Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 11, November 2013, pg.175 – 189

RESEARCH ARTICLE

Container-Hiding Methods for Preventing Discriminating Blocking Attacks

R. SAILAJA[1], G. VENKATAPRASAD[2], E. HARI PRASAD[3]

M.Tech CSE Student, Sphoorthy Engineering College, JNTU Hyderabad, Hyderabad, Andhra Pradesh, India

Assistant Professor, Department of CSE, Sphoorthy Engineering College, Hyderabad, Andhra Pradesh, India

Head of the Department of CSE & IT, Sphoorthy Engineering College, Hyderabad, Andhra Pradesh, India

sailuravuri@gmail.com

Abstract—The open nature of the wireless medium leaves it vulnerable to intentional interference attacks, typically referred to as Blocking. This intentional interference with wireless transmissions can be used as a Launchpad for mounting Denial-of-Service attacks on wireless networks. Typically, Blocking has been addressed under an external threat model. However, adversaries with internal knowledge of protocol specifications and network secrets can launch low-effort Blocking attacks that are difficult to detect and counter. In this work, we address the problem of discriminating Blocking attacks in wireless networks. In these attacks, the adversary is active only for a short period of time, discriminatingly targeting messages of high importance. We illustrate the advantages of discriminating Blocking in terms of network performance degradation and adversary effort by presenting two case studies; a discriminating attack on TCP and one on routing. We show that discriminating Blocking attacks can be launched by performing real-time Container classification at the physical layer. To mitigate these attacks, we develop three schemes that prevent real-time Container classification by combining cryptographic primitives with physical-layer attributes. We analyze the security of our methods and evaluate their computational and communication overhead.

Keywords—Discriminating Blocking; denial-of-service; wireless networks; Container classification

© 2013, IJCSMC All Rights Reserved

I. 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 eaves-dropping and message injection can be prevented using cryptographic methods, Blocking 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 Blocking, the adversary interferes with the reception of messages by transmitting a continuous Blocking signal, or several short Blocking pulses

Typically, Blocking attacks have been considered under an external threat model, in which the jammer is not part of the network. Under this model, Blocking strategies include the continuous or random transmission of high-power interference signals. However, adopting an "al-ways-on" strategy has several disadvantages. First, the adversary has to expend a significant amount of energy to jam frequency bands of interest. Second, the continuous presence of unusually high interference levels makes this type of attacks easy to detect

Conventional antiBlocking techniques rely extensively on spread-spectrum (SS) communications, or some form of Blocking evasion (e.g., slow frequency hopping, or spatial retreats). SS techniques provide bit-level protection by spreading bits according to a secret pseudonoise (PN) code, known only to the communicating parties. These methods can only protect wireless transmissions under the external threat model. Potential disclosure of secrets due to node compromise neutralizes the gains of SS. Broadcast communications are particularly vulnerable under an internal threat model because all intended receivers must be aware of the secrets used to protect transmissions. Hence, the compromise of a single receiver is sufficient to reveal relevant cryptographic information.

Address the problem of Blocking under an internal threat model. We consider a sophisticated adversary who is aware of network secrets and the implementation details of network protocols at any layer in the network stack. The adversary exploits his internal knowledge for launching discriminating Blocking attacks in which specific messages of "high importance" are targeted. For example, a jammer can target route-request/route-reply messages at the routing layer to prevent route discovery, or target TCP acknowledgments in a TCP session to severely degrade the throughput of an end-to-end flow.

To launch discriminating Blocking attacks, the adversary must be capable of implementing a "classify-then-jam" strategy before the completion of a wireless transmission. Such strategy can be actualized either by classifying transmitted Containers using protocol semantics [1], [33], or by decoding Containers on the fly [34]. In the latter method, the jammer may decode the first few bits of a Container for recovering useful Container identifiers such as Container type, source and destination address. After classification, the adversary must induce a sufficient number of bit errors so that the Container cannot be recovered at the receiver [34]. Discriminating Blocking requires an intimate knowledge of the physical (PHY) layer, as well as of the specifics of upper layers.

1.1 Our Contributions

We investigate the feasibility of real-time Container classification for launching discriminating Blocking attacks, under an internal threat model. We show that such attacks are relatively easy to actualize by exploiting knowledge of network protocols and cryptographic primitives extracted from compromised nodes. We investigate the impact of discriminating Blocking on critical network functions. Our findings indicate that discriminating Blocking attacks lead to a DoS with very low effort on behalf of the jammer. To mitigate such attacks, we develop three schemes that prevent classification of transmitted Containers in real time. Our schemes rely on the joint consideration of crypto-graphic mechanisms with PHY-layer attributes. We analyze the security of our schemes and show that they achieve strong security properties, with minimal impact on the network performance.

© 2013, IJCSMC All Rights Reserved

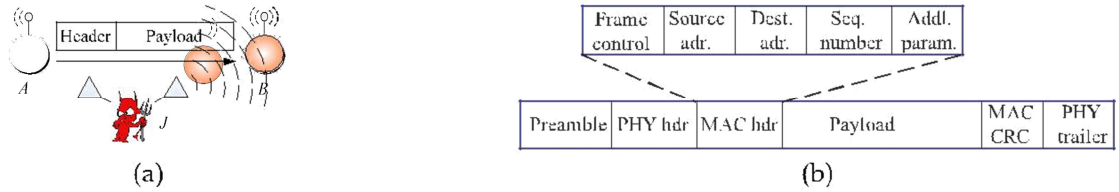


Fig. 1. (a) Realization of a discriminating Blocking attack. (b) A generic frame format for a wireless network.

II TROUBLE STATEMENT AND ASSUMPTIONS

2.1 Problem Statement

Consider the scenario depicted in Fig. 1a. Nodes A and B communicate via a wireless link. Within the communication range of both A and B, there is a Blocking node J. When A transmits a Container m to B, node J classifies m by receiving only the first few bytes of m. J then corrupts m beyond recovery by interfering with its reception at B. We address the problem of preventing the Blocking node from classifying m in real time, thus mitigating J's ability to perform discriminating Blocking. Our goal is to transform a discriminating jammer to a random one. Note that in the present work, we do not address Container classification methods based on protocol semantics, as described in [1], [4], [11], [33].

2.2 System and Adversary Model

2.2.1 **Network Model**

The network consists of a collection of nodes connected via wireless links. Nodes may communicate directly if they are within communication range, or indirectly via multiple hops. Nodes communicate both in unicast mode and broadcast mode. Communications can be either unencrypted or encrypted. For encrypted broadcast communications, symmetric keys are shared among all intended receivers. These keys are established using preshared pairwise keys or asymmetric cryptography.

2.2.2 **Communication Model**

Containers are transmitted at a rate of R bauds. Each PHYlayer symbol corresponds to q bits, where the value of q is defined by the underlying digital modulation scheme. Every symbol carries $\frac{\alpha}{\beta}$q data bits, where α/β is the rate of the PHY-layer encoder. Here, the transmission bit rate is equal to q R bps and the information bit rate is $\frac{\alpha}{\beta}$q R bps. Spread-spectrum techniques such as frequency hopping spread spectrum (FHSS), or direct sequence spread spec-trum (DSSS) may be used at the PHY layer to protect wireless transmissions from Blocking. SS provides immu-nity to interference to some extent (typically 20 to 30 dB gain), but a powerful jammer is still capable of Blocking data Containers of his choosing.

Transmitted Containers have the generic format depicted in Fig. 1b. The preamble is used for synchronizing the sampling process at the receiver. The PHY-layer header contains information regarding the length of the frame, and the transmission rate. The MAC header determines the MAC protocol version, the source and destination ad-dresses, sequence numbers plus some additional fields. The MAC header is followed by the frame body that typically contains an ARP Container or an IP datagram. Finally, the MAC frame is protected by a cyclic redundancy check (CRC) code. At the PHY layer, a trailer may be appended for synchronizing the sender and receiver.

2.2.3 **Adversary Model**

We assume the adversary is in control of the communication medium and can jam messages at any part of the network of his choosing (similar to the Dolev-Yao model). The adversary can operate in full-duplex mode, thus being able to receive and transmit simultaneously. This can be achieved,for example, with the use of multiradio transceivers. In addition, the adversary is equipped with directional antennas that enable the reception of a signal from one node and Blocking of the same signal at another. For analysis purposes, we assume that the adversary can proactively jam a number of bits just below the ECC capability early in the transmission. He can then decide to irrecoverably corrupt a transmitted Container by Blocking the last symbol. In reality, it has been demonstrated that discriminating Blocking can be achieved with far less resources [32], [34]. A jammer equipped with a single half-duplex transceiver is sufficient to classify and jam transmitted Containers. However, our model captures a more potent adversary that can be effective even at high transmission speeds.

© 2013, IJCSMC All Rights Reserved

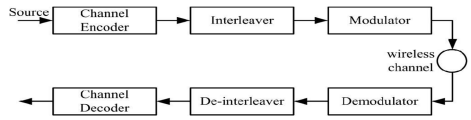


Fig. 2. A generic communication system diagram

The implementation details of every layer of the network stack are assumed to be public. Furthermore, the adversary is capable of physically compromising network devices and recovering stored information including cryptographic keys, PN codes, etc. This internal adversary model is realistic for network architectures such as mobile ad hoc, mesh, cognitive radio, and wireless sensor networks (WSNs), where network devices may operate unattended, thus being susceptible to physical compromise.

III REAL-TIME CONTAINER CLASSIFICATION

In this section, we describe how the adversary can classify Containers in real time, before the Container transmission is completed. Once a Container is classified, the adversary may choose to jam it depending on his strategy.

Consider the generic communication system depicted in Fig. 2. At the PHY layer, a Container m is encoded, interleaved, and modulated before it is transmitted over the wireless channel. At the receiver, the signal is demodulated, deinterleaved, and decoded to recover the original Container m.

The adversary's ability in classifying a Container m depends on the implementation of the blocks in Fig. 2. The channel encoding block expands the original bit sequence m, adding necessary redundancy for protecting m against channel errors. For example, an α/β block code may protect m from up to e errors per block. Alternatively, an α/β-rate convolutional encoder with a constraint length of L_{max}, and a free distance of e bits provides similar protection. For our purposes, we assume that the rate of the encoder is α/β. At the next block, interleaving is applied to protect m from burst errors. For simplicity, we consider a block interleaver that is defined by a matrix $A\alpha/\beta$ [1] The deinterleaver is simply the transpose of A. Finally, the digital modulator maps the received bit stream to symbols of length q, and modulates them into suitable waveforms for transmission over the wireless channel. Typical modulation techniques include OFDM, BPSK, 16(64)-QAM, and CCK.

- Symbols 3-10 contain the source and destination MAC addresses, and the length of the IP Container header.
- Symbols 11-17 contain the source and destination IP addresses, the size of the TCP datagram carried by the IP Container, and other IP layer information. The first two bytes of the TCP datagram reveal the source port.
- Symbols 18-23 contain the TCP destination port, sequence number, acknowledgment number, TCP flags, window size, and the header checksum.
- Symbols 24-25 contain the MAC CRC code.

$$c_1 = IV, \;\; c_{i+1} = E_k(c_i \oplus m_i), \quad i = 1, 2, \ldots, x, \qquad (1)$$

where E_k (m) denotes the encryption of m with key k. The plaintext m_i is recovered by

$$m_i = c_i \oplus D_k(c_{i+1}), \quad i = 1, 2, \ldots, x. \qquad (2)$$

© 2013, IJCSMC All Rights Reserved

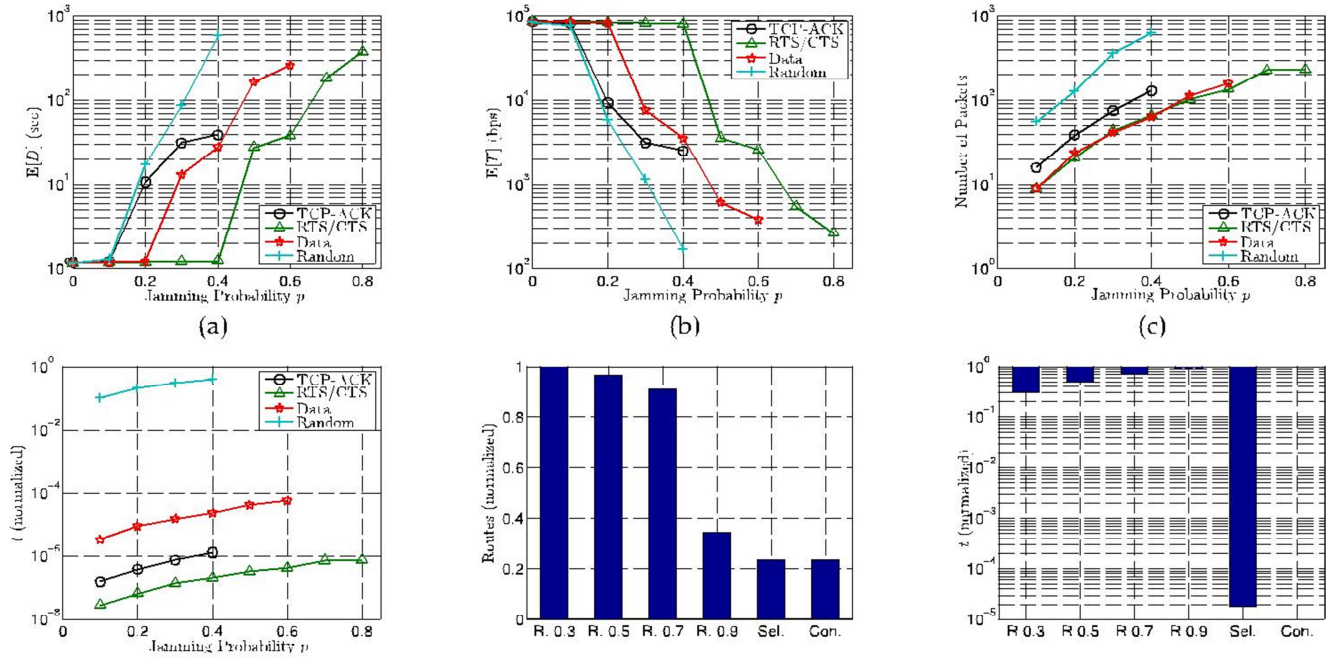


Fig. 3. (a) Average application delay E½D&. (b) Average effective throughput E½T &. (c) Number of Containers jammed. (d) Fraction of time the jammer is active. (e) Number of connections established in the network. (f) Fraction of time the jammer is active. R p: random jammer with probability p; Con.: constant jammer; Sel.: discriminating jammer.

is shared by multiple nodes. Any node that possesses the shared key is a candidate malicious node.

Moreover, even if the encryption key of a hiding scheme were to remain secret, the static portions of a transmitted Container could potentially lead to Container classification. This is because for computationally efficient encryption methods such as block encryption, the encryption of a prefix plaintext with the same key yields a static ciphertext prefix. Hence, an adversary who is aware of the underlying protocol specifics (structure of the frame) can use the static ciphertext portions of a transmitted Container to classify it.

IV IMPACT OF DISCRIMINATING BLOCKING

In this section, we illustrate the impact of discriminating Blocking attacks on the network performance. We used OPNET Modeler 14.5 [18] to implement discriminating Blocking attacks in two multihop wireless network scenarios. In the first scenario, the attacker targeted a TCP connection established over a multihop wireless route. In the second scenario, the jammer targeted network-layer control messages trans-mitted during the route establishment process.

4.1 Discriminating Blocking at the Transport Layer

In the first set of experiments, we set up a file transfer of a 3 MB file between two users A and B connected via a multihop route. The TCP protocol was used to reliably transport the requested file. At the MAC layer, the RTS/ CTS mechanism was enabled. The transmission rate was set to 11 Mbps at each link. The jammer was placed within the proximity of one of the intermediate hops of the TCP connection. Four Blocking strategies were considered:

1. Discriminating Blocking of cumulative TCP-ACKs.
2. Discriminating Blocking of RTS/CTS messages.
3. Discriminating Blocking of data Containers.
4. Random Blocking of any Container.

In each of the strategies, a fraction p of the targeted Containers is jammed.

In Fig. 3a, we show the average delay E½D& for completing the file transfer, as a function of the Blocking probability p (averaged over repeated experiments). In Fig. 3b, we show the average throughput E½T & as a function of p. It can be observed

© 2013, IJCSMC All Rights Reserved

that all Blocking attacks have significant impact on E½D& which grows several orders of magnitude larger compared to the delay in the absence of a jammer. Similarly, the effective throughput drops drastically under both random and discriminating Blocking attacks. TCP performance under Blocking of TCP-ACKs can be interpreted by the congestion control mechanism of the TCP protocol. When cumulative ACKs are lost (in our case jammed), the sender has to retransmit all unacknowledged data Containers, thus increasing the incurred delay while reducing the effective throughput. At the same time, the sender interprets the loss of ACKs as congestion and throttles its Container transmission rate by reducing the size of the transmission window. This leads to a further slow down of the application. Note that, for values of p > 0:4, the TCP connection is aborted for the case of random and TCP-ACK Blocking, due to the repeated time-outs at the sender.

Fig. 3c depicts the number of Containers that were jammed by the adversary for each value of p. Finally, Fig. 3d shows the fraction of time that the jammer remained active. Here, for discriminating Blocking attacks, we assumed that 13 percent of the Container have to be corrupted in order to be dropped [17]. In the case of random Blocking, the adversary is not aware of the type of Containers transmitted (by means of processing the header of these Containers). Hence, he is assumed to jam the entire Container in order to drop it. We observe that discriminating Blocking requires the Blocking of approximately one order of magnitude less Containers than random Blocking. This is because, as the Container transmission rate of the sender drops, fewer Containers need to be discriminatingly targeted. Moreover, in discriminating Blocking, the fraction of time the adversary remains active is several orders of magnitude less compared to random Blocking. From Fig. 3d, we observe that targeting control Containers such as RTS/CTS messages and TCP-ACKs yields the lowest Blocking activity, because control Containers are significantly smaller compared to data Containers. Such low-effort Blocking attacks are not only efficient in terms of energy expenditure, but also challenging in localizing and physically removing the Blocking devices. Typical methods of transmitter localization such as received signal strength and angle of arrival measurements require that the Blocking device remains active for extended periods of time.

4.2 Discriminating Blocking at the Network Layer

In this scenario, we simulated a multi hop wireless network of 35 nodes, randomly placed within a square area. The AODV routing protocol was used to discover and establish routing paths [19]. Connection requests were initiated between random source/destination pairs. Three jammers were strategically placed to discriminatingly jam non overlapping areas of the network. Three types of Blocking strategies were considered: 1) a continuous jammer, 2) a random jammer blocking only a fraction p of the transmitted Containers, and 3) a discriminating jammer targeting route-request (RREQ) Containers.

In Fig. 3e, we show the number of connections established, normalized over the number of connections in the absence of the jammers. Fig. 3f shows the fraction of time that the jammer was active during our simulation, for each Blocking strategy. We observe that a discriminating Blocking attack against RREQ messages is equally effective to a constant Blocking attack. However, discriminating Blocking is several orders of magnitude more efficient as it is illustrated in Fig. 3f. On the other hand, random Blocking fails to disrupt the route discovery process due to the flooding mechanism of AODV.

V HIDING BASED ON COMMITMENTS

In this section, we show that the problem of real-time Container classification can be mapped to the hiding property of commitment schemes, and propose a Container-hiding scheme based on commitments.

5.1 Mapping to Commitment Schemes

Commitment schemes are cryptographic primitives that allow an entity A, to commit to a value m, to an entity V while keeping m hidden. Commitment schemes are formally defined as follows [7]:

A releases the commitment C to V . In the open stage t_2, A releases the opening value d. Upon reception of d, V opens the commitment C, by applying function f_2 = open(), thus obtaining a value of m^{l}= open(); dÞ. This stage culminates in either acceptance (m^{l} = m) or rejection ($m^{l=}$m) of the commitment by V . Commitment schemes satisfy the following two fundamental properties:

- Hiding. For every polynomial-time party V interact-ing with A, there is no (probabilistic) polynomially efficient algorithm that would allow V to associate C with m and C^{0} with m^{0}, without access to the decommitment values d or d^{0}, respectively, and with non-negligible probability.
- Binding. For every polynomial-time party A inter-acting with V , there is no (probabilistic) polynomially efficient algorithm that would allow A to generate a triple (C; d; d^{l}), such that V accepts the commitments (C,d) and (C, d^{l}), with non-negligible probability.

© 2013, IJCSMC All Rights Reserved

For most applications, partial reveal of m with the partial release of d does not constitute a security risk. After all, the committer intends to reveal m by exposing d. However, in our context, a partial reveal of m while d is being transmitted can lead to the classification of m before the transmission of d is completed. Thus, the jammer has the opportunity to jam d instead of C once m has been classified. To prevent this scenario, we introduce the strong hiding property:

In the above definition, it is easily seen that the release of d_{part} must be limited to a fraction of d, in order for m to remain hidden. If a significant part of d becomes known to the verifier, trivial attacks, such as brute forcing the unknown bits of d, become possible.

5.2 A Strong Hiding Commitment Scheme (SHCS)

We propose a strong hiding commitment scheme, which is based on symmetric cryptography. Our main motivation is to satisfy the strong hiding property while keeping the computation and communication overhead to a minimum..

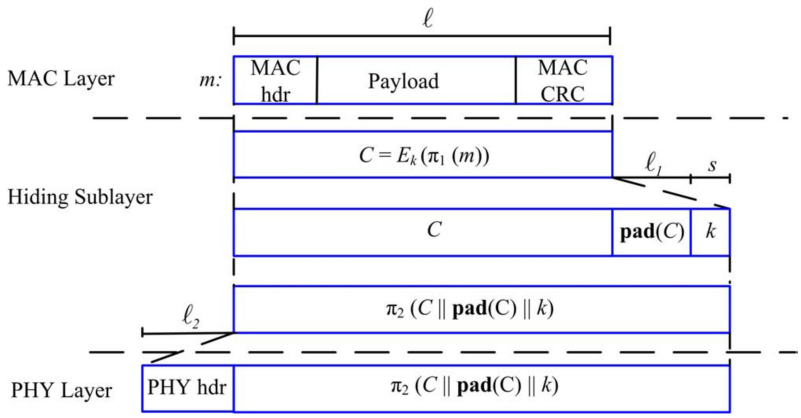


Fig. 4. Processing at the hiding sublayer.

Assume that the sender S has a Container m for R. First,S constructs (C, d)= commit(m), where

$C= E_k(\mu_1(m(m)), d = k.$

Here, the commitment function $E_k()$ is an off-the-shelf symmetric encryption algorithm (e.g., DES or AES [27]), μ_1 is a publicly known permutation, and $k \epsilon \{0, 1\}^s$ is a randomly selected key of some desired key length s (the length of k is a security parameter). The sender broadcasts (C||d), where "||" denotes the concatenation operation. Upon reception of d, any receiver R computes

$$m = \square_1^{-1}(D_k(C)),$$

where $\square_1^{-1}$ denotes the inverse permutation of $\square_1$. To satisfy the strong hiding property, the Container carrying d is formatted so that all bits of d are modulated in the last few PHY-layer symbols of the Container. To recover d, any receiver must receive and decode the last symbols of the transmitted Container, thus preventing early disclosure of d. We now present the implementation details of SHCS.

5.3 Implementation Details of SHCS

The proposed SHCS requires the joint consideration of the MAC and PHY layers. To reduce the overhead of SHCS, the decommitment value d (i.e., the decryption key k) is carried in the same Container as the committed value C. This saves the extra Container header needed for transmitting d individually. To achieve the strong hiding property, a sublayer called the "hiding sublayer" is inserted between the MAC and the PHY layers. This sublayer is responsible for formatting m before it is processed by the PHY layer. The functions of the hiding sublayer are outlined in Fig. 4.

© 2013, IJCSMC All Rights Reserved

Consider a frame m at the MAC layer delivered to the hiding sublayer. Frame m consists of a MAC header and the payload, followed by the trailer containing the CRC code. Initially, m is permuted by applying a publicly known permutation □1. The purpose of □1 is to randomize the input to the encryption algorithm and delay the reception of critical Container identifiers such as headers. After the permutation, □1(m) is encrypted using a random key k to produce the commitment value C = Ek (□1(m)).

5.3.1 Padding

The purpose of padding is to ensure that k is modulated in the minimum number of symbols needed for its transmission. This is necessary for minimizing the time for which parts of k become exposed. Let ‘1 denote the number of bits padded to C. For simplicity, assume that the length of C is a multiple of the block length of the symmetric encryption algorithm and hence, has the same length ‘ as the original message m. Let also ‘2 denote the length of the header added at the PHY layer.

The frame carrying (C,d) before the encoder has a length of (l+l1+l2+s) bits. Assuming that the rate of the encoder is□/□, the output of the encoder will be of length (l+l1+l2+s). For the last symbol of transmission to include □/□q bits of the key k, it must hold that

$$l1=\frac{\square}{\square}(q-((l+l2)\frac{\square}{\square})\bmod q). \qquad (3)$$

5.4 Security Analysis

In this section, we analyze the security of SHCS by evaluating the ability of J in classifying a transmitted Container at different stages of the Container transmission.

5.4.1 Release of C

We first examine if J can classify m by observing the commitment value C. Though C and k are part of the same Container, symbols corresponding to C are received first. The jammer can attempt to classify m by launching a cipher text only attack on C as early as the reception of the first cipher text block. Because the encryption key is refreshed at every transmission, a very small number of cipher text blocks are available for cryptanalysis. Appropriate selection of the key length s can prevent this type of attack. Note that
s can be well below the cryptographic standards, due to the limited time available to the adversary (until the transmission is completed). For instance, a 56-bit long DES key is more than adequate for our purposes, since the fastest known brute force attack on DES takes almost a day [24]. Other types of known attacks such as differential and linear cryptanalysis are not applicable, because they require the collection of a large number of chosen or known plaintext/ cipher text pairs .

As an example, consider the encryption of a message m = (m_1, m_2, . . . m) with a key k of length 56 bits, using blocks of 128 bits. For a fairly small plaintext space, the probability of cipher text uniqueness is equal to 99.8 percent. Thus, the adversary can recover k, by launching a codebook attack on m_1. The remaining c_is are decrypted in real time, using the known value of k. Here, the plaintext space for m_1 is considered to be small because of the structure imposed by the static header of a Container (all fields of the header are known to the adversary). Randmization of the plaintext ensures that all plaintexts are possible, thus equating the plaintext space with the cipher text space.

5.4.2 Partial Release of d

Depending on the PHY-layer implementation, d ¼ k re-quires n _ 1 symbols for its transmission. Hence, part of k may become known before the completion of the transmission of the Container at hand. This release reduces the search space for a brute force attack on k. Assume that the adversary proactively jams a few symbols below the ECC capability of the receiver during the transmission of C. In the best case, he can postpone his decision to jam a transmitted Container until the transmission of the last symbol (jam one more symbol to drop the Container). He must therefore complete the classification process before the last symbol is transmitted. Assuming that the adversary waits until the maximum number of bits of k are released, the key search space before the transmission of the last two symbols is equal to 2^{2-q} keys. The adversary must be capable of performing on average N ¼ $2^{\delta 2-q-1p}$R decryptions per second in order to find k before the last symbol is transmitted.[2] Here, we have assumed that, on average, half the key space must be searched.

For example, assume an 802.11a PHY layer operating at 6 Mbps, with every symbol carrying 24 bits of information. Consider k to be a 56-bit DES key, fitting in three symbols. The computational capability of the adversary must be equal to N _ 3:52 _ 10^{19} decryptions/sec in order to recover k before the completion of the Container transmission. The fastest known hardware implementation of a DES cracker achieves a throughput of 2:92 _ 10^{11} keys per second [24]. For an operating rate of 54 Mbps, all 56 bits

© 2013, IJCSMC All Rights Reserved

2. A more accurate calculation of N would assume an adversary trying a brute force attack on k, with the reception of the first ciphertext block, and adjusting the searching space according to the partial release of k.

5.4.3 Binding Property

The binding property is not a security requirement of SHCS under our adversary model. Since the primary goal of any sender S in the network is to communicate m, S has no interest in modifying m after he has committed to it. However, under a more general adversary model, the jammer may launch denial-of-service attacks by making the receiver R to accept a k^0 6¼ k such that m^0 ¼ $D^0{}_k$ðCÞ is a meaningful message. Even though SHCS is not designed to ensure the binding property of commitment schemes, generating a k^0 ¼6 k that opens a valid value of m^0 6¼ m is a computationally hard task. In order to find such a k^0, the jammer has to launch a brute force attack on C. Here, not only the attack must be performed in a timely manner, but m^0 has to be in the right format (source/destination address must be in the context of the communications, CRC code must be valid, etc). Given that k is transmitted right after C, the jammer has no time to find an appropriate k^0 that would lead to the decryption of an acceptable m^0, assuming that such m^0 exists. If m^0 is not meaningful, substituting k with k^0 is equivalent to a Blocking attack on m without classification (no selectivity).

The binding property can be theoretically achieved if a random string r is appended to m In this case, the commitment/decommitment pair (C, d) is

$$C = (\gamma, \delta) = (E_k(m\|r), r), \quad d = k.$$

Provided that r is sufficiently long, a computationally bounded jammer cannot find a k^0 such that D_k0 ðCÞ ¼ m^0kr. In this case, r preserves the integrity of message m. Since the addition of r is not necessary for preventing real-time classification of m, we leave the implementation of the binding property to the discretion of the system designer.

5.5 Resource Overhead of SHCS

In this section, we analyze the per-Container communication and computational overhead of SHCS.

5.5.1 Communication Overhead

For every Container m, a random key k of length s is appended. Also, ð‘$_b$ _ ð‘ mod ‘$_b$ÞÞ bits of overhead are added by the encryption algorithm, to convert a plaintext of length ‘ to a multiple of the encryption block. Thus, the communication overhead of SHCS is equal to s þ ð‘$_b$ _ ð‘ mod ‘$_b$ ÞÞ per Container. Here, we do not account for the padding string padðCÞ, because the addition of padðCÞ does not increase the number of transmitted symbols.

5.5.2 Computation Overhead

The computation overhead of SHCS is one symmetric encryption at the sender and one symmetric decryption at

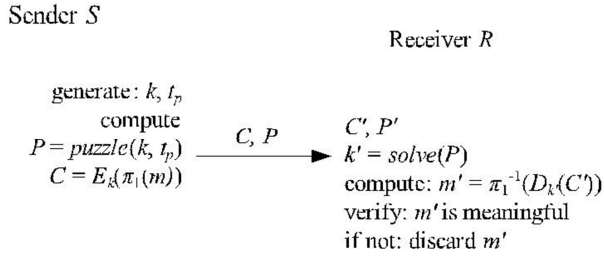


Fig. 6. The cryptographic puzzle-based hiding scheme

VI Hiding Based On Cryptographic Puzzles

In this section, we present a Container-hiding scheme based on cryptographic puzzles. The main idea behind such puzzles is to force the recipient of a puzzle execute a predefined set of computations before he is able to extract a secret of interest. The time required for obtaining the solution of a puzzle depends on its hardness and the computational ability of the solver. The advantage of the puzzle-based scheme is that its security does not rely on the PHY-layer parameters. However, it has higher computation and communication overheads.

© 2013, IJCSMC All Rights Reserved

6.1 Cryptographic Puzzle Hiding Scheme (CPHS)

Let a sender S have a Container m for transmission. The sender selects a random key k 2 f0; $1g^s$ of a desired length. S generates a puzzle P = puzzle(k; t_p), where puzzle() denotes the puzzle generator function, and t_p denotes the time required for the solution of the puzzle. Parameter t_p is measured in units of time, and it is directly dependent on the assumed computational capability of the adversary, denoted by N and measured in computational operations per second. After generating the puzzle P , the sender broadcasts (C, P), At the receiver side, any receiver R solves the received puzzle P^{0} to recover key k^0 and then computes $m^0 =^{-1}$ $(D_k0\ )C^0$. If the decrypted Container m^0 is meaningful (i.e., is in the proper format, has a valid CRC code, and is within the context of the receiver's communication), the receiver accepts that m^0 ¼ m. Else, the receiver discards m^0. Fig. 6 shows the details of CPHS.

6.2 Implementation Details of CPHS

In this section, we consider several puzzle schemes as the basis for CPHS. For each scheme, we analyze the implementation details which impact security and performance. Cryptographic puzzles are primitives originally suggested by Merkle as a method for establishing a secret over an insecure channel [16]. They find a wide range of applications from preventing DoS attacks to providing broadcast authentication and key escrow schemes.

6.2.1 **Time-Lock Puzzles**

Rivest et al. proposed a construction called time-lock puzzles, which is based on the iterative application of a precisely controlled number of modulo operations [22]. Time-lock puzzles have several attractive features such as the fine granularity in controlling t_p and the sequential nature of the computation. Moreover, the puzzle generation requires significantly less computation compared to puzzle solving.

In a time-lock puzzle, the puzzle constructor generates a composite modulus g= u . v, where u and v are two large random prime numbers. Then, he picks a random a; 1 < a < g and hides the encryption key in $K_h = k + a^{2t} \bmod g$, where t =t_p.N is the amount of time required to solve for k. Here, it is assumed that the solver can perform N squarings modulo g per second. Note that K_h can be computed efficiently if Φ(g)= (u _ 1)(v _ 1) or the factor-ization of g are known, otherwise a solver would have to perform all t squarings to recover k. The puzzle consists of the values P=(g,K_h, t, a).

In our setup, the value of the modulus g is known a priori and need not be communicated (may change periodically). The sender reveals the rest of the puzzle information in the order(K_h, t, a). Note that if any of t, a are unknown, any value of k is possible [22].

6.2.2 **Puzzles Based on Hashing**

Computationally limited receivers can incur significant delay and energy consumption when dealing with modulo arithmetic. In this case, CPHS can be implemented from cryptographic puzzles which employ computationally effi-cient cryptographic primitives. Client puzzles proposed in [10] use one-way hash functions with partially disclosed inputs to force puzzle solvers search through a space of a precisely controlled size. In our context, the sender picks a random key k with k = $k_1 \| k_2$. The lengths of k_1 and k_2 are s_1, and s_2, respectively. He then computes C = E_k ($\square_1$(m)) and transmits (C, k_1, h(k)) in this particular order. To obtain k, any receiver has to perform on average $2^{s2^{-1}}$ hash operations (assuming perfect hash functions). Because the puzzle cannot be solved before h(k) has been received, the adversary cannot classify m before the completion of m's transmission.

6.3 Security Analysis of CPHS

With the completion of the transmission of P , any receiver can recover m. Therefore, a discriminating jammer must attempt to classify m before the transmission of P has been completed. We analyze the security of CPHS at different stages of its execution.

VII HIDING BASED ON ALL-OR-NOTHING TRANSFORMATIONS (AONTS)

we propose a solution based on All-or-Nothing Transformations that introduces a modest communication and computation overhead. Such transformations were originally proposed by Rivest to slow down brute force attacks against block encryption algorithms .

7.1 An AONT-Based Hiding Scheme (AONT-HS)

In our context, Containers are preprocessed by an AONT before transmission but remain unencrypted. The jammer cannot perform Container classification until all pseudomessages corresponding to the original Container have been received and the inverse transformation has been applied. Container m is partitioned to a set of x input blocks m = (m_1, . . . ; m_x)

© 2013, IJCSMC All Rights Reserved

7.2 Implementation Details of the AONT-HS

In this section, we describe two AONTs which can be employed in AONT-HS; a linear transformation , and the original package transformation .

7.2.1 Linear AONT

In [26], Stinson showed how to construct a linear AONT when the alphabet of the input blocks is a finite field IF_u, with the order u being a prime power. He showed that if an invertible matrix M = (mij||mij belongs to IF_u; m_{ij} != 0)$_{x_x}$ exists, then the transformation (m)= mM^{-1} is a linear AONT. He also provided a method for constructing such M which is as follows:

Let $u = v^i$, where v is prime and i is a positive integer. Choose λ€ IF_u

$$LT = \begin{bmatrix} 1 & 0 & \cdots & 0 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 1 \\ 1 & 1 & \cdots & 1 & \lambda \end{bmatrix}. \qquad (4)$$

Given $m = \{m_1, \ldots, m_x\}$,

$$m'_x = \lambda m_x + \sum_{j=1}^{x-1} m_j, \quad m'_i = m_i + m'_x, \quad 1 \le i \le (x-1). \qquad (5)$$

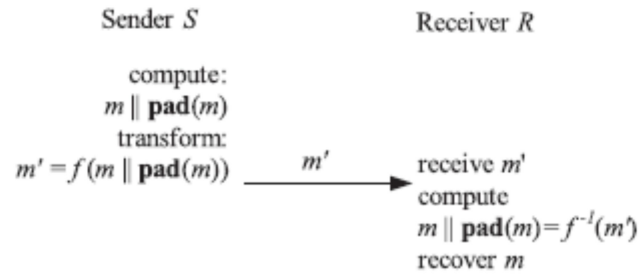


Fig. 7. The AONT-based hiding scheme.

Conversely, given $m^I = \{m^I 1; \ldots; m^I x\}$ the original input m ={m1, . . .,mx } is recovered as follows:

$$m_i = m'_i - m'_x, \quad 1 \le i \le (x-1), \qquad (6)$$

$$m_x = \gamma(m'_1 + \cdots m'_{x-1} - m'_x), \quad \gamma = \frac{1}{n - \lambda - 1}. \qquad (7)$$

Note from (6) and (7) that if any of the{ $fm^I{}_I$ }is missing, all values of m_i are possible, for every i. Thus, the linear AONT provides undeniable security.

7.2.2 The Package Transform

In the package transform [21], given a message m, and a random key k^1, the output pseudomessages are computed as follows:

$$m'_i = m_i \oplus E_k(i), \quad \text{for } i = 1, 2, \ldots, x, \qquad (8)$$

$$m'_{x+1} = k' \oplus e_1 \oplus e_2 \oplus \cdots \oplus e_x, \qquad (9)$$

© 2013, IJCSMC A

where $e_i = E_{k0}(m^0_i \oplus i)$; for i = 1, 2, . . . , x, and k_0 is a fixed publicly known encryption key. With the reception of all pseudomessages, message m is recovered as follows:

$$k' = m'_{x+1} \oplus e_1 \oplus e_2 \oplus \cdots \oplus e_x, \qquad (10)$$

$$m_i = m'_i \oplus E_{k'}(i), \text{ for } i = 1, 2, \ldots, x. \qquad (11)$$

Note that if any m^0_i is unknown, any value of k^0 is possible, because the corresponding e_i is not known. Hence, E_k0 ðiÞ cannot be recovered for any i, making it infeasible to obtain any of the m_i.

7.2.3 Hiding Sublayer Details

AONT-HS is implemented at the hiding sublayer residing between the MAC and the PHY layers. In the first step, m is padded by applying function pad() to adjust the frame length so that no padding is needed at the PHY layer, and the length of m becomes a multiple of the length of the pseudomessages m^0_i. This will ensure that all bits of the transmitted Container are part of the AONT. In the next step, m||pad(m) is partitioned to x blocks, and the AONT f is applied. Message m^0 is delivered to the PHY layer. At the receiver, the inverse transformation f^{-1} is applied to obtain m||pad(m). The padded bits are removed and the original message m is recovered. The steps of AONT-HS are shown in Fig. 7.

7.3 Security Analysis of the AONT-HS

7.3.1 Partial Reception of m^I_i; $i < x^I$

In the AONT-HS, the jammer may attempt to classify m without receiving all m^I_i ($1 \leq i \leq x^I$). By definition, AONTs prevent the computation of any part of m without the reception of all the pseudo messages. In fact, for the linear AONT, undeniable security is achieved. The jammer can launch a brute force attack on m as early as the reception of m^I_1. However, the system of equations formed by m^I_is when at least one is missing has a number of solutions equal to the message space. All these solutions are equiprobable.

7.4.1 Communication Overhead

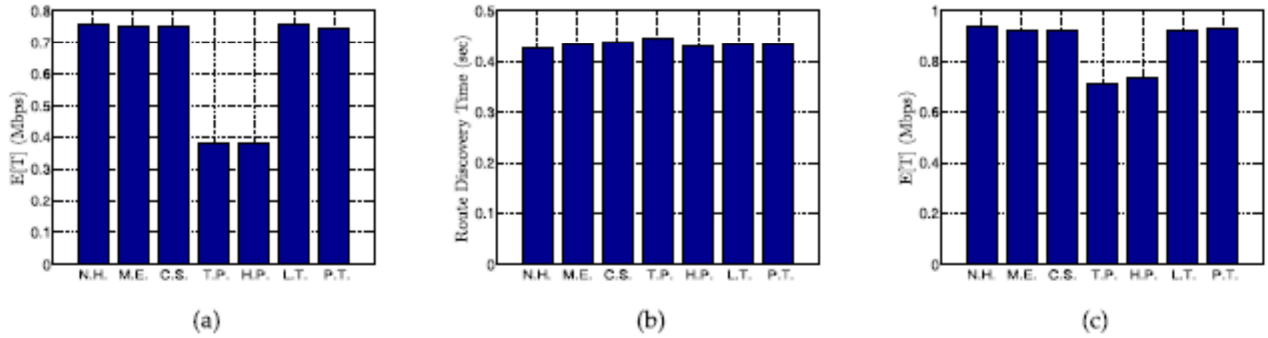
In AONT-HS, the original set of x messages is transformed to a set of x^I pseudo messages, with $x^I \geq x$. Additionally, the function pad( ) appends (ls-(l mod ls)) bits in order to make the length of m a multiple of the length '$_b$ of the pseudo messages m^I. Hence, the communication overhead introduced is ($l_b(x^I$ _ x) + l_b – (l mod l_b)) bits. For the linear AONT, x – x^I, and therefore, only the padding communication overhead is introduced.

7.4.2 Computation Overhead

The linear AONT requires only elementary arithmetic operations such as string addition and multiplication, making it particularly fast due to its linear nature. The package transform requires x^I symmetric encryptions at the sender and an equal amount of symmetric decryptions at the receiver. Note that the length of the plaintext for the x^I encryptions is relatively small compared to the length of message m (indexes 1, . . . , x are encrypted). Therefore, only one ciphertext block is produced per pseudomessage. Assuming a pseudomessage block size equal to the ciphertext block size l_b, the computational overhead of the x^I encryptions required by the package transform is equivalent to the overhead of one encryption of a message of length $l+l_b$.

© 2013, IJCSMC All Rights Reserved

VIII **EVALUATION OF CONTAINER-HIDING TECHNIQUES**



8.1 Impact on Real-Time Systems

Our Container-hiding methods require the processing of each individual Container by the hiding sublayer. We emphasize that the incurred processing delay is acceptable, even for real-time applications. The SCHS requires the application of two permutations and one symmetric encryption at the sender, while the inverse operations have to be performed at the receiver. Such operations can be implemented in hardware very efficiently. Symmetric encryption such as AES can be implemented at speeds of tens of Gbps/sec when realized with Application Specific Integrated Circuits (ASICs) or Field Programmable Gate Arrays (FPGAs) These processing speeds are orders of magnitude higher than the transmission speeds of most current wireless technologies, and hence, do not impose a significant delay.

Similarly, the AONT-HS performs linear operations on the Container that can be efficiently implemented in hardware. We note that a non-negligible processing delay is incurred by the CPHS. This is due to the cryptographic puzzle that must be solved at the receiver. As suggested in Section 6, CPHS should only be employed when the symbol size at the PHY layer is too small to support the SHCS and AONT-HS solutions. The processing delays of the various schemes are taken into account in our experimental evaluations.

8.2 Experimental Evaluation

In the first set of experiments, we set up a single file transfer between a client and server, connected via a multihop route. The client requested a 1 MB file from the server. We evaluated the effects of Container hiding by measuring the effective throughput of the TCP connection in the following scenarios:

1. No Container hiding (N.H.).
2. MAC-layer encryption with a static key (M.E.).
3. SHCS (C.S.).
4. Time-lock CPHS (T.P.).
5. Hash-based CPHS (H.P.).
6. Linear AONT-HS (L.T.).
7. AONT-HS based on the package transform (P.T.).

In Fig. 8a, we show the effective throughput averaged over 100 different traces. We observe that MAC-layer encryption, SHCS, and the linear AONT-HS achieve an effective throughput close to the throughput in the absence of Container hiding. This is justified by the relatively small communication overhead of each hiding method and the small queuing delay at intermediate routers due to the absence of any cross traffic. The AONT-HS based on the package transform achieved slightly lower throughput, because it occurs a per-Container overhead of 128 bits as opposed to 56 bits for SHCS. We also observe that hiding techniques based on cryptographic puzzles decrease the effective throughput of the TCP connection to half, compared to the no hiding case.

© 2013, IJCSMC All Rights Reserved

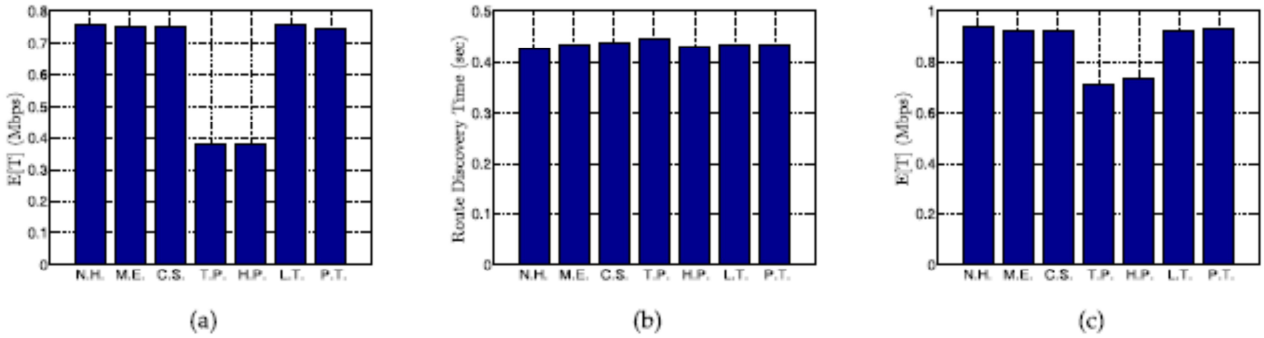


Fig. 8. (a) Average effective throughput (line network topology). (b) Average route discovery time (noncongested network). (c) Average effective throughput (congested network). N.H.: No Container hiding, M.E.: MAC-layer encryption with a static key, C.S.: SHCS, T.P.: Time-lock CPHS, H.P.: Hash-based CPHS, L.T.: Linear AONT-HS, P.T.: Package transform.

The average route discovery delay is shown in Fig. 8b. This delay is defined as the time difference between the transmission of the first RREQ from a source and the reception of the corresponding RREP from the destination. We observe that the impact of Container hiding on the route discovery delay is minimal compared to the case where no Container hiding is employed. This similarity in performance is due to the expanding ring search technique of AODV, which is used to prevent unnecessary network-wide dissemination of RREQs [19]. In order to discover a route, the originating node sends an RREQ with a time-to-live (TTL) value equal to one hop, and waits for the correspond-ing RREP. If the RREP is not received before a time-out value (t_o) expires, the originating node increases the TTL and the time-out t_o, and rebroadcasts the RREQ. This process is repeated until a valid RREP is received, or the TTL value exceeds the maximum diameter of the network. The expanding ring search technique introduces a dominant delay in comparison to the delay introduced by the Container-hiding techniques. For example, in the case of time-lock CPHS, the per-Container delay overhead is t_p ¼ 448 sec. Using the default values specified by RFC 3561 [19], the value of the first time-out is t_o ¼ 240 msec, which is 535 times higher than t_p, making the total delay introduced by time-lock CPHS insignificant.

In the third set of experiments, we evaluated the performance of TCP in a congested ad hoc network. We considered the same network topology used in the second set of experiments. Twenty source/destination pairs simul-taneously exchanged 2 MB files using TCP.

CONCLUSION

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 Containers in real time by decoding the first few symbols of an ongoing transmission. We evaluated the impact of discriminating Blocking attacks on network protocols such as TCP and routing. Our findings show that a discriminating jammer can significantly impact performance with very low effort. We developed three schemes that transform a discriminating jammer to a random one by prevent-ing real-time Container classification. Our schemes combine cryptographic primitives such as commitment schemes, cryptographic puzzles, and all-or-nothing transformations with physical-layer characteristics. We analyzed the security of our schemes and quantified their computational and communication overhead.

AUTHORS

1).R.SAILAJA M.TECH student in M.Tech CSE student , Sphoorthy Engineering college ,JNTU Hyderabad, Andhra Pradesh, India

2).G.VENKATAPRASAD Assistant Professor, Department of CSE Sphoorthy Engineering College, Hyderabad, Andhra Pradesh, India.

3). E.HARI PRASAD Head of the Department of CSE & IT, Sphoorthy Engineering College, Hyderabad, Andhra Pradesh, India

© 2013, IJCSMC All Rights Reserved

REFERENCES

[1] T.X. Brown, J.E. James, and A. Sethi, "Blocking and Sensing of Encrypted Wireless Ad Hoc Networks," Proc. ACM Int'l Symp. Mobile Ad Hoc Networking and Computing (MobiHoc), pp. 120-130, 2006.

[2] M. Cagalj, S. Capkun, and J.-P. Hubaux, "Wormhole-Based Anti-Blocking Techniques in Sensor Networks," IEEE Trans. Mobile Computing, vol. 6, no. 1, pp. 100-114, Jan. 2007.

[3] A. Chan, X. Liu, G. Noubir, and B. Thapa, "Control Channel Blocking: Resilience and Identification of Traitors," Proc. IEEE Int'l Symp. Information Theory (ISIT), 2007.

[4] Robustness of IEEE802.11 Rate Adaptation Algorithms against Smart Blocking," Proc. ACM Conf. Wireless Network Security (WiSec), 2011.

© 2013, IJCSMC All Rights Reserved