Efficient Uncoordinated FHSS Anti-jamming Communication

Mario Strasser
Communication Systems Group
ETH Zurich, Switzerland
strasser@tik.ee.ethz.ch

Christina Pöpper
System Security Group
ETH Zurich, Switzerland
poepperc@inf.ethz.ch

Srdjan Čapkun
System Security Group
ETH Zurich, Switzerland
capkuns@inf.ethz.ch

ABSTRACT

We address the problem of jamming-resistant communication in scenarios in which the communicating parties do not share secret keys. This includes scenarios where the communicating parties are not known in advance or where not all parties can be trusted (e.g., jamming-resistant key establishment or anti-jamming broadcast to a large set of unknown receivers). In these cases, the deployment of shared secret keys is unrealistic, and therefore this problem cannot be solved using existing anti-jamming solutions like FHSS and DSSS that depend on pre-shared keys. Recently, a solution to this problem has been proposed that introduces Uncoordinated Frequency Hopping (UFH), a new spread-spectrum anti-jamming technique that does not rely on secret keys. In this work, we investigate the efficiency of UFH-based communication: we identify optimal strategies for the UFH frequency channel selection and we propose a set of new UFH-based anti-jamming schemes that, compared to the original UFH proposal, reduce the communication latency up to one-half (i.e., increase UFH communication throughput up to two times).

Categories and Subject Descriptors

C.2.0 [**Computer-communication networks**]: General—*Security and protection*

General Terms

Algorithms, Performance, Security

Keywords

Anti-jamming, Frequency Hopping, Wireless Security

1. INTRODUCTION

A major limitation of common anti-jamming techniques (such as FHSS and DSSS) is their dependency on a secret shared by the sender and the receiver; the secret is required to coordinate the used frequency channels or code sequences and must not be known to an attacker. This dependency precludes the application of common anti-jamming techniques in scenarios where the parties cannot resort to shared secrets; these include anti-jamming key establishment and anti-jamming broadcast to a (partially) unknown or untrusted group of receivers. More precisely, consider the following example: If a sender wants to broadcast a message to a set of receivers in a jamming-resistant manner, he would need to share a secret code with all receivers, and the code would need to be hidden from the attacker. In a number of scenarios—such as those where (some) receivers cannot be trusted or where they are not known in advance—the assumption about shared secret codes is unrealistic and will prevent the application of anti-jamming communication.

This problem has recently been identified [26] and a solution has been proposed that introduces Uncoordinated Frequency Hopping (UFH), a new spread-spectrum anti-jamming technique that does not rely on shared secret keys. With UFH, two communicating nodes hop among a set of known frequency channels in an uncoordinated and random manner. The communication is based on the observation that, at some points in time, the sender and the receiver will be sending and listening on the same frequency channel. Due to the sender's rapid change of output channels (as countermeasure against a jammer), the transmitted message does not fit in one transmission slot, but has to be split into fragments. The fragments are then transmitted in packets one after another with a high number of repetitions. Although splitting the message into fragments is a straight-forward operation, the reassembly of the message at the receiver is non-trivial if an attacker inserts additional fragments or modifies transmitted ones. This insertion (pollution) attack can exponentially increase the computational complexity of the message reassembly process (see Figure 2), making it infeasible for the receiver to reconstruct the message [12]. In [26], this attack is prevented by means of hash links which link individual message fragments.

In this work, we focus on the efficiency of UFH-based communication and make the following contributions: First, we describe a frequency channel selection strategy for UFH communication and prove its optimality. In particular, we show that, in UFH, the probability that a packet is successfully received is maximized if the sender and receiver choose the frequency channels on which they send and receive uniformly at random from a set of common frequency channels. We further derive an expression for the optimal size of the set of channels as a function of the attacker's strength.

MobiHoc'09, May 18–21, 2009, New Orleans, Louisiana, USA.

Second, we review possible approaches for message fragmentation (encoding) in UFH communication. We propose several new UFH communication schemes based on Erasure codes [16], cryptographic accumulators [6], and short signatures [7]. We show that, compared to the original proposal based on hash links, these schemes reduce the UFH communication latency up to one-half (i.e., increase UFH throughput up to two times).

Third, we analyze the anti-jamming protection of individual packet transmissions. For different attacker (jammer) types, we identify error-correcting schemes that offer an appropriate packet anti-jamming protection.

The remainder of the paper is organized as follows: In Section 2, we describe our system and attacker model. We present background on UFH and outline our solution in Section 3. In Section 4, we evaluate verifiable message coding techniques and discuss packet codings in Section 5. We identify the optimal channel selection strategy in Section 6, discuss related work in Section 7, and conclude in Section 8.

2. SYSTEM AND ATTACKER MODEL

We adopt the system and attacker models from [26] and consider a sender and a (group of) receiver(s) that can hop within a set $\mathcal{C}$ of $c = |\mathcal{C}|$ available frequency channels (spanning a band of typically several hundred channels). The number of channels on which the sender and a receiver can send and receive on in parallel is denoted by c_n and c_m, respectively.

The attacker aims at interfering with the sender's transmission such that she prevents the receiver(s) from obtaining any useful information. For this purpose, the attacker adds her own signals to the radio channel and thus can insert self-composed and replayed packets or can jam and modify ongoing packet transmissions. We denote by c_j (c_s) the number of channels on which the attacker can jam (sense) in parallel and by t_j (t_s) the time that she needs to switch the jamming (sensing) channels.

Following prior classifications [22, 26], we distinguish between static, sweep, random, responsive, and hybrid jammers. Static, sweep, and random jammers do not sense for ongoing transmissions but permanently jam on c_j channels. *Static* jammers remain on the same channels for much longer than the transmission time of a packet. *Sweep* jammers systematically update the jamming channels in a way that after $\lceil \frac{c}{c_j} \rceil$ jamming cycles all channels have been jammed once (but they do not have to follow a particular order). *Random* jammers always choose c_j channels at random and might thus jam the same channels several times before having hit all channels. *Responsive* jammers, on the other hand, differ in that they initially solely sense for ongoing transmissions and enable the jamming channels only when a signal has been detected. *Hybrid* jammers, finally, are a combination of responsive and permanent jammers that have their jamming channels already enabled while scanning for signals, i.e., hybrid jammers can sense and transmit independently. Of the introduced jammer types, responsive-sweep jammers are the most powerful ones [26].

3. MOTIVATION & SOLUTION OUTLINE

With Uncoordinated Frequency Hopping, the communicating devices hop among a set of known frequency channels in an uncoordinated and random manner; the sender

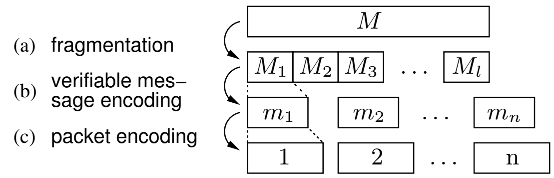

Figure 1: Message fragmentation. For the UFH transmission, the message is (a) split into fragments, then the fragments get message encoded (b), and packet encoded (c). Finally, the sender transmits the packets repeatedly on randomly selected frequency channels.

typically hops with a higher speed than the receiver. Information is transferred when the receiver happens to listen on the same frequency channel on which the sender is currently transmitting. In order for (coordinated or uncoordinated) frequency hopping to be effective against jamming, the time slots of the sender must be kept short (i.e., at most few hundred bits). Messages—in particular if they are authenticated—thus do typically not fit into the sender's short transmission slots and are split into fragments by the sender and reassembled by the receiver (see Figure 1). After the fragmentation, the sender encapsulates each fragment into a packet, error-encodes the packet, and repetitively transmits all message packets on randomly selected frequency channels.

3.1 Message Reassembly and Verification

Whereas UFH message fragmentation is efficient, the reassembly of the message at the receiver is non-trivial and can become very inefficient if the attacker inserts additional fragments or modifies transmitted ones. Consider that a legitimate message is divided into l fragments and that N adversarial packets successfully arrive at the receiver. Then, the number of possible messages that the receiver must reassemble and verify is in $O((\frac{N}{l})^l)$; in a typical system, where $l = 10$ and where the attacker can insert $N = 100$ unique packets during the legitimate message transmission, the receiver would need to reassemble and verify 10^{10} messages [26]. If l is not predefined, adversarial insertions may even lead to an exponential number of message reassemblies and verifications at the receiver (i.e., $O(q^{\lfloor N/q \rfloor})$, where q is the number of unique packets that the attacker inserts per legitimate message fragment, see Figure 2).

Reducing the impact of maliciously inserted fragments therefore requires measures that allow for the efficient identification of sets of fragments that belong to the same message (without using a shared key). In the basic UFH scheme [26], this is achieved by linking all message fragments to form a hash chain where each fragment is linked to its successor with a hash; given a collision-resistant hash function, the attacker cannot create branches in the sender's packet chains and is restricted to creating independent fragment chains or to merging partial chains into the sender's chain. Thus, in the basic UFH scheme, the number of messages that the receiver must reassemble and verify remains linear in the number of packets (fragments) that the receiver receives.

We point out that the UFH message transfer scheme is not intended to provide message authentication or confidentiality. These security goals can be achieved on the appli-

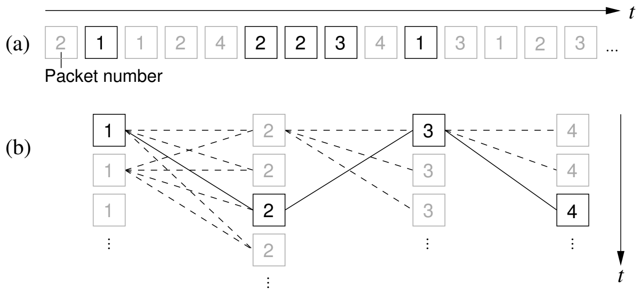

Figure 2: Message reassembly. (a) Example of the packet arrival at the receiver; black packets were sent by the sender, gray packets were inserted by the attacker (at the time of reception they cannot be distinguished). (b) The receiver sorts unique packets according to their fragment number (and message id). Without verifiable message coding, the receiver must reassemble and check all combinations of packets (indicated by dashed lines, only a subset is shown). Verifiable message coding enables an efficient reassembly of valid combinations (solid lines).

cation layer, which runs on top of the UFH scheme, e.g., by making use of public key cryptography, timestamps, and message buffering. One application of UFH in combination with digital signatures is described in [26] in the context of key establishment.

3.2 Solution Overview

For UFH-based communication, we consider two parameters fixed: the message size $|M|$ (which is usually given by the application) and the size s of the frequency hopping slots (usually given by the radio device). With these parameters, the throughput/latency of UFH communication depends on (i) the number of packets that the receiver must receive to reconstruct the message and (ii) the probability that a packet sent by the sender is successfully received by the receiver; the latter is again determined by the strength of the attacker, the number of frequency channels used, and the jamming resistance of the packets. We next outline how UFH communication performance can be improved/optimized (i.e., how higher throughput and lower transmission latency can be achieved) by a proper choice and parametrization of the message coding, packet coding, and channel selection.

Message Coding: An obvious limitation of the basic hash-linked scheme is that all fragments of a message must be received before they can be verified and the message be reassembled. The new message coding schemes that we propose in this paper use erasure codes to avoid this limitation and allow for the verification of fragments and for message reassembly even if only a subset of all fragments is available (Section 4). Common to these and the basic UFH scheme is that they must be efficient in verifying if received fragments belong to a specific message.

Packet Coding: Without packet encoding, the attacker could destroy each packet with minimal effort (e.g., by jamming one bit of the packet). Packet encoding makes the packets resistant to a certain number of bit errors, forcing the attacker to jam a minimal fraction ρ of a packet to make it undecodable by the receiver. We discuss suitable packet coding schemes in Section 5.

Frequency Channel Selection: We show in Section 6 that, given an attacker that blocks c_b channels, the optimal strategy (in terms of throughput) for the sender and receiver is to hop randomly within a set of $\approx 2c_b$ channels. We also show that the performance of UFH gracefully degrades the less accurate the assessment of c_b is.

4. VERIFIABLE MESSAGE CODING

As mentioned in Section 3, a limitation of the basic hash-linked UFH message transfer scheme is that all fragments of a message must be received before the message can be reconstructed. Erasure codes [16] allow for schemes where a message can be reconstructed if only a subset of all fragments is available. We distinguish:

Erasure Codes: Optimal erasure codes encode a message M into n fragments of size $|M|/l$ such that any subset of l fragments can be used to reconstruct M. Near optimal erasure codes are more efficient than optimal codes in terms of coding complexity and memory usage but require a fragment size of $|M|/(l-\varepsilon)$ in order to reconstruct M using l fragments. The constant ε is a code parameter that can usually be reduced at the expense of a higher coding complexity. Examples of (near) optimal erasure codes are: Reed-Solomon [28] and Tornado [16] codes.

Rateless Erasure Codes: Rateless erasure codes (also called fountain codes) do not generate a finite set of n fragments but a (potentially) infinite fragment sequence. The encoded message can be reconstructed from any set of l different fragments. Examples of efficient near optimal fountain codes are: Online [18], LT [15], and Raptor [24] codes.

In what follows, we denote by $\mathcal{E}(n, l, \varepsilon)$ an erasure code that encodes a message M into n fragments $M_1, M_2, \ldots, M_n$, $|M_i| = |M|/(l-\varepsilon)$, such that any subset of $l \leq n$ fragments can be used to reconstruct M; for optimal codes $\varepsilon = 0$ and for rateless erasure codes $n = \infty$. Despite their ability to cope with fragment losses, erasure codes are susceptible to intentional fragment insertions or modifications (pollution attack [12]); that is, they cannot distinguish correct from modified or phony fragments. Hence, the receiver cannot do better than to try all possible fragment combinations (Figure 2). Limiting the impact of maliciously inserted fragments thus requires measures that allow for the efficient identification of sets of fragments that belong to the same message (without relying on a shared key). The therefore required overhead per packet must be kept as low as possible to avoid that the advantage gained by the erasure coding is nullified by a largely increased number of required packet receptions (due to more fragments that the message needs to be split into). Given these constraints, we require a packet verification technique to fulfill the following requirements:

Time Efficiency: The time to verify (i.e., identify as either belonging to a specific message or being invalid) a packet m_i must be in $O(N+n)$, where N is the total number of received packets and n is the number of fragments per message.

Space Efficiency: The overhead per packet that is required for the verification must be in $O(n)$.

We define a set of packets $\mathcal{M}$ as being *verifiable* with respect to an erasure code $\mathcal{E}(n, l, \varepsilon)$ and a message M if at least l of the packets in $\mathcal{M}$ can be verified as belonging to the message M. We next present three packet verification techniques that fulfill the requirements above and can be used in combination with (rateless) erasure codes to build verifiable message coding. We assume a security level of

k bits for the cryptographic primitives (i.e., a strength comparable to a symmetric key of k bits) and denote by $h(\cdot)$ a weak collision-resistant cryptographic hash function with an output length of k bits.

4.1 Multiple Hash Links

This verification technique generalizes the hash-linking approach of the basic UFH scheme [26]. However, here each packet is linked not only to its successor but to the next α packets (see Figure 3). More precisely, once a message M has been erasure-encoded into the fragments $M_1, M_2, \ldots, M_n$ using $\mathcal{E}(n, l, \varepsilon)$, each fragment M_i is encapsulated into a packet $m_i := id||i||l||M_i||h_{i+1}|| \ldots ||h_{i+\alpha}$ by adding the message id, the fragment number i, the required number of fragments l, and the hash values of the packets m_{i+1} to $m_{i+\alpha}$. For the non-existing packets m_{n+1} to $m_{n+\alpha-1}$ the hash value of the entire message is used (i.e, $h_i := h(m_i)$ for $1 \leq i \leq n$, $h_i := h(M)$ otherwise).

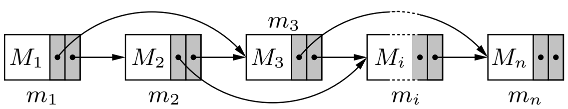

Figure 3: Multiple hash links. Each packet contains the hash value of the next α packets.

A set $\mathcal{M}$ of $|\mathcal{M}| \geq l$ packets is verifiable if at least l packets in the set are connected (i.e., have gaps of $< \alpha$ missing packets between them). More formally, the set $\mathcal{M}$ is verifiable if and only if $\exists \mathcal{M}'' \subset \mathcal{M}' \subseteq \mathcal{M}$ such that $|\mathcal{M}''| = l-1$, $|\mathcal{M}'| = l$, and $\forall m_i \in \mathcal{M}'' : \exists j \in \{i+1, i+2, \ldots, i+\alpha\} : m_j \in \mathcal{M}'$ (i.e., all but the last packet in a chain must have a valid successor to which they link).

Upon reception of a new packet m_i, the receiver must identify all packets that link to m_i or to which m_i links. This can be done by traversing all N already received packets once. Note that although each packet links to α other packets, all these α packets are successors of m_i and part of the same chain. Each packet is thus the head of exactly one unique sub-chain and at most $N-1$ chains join at a packet. These joining chains build a reverse tree that is rooted at the packet. Finding the heads of these chains (i.e., finding the leaves of the reverse three) can be done in $O(N)$ steps. Hence, the cost to verify m_i (i.e., to find all chains the packet is part of) is in $O(N)$. The verification-related overhead per packet is αk bits, where k is the length of a hash value.

In order to violate the integrity of this verification scheme, the attacker must create a branch in a chain by inserting a packet m'_i such that both m_i and m'_i are accepted by the receiver as valid successors of m_{i-1}, i.e., the attacker must find a packet $m'_i := id||i||l||M'_i||h'_{i+1}|| \ldots ||h'_{i+\alpha}$ such that $h(m'_i) = h(m_i)$. However, given that $h(\cdot)$ is a second pre-image resistant hash function, finding such an m'_i is considered infeasible for a computationally bounded attacker. Chains that contain such collisions (i.e., split into branches) will therefore be dropped by the receiver. Note that if $h(\cdot)$ is also collision-resistant, then the attacker cannot even find such collisions in the packet chains that she created herself.

4.2 Cryptographic Accumulators

Cryptographic accumulators [6] combine a (large) set of values U into one (short) accumulator y such that for each

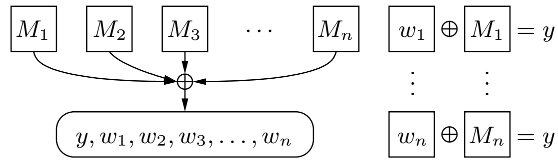

Figure 4: Cryptographic one-way accumulator. The fragments $M_1, M_2, \ldots, M_n$ are accumulated into an accumulator y and the witnesses $w_1, w_2, \ldots, w_n$; the witness w_i proves that M_i was accumulated into y.

value $u_i \in U$ there is a (short) witness w_i that u_i was indeed incorporated into the accumulator (see Figure 4). A collision-resistant one-way accumulator is a collision-resistant hash function $f : X \times U \mapsto X$ that is also quasi-commutative [5]. That is,

$$\forall x \in X : \forall u_1, u_2 \in U : f(f(x, u_1), u_2) = f(f(x, u_2), u_1).$$

In order to protect a message M, the sender computes for each fragment M_i generated by $\mathcal{E}(n, l, \varepsilon)$ the witness

$$\begin{aligned} w_i &:= f(id, \mathcal{M} \setminus \{M_i\}) \\ &= f(\ldots f(f(\ldots f(id, M_1) \ldots, M_{i-1}), M_{i+1}) \ldots, M_n), \end{aligned}$$

where id is the message id and $\mathcal{M}$ is the set of all fragments. Fragment M_i is then encapsulated into the packet $m_i := id||i||l||M_i||w_i$ by adding the message id, the fragment number i, the number of required fragments l, and the witness w_i.

For each new packet m_i, the receiver computes the accumulator $y := f(w_i, M_i) = f(f(id, \mathcal{M} \setminus \{M_i\}), M_i) = f(id, \mathcal{M}) = f(\ldots f(f(id, M_1), M_2) \ldots, M_n)$. Due to the quasi-commutative property of the one-way accumulator, the accumulator y is identical for all fragment/witness pairs of the same message and thus identifies the message the fragment belongs to. Verifying a packet involves the computation of the accumulator and its comparison with the values of all N already received packets. The cost to verify m_i is thus in $O(N)$. In order for the receiver to accept a phony inserted fragment M'_i, the attacker must find a witness w'_i such that $f(w_i, M_i) = f(w'_i, M'_i)$. Given that $f(\cdot, \cdot)$ is a collision-resistant hash function, finding such a collision is considered infeasible for a computationally bounded attacker.

The verification-related overhead per packet is given by the size of w_i. A fairly space-efficient one-way accumulator based on bilinear pairings was proposed by Nguyen [19]. For this accumulator, the size of the witness is equal to the prime order of the used group; that is, about $2k$ bits for a security level of k bits.

Another type of space-efficient one-way accumulators are Merkle trees [12]. Here, the hash values of $id||i||M_i$ are the leaves of the hash tree and the accumulator is given by the root value of the tree. The witness w_i for a fragment M_i is the set of the $\log_2(n)$ sibling nodes on the path from M_i to the root and is of size $\log_2(n)k$ bits.

4.3 Short Signatures

If the sender and receiver want to take full advantage of rateless erasure coding, the packet verification technique must allow for verifying each packet of a continuous packet stream individually. We propose a scheme based on short signatures that meets these requirements. In our scheme, the sender generates a new public/private key pair (K_M, K_M^{-1}) for every message M that he transmits. The length of these keys must be such that they resist an attack for the duration of the message transmission. Once the message transmission is over, the keys become useless for the attacker. After the keys have been created, the sender encapsulates each fragment M_i generated by $\mathcal{E}(n, l, \varepsilon)$ into a packet $m_i := K_M||i||l||M_i||Sig_{K_M^{-1}}(K_M||i||l||M_i)$ by adding the public key K_M, the fragment number i, the number l of required fragments, and the signature of $K_M||i||l||M_i$. The receiver uses the included public key to verify the signature of each received packet and drops packets with an invalid signature. Packets that are signed with the same private key belong to the same message. Verifying a new packet thus requires to verify the signature once and compare the included key with the keys of the N already received packets.

The verification-related overhead per packet consists of the public key and the signature. For the short signature scheme based on bilinear maps proposed in [7], the size of the signature and of the public key is equal to the prime order of the used group. The total overhead per packet is thus $4k$ bits for a security level of k bits.

4.4 Performance Evaluation

In this section we analyze the performance of the proposed packet verification techniques. As a metric for their performance, we use the expected number of required packet reception attempts until the receiver can successfully decode the message. We assume that all erasure codes are optimal (i.e., $\varepsilon = 0$) and we neglect differences in the verification and decoding speed of the proposed schemes. This is reasonable since the message reassembly is performed only once per message due to the packet verification and takes significantly less time (in the order of milliseconds[1]) than the time required for the message packets to be acquired at the receiver (order of seconds). The packet verification time of our schemes is also in the order of milliseconds[2] and the verification can be performed in parallel to the packet gathering.

Rateless Erasure Codes

We first consider the case of a rateless erasure code $\mathcal{E}(\infty, l, \varepsilon)$ in combination with short signatures. This combination is optimal in the sense that there are no duplicate or non-verifiable packets and every successfully received fragment contributes to the message. Hence, a message can be reassembled as soon as l fragments have been received. Let p_m be the probability that a packet sent by the sender is successfully received by the receiver. Let further X_i denote the event that the last of the required l packets is received after i (successful and unsuccessful) packet reception attempts. The expected number of required packet reception attempts until a message can be reconstructed by the receiver is then $N(p_m) = \sum_{i=0}^{\infty} \mathsf{P}[X_i] i$ and thus

$$N(p_m) = \sum_{i=l}^{\infty} \binom{i-1}{l-1} (p_m)^l \, (1 - p_m)^{i-l} \, i = \frac{l}{p_m} \in O(l). \quad (1)$$

Note that Equation (1) also applies to erasure codes where the number of fragments n is finite but larger than the expected number of required reception attempts $N(p_m)$.

Erasure Codes

If the number of fragments n per message is finite (and smaller than $N(p_m)$), the sender repeatedly sends the sequence of n packets. We observe that after i such sequence transmissions the expected number of missing packets is $n(1 - p_m)^i$. For the case that any set of l fragments can be verified we get the approximation

$$N(p_m) \approx \frac{\log(n - l) - \log(n)}{\log(1 - p_m)} n \in O(\log(\tfrac{n}{n-l})n) \quad (2)$$

by solving $n(1-p_m)^i = n-l$ for i. More precisely, the probability that exactly j out of n fragments have been received after i rounds is $\mathsf{P}[X_{ji}] = ((1 - p_m)^i)^{n-j}(1 - (1 - p_m)^i)^j$. In the final round, the last packet will be received after an average of $n/2$ attempts. Let Y_j denote the event that the set of j fragments is verifiable. For the erasure code $\mathcal{E}(n, l, \varepsilon)$ the expected number of required packet reception attempts is then

$$N(p_m) = \Big(\sum_{i=0}^{\infty} \sum_{j=0}^{n} \mathsf{P}[X_{ji}] \mathsf{P}[\neg Y_j] - 1 \Big) n + \frac{n}{2} \quad (3)$$

$$= \sum_{i=0}^{\infty} \sum_{j=0}^{n} D(j) \left((1 - p_m)^i \right)^{n-j} \left(1 - (1 - p_m)^i \right)^j n - \frac{n}{2} ,$$

where $D(j)$ is the number of *non-verifiable* subsets of size j. Clearly, $D(j) = \binom{n}{j}$ if $j < l$ since a message cannot be reconstructed with less than l fragments. Moreover, if an accumulator-based verification is used then $D(j) = 0$ if $j \geq l$ as each packet can be verified individually and thus any set of $j \geq l$ genuine fragments is verifiable.

In the case of multiple hash-link verification, a set of $j \geq l$ packets is verifiable if at least l packets in the set are connected without gaps of α or more missing packets between them. In order to compute $D(j)$ (i.e., the number of sets with j packets such that no l packets in the set are connected) we first introduce the following two lemmas:

Lemma 1. *The number of allocations of b indistinguishable balls into r distinguishable bins such that the first m, $0 \leq m \leq r$, bins contain at most k balls is*

$$A(b, r, m, k) = \sum_{i=0}^{\lfloor \frac{b}{k+1} \rfloor} (-1)^i \binom{m}{i} \binom{b - (k+1)i + r - 1}{r - 1} .$$

Proof. The proof can be found in [17]. □

Lemma 2. *The number of sequences of length n with j black and $n-j$ white balls that contain a sub-sequence of $\geq l$, $2l > n$, black and m white balls such that no two subsequent black balls in the sub-sequence are separated by more than b white balls is $A(n - j, j + 1, l - 1, b) + (j - l)A(n - j - b - 1, j + 1, l - 1, b)$.*

Proof sketch. We identify each sub-sequence fulfilling the given requirements by the first black ball in the sequence. If the sub-sequence starts at the first black ball in the sequence, according to Lemma 1, there exist $A(n-j, j+1, l-$

[1]The raptor code implementation in [24], for instance, achieves a decoding speed of several Gbit/s.

[2]Of the used cryptographic primitives, signatures are the most expensive (compared to computing hash values or to verifying accumulators). Verifying a single 160 bit short signature takes about 48 ms on a current general purpose CPU and becomes faster if several packets are verified in a single batch verification (less than 3 ms per signature for a batch of 200 packets) [11].

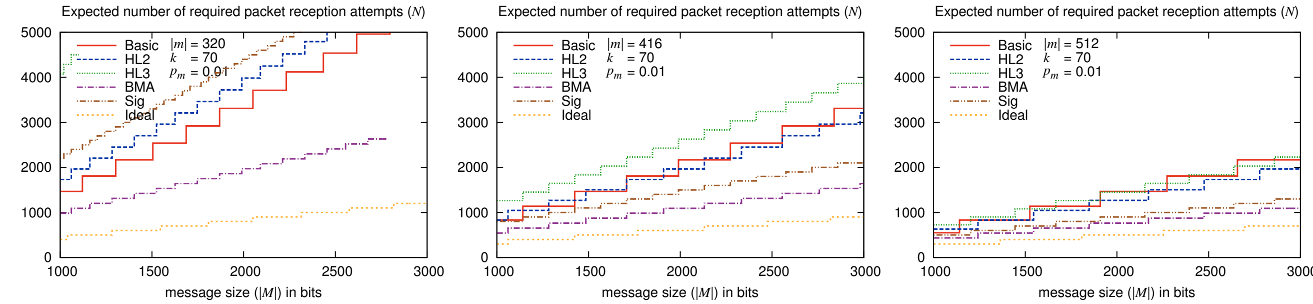

Figure 5: Performance of the presented message coding schemes as a function of message and packet size. We compare the schemes to the single hash-link scheme from [26] (Basic) and to an ideal scheme without verification overhead. We observe that BMA and Sig perform best whereas the advantage provided by two hashes in HL2 is more or less nullified by the larger overhead per packet. This is even worse for HL3 which is always less efficient than the basic scheme. By comparing the three plots for the packet sizes 320, 416, and 512 bit, we observe that the difference between the proposed schemes and the ideal scheme decreases for larger *packet* sizes and increases for larger *message* sizes.

$1, b)$ ways to distribute the $n-j$ white balls before, between, or after the j black balls such that there are no more than b white balls between the first l black balls. In the $j-l$ cases where the wanted sub-sequence starts at the second to $(j-l+1)$-th black ball, the sub-sequence must be preceded by (at least) $b+1$ white balls. Thus, in each of these cases there exist $A(n-j-b-1, j+1, l-1, b)$ ways to distribute the remaining $n-j-b-1$ white balls. □

From Lemma 2 it follows that there exist $D(j) = \binom{n}{j} - A(n-j, j+1, l-1, \alpha-1) - (j-l)A(n-j-\alpha, j+1, l-1, \alpha-1)$ sets of j packets such that no l packets in the set are connected. For the single hash link scheme proposed in [26] we have $\alpha = 1$ and $n = l$ and thus:

$$N(p_m) = \sum_{i=0}^{\infty} \left(1 - \left(1 - (1-p_m)^i\right)^l\right) l - \frac{l}{2} \qquad (4)$$
$$\approx \frac{\log(0.5) - \log(l)}{\log(1-p_m)} l \in O(\log(l) l) .$$

Performance Comparison and Discussion

We next compare the performance of the above presented message coding schemes among each other and with the basic single hash link scheme. In particular we compare the following schemes:

Basic The basic single hash-link scheme from [26].
HL2 Erasure coding combined with double hash links.
HL3 Erasure coding combined with triple hash links.
BMA Erasure coding combined with the one-way authenticator based on bilinear maps. Since this accumulator introduces strictly less overhead per packet than a Merkle tree based accumulator, we do not consider the latter in this comparison.
Sig Rateless erasure coding combined with short signatures.
Ideal An ideal scheme that uses rateless erasure coding but has no verification overhead. This scheme constitutes an upper bound on the performance.

The performance of the considered schemes as a function of the message and packet size is depicted in Figure 5. We observe that overall BMA and Sig perform best whereas the advantage of the second hash link for HL2 is more or less nullified by the larger overhead per packet (i.e., the larger number l of required packets). This is even worse for HL3 which is always less efficient than the basic scheme. We also observe that the difference between the proposed schemes and the ideal scheme decreases for larger packet sizes (i.e., a smaller number l of required packets per message), and increases for larger message sizes (i.e., a larger number l of required packets per message).

In this evaluation, the size of the message ids and the fragment numbers is 48 and 16 bit, respectively. We choose $n = 500$ fragments for BMA and derive the optimal n for the hash link schemes numerically. The number of required packets is computed as $l = |M|/(|m|-o)$, where o is the overhead per packet (message id, fragment number, verification data). We further choose a (short term) security level of $k = 70$ bits, which ensures that the cryptographic primitives (hashes, accumulators, signatures) can resist an attack for a period of several weeks to months [1]. We point out that this is more than sufficient since the primitives must resist an attack only for the duration of the message transmission, which is in the order of seconds.

The expected throughput and message transmission times as a function of the message and slot size, respectively, for consumer-class and advanced radio transceivers are shown in Figure 6. The simulation results presented in this figure were obtained with a purpose-built Java application that simulates the message transmissions and uses a simplified communication model for the physical layer. The application assumes a perfect jammer that jams a packet for the minimal required amount of time and whose interference with a packet is always destructive. Hence, a packet is successfully received if the sender and receiver are sending and receiving on at least one unjammed channel, the packet is dropped otherwise. The results show that, compared to the basic scheme, BMA reduces the UFH communication latency up to one-half (i.e., increases the throughput up to two times).

We further observe from Figure 6 that the throughput initially increases for larger slot lengths as less fragments (and thus less reception attempts) per message are needed. Since the jamming probability also increases for larger slots, the throughput starts decreasing once the advantage of less fragments is outweighed by the increased jamming probability.

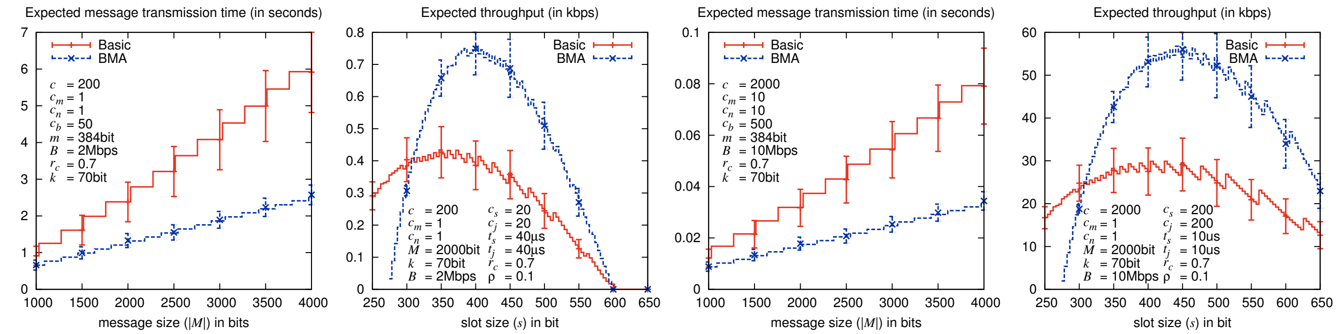

Figure 6: Expected message transmission time and expected throughput as a function of the message and slot size, respectively, for consumer-class ($c = 200$, $c_m = c_n = 1$, $B = 2$ Mbps) and advanced radio transceivers ($c = 2000$, $c_m = c_n = 10$, $B = 10$ Mbps). The plots assume a packet encoding with code rate $r_c = 0.7$ and jamming resistance $\rho = 0.1$ (Section 5), which is approximated, e.g., by repeating a binary (28,22,4) code. We observe that the throughput initially increases for larger slot lengths as less fragments (and thus also less reception attempts) per message are needed. Since the jamming probability also increases for larger slots, the throughput finally starts decreasing once the advantage of less fragments is outweighed by the increased jamming probability. We further observe that our BMA scheme (erasure coding combined with a one-way authenticator based on bilinear maps) roughly doubles the throughput and halves the transmission latency of UFH communication compared to the basic scheme. In this figure, the lines show the analytical results, the points and σ-confidence intervals display the findings of our simulations.

5. ERROR-RESISTANT PACKET CODING

During their transmission over the wireless channel, the packets resulting from the message encoding (Section 4) need to be protected against jamming attacks and bit errors. Error-correction is a well-understood problem in wireless communication and existing solutions use a variety of (block or convolutional) error-correcting codes. These codes differ in the redundancy that they introduce, in the error-protection that they achieve, and in their decoding performance. A packet coding scheme $\mathsf{PC}(\rho, r_c)$ is parametrized by its jamming resistance ρ, $0 < \rho \leq 1$, and its code rate r_c, $0 < r_c \leq 1$: it encodes data of length $|m|$ into a packet of length $|m|/r_c$ and more than $\rho|m|/r_c$ packet bits have to be disrupted so that the receiver cannot correctly decode the packet. We say that a $\mathsf{PC}(\rho, r_c)$ is *efficient* if it allows for a decoding time that is polynomial in the length of the packet and if the packet length resulting from the encoding remains linear in the data length.

It is particular to UFH that the length of the packets resulting from the packet encoding must not exceed the size $s = B/f_h$ of the hopping slots (typically in the order of few hundred bits), where B is the data rate and f_h is the hopping frequency of the sender. The shorter the slots (shorter packets), the better the protection against responsive jamming, but more packets need to be successfully transmitted. On the other hand, the longer the slots are, the more redundancy can be added to the packets (allowing to choose a $\mathsf{PC}(\rho, r_c)$ with smaller r_c and larger ρ) and hence the better is the protection against non-responsive jamming.

As a metric for evaluating and comparing different packet encodings, we express the attacker's jamming strength with respect to a packet encoding by the number of channels c_b that the attacker can effectively block during the transmission of a packet. The probability p_j that a packet is jammed then simplifies to $p_j = \frac{c_b}{c}$. Let $n_j := \frac{t_m}{\rho t_m + t_j}$ and $n_s := \frac{t_m - \rho t_m - t_j}{t_s}$ denote the number of jamming and sensing cycles per packet that the attacker can achieve, where t_s (t_j) is her required time to switch the sensing (jamming) channels. Using the number of channels c_j (c_s) on which the attacker can jam (sense) simultaneously, we can compute the number of blocked channels c_b for different types of attackers. For the jammer types defined in Section 2 we obtain: $c_b = c_j$ for static jammers, $c_b = n_j c_j$ for sweep jammers, $c_b = c(1 - (1 - \frac{c_j}{c})^{n_j})$ for random jammers, $c_b = n_s c_s$ for responsive jammers, $c_b = c_j + n_s c_s$ for responsive-static jammers, $c_b = n_j c_j + n_s c_s$ for responsive-sweep jammers, and $c_b = c(1 - (1 - \frac{c_j}{c})^{n_j}) + n_s c_s$ for responsive-random jammers.

Now, we can compare possible packet encodings. We distinguish two types: (1) block codes (e.g., Reed-Solomon, BCH, or Preparata codes) that encode entire data blocks into packets, which requires that the length of the produced codewords is in the size of the hopping slots (several hundred bits) and (2) concatenated codes, combining an outer and an inner code (typically Reed-Solomon with a short block code) [28]. Figure 7 displays the number of blocked channels c_b as a function of the selected $\mathsf{PC}(\rho, r_c)$ and the attacker type for packets of size 255 bits. We see that the packet encoding only marginally affects the jamming-resistance for purely responsive jamming but is more effective against non-responsive or hybrid jammers. Depending on the expected attacker and given their efficiency in decoding, we suggest to use BCH codes (e.g. BCH(255,131,30) or BCH(255,191,8)), because they offer an effective ρ (7.5% or 4%, compared to 0.4% without encoding) while keeping r_c sufficiently large (0.51 or 0.75).

If block or concatenated codes of appropriate lengths are hard to find or become inefficient in the decoding performance, we propose to apply block codes (e.g., Hamming codes) multiple times for each data packet and to interleave the resulting codewords. This has been described and analyzed for exemplary codes in [14] where shared secret keys are used for cryptographically interleaving the bits. In UFH, we cannot leverage shared secrets, still the interleaving

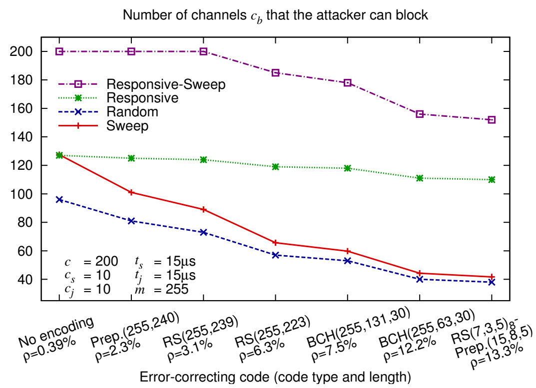

Figure 7: Number of blocked channels c_b as a function of the attacker type and the packet coding (block codes and a concatenated 'Reed-Solomon'-Preparata-code [14]). We observe that the effect of the packet encoding is only marginal for purely responsive attackers whereas the encoding considerably decreases the impact of non-responsive and hybrid attackers.

must be deterministic for the receiver to be able to despread the codewords (without interleaving, the attacker could jam the percentage ρ of one short block code to destroy the entire packet). Fortunately, we can simulate a property comparable to cryptographic interleaving as follows: after interleaving the data using a public seed, the sender delays the transmission of each packet by a short random time $\delta \leftarrow_R \{0, \delta_{max}\}$. This does not affect the receiver due to his longer dwelling time per channel, but since the attacker cannot predict the delay, she cannot jam targeted bits (without scanning for packet starts). Using random transmission delays, the expected total message transmission time then becomes $\mathsf{E}[t_M] = N(p_m) \cdot (t_m + \delta_{max}/2)$ where $t_m = |m|B/r_c$ is the packet transmission time; we note that the probability p_m of a successful packet transmission depends both on the packet encoding and on the attacker strength.

6. OPTIMAL CHANNEL NUMBER AND CHANNEL SELECTION

In UFH, the frequency channels on which the sender and the receiver send and receive are chosen randomly from the set of available channels. In contrast to coordinated FH where the jamming resistance and thus the throughput of the communication increases with the number of channels used, using all available channels might not be optimal for UFH: while the jammer's chances to jam the right channel decrease the more channels are used, so do an uncoordinated receiver's chances to listen on the right channel. As an example, consider the case where the sender and receiver can send and receive on one channel (i.e., $c_n = c_m = 1$) and select the frequency channels uniformly at random from a set of c channels. Assuming further that the attacker blocks c_b channels (see Section 5), the probability that a packet is successfully received with UFH is $p_m = \frac{1}{c}(1 - \frac{c_b}{c})$, which is maximized for $c = 2c_b$. We next show that a similar result holds for the general case where the sender (receiver) sends (receives) on c_n (c_m) channels:

Theorem 1. *In the presence of an attacker that prevents communication on $c_b \leq c$ channels, the probability p_m that a packet is successfully received with UFH is maximized if the sender and receiver choose the c_n (c_m) frequency channels on which they send (receive) uniformly at random from a set of size c^*, where*

$$c^* = \begin{cases} c_b + 1 & \text{if } c_b < c_m \text{ and } c_b < c_n \\ \max\{\approx 2c_b, c_m, c_n\} & \text{otherwise.} \end{cases}$$

To prove Theorem 1, we first introduce three lemmas:

Lemma 3. *If $a_1, \ldots, a_c \in [0, 1]$ such that $\sum_{i=1}^{c} a_i \leq c_n$, then $\sum_{i=1}^{c} \frac{1}{a_i} \geq \frac{c^2}{c_n}$.*

Proof Sketch. Consider the values $a_i := \frac{c_n}{c} + \varepsilon_i$ and $a_j := \frac{c_n}{c} - \varepsilon_i + \varepsilon_j$. The sum $y := \frac{1}{a_i} + \frac{1}{a_j}$ is minimized if $\frac{dy}{d\varepsilon_i} = -(\frac{c_n}{c} + \varepsilon_i)^{-2} + (\frac{c_n}{c} - \varepsilon_i + \varepsilon_j)^{-2} = 0$; that is, if $\varepsilon_i = \frac{1}{2}\varepsilon_j$ and thus $a_i = a_j$. Since this holds for all pairs a_i and a_j, $\sum_{i=1}^{c} \frac{1}{a_i}$ is minimized if $\forall i, j : a_i = a_j = \frac{c_n}{c}$. $\square$

Lemma 4. *Let c_n (c_m) be the number of channels on which the sender (receiver) sends (receives) in parallel and c_b be the number of channels that the attacker jams. If the sender and receiver select their channels uniformly at random from a set of c channels, the probability that the sender and receiver select at least one common non-jammed channel is*

$$p_m = 1 - \sum_{i=0}^{c_m} \frac{\binom{c-c_n}{c_m-i}\binom{c_n}{i}}{\binom{c}{c_m}} \frac{\binom{c-i}{c_b-i}}{\binom{c}{c_b}} . \quad (5)$$

Proof Sketch. Let X_i denote the event that there is a transmission on exactly i out of the c_m channels chosen by the receiver. There exist $\binom{c-c_n}{c_m-i}$ possibilities to chose the $c_m - i$ channels without a transmission and $\binom{c_n}{i}$ possibilities to chose the i channels with a transmission. Hence, $\mathsf{P}[X_i] = \binom{c-c_n}{c_m-i}\binom{c_n}{i}/\binom{c}{c_m}$. The attacker, in turn, blocks communication on c_b out of c channels. Given a set of i channels, the probability that the attacker blocks them all is $\mathsf{P}[\text{the } i \text{ channels are jammed}] = \binom{c-i}{c_b-i}/\binom{c}{c_b}$. The proof follows from the observation that $p_m = 1 - \sum_{i=0}^{c_m} \mathsf{P}[X_i]\mathsf{P}[\text{the } i \text{ channels are jammed}]$. $\square$

Lemma 5. *For $u \geq v \geq 0$: $\binom{u}{v} \leq \frac{u^v}{v!}$.*

Proof. By definition $\binom{u}{v} = \frac{u(u-1)(u-2)\cdots(u-v+1)}{v!} = (1 - \frac{1}{u})(1 - \frac{2}{u}) \cdots (1 - \frac{v-1}{u}) \frac{u^v}{v!} \leq \frac{u^v}{v!}$. $\square$

Proof of Theorem 1. First we show that selecting the input and output channels uniformly at random is an optimal strategy for the UFH sender and receiver. We will show that, by jamming the channels that are more likely to be selected by the sender more intensive than the rarely selected channels, the attacker can nullify any advantage that the sender and receiver might obtain by using a non-uniform channel selection. Let a_i (b_i) be the probability that the sender (receiver) sends (receives) on channel $i \in \{1, 2, \ldots, c\}$. Let further x_i be the probability that the attacker blocks channel i. Without loss of generality we assume that $a_1 \geq a_2 \geq \ldots \geq a_{c'} > 0$ and $a_{c'+1} = a_{c'+2} = \ldots = a_c = 0$. Now consider the jammer strategy where the attacker jams channel i with probability $x_i = 1 - \frac{c_n(c'-c_b)}{a_i c'^2}$ if $a_i \geq \frac{c_n(c'-c_b)}{c'^2}$ and $x_i = 0$ otherwise. With this strategy, the probability that a packet is not jammed and thus can be successfully received on channel i is equal to $a_i(1-x_i) \leq a_i\frac{c_n(c'-c_b)}{a_i c'^2} = \frac{c_n}{c'}(1-\frac{c_b}{c'})$. Given that $\sum_{i=1}^{c'} a_i \leq c_n$ and $\sum_{i=1}^{c'} \frac{1}{a_i} \geq \frac{c'^2}{c_n}$ (Lemma 3), this attacker strategy is indeed valid because $\sum_{i=1}^{c} x_i \leq \sum_{i=1}^{c'}(1-\frac{c_n(c'-c_b)}{a_i c'^2}) = c' - \frac{c_n(c'-c_b)}{c'^2}\sum_{i=1}^{c'}\frac{1}{a_i} \leq c' - \frac{c_n(c'-c_b)}{c'^2}\frac{c'^2}{c_n} = c_b$. Hence, the receiver's chances to successfully receive a packet are the same for the first c' channels and zero for the remaining $c-c'$ channels. It follows that in this case every selection strategy—and thus also selecting the channels uniformly at random—for which $\sum_{i=1}^{c'} b_i = c_m$ is an optimal receiver strategy. We note that jamming channel i with probability $x_i = 1 - \frac{c_n(c-c_b)}{a_i c^2}$ might not be the optimal strategy for the attacker with respect to a particular receiver (e.g., if she knew that the receiver would listen on one channel only, focusing the jamming on this channel would be optimal). This strategy is, however, optimal for the attacker in the sense that it limits the performance of the sender for any number of receivers with arbitrary channel selection schemes.

We next deduce the optimal set size c^* from which the sender and receiver should uniformly choose their channels. We first consider the special case where the attacker is weaker than the sender and the receiver: If $c_b < c_m$ and $c_b < c_n$, for all $c \in \{c_b+1, c_b+2, \ldots, \min\{c_m, c_n\}\}$ there exists at least one non-jammed channel on which the sender is sending and the receiver is receiving. In particular we have $p_m = 1$ for $c^* = c_b+1$. Otherwise, if either $c_m \leq c_b$ or $c_n \leq c_b$, a second bound on c^* is given by $c^* \geq \max\{c_n, c_m\}$. If $c = c_n \geq c_m$, the receiver always listens on a channel on which a packet is transmitted. Using less than c_n channels would therefore increase the attacker's changes to successfully jam all transmissions without being beneficial for the receiver. Likewise, if $c = c_m \geq c_n$, the receiver receives all transmissions. Using less than c_m channels would again increase the attacker's jamming performance without any benefit for the receiver. Finally, we consider the general case. According to Lemmas 4 and 5, the probability that a packet sent by the sender is successfully received by the receiver is

$$\begin{aligned} p_m &= 1 - \sum_{i=0}^{c_m} \frac{\binom{c-c_n}{c_m-i}\binom{c_n}{i}}{\binom{c}{c_m}} \frac{\binom{c-i}{c_b-i}}{\binom{c}{c_b}} \\ &\gtrapprox 1 - \sum_{i=0}^{c_m} \frac{\frac{(c-c_n)^{m-i}}{(c_m-i)!}\frac{c_n^i}{i!}}{\frac{c^{c_m}}{c_m!}} \left(\frac{c_b}{c}\right)^i \\ &= 1 - \left(1 - \frac{c_n}{c}\left(1-\frac{c_b}{c}\right)\right)^{c_m} =: \hat{p}_m \,. \qquad (6) \end{aligned}$$

For a given c_n and c_m, $\hat{p}_m$ is maximized if $\frac{1}{c}(1-\frac{c_b}{c})$ is maximized. We obtain $\frac{d}{dc}\frac{1}{c}(1-\frac{c_b}{c}) = 0$ if $c = 2b$ and thus $c^* \approx 2b$. For particular values of c_m, c_n and c_b, a more precise result can be obtained by means of some standard (numerical) optimization technique. Our simulations showed that using the approximation $c^* = 2b$ already leads to very accurate results: for $1 \leq c_b \leq 100$ the relative error in p_m was always less than 0.003. □

6.1 Adaptive Channel Selection

Achieving an optimal throughput with UFH requires the sender and receivers to accurately assess c_b and agree on a set of $c = c^*$ frequency channels. Especially in the absence of jamming, any selection of $c \geq 1$ channels will lead to a suboptimal performance. An optimal adaptive scheme

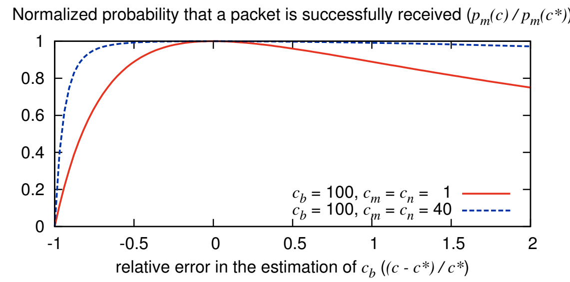

Figure 8: Graceful degradation of UFH performance for an imprecise assessment of the attacker's strength (in terms of c_b). We observe that overestimating the attacker's strength is less harmful than underestimating it. We further observe that the larger c_m and c_n are, the less impact an imprecise estimation has.

in which both the sender and receiver(s) adapt their channels depending on the encountered jamming is, however, not practical. As jamming occurs at the receiver of a transmission, the sender would have to reliably obtain the feedback of the (maybe unknown) receiver(s); since different receivers are likely to observe different jamming strengths, this would further require to adapt to the worst case receiver. To avoid that the attacker can exploit this feedback, the feedback channel would have to be authentic. Providing the sender with the required feedback is therefore the same problem as the one we intend to solve with UFH in the first place.

Consequently, only those adaptive schemes are feasible where the receivers adapt their behavior but not the sender. However, as shown in the proof of Theorem 1, the attacker can nullify any possible advantage of an adaptive scheme that leverages a non-uniform channel selection by jamming more often those channels that are selected by the sender more likely than the rarely selected channels. This result is independent of the receiver's behavior and even holds for multiple senders: if we neglect the impact of the senders' position and of the physical environment, from the attacker's point of view there is no difference between the cases of multiple senders and a single sender that sends on multiple channels in parallel.

In conclusion, the best strategy for the sender is to assess the jamming strength c_b of the expected attacker as accurately as possible and then select all output channels uniformly at random from a set of size $\max\{c^*, c\}$, where c is the maximal number of available channels and c^* is the optimal number of channels as specified in Theorem 1. The only exception might be a situation where the attacker (although being strong) is absent (or inactive) most of the time. In such a situation it is advisable to send permanently on one channel and select the other output channels uniformly at random from the remaining channels. The receivers first try to receive the message on the first channel and only switch to UFH if they are not successful. Permanently sending on the first channel ensures that the scheme is as efficient as coordinated frequency hopping if the attacker is absent.

In practice, assessing the attacker's strength means estimating the scanning and sending performance of her jamming device. Given these parameters, the number of channels c_b that the attacker can block can be derived as described in Section 5. How accurately the sender can estimate the capabilities of the jamming device depends on the considered attacker model. If, for example, the attacker is assumed to compromise t nodes in a network and abuses them as static jammers each sending on c_n different channels in parallel, $c_b = \min\{t \cdot c_n, c\}$ will be a fairly accurate approximation of the number of blocked channels. The impact of an imprecise assessment of the attacker's strength is shown in Figure 8. We observe that the performance of UFH gracefully degrades the less accurate the assessment of c_b is and that overestimating the attacker's strength is less harmful that underestimating it. Hence, using all available channels is a reasonable fallback strategy under the threat of a completely unknown attacker.

7. RELATED WORK

Spread-spectrum (SS) techniques such as FHSS, DSSS, and chirp SS [3,22] are well-studied countermeasures against communication jamming attacks. Jamming and (unintentional) interference are further thwarted by the application of forward error-correcting codes and special coding strategies [14,28].

Over the last years, denial-of-service attacks on wireless local area, ad-hoc and sensor networks have widely been analyzed [2,13,14,20], and various countermeasures for their detection and prevention have been proposed [8, 9, 20, 27, 29]. However, all these countermeasures rely on secrets that were pre-established between sender and receiver(s), or are very specific regarding the attacker's capabilities and the considered scenarios.

Recent observations [4,10,23,26] identify the shortcoming of non-existing methods for jamming resistance communication without shared secrets and propose solutions to this problem. Dolev et al. present f-AME [10], a round-based, randomized protocol to set up group keys in the presence of message collisions and insertions. A major limitation of f-AME is its requirement of a (fully connected) group of size $> 3(t+1)^2+2(t+1)$, where t is the number of channels that the attacker can jam (usually t is in the order of tens or even hundreds of channels requiring a group of hundreds or even thousands of nodes). The solution proposed by Baird et al. [4] uses concurrent codes in combination with UWB pulse transmissions. The jamming resistance achieved by their scheme is, however, not one-to-one comparable to common spread-spectrum-based techniques: While the attacker of SS techniques must have enough transmission power to overcome the processing gain, in [4] the limiting factor is the number of pulses that the attacker can insert, i.e., the energy of the attacker. In [23] Uncoordinated Direct Sequence Spread Spectrum (UDSSS) communication was introduced which, similar to UFH, enables jamming-resistant broadcast without shared keys. In UDSSS, the sender transmits using a randomly chosen spreading code from a publicly known code set and the receivers decode the transmitted message by trying out codes from the set. Since UDSSS enables the receiver to record the precise reception time of a message, it is well suited for anti-jamming navigation (i.e., localization and time synchronization).

The use of erasure codes for more reliable packet transmissions has also been considered in other wireless scenarios including routing [21] and authenticated multicast [12]. The distillation codes introduced in [12] have a similar goal as the verification techniques presented in this work. However, because UFH communication relies on short packets, the distillation codes based on Merkle trees proposed in [12] are not well suited for our application.

Independent of this work, Slater et al. [25] proposed two additional packet verification techniques (Hashcluster and Merkleleaf) and one similar to BMA (Witnesscode). Their results confirm our finding that BMA is a very efficient message coding scheme.

8. CONCLUSION

In this work, we addressed the efficiency of Uncoordinated Frequency Hopping communication, a novel spread spectrum scheme that enables anti-jamming communication between parties that do not share secret keys. We proved that an optimal frequency channel selection strategy for UFH communication is the one in which the sender and the receiver choose the frequency channels uniformly at random from a set of common frequency channels. We further showed that, given an attacker that blocks c_b frequency channels, the optimal strategy (in terms of communication throughput) for the sender and the receiver is to hop within a set of $\approx 2c_b$ channels. Furthermore, we proposed and evaluated several new UFH communication schemes. Our evaluations show that among the proposed schemes, BMA (erasure coding combined with a one-way authenticator based on bilinear maps) performs best. Compared to the basic scheme, BMA reduces the UFH communication latency up to one-half (i.e., increases UFH communication throughput up to two times). Finally, we analyzed the anti-jamming protection of individual packet transmissions and identified appropriate error-correcting schemes. We observed that for Reed-Solomon, BCH, and Preparata codes the effect of the packet coding is only marginal for purely responsive attackers while it may considerably decrease the impact of non-responsive and hybrid attackers.

9. REFERENCES

[1] ECRYPT Yearly Report on Algorithms and Keysize. D.SPA.28, July 2008. IST-2002-507932.

[2] Imad Aad, Jean-Pierre Hubaux, and Edward W. Knightly. Denial of service resilience in ad hoc networks. In *Proceedings of the 10th ACM International Conference on Mobile computing and networking (MobiCom)*, New York, 2004.

[3] David Adamy. *A first course in electronic warfare.* Artech House, 2001.

[4] Leemon C. Baird, William L. Bahn, Michael D. Collins, Martin C. Carlisle, and Sean C. Butler. Keyless Jam Resistance. In *Proceedings of the IEEE Information Assurance and Security Workshop (IAW)*, 2007.

[5] Niko Bari and Birgit Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In *Advances in Cryptology EUROCRYPT*, volume 1233/1997 of *Lecture Notes in Computer Science*, pages 480–494. Springer Berlin / Heidelberg, 1997.

[6] Josh Benaloh and Michael de Mare. One-Way Accumulators: A Decentralized Alternative to Digital Signatures. In *Advances in Cryptology EUROCRYPT*, volume 765/1994 of *Lecture Notes in Computer*

Science, pages 274–285. Springer Berlin / Heidelberg, 1994.

[7] Dan Boneh, Ben Lynn, and Hovav Shacham. Short Signatures from the Weil Pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.

[8] Jerry T. Chiang and Yih-Chun Hu. Dynamic Jamming Mitigation for Wireless Broadcast Networks. In *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[9] Yvo Desmedt, Rei Safavi-Naini, Huaxiong Wang, Chris Charnes, and Josef Pieprzyk. Broadcast Anti-Jamming Systems. In *Proceedings of the 7th IEEE International Conference on Networks (ICON)*, Washington, 1999.

[10] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, and Calvin Newport. Secure communication over radio channels. In *Proceedings of the 27th ACM symposium on Principles of distributed computing (PODC)*, 2008.

[11] Anna Lisa Ferrara, Matthew Green, Susan Hohenberger, and Michael Østergaard Pedersen. Practical Short Signature Batch Verification. *Topics in Cryptology (CT-RSA)*, 2009.

[12] Chris Karlof, Naveen Sastry, Yaping Li, Adrian Perrig, and J. Doug Tygar. Distillation Codes and Applications to DoS Resistant Multicast Authentication. In *Proceedings of the 11th Network and Distributed Systems Security Symposium (NDSS)*, 2004.

[13] Mingyan Li, Iordanis Koutsopoulos, and Radha Poovendran. Optimal Jamming Attacks and Network Defense Policies in Wireless Sensor Networks. In *Proceedings of the 26th IEEE Conference on Computer Communications (INFOCOM)*, 2007.

[14] Guolong Lin and Guevara Noubir. On link layer denial of service in data wireless LANs. *Wireless Communications & Mobile Computing*, 5(3):273–284, 2005.

[15] Michael Luby. LT Codes. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2002.

[16] Michael Luby, Michael Mitzenmacher, M. Amin Shokrollahi, Daniel A. Spielman, and Volker Stemann. Practical Loss-resilient Codes. In *Proceedings of the 29th annual ACM symposium on Theory of computing (STOC)*, 1997.

[17] Frosso S. Makri, Andreas N. Philippou, and Zaharias M. Psillakis. Success run statistics defined on an urn model. *Advances in Applied Probability*, 39(4):991–1019, 2007.

[18] Petar Maymounkov. Online Codes. Technical Report TR2002-833, New York University, Nov 2002.

[19] Lan Nguyen. Accumulators from Bilinear Pairings and Applications. In *Topics in Cryptology – CT-RSA*, volume 3376/2005 of *Lecture Notes in Computer Science*, pages 275–292. Springer Berlin / Heidelberg, 2005.

[20] Guevara Noubir and Guolong Lin. Low-power DoS attacks in data wireless LANs and countermeasures. *SIGMOBILE Mobile Computing and Communications Review*, 7(3):29–30, 2003.

[21] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure data transmission in mobile ad hoc networks. In *Proceedings of the 2nd ACM workshop on Wireless Security (WiSe)*, 2003.

[22] Richard A. Poisel. *Modern Communications Jamming Principles and Techniques*. Artech House, 2004.

[23] Christina Pöpper, Mario Strasser, and Srdjan Čapkun. Jamming-resistant broadcast communication without shared keys. Technical Report 609, ETH Zurich, 2008.

[24] Mohammad Amin Shokrollahi. Raptor Codes. *IEEE Transactions on Information Theory*, 52(6):2551–2567, June 2006.

[25] David Slater, Patrick Tague, Radha Poovendran, and Brian J. Matt. A Coding-Theoretic Approach for Efficient Message Verification Over Insecure Channels. In *Proceedings of the ACM Conference on Wireless Network Security (WiSec)*, 2009.

[26] Mario Strasser, Christina Pöpper, Srdjan Čapkun, and Mario Čagalj. Jamming-resistant Key Establishment using Uncoordinated Frequency Hopping. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2008.

[27] Mario Čagalj, Srdjan Čapkun, and Jean-Pierre Hubaux. Wormhole-Based Antijamming Techniques in Sensor Networks. *IEEE Transactions on Mobile Computing*, 2007.

[28] Stephen G. Wilson. *Digital Modulation and Coding*. Prentice-Hall, 1996.

[29] Wenyuan Xu, Wade Trappe, Yanyong Zhang, and Timothy Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, 2005.