

Low Power DoS Attacks in Data Wireless LANs and Countermeasures

G. Lin, G. Noubir

Wireless Security Laboratory
College of Computer Science
Northeastern University
{lingl, noubir}@ccs.neu.edu

Abstract:

In this paper we investigate the resiliency to jamming of data protocols, such as IP, over Wireless LAN. We show that, on existing WLAN, an adversary can jam this protocol at a very low energy cost. Such attacks enable a set of adversary nodes disseminated over a geographical area to prevent communication, partition an ad hoc network, or force packets to be routed over adversary chosen paths. The ratio of the jamming pulses duration to the transmission duration can be as low as 10^{-4} . We investigate and analyze the performance of using various coding schemes to improve the robustness of wireless LANs for IP packets transmission. We propose a concatenated code that is simple to decode and can maintain a low Frame Error Rate (FER) under a jamming effort ratio of 15%. We investigate the theoretical limits by analyzing the performance derived from upper bounds on binary error-control codes. We also propose an efficient anti-jamming technique for IEEE802.11b standard.

1 Introduction

Current standards for wireless data communications such as IEEE802.11 and Bluetooth are easy targets of denial of service attacks. For example, the physical layers of IEEE802.11 and IEEE802.11b do not have any error-correction scheme. If an attacker sends a strong jamming signal of duration *one* bit/symbol it will make the CRC computation wrong. Therefore the whole packet will be lost. If we assume that this wireless link is used to transmit an IP data packet (usually 12000 bits long), the energy ratio between a jammer and user can be of the order of 1/10000 (which is equivalent to 40 dB gain for the jammer). Other wireless data standards that make use of error-correction codes can also be easily defeated as we will show in Section 2.2. The reason is that current systems are designed to resist non-malicious interference and noise. Even robust wireless links designed to resist jamming do not fully take into account the data aspect of the communication. Existing anti-jamming systems rely on an extensive use of spread-spectrum techniques [1]. These techniques *separately* protect *bits* against jammers. They are adequate for voice communication where the jammer has to keep jamming the channel to prevent a communication. In voice communication, when the communicating nodes use a high-gain spreading sequence, the energy of a jammer can be easily exhausted for a continuous jamming of the voice communication. Non-continuous jamming only results in a graceful degradation of the voice quality. In the context of data communication, spread-spectrum techniques are not sufficient because the jammer does not need to jam a data packet for a long period of time to be able to destroy it. In a “non error-correction” encoded data packet a single bit error generates a CRC error, leading to the loss of the entire packet. Our work aims at building on top of traditional anti-jamming techniques, used at the bit level (such as spread spectrum), to protect data packets.

In the context of a multihop ad hoc network a small number of smart jammers disseminated across a geographical area can last for a long period of time with limited energy resource. Since they only need short jamming durations, the remaining time and energy can also be used to jam other communication channels. They can even be coordinated to create an attack network

targeting traffic between specific nodes. They can achieve several goals such as preventing all communication, partitioning a network at low energy cost, or forcing all packets to be routed over chosen areas. In the last case, the traffic will be forced over an area where the adversary has powerful nodes that do better channel decoding and traffic analysis. The adversary nodes can stay in sleep mode most of the time and be triggered to jam some communication between specific nodes. In this case, the attackers would only wake-up to detect some MAC/IP address and, if needed, jam only few bits of the packet to destroy. The attacking nodes receivers can be designed to consume very little energy because the goal is not to demodulate/decode correctly a packet but only to detect (or carrier sense), with a reasonable probability, ongoing communication. These low-power jammers will be referred to as *cyber-mines*. Another scenario is in a building where most of the communication is wireless. A small number of portable devices can prevent all communications at a very low cost and without being easily detected. Even if anti-jamming techniques, such as spread-spectrum, are used, the substantial gain achieved by having to jam only few bits out of 1500 bytes IP packets can be invested in a higher signal power (for direct-sequence spread spectrum) or multi-channel jamming (for frequency hopping spread spectrum). This gain in jamming effort can be invested by the attacker to circumvent the processing gain (usually 20 to 30 dB in the context of military communications) achieved by spread spectrum techniques.

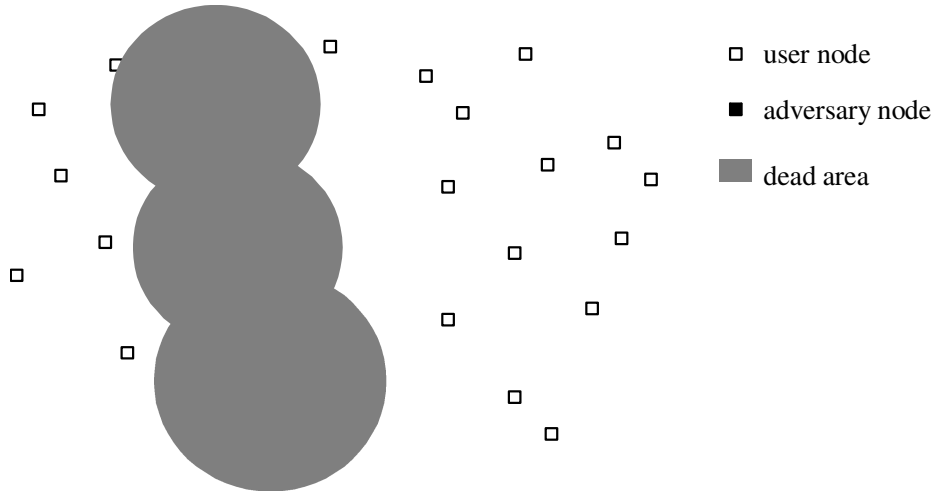


Figure 1. Adversary nodes disseminated over an area can prevent communications and also partition a multihop ad hoc network. They can last for long durations of time because they only consume a fraction of the energy of a normal node.

In this paper, we show that it is easy to jam existing wireless data systems at low power cost. We will propose and analyze the performance of various techniques for making data communications reliable in the presence of such malicious attackers. Our techniques are based on the combination of error-correction codes and cryptographically strong interleavers (i.e., adversaries cannot guess the interleaving function). The underlying assumption to our work is that jamming a single bit has a constant cost. We investigate how this cost scales to destroying a complete packet. All existing techniques, such as spread spectrum, can be transparently combined with our approach for an increased resiliency.

We will focus on DoS targeting the physical layer. Previous research on physical layer jamming has only focused on bit anti-jamming [1, 2] and not on packet level anti-jamming. Other DoS techniques can be applied at higher protocol layers of systems such as IEEE802.11 (e.g., by forcing the backoff window to remain at its maximum) or Bluetooth MAC (e.g., by destroying

some control packet), routing (e.g., by injecting erroneous or destroying control routing packets), and transport protocols (e.g., by forcing TCP multiplicative decrease to keep the congestion window small) [3-10]. In our future investigations, we will address multi-layer DoS attacks on wireless networks.

In the rest of this section, we introduce the concepts behind reliable communications. In Section 2, we show how existing WLAN standards, such as IEEE802.11 and Bluetooth, can be jammed at very low energy cost. In Section 3, we analyze the performance of directly using known binary codes against jamming. In Section 4, we propose and analyze the performance and the tradeoffs of two concatenated codes against jamming. Then, in Section 5, we address some practical considerations for implementing the proposed schemes. Finally, we conclude and propose directions for future research.

1.1 Channel coding

Figure 2 describes a simplified architecture for the transmitter and receiver of a digital communication [1]. We only show the components relevant to our paper. Components such as equalizers, amplifiers, upconversion mixers, and antennas are omitted. We only consider block codes, but convolutional codes use similar design. The stream of data bits is first encoded, then interleaved, and finally modulated for transmission over the channel. The receiver first demodulates the incoming signal, then it de-interleaves the bits, and finally error-decodes them.

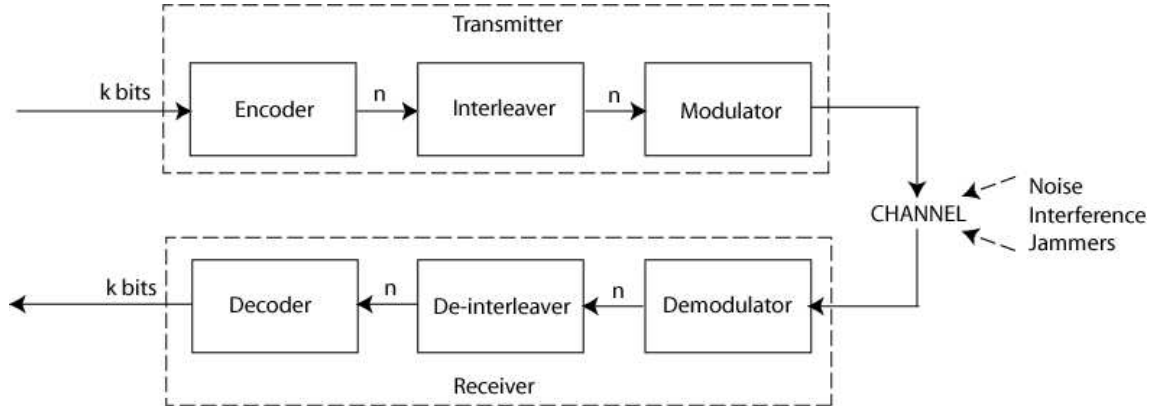


Figure 2: Simplified architecture of a communication link.

The channel encoding is achieved using an error-control code (ECC) [11, 12]. An error-control code can be defined as follows. Let's consider a set of symbols Σ , with cardinality q . A block error-control coding scheme is a function that maps a vector $u = (u_1, \dots, u_k) \in \Sigma^k$ into a codeword $v = (v_1, \dots, v_n) \in \Sigma^n$. When $q = 2$ the scheme is called a binary error-control code or binary code.

The Hamming distance between two words $x, y \in \Sigma^n$ is the number of positions where x differs from y . It is denoted by $\Delta(x, y)$. A code C is a subset of Σ^n , whose minimum

distance is defined by $\Delta(C) \stackrel{\text{def}}{=} \min_{x, y \in C; x \neq y} \{\Delta(x, y)\}$.

A code C is typically characterized by four parameters $(n, k, d)_q$. n denotes the codeword length, $k = \log_q |C|$ the uncoded word length, $d = \Delta(C)$ the code minimum distance, and $q = |\Sigma|$ the code alphabet size. To simplify the notation, we will omit q when addressing binary codes (i.e., $q = 2$).

We usually also characterize a code by its code rate $r = k/n$, and its relative distance $\delta = d/n$. A $(n, k, d)_q$ code can correct up to $\lfloor (d-1)/2 \rfloor$ symbols in error.

1.2 Packet encoding

Before transmission, the data information is formatted and processed for reliable error detection and correction. Figure 3 gives a simplified view of this process. First, a checksum (or CRC) is appended to the data bits. Then, the data sequence is divided into one or several blocks of k bits. Each block is encoded into a codeword of n bits. Finally, the encoded bits are interleaved before being transmitted. The checksum is used by the receiver to verify that the de-interleaved/decoded steps did not lead to an uncorrectable error. We assume that the checksum length is s . In practice, s is usually between 16 and 32 bits.

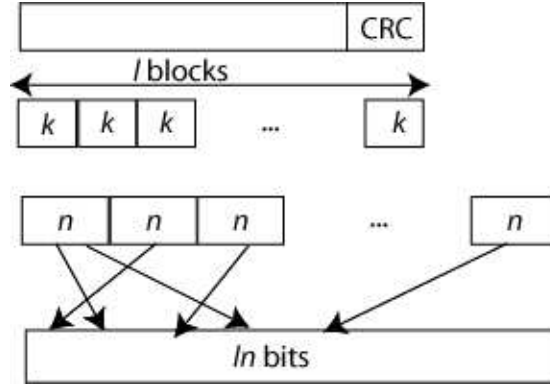


Figure 3: Encoding of a packet using a block code.

1.3 Adversarial model

In our discussion, we assume that the physical communication channel is noiseless¹. We also assume that there is an attacker that will try to jam the channel, using a strategy best suited to its interests. The attacker is capable of sending jamming signals of arbitrary length at any time. And for any bit that the attacker jams, that bit is flipped with probability 1. A more realistic assumption would be to assume that the flipping probability is 0.5. However, both for sake of simplicity and as a worst-case analysis, we assume that all jammed bits are flipped with probability 1. Considering a flipping probability of 0.5 will lead to a higher throughput under the same jamming effort.

Parameters: To quantify the cost (jamming effort) of the attacker, we use the sum of the duration of all the jamming signals sent by the attacker when a packet of length nl is being sent. This total duration, measured in bits, is denoted as e . In addition, we define the jamming effort τ as: $\tau = e/nl$. τ is constrained by the code rate and the relative distance of the code $C(n, k, d)$ being used. We will analyze this relation in the subsequent sections. Our goal is to analyze the various techniques we proposed in terms of achievable throughput under a given jamming effort. We also talk about jamming efficiency which is defined as $1/\tau$.

¹ We plan to extend our results to combined noise, interference, and jamming.

1.4 Performance evaluation

We will evaluate the performance of the various jamming/anti-jamming schemes based on the overall achievable throughput. The throughput is the product of the code rate and the resulting frame success rate (i.e., 1-FER):

$$\text{Throughput} = \frac{(lk - s)(1 - FER)}{nl}.$$

(n, k, d) is the error-control code being used, l is the number of block in the packet, and s is the checksum length. The frame error rate is the probability that a packet cannot be correctly decoded. This is detected by checking the checksum. We will assume that the checksum is long enough such that all incorrectly decoded packets are detected by the checksum.

1.5 Traditional anti-jamming techniques

The jamming capability of a single symbol is a function of the the jammer power, the transmitter power, the antennas gains (from jammer to receiver, receiver to jammer, transmitter to receiver, and receiver to transmitter), the communication receiver bandwidth, the jamming transmitter bandwidth, the range between the transmitter and receiver, the range between the jammer and receiver, the jammer signal loss, and the communication signal loss [2]. Classical jamming consists in injecting an interfering signal that submerges the signal at the receiver. Several interfering waveforms can be used such as noise modulated FM, noise bursts, or continuous wave (CW) tone. The jammer can also play-back a previously recorded signal. Resistance to jamming is traditionally achieved by tuning various parameters such as transmission power, directional antennas, and receiver communication bandwidth. In the next paragraph, we describe one of the most common and efficient bit-level anti-jamming techniques.

Protection against jamming in wireless communication is usually achieved by using *spread spectrum* techniques [1]. These techniques force the jammer to spend much more energy than the sender. The typical value of the spread spectrum processing gain in military communication is between 20 dB and 30 dB. Spread spectrum technology uses a pseudorandom sequence to spread a signal over a much larger frequency band than what is required for its transmission. Correlating the received signal with the pseudorandom sequence carries out the despreading operation. There are two main spread spectrum techniques, namely: the direct sequence technique and frequency hopping. If the pseudorandom sequence is unknown to the jammer, then the spreading operation achieves a processing gain G in the signal-to-noise ratio. To successfully jam a communication the adversary would have to compensate this processing gain by increasing its transmission power.

Previous research in the area on anti-jamming has mainly focused on bit error probability of anti-jamming systems [13]. The main application being voice communication. In this paper, we are interested in techniques for *data packet* jamming. We assume that a bit-level anti-jamming technique, such as spread spectrum, can be used. We assume that jamming a single bit requires some constant effort. We investigate how this effort scales when a data packet such as in the IP protocol is transmitted.

2 Jamming Data Communication

In this section, we show how an adversary can jam existing WLAN when used to transmit IP packets. We also present the jamming effort for various modes of IEEE802.11, IEEE802.11a, IEEE802.11b, and Bluetooth.

2.1 Technique [Jamming no-ecc, ecc, interleaver+ecc]

A communication that is not protected with error-control codes (ECC) can be denied by destroying a single bit in each packet. Protection against single-bit errors is traditionally achieved using error-control codes. However, even error-correcting codes have a bounded error-correction capability (i.e., one-half of the minimum Hamming distance of the code). Practical codes cannot tolerate bursts of errors that exceed some small bound (e.g., a Hamming code is only able to correct a single bit and cannot tolerate two bit errors in the same block). In practice, a combination of an interleaver and an error-correction code is used. The interleaver spreads the burst of errors over multiple blocks, which allows reducing the number of errors per time window (or block) below the error-correction capability of the code. These are known techniques in the context of non-malicious interference. In traditional communication systems, the structures of the interleaver and ECC are publicly known. Therefore the attacker can choose which bits to jam such that, when de-interleaved, they will result in a burst of errors that exceeds the ECC capability.

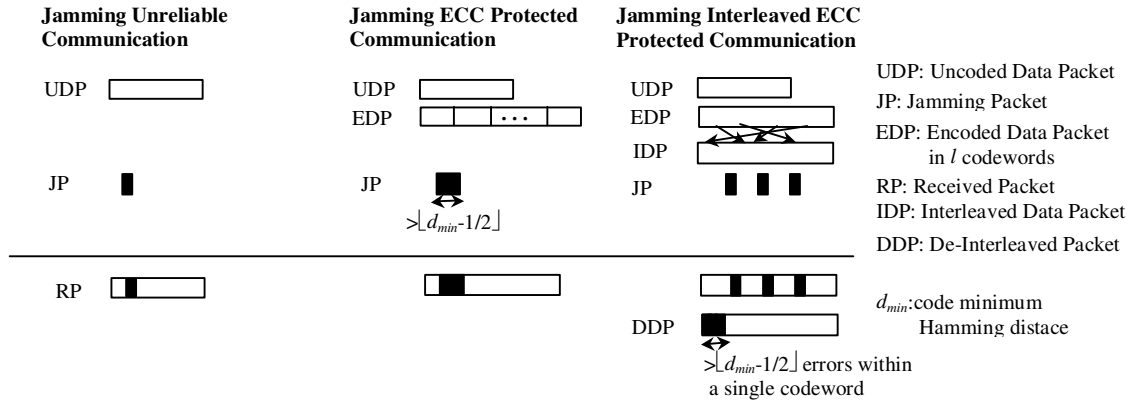


Figure 4. Low-power jamming of a data packet.

Figure 4 shows how an adversary can corrupt a data packet for three types of communication. A single interference pulse corrupts the whole packet when no error-correction is used. A jamming burst exceeding the error-correction capability of the code results in an unrecoverable error. Finally, if the structure of the interleaver is publicly known, the adversary can choose a sequence of interfering pulses that would result in an uncorrectable error after de-interleaving.

2.2 Jamming existing systems

2.2.1 IEEE802.11 and IEEE802.11b

IEEE802.11 sends the data using a Differential Binary Phase Shift Keying (DBPSK) modulation (1 Mbps) or Differential Quaternary Phase Shift Keying (DQPSK) (2 Mbps) [1, 14]. The bits are spread using an 11 chips Barker code. IEEE802.11 does not use any error-correction scheme. Therefore, a single interference pulse of length 1 bit (i.e., duration 1 μ s) can destroy an IP packet of size 1500 bytes (duration 12 ms or 6 ms depending on the modulation type). As a result, the jammer saves energy by a factor of 1/12000 or 1/6000. Table 1 summarizes the jamming efficiency (i.e., 1/(jamming effort)) of an adversary for IEEE802.11 modes.

IEEE802.11b uses a *complementary code keying* (CCK) modulation [15]. CCK allows transmissions at data rates of 5.5 Mbps and 11Mbps. The data stream is divided into symbols of 4 bits for the 5.5 Mbps data rate or symbols of 8 bits for the 11 Mbps data rate. If the jammer

destroys one symbol, it will succeed in destroying the whole packet. Therefore the jammer effort is 4/12000 for the 5.5 Mbps data rate and 8/12000 for the 11 Mbps data rate.

Modulation/coding Rate	Packet length IP packet	Number of bits needed to jam	Jammer Efficiency
BPSK	1500*8	1	12000
QPSK	1500*8	2	6000
CCK (5.5Mbps)	1500*8	4	3000
CCK (11Mbps)	1500*8	8	1500

Table 1: Jammer efficiency against IEEE802.11.

2.2.2 IEEE802.11a

IEEE802.11a has 8 possible data rates (i.e., 6 Mbps, 9 Mbps, 12 Mbps, 18 Mbps, ... , 54 Mbps). It uses various modulation techniques (i.e., BPSK, QPSK, 16QAM, 64QAM) and convolutional coding with various coding rates (i.e., 1/2, 2/3, 3/4). IEEE802.11a also uses an interleaver. Both the convolutional code and the interleaver are applied to blocks of bits. Each block of bits is separately encoded as an OFDM symbol (Orthogonal Frequency Division Multiple Access) [16]. The size of these blocks varies from 48 to 288 depending on the modulation and coding rates. The 48 bits per symbol encoding provides a 6Mbps data rate, while the 288 bits per symbol provides 54 Mbps data rate. If the adversary successfully jams a whole OFDM symbol, the whole IP packet will be lost. Table 2 summarizes the jamming efficiency against IEEE802.11a modes for an adversary to successfully destroy a typical IP packet. To compute the jamming efficiency we divide the size of an encoded IP packet by the number of bits per OFDM symbol. This is only the worst case scenario from the jammer's perspective. More efficient jamming can be achieved by destroying sub-OFDM symbols to exceed the error correction capability of the used codes.

Data Rate (Mbps)	Modulation	Coding Rate	Bits per Symbol = Bits to be Jammed	Encoded Packet length IP packet	Jammer Efficiency
6	BPSK	1/2	48	1500*8*2	500
9	BPSK	3/4	48	1500*8*4/3	333
12	QPSK	1/2	96	1500*8*2	250
18	QPSK	3/4	96	1500*8*4/3	167
24	16QAM	1/2	192	1500*8*2	125
36	16QAM	3/4	192	1500*8*4/3	83
48	64QAM	1/2	288	1500*8*2	62.5
54	64QAM	3/4	288	1500*8*4/3	55.5

Table 2. Jamming efficiency against IEEE802.11a.

2.2.3 Bluetooth

Bluetooth uses a Gaussian Frequency Shift Keying (GFSK) modulation combined with slow frequency hopping spread-spectrum technique [17]. Since it is simple for an attacker to recover the frequency hopping sequence, we will ignore the spreading gain against a malicious attacker. Bluetooth recovers from errors using three techniques: ARQ retransmissions, (15, 10, 4) shortened Hamming code, or 1/3 repetition code. Only the (15, 10, 4) code and ARQ are used with data packets. The data packets have various sizes and error-coding schemes. They are designated by the standard as DH1, DH3, DH5, DV, DM3, and DM5. Table 3 summarizes these packet sizes and error-coding schemes. These coding schemes are easy to overcome. When the

ARQ scheme is used, it is sufficient to destroy a single bit in order to systematically generate a CRC error. The (15, 10, 4) code has a minimum distance of 4 and therefore can be exceeded by jamming two bits. Bluetooth does not have any interleaving² scheme.

Packet Type (data only)	Number of bits	Number of bits needed to jam	Jammer Efficiency
DH1 (no ECC)	$28 \times 8 = 224$	1	224
DM3 (15, 10, 4)	$123 \times 8 = 984$	2	$984/2 = 492$
DH3 (no ECC)	$185 \times 8 = 1480$	1	1480
DM5 (15, 10, 4)	$226 \times 8 = 1808$	2	$1808/2 = 904$
DH5 (no ECC)	$341 \times 8 = 2728$	1	2728
DV (15, 10, 4)	150	2	75

Table 3. Jamming efficiency against Bluetooth data packets.

3 Direct application of binary codes

In this section, we investigate a direct use of the performance of binary error-correction codes. We are interested in figuring out the best performance that could be achieved. Therefore, we do not consider issues related to actually constructing the best codes or being able to decode them. Subsequent sections will consider more practical constraints.

3.1 Single codeword binary code

The most direct approach to resist to jamming is to use the best known codes. In [18], a compilation of upper bounds on binary codes for values of n below 28 is presented. [12] provides a table of upper bounds on best known binary codes for values of n within the interval $[28, 512]$ and for values of minimum distance $d \leq 29$. In order to assess the best we can do against jamming, we have plotted the coding rate required to resist a jamming effort of 15% and 20%. To be able to resist to a jamming effort of τ , the error code has to verify the following constraint: $d > 2 \times \tau \times n$. Figure 5 shows the upper bound on the coding rate derived from the upper bound on binary codes. Here the coding rate is computed as the ratio of k and n . We do not take into account the checksum overhead given the short length of the codes. Using only short codes per checksum would be extremely inefficient. In Section 3.2, we will analyze the use of multiple short codes combined with a single checksum. Only values up to $n = 95$ are used because longer codes, that resist to the jamming effort we are considering, require higher minimum distance than given by the tables (i.e., the maximum minimum distance given in [12] is 29). The up and downs in the curves are a result of the discrete characteristic of the codes.

² Whitening is used against DC bias and is applied before encoding. Therefore it doesn't help against errors.

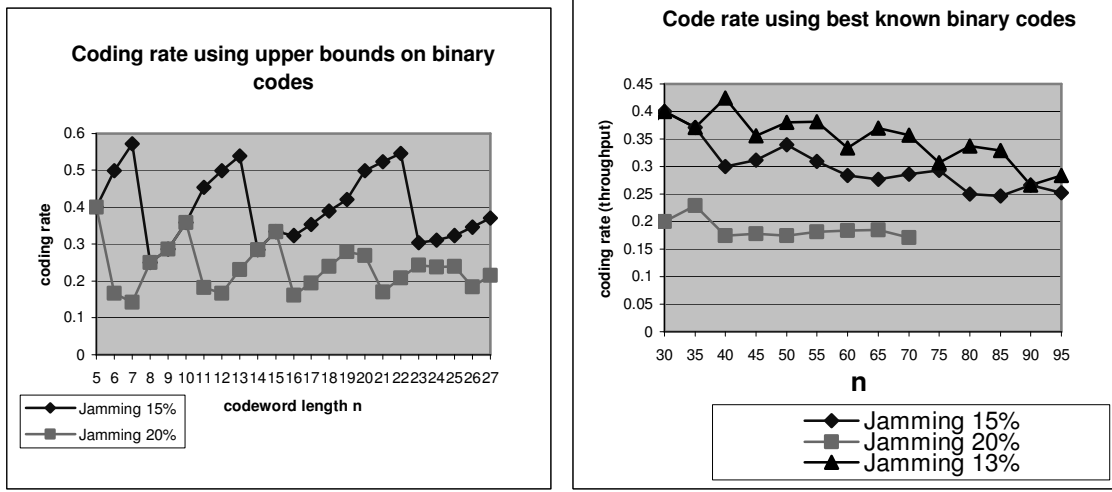


Figure 5: Upper bound on coding rate against jamming effort of 15% and 20%. The first figure is derived from [18] for $n < 28$, the second figure is derived from [12] and provides bounds for best known codes with $n < 100$.

We can conclude that reasonable throughput can be maintained against 20% jamming effort. This should be contrasted with the performance of existing WLAN. The latter would lead to a throughput that is almost zero for a jamming effort much lower than 10%. Also to understand why we claim that this performance is good, one can consider the theoretical bound of capacity for a 50% jamming. From Shannon's theorem, if the jamming effort is 50% (with flip probability 1) then the channel capacity is 0 whatever the coding scheme used.

Even if the achievable throughput can be maintained at a reasonable value using best codes, it is both very difficult to find these codes, and also difficult to decode them efficiently. This becomes specially true when n is increased beyond 100 bits. Using only short codes (i.e., $n < 20$) is not reasonable because of the checksum overhead. The following sections will investigate some alternatives.

From a theoretical perspective, an alternative approach to estimate the best we can do against a given jamming effort would be to use Shannon's theorem to determine the capacity of the channel under jamming. The underlying assumption would be that packets are arbitrarily long, the jamming effort is constant, and that the code construction and decoding complexity are not an issue.

3.2 Multiple codeword binary code

Because of the limitations of using a single codeword, it is natural to think of using multiple codewords that are interleaved together. The structure of the interleaver can be cryptographically designed to prevent the adversary from jamming selected bits that would be de-interleaved within the same codeword block. The communicating nodes can share a secret key that determines the structure of the interleaver at each communication. We assume that there exists a cryptographically strong interleaving function and that, for each transmission, the packet is seen as randomly interleaved. Therefore the effect of the jammer is seen as randomly picking e bits and flipping their values. We will keep this assumption in the remaining of the paper. The coding procedure is as follows:

1. The data packet (including a checksum) is divided into l blocks of k bits each.

2. Each block is encoded using a (n, k, d) binary code.
3. The nl bits are cryptographically interleaved with a secret shared key³.

The decoding procedure is as follows:

1. The packet is deinterleaved.
2. Each block is separately decoded.
3. The packet is kept only if the checksum is correct.

The error-detection is achieved using the checksum. The probability of not detecting an erroneous decoding is considered to be negligible (in the order of 2^{-16}).

Proposition 1: Let e be the total number of bits that an attacker jammed during the packet transmission, and let X_i be the number of error bits in codeword i ($1 \leq i \leq l$). Then, the distribution of X_i is multivariate hypergeometric:

$$\Pr\left[\bigcap_{i=1}^l X_i = e_i\right] = \frac{\prod_{i=1}^l \binom{n}{e_i}}{\binom{nl}{e}}; \left(\sum_{i=1}^l e_i = e; 0 \leq e_i \leq n\right).$$

The probability that some specified codeword contains x errors is:

$$\Pr[X = x] = \frac{\binom{l}{x} \binom{(l-1)n}{e-x}}{\binom{nl}{e}}.$$

Proposition 2: The probability that a packet is dropped when e bits are jammed is the probability that at least one codeword has more than $t = \lfloor (d-1)/2 \rfloor$ errors (error-correction capability of the code) is:

$$\Pr\left[\bigcup_{i=1}^l X_i > t; \left(\sum_{i=1}^l X_i = e; 0 \leq X_i \leq n\right)\right].$$

The probability in Proposition 2 can be evaluated using Proposition 1, which unfortunately is tedious [19]. Since it is very difficult to compute the packet error probability using a closed form formula or an analytical approach, we have simulated this probability for various values of n , k , d , and l . Figure 6 shows that the probability of packet error increases rapidly to 1 for a constant jamming effort (7%, 14%, and 28%). We simulated two short codes with good minimum distance and coding rates: the Hamming code which is perfect and a Preparata code (15, 8, 5).

³ In this paper we do not address how this shared key can be established. In the worst case it could be setup offline.

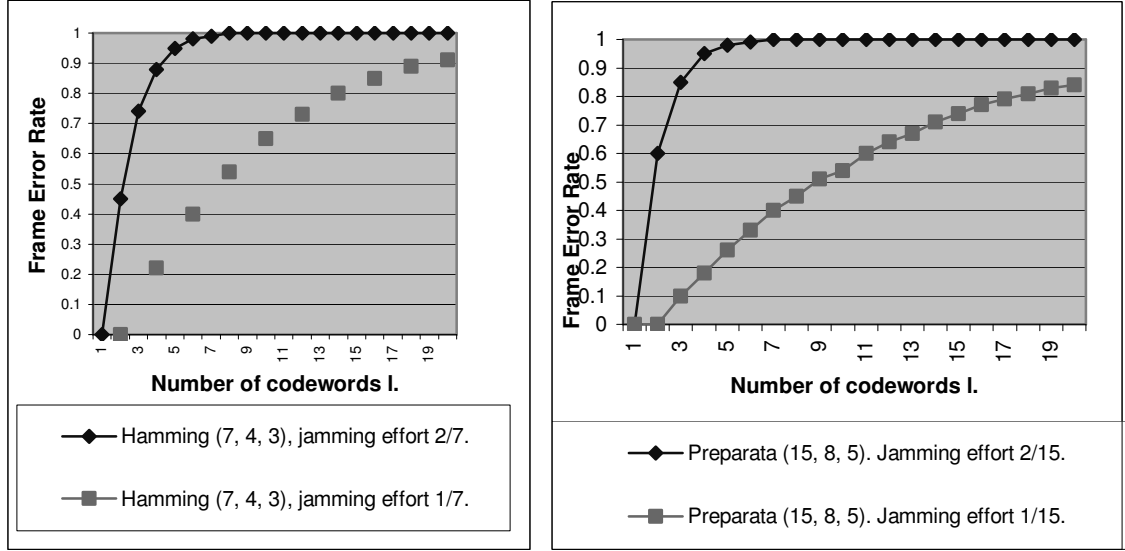


Figure 6: Frame error as a function of the number of codewords.

4 Concatenated Codes

Because of the difficulty to find and decode reasonably long binary codes and because of the poor performance of multiple short codewords, we propose to investigate a set of concatenated codes. The concatenated codes are defined as the use of two error correction codes in sequence. The first code $(n_2, k_2, d_2)_{q_2}$ is called an outer code and the second code to be used is called an inner code $(n_1, k_1, d_1)_{q_1}$ [1]. The sequence of symbols of the outer codeword is then encoded using the inner code.

We propose a concatenated code using a Reed-Solomon (RS) code as the outer code and a good short block code as the inner code. The advantages of using RS codes [20] are its flexibility and easy decoding. The flexibility of RS codes allows us to adapt the code rate/error correction capability to the jamming effort. This can be done by retransmitting additional redundancy only when necessary (i.e., channel being jammed). This process is usually called Hybrid-ARQ type II. The overall coding rate of the concatenated code is the product of the coding rate of the inner code by the coding rate of the outer code: $r = r_1 * r_2$ (i.e., $k_1 * k_2 / n_1 / n_2$). The throughput of a concatenated code is $r_1 * r_2 * (1 - FER)$. Finally, for a RS outer code, the frame-error rate of the concatenated code is $FER = \text{Prob}[(\text{number of inner code decoding errors}) > (n_2 - k_2) / 2]$.

We consider the generalization of RS-code that describes them as non-binary Maximum Distance Separable (MDS) code that can be characterized as $(n, k, n - k + 1)_{q_2}$, such that $k \leq n \leq q$ [21]. For more information on the construction of RS-codes, the reader is referred to [20]. In this paper, we only consider their properties. RS codes flexibility proved to be useful in various contexts [22, 23]. When combined with an inner code, the number of codewords of the inner code limits the number of RS code symbols. Therefore the inner code cannot be too short. We have considered two inner codes taken from [18], namely: the Preparata code (15, 8, 5) and the (26, 10, 9).

The coding procedure is as follows:

1. The data packet (including a checksum) is divided into k_2 blocks of k_1 bits each.

2. Each block of k_1 bits is seen as a symbol of the outer-code (RS). These k_2 symbols are encoded using the $(n_2, k_2, n_2-k_2+1)_2^{k_1}$ RS code. This results into n_2 symbols (blocks) of k_1 bits each. This can only be done if $n_2 \leq 2^{k_1}$.
3. Each block of k_1 bits is now encoded using the inner code leading to blocks of n_1 bits.
4. The resulting $n_1 * n_2$ bits are cryptographically interleaved.

The decoding procedure is as follows:

1. The packet is deinterleaved.
2. Each (n_1, k_1, d_1) block is separately decoded.
3. The RS code is decoded.

We simulated this scheme under various jamming efforts. Figure 7 shows the performance of the two concatenated codes under 13% and 15% jamming effort. For each value of n_2 we choose the value k_2 that maximizes the throughput function assuming a constant jamming effort of 13% or 15%.

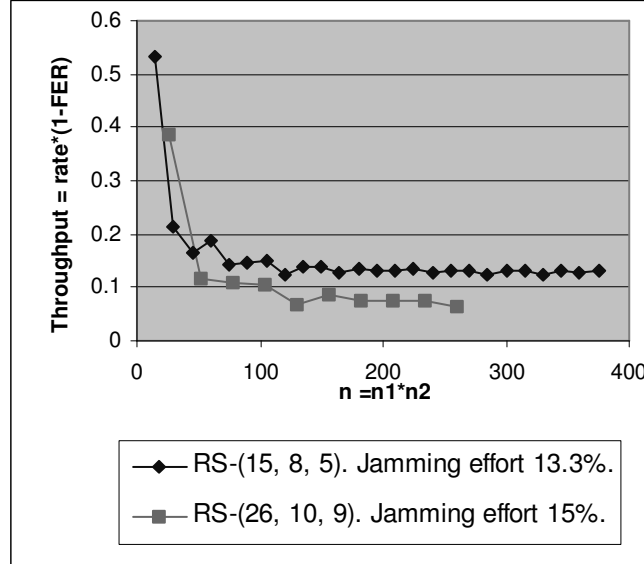


Figure 7: Concatenating Reed-Solomon codes with a short code maintains a 8% throughput even under 15% jamming effort, and a 13% throughput under 13.3% jamming effort.

Figure 7 shows that the concatenated codes maintain good performance even for large values of $n=n_1*n_2$. For example, if we compare the best known code for jamming effort 13.3% to the performance of the concatenated RS-(15, 8, 5) code for length 105 bits we obtain the following⁴:

- The best known code that tolerates 13.3% jamming effort needs a minimum Hamming distance of 29 ($105*0.133*2+1$). From [12] this would correspond to a (105, 29, 29). Therefore its coding rate (or throughput under 13.3% jamming) would be $29/105 = 0.276$.
- The best combination of RS-(15, 8, 5) would be with a $(7, 3, 5)_8$ RS code with symbols of 8 bits concatenated with the (15, 8, 5) code. From our simulation of 13.3% bits in error this code provides a throughput of 0.15 (see Figure 7).

⁴ The values of $n = 105$, and jamming effort 13.3% are chosen in order to simplify the comparison by getting integer values.

The performance of the proposed concatenated code is 0.15, which is better than half the performance of the best known code ($0.15 > \frac{1}{2} \cdot 0.276$). Although the concatenated code is not the best in performance it has several advantages. It is much easier to decode (even in software) than a long best known code. Some of the best known codes can only be decoded using exhaustive search, which is unrealistic for practical applications. One of the most important advantages of RS codes is their flexibility in providing more error correction capability on demand. In the first transmission the sender only needs to send a small number of redundancy symbols. If the receiver is unable to decode the packet than the transmitter can send additional redundancy symbols [20]. This property can be used to design an anti-jamming hybrid ARQ protocol that is adaptive to the jamming effort of the adversary.

5 The case of IEEE802.11b

Reed-Solomon-like codes (RS-codes) are particularly efficient codes, however they are non-binary. A single bit error has the same effect as a symbol error. Therefore they are not suitable for correcting bit errors. However, combining RS-codes with modulation schemes that transmit multibit symbols can lead to efficient anti-jamming techniques. The assumption here is that the adversary destroys the whole symbol.

CCK communication used in IEEE802.11b transmits 8 bits in each symbol (when used at 11Mbps). Thus, we can combine it with an RS-code of symbol size 8 bits. Since the symbol size is 8 bits, it is possible to create an RS-code of maximum length 256 symbols. For example, if k is taken equal to 85, the adversary needs to jam $(256-85-1)/2 = 85$ bytes to destroy the data packet. Therefore, the jamming effort has to be $1/3$. Furthermore, the data rate (or throughput) is still reasonable at $85/256 = 1/3$. Figure 8 shows the jamming effort that can be tolerated and the corresponding data rate for various values of k . In other words, the throughput is a linear function of the jamming effort. It decreases from 1 to 0 when the jamming effort increases from 0 to 0.5.

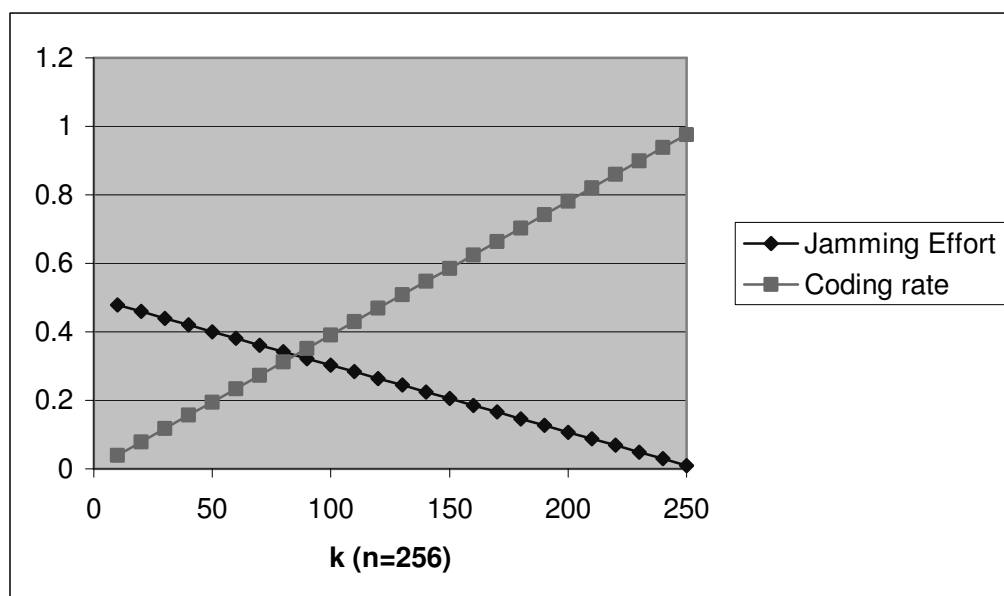


Figure 8: Jamming effort and coding rate (or throughput) for various RS encoding schemes.

6 Conclusion and future work

In this paper, we have investigated the problem of denial of service against data packets (e.g., IP packets) transmitted over WLAN protocols (i.e., IEEE802.11 and Bluetooth). Our results are as follows:

- We have shown that it is easy for an attacker to jam such a transmission at an energy cost that is much lower than the transmitter's cost. Such attacks cannot only prevent communication within large areas for long periods of time but can also lead to other more elaborate and coordinated attacks such as partitioning of a multihop ad hoc network or forcing packets to be routed over chosen paths.
- We have analyzed the performance of the best known binary codes.
- Finally, we have proposed and analyzed the performance of some Reed-Solomon concatenated codes. The advantages of such codes are their flexibility to achieve adaptive anti-jamming, long codewords, and simple decoding.

As future directions for research we plan to investigate the performance of hybrid-ARQ type II based on Low Density Parity Check codes (LDPC) against dynamic jamming efforts. In a practical setting, the communication is not always under attack. If the communicating nodes are always using excessive error-correction codes, then they will waste bandwidth. Therefore, the communicating nodes should use an adaptive scheme that increases the resistance to jamming whenever an attack is detected. The proposed approach is to use a Hybrid-ARQ scheme in the setting of adaptive anti-jamming. Because of the inter-dependence of the protocol layers in wireless networks, we will also investigate the impact of a multi-layer DoS attack in MANET.

Reference:

1. Bernard Sklar, "*Digital Communications, Fundamentals and Applications*". 2nd ed. 2001: Prentice-Hall.
2. Curtis D. Schleher, "*Electronic Warfare in the Information Age*". 1999, Norwood, MA: Artech House.
3. Yih-Chun Hu, Adrian Perrig, and D.B. Johnson. "*Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks*". in *Proceedings of ACM Mobicom*. 2002. Atlanta, GA: ACM Press.
4. P. Papadimitratos and Z.J. Haas, *Securing Mobile Ad Hoc Networks*, in *Handbook of Ad Hoc Wireless Networks*, M. Ilyas, Editor. 2002, CRC Press.
5. Pradeep Kyasanur and N. Vaidya, "*Detection and Handling of MAC Layer Misbehavior in Wireless Networks*". August 2002, UIUC.
6. Bridget Dahill, et al., "*A Secure Routing Protocol for Ad Hoc Networks*". 2001, Electrical Engineering and Computer Science, University of Michigan.UM-CS-2001-037,
7. Jean-Pierre Hubaux, Levente Buttyan, and S. Capkun. "*The Quest for Security in Mobile Ad Hoc Networks*." in *Proceedings of MobiHoc'01*. 2001: ACM Press.
8. Sergio Marti, et al. "*Mitigating Routing Misbehavior in Mobile Ad Hoc Networks*". in *Proceedings of Sixth Annual IEEE/ACM International Conference on Mobile Computing and Networking (MobiCom 2000)*. 2000: ACM Press.
9. Frank Stajano and R. Anderson. "*The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks*." in *Proceedings of Security Protocols, 7th International Workshop*. 1999: Lecture Notes in Computer Science, Springer Verlag.

10. Lidong Zhou and Z.J. Haas, "*Securing Ad Hoc Networks*". IEEE Networks Magazine, 1999. **13**(6): p. 24-30.
11. Shu Lin and D.J. Costello, "*Error Control Coding : Fundamentals and Applications*". 1983: Pearson Education.
12. W. C. Huffman and V.S. Pless, eds. "*Handbook of Coding Theory*". Vol. 1. 1998, Elsevier Science.
13. Jim K. Omura and B.K. Levitt, "*Coded Error Probability Evaluation for Antijam Communication Systems*". IEEE Transactions on Communications, 1982. **30**(5): p. 896-903.
14. IEEE, "*Draft Supplement to Standard. Part 11: Wireless Lan MAC and PHY Specifications: High Speed Physical Layer in the 5GHz Band*". 1999, IEEE Press
15. Bob Pearson, "*Complementary Code Keying Made Simple*". 2001, Intersil.AN9850.2 <http://www.intersil.com/data/an/an9/an9850/AN9850.pdf>.
16. Juha Heiskala and J. Terry, "*OFDM Wireless LANs: A Theoretical and Practical Guide*". 2001: Pearson Education.
17. Bluetooth. <http://www.bluetooth.com>.
18. Erik Agrell, Alexander Vardy, and K. Zeger, "*A Table of Upper Bounds for Binary Codes*". IEEE Transactions on Information Theory, 2001. **47**(7): p. 3004-3006.
19. Norman Johnson, Samuel Kotz, and N. Balakrishnan, "*Discrete Multivariate Distributions*". Wiley Series in Probability and Mathematical Statistics. 1997, New York: John Wiley & Sons.
20. Stephen B. Wicker and V.K. Bhargava, eds. "*Reed-Solomon Codes and Their Applications*". 1999, IEEE Press.
21. Lloyd R. Welch and E.R. Berlekamp, "*Error correction of algebraic block codes*." US Patent, 4,633,470, 1986.
22. Jörg Nonnenmacher, Ernst W. Biersack, and D. Towsley, "*Parity-Based Loss Recovery for Reliable Multicast Transmission*". IEEE/ACM Transactions on Networking, 1998.
23. Guevara Noubir. "*Collision-Free One-Way Communication Using Reed-Solomon Codes*". in *Proceedings of IEEE International Symposium on Information Theory and Applications*. 1998. Mexico City.