either used for parity (odd for DES) or set arbitrarily and does not increase the security in any way. All blocks are numbered from left to right which makes the eight bit of each byte the parity bit. Once a plain-text message is received to be encrypted, it is
arranged into 64 bit blocks required for input. If the number of bits in the message is not evenly divisible by 64, then the last block will be padded. Multiple
permutations and substitutions are incorporated throughout in order to increase the difficulty of performing a cryptanalysis on the cipher. However, it is generally accepted that the initial and final permutations offer little or no contribution to the security of DES and in fact some software implementations omit them (although strictly speaking these are not DES as they do not adhere to the standard).

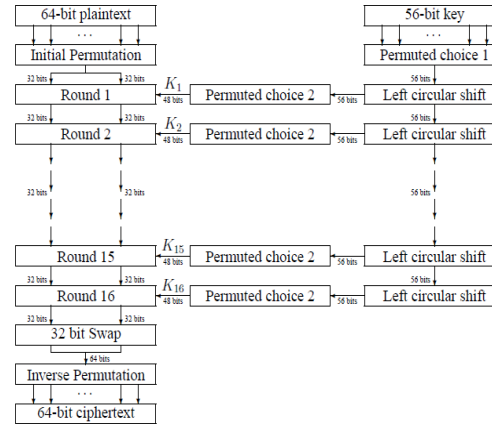

Figure 2.2: Flow Diagram of DES algorithm for encrypting data.

Figure 2.2 shows the sequence of events that occur during an encryption operation.DES performs an initial permutation on the entire 64 bit block of data. It is then split into 2, 32 bit sub-blocks, Li and Ri which are then passed into what is known as a **round** (see figure 2.3), of which there are 16 (the subscript i in Li and Ri indicates the current round). Each of the rounds are identical and the effects of increasing their number is twofold - the algorithms security is increased and its temporal efficiency decreased. Clearly these are two conflicting outcomes and a compromise must be made. For DES the number chosen was 16, probably to guarantee

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Permuted Choice One (PC-1)

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(d) Schedule of Left Shifts

Round number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Table 2.2: DES key schedule.

the elimination of any correlation between the ciphertext and either the plaintext or key6. At the end of the 16th round, the 32 bit Li and Ri output quantities are swapped to create what is known as the **pre-output**. This [R16, L16] concatenation is permuted using a function which is the exact inverse of the initial permutation. The output of this final permutation is the 64 bit cipher text.

So in total the processing of the plaintext proceeds in three phases as can be seen from the left hand side of figure 2.2:
1. Initial permutation (**IP** - defined in table 2.1) rearranging the bits to form the "permuted input".
2. Followed by 16 iterations of the same function (substitution and permutation). The output of the last iteration consists of 64 bits which is a function of the
plaintext and key. The left and right halves are swapped to produce the preoutput.
3. Finally, the preoutput is passed through a permutation (**IP**−1 - defined in table 2.1) which is simply the inverse of the initial permutation (**IP**). The output of**IP**−1 is the 64-bit cipher text**.**

(a) Initial Permutation (IP)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

(b) Inverse Initial Permutation (IP^{-1})

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

(c) Expansion Permutation (E)

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

(d) Permutation Function (P)

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Table 2.1: Permutation tables used in DES.

As figure 2.2 shows, the inputs to each round consist of the Li,Ri pair and a 48 bit **subkey** which is a shifted and contracted version of the original 56 bit key. The use of the key can be seen in the right hand portion of figure 2.2:

• Initially the key is passed through a permutation function (**PC**1 - defined in table 2.2)
• For each of the 16 iterations, a subkey (**K**i) is produced by a combination of a left circular shift and a permutation (**PC**2 - defined in table 2.2) which is the same for each iteration. However, the resulting sub key is different for each iteration because of repeated shifts.

5 Details of individual rounds

Details of an individual round can be seen in figure 2.3. The main operations on the data are encompassed into what is referred to as the **cipher function** and is labeled ***F***. This function accepts two different length inputs of 32 bits and 48 bits and outputs a single 32 bit number. Both the data and key are operated on in parallel, however the operations are quite different. The 56 bit key is split into two 28 bit halves Ci and Di (C and D being chosen so as not to be confused with L and R). The value of the key used in any round is simply a left cyclic shift and a

permuted contraction of that used in the previous round. Mathematically, this can be written as

$$Ci = Lcsi(Ci-1), Di = Lcsi(Di-1) \quad (2.1)$$
$$Ki = PC2(Ci,Di) \quad (2.2)$$

where Lcsi is the left cyclic shift for round i, Ci and Di are the outputs after the shifts,PC2(.) is a function which permutes and compresses a 56 bit number into a 48 bit number and Ki is the actual key used in round i. The number of shifts is either one or two and is determined by the round number i. For i = {1, 2, 9, 16} the number of shifts is one and for every other round it is two (table 2.2).

The common formulas used to describe the relationships between the input to one round and its output (or the input to the next round) are:

$$Li = Ri-1 \quad (2.3)$$

$$Ri = Li-1 \_ \boldsymbol{F}(Ri-1,Ki) \quad (2.4)$$

where L and R have their usual meaning and ***F***(.) is the cipher function. This function ***F*** is the main part of every round and consists of four separate stages (see figure 2.4):

1. The E-box expansion permutation - here the 32-bit input data from Ri−1 is expanded and permuted to give the 48 bits necessary for combination with the
48 bit key (defined in table 2.1). The E-box expansion permutation delivers a larger output by splitting its input into 8, 4-bit blocks and copying every first and
fourth bit in each block into the output in a defined manner. The security offered by this

operation comes from one bit affecting two substitutions in the S-boxes.
This causes the dependency of the output bits on the input bits to spread faster,
and is known as the avalanche affect.

2. The bit by bit addition modulo 2 (or exclusive OR) of the E-box output and 48 bit sub key Ki.

3. The S-box substitution - this is a highly important substitution which accepts a 48-bit input and outputs a 32-bit number (defined in table 2.3). The S-boxes are the only non-linear operation in DES and are therefore the most important part of its security. They were very carefully designed although the conditions they were designed under has been under intense scrutiny since DES was released.
The input to the S-boxes is 48 bits long arranged into 8, 6 bit blocks (b1, b2, . . . , b6). There are 8 S-boxes (S1, S2, . . . , S8) each of which accepts one of the 6 bit blocks. The output of each S-box is a four bit number. Each of the S-boxes can be thought of as a 4 × 16 matrix. Each cell of the matrix is identified by a coordinate pair (i, j), where 0 _ i _ 3 and 0 _ j _ 15. The value of i is taken as the decimal representation of the first and last bits of the input to each S-box, i.e. Dec(b1b6) = i and the value of j is take from the decimal representation of the inner four bits that remain, i.e. Dec(b2b3b4b5) = j. Each cell within the S-box matrices contains a 4-bit number which is output once that particular cell is selected by the input.

4. The P-box permutation - This simply permutes the output of the S-box without changing the size of the data (defined in table 2.1). It is simply a permutation
and nothing else. It has a one to one mapping of its input to its output giving a 32 bit output from a 32 bit input.

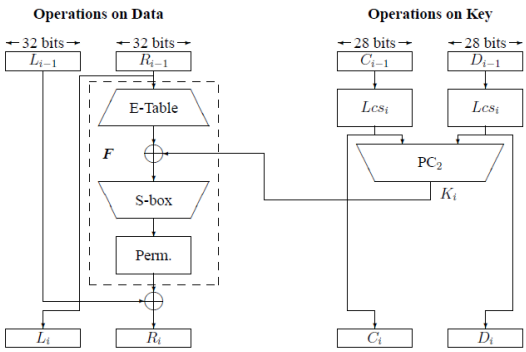

Figure 2.3: Details of a single DES round.

Other points of note
Having looked at DES in some detail a brief look at some other points is in order These include decryption, modes of operation, security etc.

5.1 Modes of operation
The DES algorithm is a basic building block for providing data security. To apply DES in a variety of applications, five modes of operation have been defined which cover virtually all variation of use of the algorithm and these are shown in table 2.4.

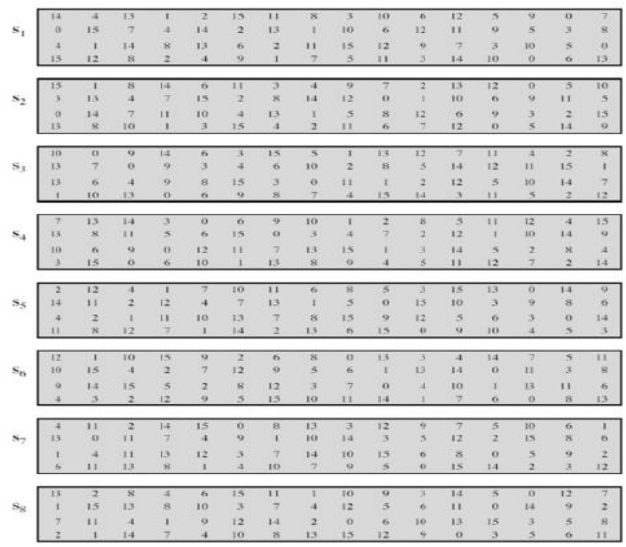

Table 2.3: S-box details.

5.2 DES decryption
The decryption process with DES is essentially the same as the encryption process and is as follows:
• Use the ciphertext as the input to the DES algorithm but *use the keys* Ki *in reverse order*. That is, use K16 on the first iteration, K15 on the second until K1 which is used on the 16th and last iteration

6 Signing:

- To sign a message *m* the signer *S* picks random padding *U* and calculates *H(mU)*
  - *S* then solves
    $$x(x+b) = H(mU) \mod n$$

- If there is no solution *S* picks a new pad *U* and tries again. If *H* is truly random the expected number of tries is 4.
- The signature on *m* is the pair *(U,x)*

Verification:

- Given a message *m* and a signature *(U,x)* send to the verifier.

The verifier *V* calculates *x(x+b)* and *H(mU)* and verifies that they are equal then retrieve the packet . If not equal discard packets sended by sender to the verifier. This way we can preventing the selective jamming attack.

6.1 DETECTION OF JAMMING

The network employs a monitoring mechanism for detecting potential malicious activity by a jammer. The monitoring mechanism consists of the following: (i) determination of a subset of nodes M that will act as network monitors, and (ii) employment of a detection algorithm at each monitor node. The assignment of the role of monitor to a node can be affected by energy limitations and detection performance specifications. In this work, we fix M and formulate optimization problems for one or more monitor nodes. We now fix attention to detection at one monitor node. First, we define the quantity to be observed at each monitor node. In our case, the readily available metric is probability of collision that a monitor node experiences, namely the percentage of packets that are erroneously received. During normal network operation, and in the absence of a jammer, we consider a large enough training period in which the monitor node "learns" the percentage of collisions it experiences as the long-term average of the ratio of number of slots in which there was a collision over total number of slots of the training period. Assume now the network operates in the open after the training period and fix attention to a time window much smaller than the training period. An increased percentage of collisions over this time window compared to the learned long-term average may be an indication of an ongoing jamming attack or only a temporary increase of percentage of collisions compared to the average during normal network operation. A detection algorithm takes observation samples obtained at the monitor node (i.e, collision or not collision) and decides whether there exists an attack. On one hand, the observation window should be small enough, such that the attack is detected on time and appropriate countermeasures are initiated. On the other hand, this window should be sufficiently large, such that the chance of a false alarm notification is minimized

6.2. PREVENTION OF JAMMING

A. Rate Adaptation Scheme

Most of widely used jamming attack solutions have some limitations. Those solutions use spatial or spectrum diversity to cope with the jamming attack. These schemes do not utilize the jammed channels, though they have enough bandwidth for the data transmission. Rate adaptation scheme to overcome problems in previous works. The most important goal of the proposed scheme is to achieve high link utilization by adjusting the transmission mode based on the expected maximum throughput. The expected maximum throughput must consider the successful transmission probability. Suppose that L is the length of data frame and Tm is the transmission time of data frame in a specific transmission mode, m. Each transmission mode specifies the transmission rate appropriately adapted to network condition. The successful transmission probability can be calculated using error probabilities for a data frame and ACK frame. An ACK frame which is usually much shorter than the data frame is transmitted at the rate equal to or lowers than the data frame rate. Therefore, the error probability of the ACK frame is much lower than that of the data frame. Hence we can approximate the successful transmission probability. The error probability for a data frame can be calculated using error probability of the PLCP (Physical Layer Convergence Procedure) scheme selects the transmission mode based on the expected maximum throughput. Each node is able to calculate the expected maximum throughput for each transmission mode m from a set of available transmission modes, M. Finally we can choose

the optimal transmission mode.

B. Mapping to Commitment Scheme for Selective
Jamming attack prevention

For Countering selective jamming, the goal of this
scheme is to transform a selective jammer to a random one. This can be achieved by overwhelming the adversary's computational ability to perform real-time packet classification. It first show that our problem can be mapped to the hiding property of commitment schemes Commitment schemes are fundamental cryptographic primitives that allow a committer P, commit to a value m to a verifier V while keeping m hidden. Initially, P provides V with a commitment C = commit(m, r), where commit is some commitment operation, and r is a random number. At a later stage, P can release additional information that reveals m. A scheme that does not allow the computation of m from C without additional information from P is called perfect or hiding, while a scheme that does not allow P to change m to a value m_ once C is released, is called binding. The role of the committee is assumed by the transmitting node S. The role of the verifier V is assumed by any receiver R within the communication range of S, including the jammer J.

Note that S has no particular interest in modifying m after he has committed to it, since its primary goal is to communicate m. However, satisfying the binding property ensures that, (a) only S can release information that reveals m, and (b) the only value
that R can accept is m. To prevent selective jamming, S first transmits C that hides m from any receiver, including J. Once the transmission of C is completed, S reveals additional information that "opens" C. Intended receivers are able to read m. We now provide a scheme that prevents packet classification based on the idea of commitments**.**

7 CONCLUSION :

We addressed the problem of selective jamming attacks in wireless networks. We considered an internal adversary model in which the jammer is part of the network under attack, thus being aware of the protocol specifications and shared network secrets. We showed that the jammer can classify transmitted packets in real time by decoding the first few symbols of an ongoing transmission. We evaluated the impact of selective jamming attacks on network protocols such as TCP and routing. Our findings show that a selective jammer can significantly impact performance with very low effort. We developed three schemes
that transform a selective jammer to a random one by preventing real-time packet classification. Our schemes combine cryptographic primitives such as commitment schemes, cryptographic puzzles, and all-or-nothing transformations (AONTs) with physical layer characteristics. We analyzed the security of our schemes and quantified their computational and communication overhead

8 REFERENCES

[1] T. X. Brown, J. E. James, and A. Sethi. Jamming and sensing of encrypted wireless ad hoc networks. In Proceedings of MobiHoc, pages 120–130, 2006..
[2] M. Wilhelm, I. Martinovic, J. Schmitt, and V. Lenders. Reactive jamming in wireless networks: How realistic is the threat? In Proceedings of WiSec, 2011.
[3] W. Xu, W. Trappe, Y. Zhang, and T.Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In Proceedings of MobiHoc, pages 46–57, 2005.
[4] IEEE. IEEE 802.11 standard. http://standards.ieee.org/getieee802/download/802.11-2007.pdf, 2007.
[5] Akyildiz, I. F., Wang, W., & Wang, W. (2005, January). Wireless mesh networks: a survey. Computer Networks Journal,47(4), 445-487.
[6] D. Stinson. Cryptography: theory and practice. CRC press, 2006.
[7] Eriksson, J. and Koivunen, V.: Identi¯ability, separability, and uniqueness of linear ica models. IEEE Signal Processing Letters, 11(7), July 2004.
[8] P. Tague, M. Li, and R. Poovendran. Mitigation of control channel jamming under node capture attacks. IEEE Transactions on Mobile Computing, 8(9):1221–1234, 2009.

Mr.B.R.S.S.Raju has completed his M.C.A in Noble Institute of Science & Technology, Visakhapatnam, Andhra University in 2009 and he is pursuing M.Tech (CSE) in Dadi Institute of Engineering Technology, Anakapalli, Vizag,JNTUK interested in Computer Networks.

Mr.A.Vasudeva Rao,currently working as an Associate Professor in CSE Department , in Dadi Institute of Engineering Technology, Anakapalli, with 8 years of experience.He completed M.Tech(Computer Science &Technology) from College of Engineering, Andhra Universitv 2008. His