# Jamming-resistant Key Establishment using Uncoordinated Frequency Hopping

Mario Strasser*
*Computer Eng. and Networks Laboratory*
*ETH Zurich, Switzerland*
*strasser@tik.ee.ethz.ch*

Christina Pöpper*
*Department of Computer Science*
*ETH Zurich, Switzerland*
*poepperc@inf.ethz.ch*

Srdjan Čapkun
*Department of Computer Science*
*ETH Zurich, Switzerland*
*capkuns@inf.ethz.ch*

Mario Čagalj
*FESB*
*University of Split, Croatia*
*mario.cagalj@fesb.hr*

## Abstract

*We consider the following problem: how can two devices that do not share any secrets establish a shared secret key over a wireless radio channel in the presence of a communication jammer? An inherent challenge in solving this problem is that known anti-jamming techniques (e.g., frequency hopping or direct-sequence spread spectrum) which should support device communication during the key establishment require that the devices share a secret spreading key (or code)* prior *to the start of their communication. This requirement creates a circular dependency between anti-jamming spread-spectrum communication and key establishment, which has so far not been addressed. In this work, we propose an Uncoordinated Frequency Hopping (UFH) scheme that breaks this dependency and enables key establishment in the presence of a communication jammer. We perform a detailed analysis of our UFH scheme and show its feasibility, both in terms of execution time and resource requirements.*

## 1. Introduction

The broadcast nature of wireless radio transmissions makes them particularly vulnerable to communication jamming Denial-of-Service (DoS) attacks. The aim of these attacks is to prevent devices from exchanging any useful information by interfering with their communication. Possible communication jamming attacks include signal annihilation, modification (bit-flipping, overshadowing) and jamming as well as the insertion of forged or replayed signals [2, 16, 20].

A class of well-known countermeasures against communication jamming attacks are *spread-spectrum* techniques such as frequency hopping, direct-sequence spread spectrum, and chirp spread spectrum [16]. Common to all these techniques is that they rely on secret (spreading) codes that are shared between the communication partners. These secret codes enable the sender to spread the signal (in time and/or frequency) such that its transmission becomes unpredictable for a third party, thus reducing the probability of interference. However, for these schemes to work, the required secret code must be shared between the partners *prior to their communication*, generally precluding unanticipated transmissions between unpaired devices. The requirement of a shared code has so far been fulfilled by out-of-band code pre-distribution on the devices. This approach has scalability disadvantages in environments where a large number of nodes potentially take part in a pairwise communication.

If pre-sharing the codes is not adequate or even infeasible (e.g., due to a large number of nodes or high network membership dynamics) the devices must agree on a secret code in an ad-hoc manner using the wireless channel. This observation leads to the following problem: *How can two devices that do not share any secrets establish a shared secret key over a wireless radio channel in the presence of a communication jammer (in order to derive a secret spreading code from the established key)?* The execution of a key establishment protocol relies on jamming-resistant communication which, in turn, requires the availability of a shared secret code. In other words, the dependency of spread-spectrum techniques on a shared key (or code) and the dependency of key establishment on a jamming-resistant communication create a circular dependency, which we call *anti-jamming/key-establishment dependency* (see Fig-

*Equally contributing authors.

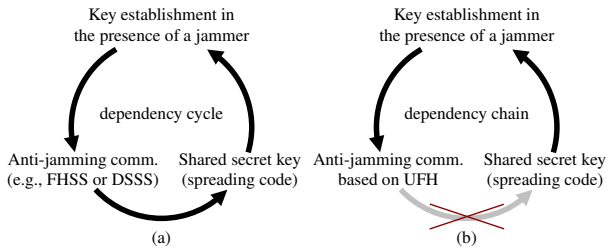



**Figure 1. Anti-jamming/Key-establishment dependency graphs. (a) If two devices do not share any secret keys or codes and want to execute a key establishment protocol in the presence of a jammer, they have to use a jamming-resistant communication. However, known anti-jamming techniques such as frequency hopping and direct-sequence spread spectrum rely on secret (spreading) codes that are shared between the communication partners prior to the start of their communication. (b) In this work, we break this dependency and propose a novel frequency hopping scheme called Uncoordinated Frequency Hopping (UFH) that enables two parties to execute a key establishment protocol in the presence of a jammer, even if the parties do not yet share a secret key or code.**

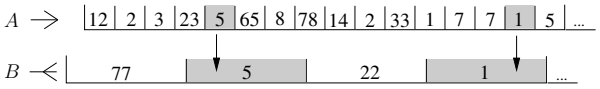



**Figure 2. Example of UFH. The numbers indicate the frequency channels where sender $A$ is sending and receiver $B$ is listening over time (here both send and receive on *one* frequency at a time). If $A$ and $B$ send and receive simultaneously on the same frequency (5 and 1 in the example) the packet sent on this frequency is successfully transmitted over the undisturbed channel.**

ure 1(a)). To the best of our knowledge, this circular dependency has so far not been addressed in this setting. We point out that, even if the nodes hold their public-key certificates issued by a commonly trusted authority, they still need to communicate in order to establish the secret spreading key (code) required for the jamming-resistant communication (e.g., using an authenticated Diffie-Hellman key establishment protocol).

In our present work, we break the circular dependency between anti-jamming spread-spectrum communication, shared secret keys (or codes), and key establishment in a jammed environment (see Figure 1(b)). As a solution to this circular dependency, we propose a scheme called *Uncoordinated Frequency Hopping* (UFH) that enables the jamming-resistant communication between two nodes in the presence of a jammer *without* a pre-shared code. We further show how to use this UFH scheme for executing a key establishment protocol, which, in the presence of a jammer, enables the nodes to agree on a shared secret key. The nodes can then use this key to create a secret hopping sequence and communicate using coordinated frequency hopping, thereby abandoning the use of the UFH scheme.

UFH is closely related to coordinated frequency hopping: each message is split into multiple parts and then sent across the air on random hopping frequencies chosen from a fixed frequency band. Like coordinated frequency hopping, UFH is based on the assumption that the attacker cannot jam all frequency channels on which the nodes communicate at the same time so that the sender and the receiver can still communicate through the remaining channels. However, in UFH, the sender and the receiver do not agree on a secret channel sequence but instead transmit and listen on randomly selected channels. Hence, all communication in UFH underlies the observation that, with sufficient transmission attempts, the sender and receiver will send and listen on the same channels in a number of time slots, even if they did not agree on them beforehand (see Figure 2). Intuitively, given 500 channels and given a sender hopping among the channels at a high rate of, for instance, 1500 Hz and a receiver hopping at a low rate (e.g., 100 Hz), the receiver will be listening on the frequency where the sender is transmitting in average 1500/500 = 3 times per second. Building on this observation, we develop a UFH scheme that is highly resistant to packet losses and active interference by an attacker. It can, thus, be applied in settings where two nodes wish to establish an unanticipated and spontaneous communication without pre-shared keys, which was so far not feasible using coordinated frequency hopping.

In summary, the main contributions of this work are:

- We address and describe the anti-jamming/key-establishment circular dependency problem: anti-jamming spread-spectrum communication techniques rely on a shared (spreading) key and key establishment relies on a jamming-resistant communication. This leads to the following question: in the presence of a communication jammer, how can two devices that do not share any secrets establish a shared secret key over a wireless radio channel?
- As one solution to the addressed problem, we propose a scheme called UFH (Uncoordinated Frequency Hopping) that enables two nodes to execute a key establishment protocol in the presence of a jammer; the es-

tablished key can then be used to support later coordinated frequency hopping communication. Our UFH scheme supports the transmission of messages of arbitrary length in a jammed environment without relying on a shared secret key.
- We introduce a comprehensive DoS attacker model which captures signal jamming and overshadowing as well as message insertion and modification.
- We show that, although our UFH scheme achieves lower communication throughput and incurs higher storage and processing costs, it achieves the same level of anti-jamming protection as (coordinated) frequency hopping (which, however, unlike our UFH, cannot be used in scenarios where devices do not share secret spreading keys).

The remainder of the paper is organized as follows: In Section 2, we specify the system setting and attacker model. We describe our jamming-resistant UFH scheme in Section 3 and demonstrate its use for a key establishment protocol in Section 4. In Section 5, we identify possible attacker strategies and provide a performance analysis of the proposed scheme. We discuss related work in Section 6 and conclude in Section 7.

## 2. System and Attacker Models

### 2.1. System Model

In our system, we focus on two nodes that reside within each other's power range, but are initially unaware of their proximity. The goal of each node is to detect communication signals from other nodes in its communication neighborhood and to establish communication with the detected node. Each node is equipped with processing and storage units, a clock, and a radio transceiving module capable of frequency hopping communication. We assume that the nodes are able to store few megabytes of data and can efficiently perform ECC-based public key cryptography. The nodes share the same concept of time and their clocks are assumed to be loosely synchronized in the order of seconds, e.g., by means of GPS. The transceiver permits each node to hop within a given set $\mathcal{C}$ of available radio frequencies (spanning a large band of typically several hundred frequencies, $c = |\mathcal{C}|$). The transceiver can be narrowband or broadband, enabling the node to send and receive signals on one or more (hopping) frequencies simultaneously; by $c_n$ and $c_m$ we denote the number of channels on which a node can send and receive on, respectively. We assume that the transceiver does not leak information about its active reception channels, that is, that the channels on which the transceiver is actively listening on cannot be detected by monitoring its radio signal emissions. Furthermore, the nodes can switch their input (listen) and output (send) channels independently of each other and are capable of performing full duplex communications by sending and receiving in parallel.

We assume that sender $A$ splits its available power uniformly over its $c_n$ output channels such that it transmits with the same signal strength on all channels; $P_a$ then denotes the strength of the signal arriving at a receiver. With respect to a specific receiver $B$, we denote by $P_t$ the minimal required signal strength at $B$ such that $B$ can successfully decode a message (i.e., the sensitivity of $B$'s receiver). In this context, the transmission between $A$ and $B$ over an undisturbed channel will be successful if $P_a \geq P_t$ and if $A$ sends on a channel on which, at the same time, $B$ is listening.

We further assume that each node holds a public/private key pair ($PK$, $SK$) and is computationally capable of performing public-key operations. The system is supported by a trusted Certification Authority (CA) that issues public-key certificates binding node identities and their public keys. The CA may be off-line or unreachable by the nodes at the time of the intended communication, but we assume that each node holds a valid certificate of its own public key and the valid public key $PK_{CA}$ of the CA. Both were distributed during the system initialization phase (e.g., after the procurement of the nodes).

### 2.2. Attacker Model

We consider an omnipresent but computationally bounded adversary who controls the communication channel in the sense that she is able to eavesdrop and insert arbitrary messages, but can only modify transmitted messages by adding her own energy-limited signals to the channel. This means that the attacker's ability to alter or erase a message is restricted to interfering with the message transmission and that she cannot disable the communication channel by blocking the propagation of radio signals (e.g., by placing a node in a Faraday's cage).

The attacker's goal is to interfere with the communication of the nodes in order to prevent them from exchanging any useful information. That is, the attacker aims at increasing (possibly indefinitely) the time for the message exchange in the most efficient way. In order to achieve this goal, the attacker is not only restricted to message jamming, but can also try to disturb the nodes' communication by modifying and inserting messages or by keeping one or both nodes too busy to participate in or proceed with the protocol. She can thus choose among the following actions:

- The attacker can *insert messages* that she generated by using known (cryptographic) functions and keys as well as by reusing (parts of) previously overheard messages (constituting a replay attack). Depending on the signal strength of the inserted messages, these messages might interfere with regular transmissions.

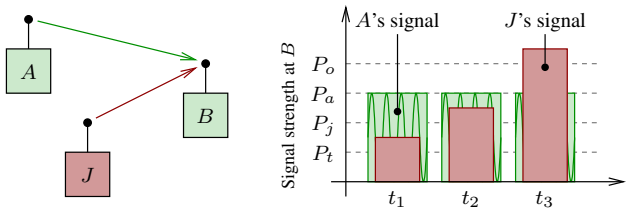



**Figure 3. Required signal strengths for different attacker strategies. Let sender $A$ transmit a message to receiver $B$ such that the corresponding signal arrives at $B$ with strength $P_a$. If an attacker $J$ interferes using a signal that, at $B$, has lower strength than $P_j$, then $B$ successfully receives $A$'s message ($t_1$ in the figure); if, however, $J$'s signal arrives at $B$ with a strength between $P_j$ and $P_o$, the transmission is jammed and $B$ receives no message ($t_2$); finally, if the strength of $J$'s signal at $B$ is even equal or greater than $P_o$ it entirely overshadows $A$'s transmission and $B$ receives $J$'s message ($t_3$).**

- The attacker can *modify messages* by either flipping single message bits or by entirely overshadowing original messages. In the former, the attacker superimposes a signal on the radio channel that converts one or several bits in the original message from zero to one or vice versa. In the latter, the attacker's signal is of such high power that it entirely covers the original signal at the receiver. As a result, the original signal is reduced to noise in the attacker's signal and the original message is replaced by the attacker's message. In either case, in this attack the messages must remain readable by the receiver.
- The attacker can *jam messages* by transmitting signals that cause the original signal to become unreadable by the receiver. The portion of a message the attacker has to interfere with in such a manner depends on the used coding scheme and can be as high as 15% of the message size [13]. As mentioned, we do not consider the case where the attacker can block a message by placing one (or both) of the nodes in a Faraday's cage.

Based on the above actions, we denote by $\mathcal{A}_I$, $\mathcal{A}_M$, and $\mathcal{A}_J$ the strategies where the attacker only inserts, modifies, or jams messages, respectively. Additionally, we use the term $\mathcal{A}_{IMJ}$ for the resulting mixed strategy. $\mathcal{A}_0$ denotes the attacker's strategy to not interfere at all.

Regarding the interception and jamming of messages, we assume that the attacker is aware of the location and configuration of all nodes so that her capabilities are only restricted by the performance of her transceiver. We can therefore abstract away from physical parameters such as node distances, node characteristics (e.g., their antenna gains), and environmental influences, and only consider the power of the original and of the attacker's signal at the receiver. For a given $P_a$ (the strength of the original signal at $B$), we denote by $P_j$ ($P_o$) the minimal required strength of the attacker's signal at $B$ in order to jam (overshadow) a message sent from $A$ to $B$ (see Figure 3). We assume that $P_t < P_a, P_j < P_o$ and that a message from $A$ is successfully received by $B$ if the strength of the attacker's signal at $B$ is less than $P_j$. In addition, we assume that the maximal transmission power of the attacker is finite and we denote by $P_T$ the signal strength that the attacker is able to achieve at the receiver $B$ if she transmits with maximal transmission power on a single channel. However, we do not assume any restrictions on the attacker's energy supply, that is, she is considered to be mains-operated. The attacker's resulting strength in terms of her ability to insert, jam, and overshadow messages will be analyzed in Section 5.2.

## 3. Enabling Robust Communication using Uncoordinated Frequency Hopping

Having described the system and attacker model, we now present the basic intuition behind UFH as countermeasure against communication jammers. We then present and analyze our UFH scheme that enables two nodes to exchange messages of arbitrary length in the presence of a jammer.

### 3.1. Uncoordinated Frequency Hopping

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 point in time, the sender and the receiver will be sending and listening on the same frequency channel $f_j$ (Figure 2). In an undisturbed setting, the receiver will receive each fragment with a small but positive probability that increases with the number of transmission attempts.

Let $M$ denote the message that the sender wants to transfer to the receiver. Due to the sender's rapid change of output channels (as countermeasure against a jammer), $M$ does not fit in one transmission slot, but has to be split into the fragments $M_1, M_2, \ldots, M_l$. These fragments are transmitted one after another with a high number of repetitions. The UFH scheme we consider is *randomized* in the sense that the sender does not relate the frequency $f_j$ for fragment $M_i$ with the channels used and the fragments sent before. Although splitting $M$ into fragments is a straight-forward operation, the reassembly of the received fragments at the receiver is non-trivial if an attacker inserts additional fragments or modifies transmitted ones (that may be hard to distinguish from legitimate fragments).

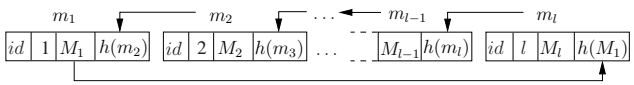



**Figure 4. Hash-linked UFH scheme. Each packet consists of an identifier ($id$) indicating the message the packet belongs to, a fragment number ($i$), the message fragment ($M_i$), and the hash of the next packet ($h(m_{i+1})$). The packets are built in reverse order starting with the last packet $m_l$ linked to the first fragment $M_1$. The linked packed sequence is called *packet chain*.**

Receiving a fragment with (coordinated or uncoordinated) frequency hopping requires the receiver to listen on the correct channel for the complete transmission of the fragment. If the sender's and receiver's hopping frequencies were identical (and with it the time that both stay on a channel before hopping to the next), the successful transmission of a fragment would require precisely synchronized transmission and reception slots to avoid only partially received fragments. In UFH, we do not require the slots to be synchronized by permitting the receiver to switch the channels less often than the sender (Figure 2), thus reducing the number of partially received fragments.

The throughput of the communication with UFH is considerably lower than for coordinated hopping. Given that the nodes did not establish a secret shared key before, the sender will need numerous sending attempts for transmitting each fragment. From the attacker's point of view, the probability to jam a transmitted fragment with randomized uncoordinated frequency hopping is equal to the jamming probability in coordinated frequency hopping, since, for each transmitted fragment, she needs to guess the transmission frequency.

### 3.2. UFH Message Transfer

In this section, we describe our UFH-based *message transfer* protocol; this protocol enables the transfer of messages of arbitrary lengths using UFH. We specify the message fragmentation and transmission carried out by the sender and the reassembly at the receiver.

**Fragmentation.** Before its transmission, a message $M$ is split into $l$ fragments of $\lceil \frac{|M|}{l} \rceil$ bits. Each fragment $M_i$ is then encapsulated in a packet $m_i := id|i|M_i|h(m_{i+1})$, comprising a message identifier ($id$), the fragment number ($i$), the fragment itself ($M_i$), and the hash value of the next packet ($h(m_{i+1})$), where $h(\cdot)$ is a collision-resistant hash function (see Figure 4). The packets are build in reverse order starting with the last packet which is, exceptionally, linked to the first *fragment* ($m_l := id|l|M_l|h(M_1)$), since a link to the complete first packet would result in an unbreakable cyclic dependency in the message fragmentation. The purpose of the hash values is to facilitate the reassembly of the fragmented message. Hash-linking each packet to its successive packet ensures that the attacker cannot excessively increase the complexity of the message reassembly process by inserting or modifying packets. The sender then applies a coding scheme to the packet and distributes the bits of each code within the packet using a pseudo random bit interleaver; here, the used codes and the seed of the bit interleaver are publicly known. The purpose of this coding and interleaving is to make the packets more resistant to bit errors [13]. The vulnerability of this scheme depends on the coding as well as on the type and strength of the jammer as discussed in Section 5.3.

**Transmission.** The message is transferred over the UFH channel with a high number of repetitions in order to compensate for the lacking coordination. The sender repeats transmitting the sequence of packets $m_1$, ..., $m_l$ on randomly selected channels and, in parallel, listens on the input channels to record all incoming packets. The sender immediately starts the reassembly phase for the received packets but continues sending the message packets either until it succeeds in reassembling an expected reply or until the execution of the protocol reaches a threshold value (in time or number of repetitions). A protocol abortion could be due to, e.g., too strong jamming capabilities of the attacker or the non-availability of the second node.

**Reassembly.** The structure of the hash-linked message avoids an exhaustive search for assembling the received fragments into reasonable messages. Received packets can directly be linked to each other when their preceding or successive packet is received ($m_1$ is linked to $M_l$). More precisely, on the reception of a new packet $m_i$ the receiver tries to identify $m_i$ as successor (predecessor) of any received packet $m_{i-1}$ ($m_{i+1}$) by comparing, first, the message identifiers and, second, the hash in $m_{i-1}$ ($m_i$) with $h(m_i)$ ($h(m_{i+1})$). This involves the computation of one hash function as well as string comparisons. If the verification succeeds for a pair of packets $m_{i-1}$ and $m_i$ ($m_i$ and $m_{i+1}$), the receiver connects the fragments to form or extend a *packet chain*. After enough repetitions, $B$ will have received all $l$ packets and the packet chain will be complete. Once the possible combinations of fragments have been reassembled to one or more messages, the receiver starts to process the semantic meaning of the message and to compose his reply message.

### 3.3. Security Analysis of the UFH Message Transfer Protocol

We now analyze the security properties of the presented UFH message transfer protocol and motivate our choice of the scheme, focusing on the properties introduced by the hashes.

Each packet in a packet chain is linked to its successor by a hash. This makes it infeasible for the attacker to create a branch in the original packet sequence $chain_A(M)$ built by the sender $A$ so that the receiver $B$ has to pursuit different successors for a single packet (see Figure 5), which would lead to an exponential growth of the search space for the reassembly process at the receiver. More precisely, given two consecutive packets $m_{i-1} = id|i|M_{i-1}|h_i$ and $m_i = id|i|M_i|h_{i+1}$ where $h_i = h(m_i)$ and $2 < i \leq l$, an attacker (as defined in Section 2.2) cannot create a packet $m'_i = id|i|M'_i|h'_{i+1}$ such that the fragment $M'_i$ is accepted as a genuine successor of $M_{i-1}$ by a correct receiver. The reason is that in order for the fragment $M'_i$ to be accepted as a genuine successor of $M_{i-1}$, the hash value of $m'_i$ must be equal to $h_i$, i.e., $h_i = h(id|i|M_i|h_{i+1}) = h(id|i|M'_i|h'_{i+1})$ must hold. However, finding a fragment $M'_i$ and a hash value $h'_{i+1}$ such that this condition is met means finding a collision for $h(\cdot)$, which is considered infeasible for a computationally bounded attacker.

Besides the regular hash links, the last packet in a chain is linked to the fragment of the first packet, thus avoiding that the attacker can insert additional chain heads that all point to the same chain. More precisely, given a packet $m_1 = id|1|M_1|h_2$ the attacker cannot create a packet $m'_1 = id|1|M'_1|h_2$ such that $M'_1$ is accepted as a genuine head fragment for the chain started by $h_2$, because she would have to find a fragment $M'_1$ such that $h(M'_1) = h(M_1) = h_l$. However, finding such a fragment $M'_1$ means finding a collision for $h(\cdot)$, which is considered infeasible for the attacker.

The attacker therefore cannot create branches neither in legitimate packet chains transmitted by the sender nor in packet chains that she created and inserted herself.

In summary, *without* linked packet chains the attacker can insert arbitrary packets that, as long as they comply with the described packet structure (Figure 5), the receiver cannot distinguish from legitimate packets. Since the number of fake packets that the attacker could insert for each legitimate packet $m_i$ is proportional to the number of transmissions, the message reassembly would require time that is exponential in the number of message repetitions (given that the receiver can identify the correctly reassembled message). On the other hand, *with* packet chains, the attacker can only introduce entire replayed or self-constructed chains, otherwise her packets will be dropped without that the receiver reassembles them. The reassembly of the legitimate packet chain is thus independent of the number of packets inserted by the attacker.

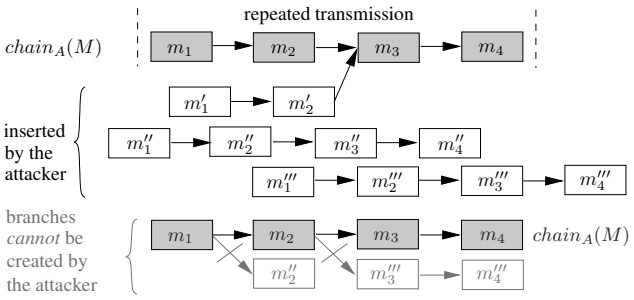



**Figure 5. Packet chains created by the message transfer protocol. $chain_A(M)$ represents the legitimate packet sequence transmitted by sender $A$ (marked in gray). The attacker can send (replayed or own) packet chains and, with sufficient power, try to prevent the reception of all repeated transmissions of each legitimate packet $m_i$. Although she can create a fusion from her own chain to the original $chain_A(M)$ ($m'_2$ linked to $m_3$), she *cannot* purposefully create a packet $m'_i$ such that $m'_i$ is linked to $m_{i-1}$ (branch at $m_1$ to $m''_2$ or at $m_2$ to $m'''_3$). A fake first packet $m'_1$ containing the correct hash but a fake message $M'_1$ will be detected when the last packet $m_l$ is received (which may even happen before the reception of $m_1$).**

We point out that the attacks in which the attacker inserts or replays messages (i.e., complete packet chains) cannot be eliminated by the UFH message transfer protocol. These attacks must thus be identified by the (key establishment) protocols running on top of the UFH protocol (Section 4), e.g., by making use of signatures, timestamps, and message buffering. The impact of these attacks on the message transfer is discussed in Sections 5.4 and 5.5.

### 3.4. Discussion of Alternative Schemes

A limitation of the proposed UFH message transfer protocol is that all packets of a message must be received before the message can be reassembled. In principle, forward error correction techniques (e.g., based on erasure [18] or fountain codes [14]) allow for schemes where a message can be reassembled if only a subset of all packets is available. However, as discussed above, to avoid that the attacker can exponentially increase the reassembly time at the receiver, each of these subsets must be identifiable as belonging to the same message *without* relying on a shared key. Using a single hash chain to link packets of the same mes-

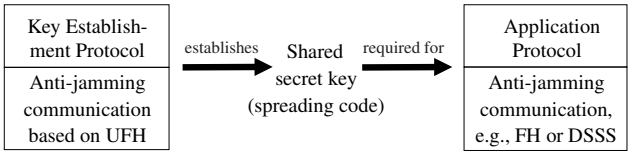



**Figure 6. Use of Uncoordinated Frequency Hopping (UFH). The jamming-resistant communication using UFH does not rely on a secret spreading code (key). Hence, UFH can be used by two nodes without pre-shared keys to execute a key establishment protocol in the presence of a jammer, resulting in a secret key shared between the nodes. This key then serves as input for a coordinated spread-spectrum system. Once the key is set up, UFH is abandoned in favor of, e.g., coordinated frequency hopping.**

sage is not beneficial because verifying this packet chain requires that the receiver receives *all* packets, which nullifies the advantage of the forward error correction. A possible solution to this problem could be to append more than one hash link to each packet such that subsets of packets will be connected to a complete chain with high probability. Another promising approach might be to use distillation codes [11] that use (RSA authenticated) Merkle hash trees as one-way accumulators. As for the multiple-chains case, here each packet would contain more than one hash value. This increases the overhead per packet (and thus the total number of packets per message) in favor of a reduced number of required successful packet receptions. We leave the evaluation and adaption of these and new fragmentation/reassembly schemes as a topic for future research.

## 4. UFH Key Establishment

Having presented our UFH message transfer protocol, we now show how it supports a jamming-resistant key establishment by which the nodes can establish a shared hopping sequence for later coordinated frequency hopping (see Figure 6).

Our UFH key establishment is divided into two stages. In stage 1, the nodes execute a key establishment protocol and agree on a shared secret key $K$ using UFH; various key establishment protocols can be used in this step. Then, in stage 2, each node transforms $K$ into a hopping sequence (using linear feedback shift registers and channel mappers [16]) and, subsequently, the nodes communicate using coordinated frequency hopping. The first message in stage 2 would typically be a key confirmation verifying the successful key agreement and, additionally, it would be used for the frequency hopping synchronization between the nodes. Note that the established key is not used for encrypting or signing sensitive data but exclusively for establishing the hopping sequence; a weak choice does thus not disclose any confidential data.

A requirement for the key establishment protocol in our scenario is the authentication of all exchanged messages in order to prevent the insertion of fake messages. The protocol we propose therefore contains the exchange of public key certificates issued by the CA, which can be omitted in the case where the nodes exchanged or preloaded their certificates prior to the protocol execution. In addition, the protocol uses timestamps to preclude replay attacks and a (short-term) history buffer to detect and drop duplicate messages during the validity of the timestamps. Messages can be received more than once during their validity either due to replay attacks or due to the repetitive message transmissions which are inherent to our UFH message transfer protocol.

As an example, we consider an authenticated Diffie-Hellman key agreement protocol in which $A$ will establish a key with any node in its power range. We focus on the Elliptic Curve Cryptography (ECC) based Station-to-Station (STS) Diffie-Hellman protocol [3]. It is executed as follows (see Figure 7): Let $P$ be the generator of a cyclic group $\mathbb{G}$ with prime order $p$. Let $Sig_X(\cdot)$ be the signature by node $X$ of the string in brackets and let $\{\cdot\}_K$ be its encryption with key $K$. $A$ selects a (pseudo-)random element $r_A \leftarrow_R \mathbb{Z}_p$ and broadcasts its public key certificate, a timestamp $T_A$, and the credential $r_AP$, including the digital signature of the message. Any node $B$ in the transmission range may reply with a symmetric message containing its credential $r_BP$ and $A$'s timestamp $T_A$. Based on the received messages, both compute the shared key, $K = r_A(r_BP)$ for $A$ and $K = r_B(r_AP)$ for $B$. Due to the nature of the DH-key exchange and the fact that public key certificates can be prepended to the DH-messages, two UFH messages are sufficient in order to agree on a key. In stage 2, $A$ uses coordinated frequency hopping to provide an authenticated proof of its secret knowledge by sending an encrypted signature $\{Sig_A(r_AP \parallel r_BP)\}_K$.

**Discussion.** The DH-protocol we consider is not mandatory for the UFH scheme and can be optimized for size (e.g., using short signature schemes [6]) or be replaced by other protocols such as a key transport protocol [8] or an ID-based key negotiation using bilinear maps [7]. Common to the design of the UFH key establishment protocols is that each message is authenticated with a digital signature and either contains a timestamp or a previously contributed random value (Figure 7). Hence, the authenticity and freshness of all received messages can be verified: fake messages will fail the signature verification and replayed messages will

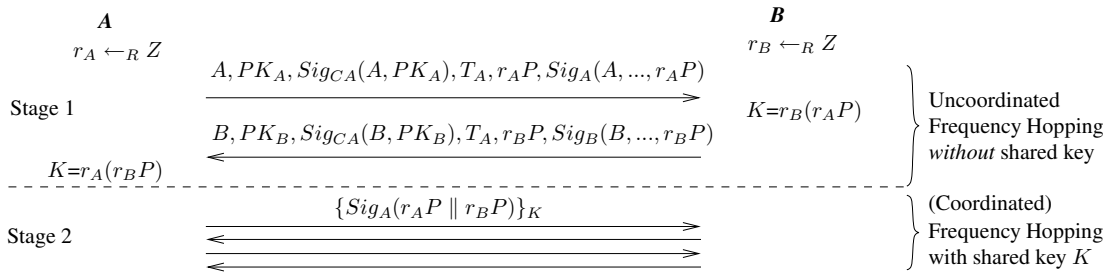



**Figure 7. UFH key establishment using an authenticated DH protocol. In stage 1, $A$ uses UFH to broadcast its certified public key and its key contribution $r_AP$ for the elliptic curve DH protocol. Replay attacks are prevented by the timestamp $T_A$ and the signature of the message. Any node $B$ in the power range may answer by sending its DH-contribution. In stage 2, $A$ transmits a key acknowledgment, then the nodes can send arbitrary messages using coordinated frequency hopping.**

be detected due to an expired timestamp or an entry in the (short-term) history buffer. The period during which a message is considered valid is defined by the receiver and is usually in the order of time that is required to successfully transmit the message using our UFH scheme. Messages which are replayed by the attacker during their validity period are thus of no harm and might even help the sender to deliver them. Although an attacker may be able to replay the authenticated message of the receiver within the acceptable time interval in another protocol session, this does still not enable her to defer the secret hopping sequence from it (the key contribution of the legitimate device remains secret).

Apart from authenticating the sender of a message, the attached signatures suit a second purpose on the UFH message transfer level. Once the possible combinations of fragments have been reassembled to one or more messages, the signature of each message is used to verify the reassembly. Thus, we do not need to add checksums to the messages because the signature provides the same type of error detection check.

## 5. Performance Analysis and Evaluation

In this section, we evaluate the efficiency of the presented UFH message transfer protocol in terms of its throughput and resource requirements. We show that, compared to insertion ($\mathcal{A}_I$), modification ($\mathcal{A}_M$), and mixed strategies ($\mathcal{A}_{IMJ}$), jamming ($\mathcal{A}_J$) is the best attacker's strategy (see Figure 10). Throughout this analysis, we assume that the hopping frequency of the receiver $f_B$ is much slower than the hopping frequency of the sender $f_A$ (Figure 2). We can therefore neglect packet losses caused by the lack of synchronization between sender and receiver as they only affect every $\frac{f_A}{f_B}$-th packet and are thus rare events compared to the likelihood that the receiver listens on an incorrect channel (i.e., $\frac{f_B}{f_A} \ll (1 - \frac{1}{c})$). We also assume that the number of devices in the neighborhood of the receiver that use the same channels is low (i.e., $\ll c$) and that the impact of unintentional interference is thus negligible.

Given an attacker strategy $\mathcal{A}_x$ ($x \in \{0, I, M, J, IMJ\}$), we use the expected number of packets that have to be sent in order for a message to be successfully received ($N_x$) as the main metric for the performance of our UFH message transfer protocol. For ease of comparison, we additionally derive therefrom the relative throughput of UFH ($\Phi_x$) with respect to coordinated frequency hopping. We note that our UFH scheme is probabilistic in nature and that the probability that a message is successfully received depends on the strength of the attacker and the number of times the message has been (re)sent (see Figure 11(a)).

### 5.1. Communication without an Attacker (Attacker Strategy $\mathcal{A}_0$)

First of all, we evaluate the performance of the presented UFH message transfer protocol in the absence of an attacker, which is equal to the situation where the attacker embarks on strategy $\mathcal{A}_0$. In this case, the probability that a particular packet is successfully received is

$$p_m^{\mathcal{A}_0} = 1 - \prod_{i=0}^{c_m-1} \left(1 - \min\left\{\frac{c_n}{c-i}, 1\right\}\right) \geq 1 - \left(1 - \frac{c_n}{c}\right)^{c_m} \quad (1)$$

where $c_n \leq c$ ($c_m \leq c$) is the number of channels on which the sender (receiver) simultaneously sends (receives). From the receivers point of view, the transfer of an entire message is completed once it has successfully received all $l$ message fragments. Let the random variable $Y$ represent the number of times that the sender is required to retransmit the sequence of packets in order to successfully transfer the

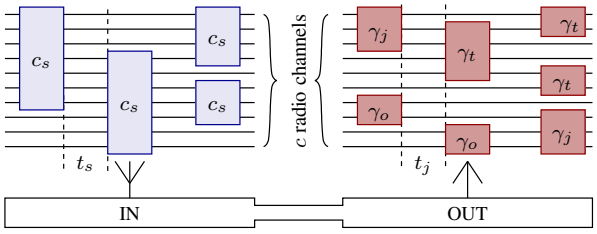



**Figure 8. Input and output channel configuration of the attacker. We denote by $c_s$ the number of channels that the attacker can sense in parallel and by $t_s$ ($t_j$) the required time to switch the frequency of the input (output) channels. The number of channels on which the attacker can send ($\gamma_t$), jam ($\gamma_j$), and overshadow ($\gamma_o$) is bounded by her transmission power.**

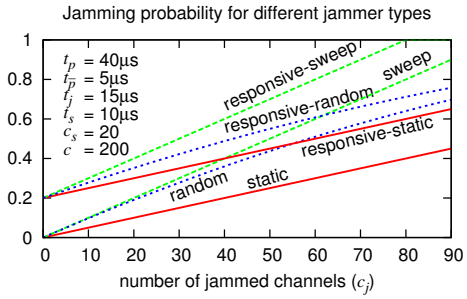



**Figure 9. Probability that a packet is jammed as a function of the number of jammed channels for different jammer types.**

message. The probability that a message transfer is still incomplete after $i$ (re)transmissions is then $\mathsf{P}[Y \geq i+1] = \mathsf{P}[Y > i] = 1-(1-(1-p_m^{\mathcal{A}_0})^i)^l$. Hence, the expected total number of packets that have to be transmitted in order to successfully transfer a message is $N_0 = N(p_m^{\mathcal{A}_0})$, where

$$\begin{aligned} N(p_m^{\mathcal{A}_0}) &:= \sum_{i=0}^{\infty} \mathsf{P}[Y=i] il \\ &= \sum_{i=0}^{\infty} \left(\mathsf{P}[Y \geq i] - \mathsf{P}[Y > i]\right) il \\ &= \sum_{i=0}^{\infty} \left(1-\left(1-\left(1-p_m^{\mathcal{A}_0}\right)^i\right)^l\right) l. \end{aligned} \quad (2)$$

### 5.2. Jamming Performance of the Attacker

As introduced in Section 2.2, we assume that the maximal transmission strength of the attacker is finite, and denote by $P_T$ the signal strength that the attacker is able to achieve at the receiver $B$ if she transmits with maximal transmission power on a single channel. We also assume that the frequency-dependent variance in the signal attenuation is negligible over the communication frequency range of $\mathcal{C}$ and that the attacker can divide her transmission power arbitrarily among all $c$ channels. The only restriction is therefore that for all combinations of output channel assignments in which the attacker sends on $\gamma_t$, jams on $\gamma_j$, and overshadows on $\gamma_o$ channels, $\gamma_t P_t + \gamma_j P_j + \gamma_o P_o \leq P_T$ and $0 \leq \gamma_t, \gamma_j, \gamma_o \leq c$ must hold at all times. Consequently, we can derive $\lfloor \frac{P_T}{P_t} \rfloor$, $\lfloor \frac{P_T}{P_j} \rfloor$, and $\lfloor \frac{P_T}{P_o} \rfloor$ as upper bounds on the number of channels on which the attacker can simultaneously send ($c_t$), jam ($c_j$), or overwrite ($c_o$), respectively. The number of channels that the attacker can concurrently sense is denoted by $c_s$. Furthermore, we assume that the attacker is able to receive and transmit in parallel and that the channels on which she receives and transmits can be switched independently of each other. The required time to switch the frequency of the input (output) channels is denoted by $t_s$ ($t_j$) (see Figure 8).

Using the above introduced terms, the jamming capabilities of the attacker can then be expressed as the probability $p_j$ with which a packet $m_i$ is jammed. This probability depends on the length of the packet and on the strength of the attacker: the longer the packet is and the stronger the attacker is, the more likely it is that the packet will get jammed. More precisely, let $t_p$ be the time to transmit a packet $m_i$ on frequency channel $f_i$ and $t_{\overline{p}}$ be the minimum jamming period during which the attacker has to interfere with the transmission of $m_i$ such that it cannot be decoded. Following previous classifications [16], we distinguish between static, sweep, random, responsive, and hybrid jammers (see Figure 9).

Static, sweep, and random jammers do not sense for ongoing transmissions but permanently jam on $c_j$ channels. However, whereas sweep and random jammers switch the output channels after a duration of $t_{\overline{p}}$, static jammers remain on the same channels for a time $t \gg t_p$. Also, sweep jammers systematically update the output channels in a way that after $\lceil \frac{c}{c_j} \rceil$ jamming cycles all channels have been jammed once (but do not have to follow a particular order). Random jammers, on the other hand, always choose $c_j$ channels at random and might thus jam the same channels more than once in subsequent jamming cycles. The number of jamming cycles per packet is in both cases $n_j := t_p/(t_{\overline{p}} + t_j)$. Hence, the probability that a packet is jammed is $p_j \leq \frac{c_j}{c}$ for a static, $p_j \leq \min\{\frac{n_j c_j}{c}, 1\}$ for a sweep, and $p_j \leq 1-(1-\frac{c_j}{c})^{n_j}$ for a random jammer.

Responsive jammers differ from the above mentioned in that they initially solely sense for ongoing transmissions

and enable the output channels only when a signal has been detected. Hybrid jammers, finally, are a combination of responsive and permanent jammers that have their output channels already enabled while they are scanning for signals. During an ongoing transmission of a packet, the attacker can switch the input channels at most $n_s := (t_p - t_{\overline{p}} - t_j)/t_s$ times such that the transmission is detected early enough to adapt the output channels and jam the packet for a duration of $t_{\overline{p}}$. Hence, for a responsive-sweep jammer the probability that a packet is successfully jammed is $p_j \leq \frac{\gamma}{c} + (1 - \frac{\gamma}{c}) \min\{\frac{n_s c_s}{c-\gamma}, 1\}$, $\gamma = \min\{n_j c_j, c\}$. Similar derivations yield a jamming probability of $p_j \leq \frac{c_j}{c} + (1 - \frac{c_j}{c}) \min\{\frac{n_s c_s}{c-c_j}, 1\}$ for responsive-static, and $p_j \leq \beta + (1-\beta) \min\{\frac{n_s c_s}{c}, 1\}$, $\beta = 1-(1-\frac{c_j}{c})^{n_j}$ for responsive-random jammers. It follows that, of the introduced jammer types, responsive-sweep jammers are the most powerful ones (Figure 9).

As illustrated in Figure 9, for all considered jammer types the jamming probability increases with the number of channels that the attacker can jam in parallel. Clearly, this probability is 1 if all channels can be jammed at once (i.e., if $c_j = c$). However, since the attacker needs to jam only a fraction of a packet to prevent the successful reception of the packet and because a reactive jammer actively searches for transmissions, the attacker's jamming probability can reach 1 even for $c_j < c$. In the example given in Figure 9 a responsive-sweep jammer is able to jam all packets if its transmission power allows the attacker to jam $\geq 80$ channels in parallel.

Following the above analysis, we can also deduce the probability $p_o$ that a packet is (systematically) overwritten by substituting $c_j$ with $c_o$ in the expressions for $p_j$.

### 5.3. Impact of the Packet Coding on the Minimum Jamming Period

The minimum jamming period $t_{\overline{p}}$ during which the attacker has to interfere with the transmission of a packet such that it cannot be decoded by the receiver depends on the coding scheme applied to the packet as well as on the type and strength of the jammer. Here, we assume that jamming always causes a bit error whereas a realistic assumption would be that jamming results in an error with a probability of 0.5 [13]; our presented results therefore represent upper bounds on the attacker's performance.

A non-responsive (i.e. static, sweep, or random) jammer does not scan the channel for transmissions and thus does not know when a packet starts. The attacker therefore cannot exploit her knowledge about the used coding schemes and bit interleaving to jam the packets more efficiently. For this jammer, the minimum jamming period $t_{\overline{p}}$ is thus determined by the number of bits that the used coding scheme can correct. More precisely, if the coding scheme can correct $t$ out of $n$ bits we get $t_{\overline{p}} \geq \frac{t+1}{n} t_p$, where $t_p$ is the time to transmit a packet.

If an attacker using a non-responsive jammer has precise information about the start of a packet (but does not know on which channel it is transmitted) she can leverage on this knowledge in order to reduce the required jamming period to a minimum. If this attacker synchronizes her jamming burst with the transmission time of $t + 1$ bits of a code word, the corresponding byte can no longer be correctly decoded by the receiver and thus the entire packet will be faulty. The required information about the positions of the bits in the packet can easily be obtained as the coding and bit interleaving schemes are publicly known. The minimum jamming period for this jammer is therefore $t_{\overline{p}} \geq (t+1)\, t_b$, where $t_b$ is the time to transmit one bit. We note that obtaining the starting time of a packet and synchronizing the jamming bursts with the bit transmissions without sensing the packet is difficult. Even if the attacker observes the transmission of one of the predecessors of a packet, variances in the radio stack of the sender and random delays due to the switching of the radio channel between two successive packet transmissions complicate the prediction of the packet starting time. Consequently, the attacker is likely to jam not only single bits but bit groups in order to account for the imprecision in her timing information. Finally, we note that this jamming attack can be mitigated by randomly delaying packet transmissions or by randomly choosing the applied bit interleaving scheme out of a set of possible schemes. Since de-interleaving a packet is considered to be an efficient operation, using more than one interleaving scheme to chose from increases the work per packet for the receiver only marginally but increases the required minimum jamming period for the attacker (almost) proportionally to the number of used schemes.

In the case of a responsive jammer, a packet is considered to be jammed if its transmission is detected before the already transmitted data allows the receiver to decode the packet. Here, $t_{\overline{p}}$ therefore corresponds to the minimum remaining transmission time of a packet without which the receiver is not able to decode the packet; that is $t_{\overline{p}} \geq \frac{t+1}{n} t_p$.

We note that with respect to a responsive jammer, sending a non-encoded, shorter packet might be beneficial for the sender and receiver. Such a scheme would, however, be vulnerable to non-responsive (proactive) jamming. Therefore, in order to maintain resistance against non-responsive jamming, coding and interleaving schemes need to be used.

### 5.4. Communication in the Presence of an Attacker

Having analyzed the capabilities of different jammer types, we next analyze the impact of the various attacker strategies on the throughput of our UFH transfer protocol (see Figure 10).

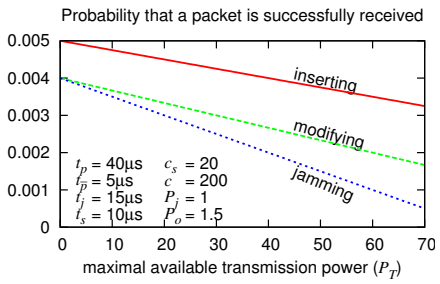



**Figure 10. Impact of different attacker strategies on the probability that a packet is successfully received by the receiver. The most effective choice for the attacker is therefore to use a hybrid responsive-sweep jammer and to choose strategy $\mathcal{A}_J$ (jamming).**

**Attacker Strategy $\mathcal{A}_J$.** We express the jamming strength of an attacker as the probability $p_j$ with which a packet is successfully jammed. Following (1), the probability that a particular packet is successfully received is in this case

$$p_m^{\mathcal{A}_J} = 1 - \prod_{i=0}^{c_m-1} \left(1 - \min\left\{\frac{c_n}{c-i}, 1\right\}(1-p_j)\right) \quad (3)$$

resulting in an expected number of required packet transmissions of $N_J = N(p_m^{\mathcal{A}_J})$.

**Attacker Strategy $\mathcal{A}_I$.** We assume that the receiver can handle all additionally inserted messages and only consider their impact on the throughput of the UFH message transfer; the additional resource requirements that maliciously inserted messages can introduce are discussed in Section 5.5. By definition of $P_j$, packets of the attacker whose corresponding signal strength at the receiver is less than $P_j$ do not interfere with regular packets and thus do not have any impact on their transmission. As the attacker can send on at most $c_j$ channels with a signal strength $\geq P_j$, the probability that a particular packet is successfully received is

$$p_m^{\mathcal{A}_I} \geq 1 - \prod_{i=0}^{c_m-1} \left(1 - \min\left\{\frac{c_n}{c-i}, 1\right\}\left(1 - \frac{c_j}{c}\right)\right) \quad (4)$$

and the expected number of required packet transmissions is $N_I = N(p_m^{\mathcal{A}_I})$.

**Attacker Strategy $\mathcal{A}_M$.** Modifying the content of a regular packet such that it is still accepted by the receiver is assumed to be infeasible for the considered attacker due to the hash links (see Section 3.3). Partially modifying packets is thus an (expensive) form of jamming and its impact is considered in the analysis of $\mathcal{A}_J$. Consequently, packets must be entirely replaced (i.e., overwritten) by the attacker with valid alternatives in order to be accepted. The attacker's ability to modify messages is therefore equal to her overwriting capabilities which, as shown in Section 5.2, can be expressed as the probability $p_o$ with which a packet is successfully overwritten. Hence, when the attacker applies strategy $\mathcal{A}_M$, the probability that a particular packet is successfully received is

$$p_m^{\mathcal{A}_M} = 1 - \prod_{i=0}^{c_m-1} \left(1 - \min\left\{\frac{c_n}{c-i}, 1\right\}(1-p_o)\right) \quad (5)$$

resulting in $N_M = N(p_m^{\mathcal{A}_M})$ required packet transmissions.

**Optimal Strategy.** Based on the above analysis, we state that jamming is the best strategy for the attacker.

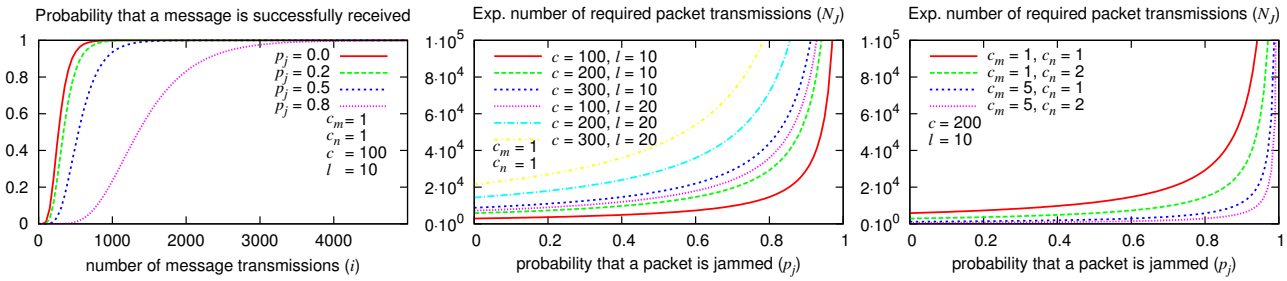
**Theorem 1.** *For all attacker types (Section 5.2), the optimal attacker's strategy, which minimizes the throughput of the UFH message transfer, is jamming ($\mathcal{A}_J$).*

*Proof.* By definition of $p_j$ and $p_o$, and as a consequence of $c_o = \lfloor \frac{P_T}{P_o} \rfloor \leq \lfloor \frac{P_T}{P_j} \rfloor = c_j$ we have $p_j \geq p_o$ and $p_j \geq \frac{c_j}{c}$. From (3), (4), and (5), we can directly deduce $p_m^{\mathcal{A}_J} \leq p_m^{\mathcal{A}_I}$ and $p_m^{\mathcal{A}_J} \leq p_m^{\mathcal{A}_M}$. Hence, it follows that $N_J \geq N_I$ and $N_J \geq N_M$, which means that jamming is the best strategy for an attacker that can only perform one single action at a time. In the general case, the task of finding the best mixed strategy can be formulated as an optimization problem: The function to be minimized is $p_m^{\mathcal{A}_{IMJ}} = 1 - \prod_{i=0}^{c_m-1}(1 - \min\{\frac{c_n}{c-i}, 1\}\psi(\gamma_t, \gamma_o, \gamma_j))$, where $\psi(\gamma_t, \gamma_o, \gamma_j) = \left(1 - \frac{\gamma_t}{c}\right)(1 - p_o|_{c_o=\gamma_o})(1 - p_j|_{c_j=\gamma_j})$ with the constraints $\gamma_t, \gamma_o, \gamma_j \in \mathbb{N}_0^+$, $0 \leq \gamma_t, \gamma_o, \gamma_j \leq c$, and $P_T \geq \gamma_o P_o + (\gamma_j + \gamma_t)P_j$. Given $c_n$ and $c_m$, $p_m^{\mathcal{A}_{IMJ}}$ is minimal if and only if $\psi(\gamma_t, \gamma_o, \gamma_j)$ is minimal. Moreover, by definition of $p_j$ and $p_o$, we have $\forall \gamma, 0 \leq \gamma \leq c : p_j|_{c_j=\gamma} = p_o|_{c_o=\gamma}$ and $p_j|_{c_j=\gamma} \geq \frac{\gamma}{c}$. Hence, $\psi(\gamma_t, \gamma_o, \gamma_j) \geq (1 - p_j|_{c_j=\gamma_t})(1 - p_j|_{c_j=\gamma_o})(1 - p_j|_{c_j=\gamma_j})$ and, because $P_j < P_o$, it follows that $p_m^{\mathcal{A}_{IMJ}}$ is minimized if $\gamma_t = 0$, $\gamma_o = 0$, and $\gamma_j = \lfloor P_T/P_j \rfloor$; that is, if the attacker solely jams. □

The impact of the attacker's jamming capabilities on the performance of our UFH message transfer protocol is depicted in Figure 11. We observe that increasing the number of channels $c$ is less harmful than increasing the number $l$ of packets per message. Also, sending or receiving on more than one channel significantly reduces the number of required transmissions.

## 5.5. Resource Requirements

In the case that an attacker inserts her own messages (for instance by replaying previously recorded packets), the




(a) (b) (c)

**Figure 11. Impact of the attacker's jamming probability on the performance of the UFH message transfer protocol (a)-(c). One can observe that increasing the number of channels $c$ is less harmful than increasing the number of packets per message $l$ (b). Also, being able to send or receive on more than one channel significantly reduces the number of required packet transmissions (c).**

number of messages that a receiver can handle in parallel becomes a critical factor. In this context, handling a message means to *receive*, *assemble*, and *store* its associated packets as well as, in the case of the presented key establishment protocol, to *verify* the appended signature.

**Reception and assembling.** As the number of received packets per time unit is less or equal to coordinated frequency hopping, we can assume that the receiver is able to receive all packets. Assembling a packet essentially means computing its hash value and comparing it to potential predecessors, which both can be implemented very efficiently (even on computationally weak devices), and is thus also considered to be feasible for all packets.

**Storage.** In order to assess the required storage space, we assume that, once there is no more space for new packets left, the oldest packets (i.e., the packets with the longest duration since their reception) will be deleted first. Thus, a receiver must at least be able to store all received packets between the first and the last packet of a regular message. The expected maximal time period between the first and the last packet of a message is $N_J$ sender hops (see Section 5.4). During this time, an expected amount of less than $N_J \sum_{i=0}^{c_m-1} \min\{\frac{c_t}{c-i}, 1\} \leq N_J c_m$ additional packets is received. To be more concrete, we give an example: For a typical packet length of $|m_i| = 40$ bytes, a fragmentation into $l = 10$ packets, a set of $c = 200$ channels, $c_m = c_n = 1$ channel for sending and receiving, $c_t = 50$ attacker channels for insertion, and a jamming probability of 80%, we get $N_J \approx 30\,000$ for the number of packets transmitted by the sender (see Figure 11(b)). This results in the reception of about 7 500 packets and corresponds to a required storage capacity of about 290 kbytes on the receiver side. We argue that nowadays this amount of space is clearly available on notebook- or handheld-class devices.

**Signature verification.** The expected number of packets that the attacker must send in order to insert a message which is received by the receiver is $N(p_t)$, where $p_t \leq 1 - \prod_{i=0}^{c_m-1}(1 - \min\{\frac{c_t}{c-i}, 1\})$ is the probability that a packet inserted by the attacker is successfully received. The expected number of additional messages that can be inserted by the attacker during the transfer of a regular message is thus $N_J/N(p_t)$, resulting in a total of $N_J/N(p_t) + 1$ signature verifications per message transfer. Using the parameters of the above example, this gives a total of about 160 signature verifications. Again, we argue that this is clearly within the realms of possibility with today's hardware. For instance, it takes only about 6 s to verify 160 ECDSA-256 bit signatures on a typical handheld device with a 206 MHz StrongARM CPU, and less than a second on a notebook-class device [17].

## 5.6. Comparison of Coordinated and Uncoordinated Frequency Hopping

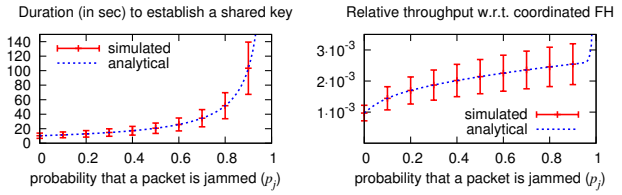
Let $f_H$ be the hopping frequency of a given transceiver and $|m_i^*| \propto 1/f_H$ be the resulting length of a packet for a coordinated frequency hopping scheme; $|M_i^*|$ then denotes the length of a fragment sent per packet in that scheme. Provided that the fragment length is large enough to comprise the shortest possible fragment of our UFH scheme (i.e., $|M_i^*| \geq 1 + |h(\cdot)|$), our UFH message transfer protocol enables anti-jamming communication without a shared key and achieves the same jamming robustness as the coordinated scheme at the expense of a lower throughput and an increased storage consumption. More precisely, using

the same coding scheme and packet length (i.e., $|m_i| = |m_i^*|$) results in the same probability that a packet gets jammed for coordinated and uncoordinated frequency hopping. The hash value added in the UFH scheme, however, decreases the payload per packet to $|M_i| = |M_i^*| - |h(\cdot)|$ and thus increases the number of packets per message to $l = \lceil |M|/|M_i| \rceil > l^* = \lceil |M|/|M_i^*| \rceil$. The relative throughput of the presented UFH protocol compared to coordinated frequency hopping is thus $\Phi_J = N_J/N_J^*$, where $N_J^* = N(1-p_j)|_{l=l^*}$ (see Figure 12(b)).

### 5.7. Illustrative Example

We illustrate the feasibility of the proposed jamming-resistant key establishment protocol by showing a possible implementation for a Bluetooth-like transceiver. Although Bluetooth was not designed to resist advanced jamming attacks, it is nevertheless a representative example for what can be achieved with low-cost hardware. According to [1], we assume a sender (receiver) hopping frequency of 1.6 kHz (160 Hz) and a symbol rate of 1 Mbit/s. Other than for Bluetooth, the number of frequency channels is assumed to be 200. Regarding the key establishment protocol, we follow the current NIST recommendations [4] and advocate a mid-term security level of 128 bits for keys and signatures as well as a short-term security level of 112 bits for hash links. Hence, we assume 256-bit prime fields for the elliptic curves ($|Sig(\cdot)| = |PK_X| = 512$ bits) and a hash size of 112 bits for the hash links. Identities and time-stamps are encoded with 64 bit integers. The size of the exchanged messages is thus $|M| = 2176$ bits $= 272$ bytes (Figure 7).

A packet $m_i$ consists of a message id (34 bits), a frame id (6 bits), the payload $M_i$ (168 bits), and the hash value of the next packet (112 bits). We assume the application of a Reed-Solomon error-correcting code that encodes 8-bit blocks into 15-bit blocks, achieving a jamming ratio of 20% [13]. The length of an encoded packet is thus $320 \cdot 15 / 8 = 600$ bits and fits well in a hopping slot of 1/1600 Hz $\cdot$ 1 Mbit/s $= 625$ bits. The number of packets per message is $l = \lceil 272 / 21 \rceil = 13$ for uncoordinated and $l^* = \lceil 272 / 35 \rceil = 8$ for coordinated frequency hopping. The resulting duration for a key establishment and the achievable throughput with respect to coordinated frequency hopping are depicted in Figures 12(a) and 12(b), respectively. We observe that even under harsh conditions where 80% of all packets are jammed, the expected duration to establish a key is well below one minute.




(a) (b)

**Figure 12. (a) Duration for a key establishment and (b) relative throughput with respect to coordinated frequency hopping as a function of the attacker's jamming probability for a Bluetooth-like transceiver using 200 channels. In addition to the analytical result, the average and standard deviation of 100 000 simulated key establishments is shown.**

## 6. Related Work

Wireless communication jammers have been widely analyzed and categorized in terms of their capabilities (e.g., broadband or narrowband) and behavior (e.g., constant, random, responsive, sweep) [12, 16, 25]. The jammer models used in prior works [12, 16, 21, 24, 25] cover the interference with transmissions in terms of signal jamming or dummy packet/preamble insertions. In our work we extend these models by covering also protocol-specific DoS attacks. Therefore, our attacker model captures signal jamming and overshadowing as well as message modification and insertion.

As well-known countermeasures against these communication jamming attacks, spread-spectrum techniques such as DSSS, chirp [2], and in particular frequency hopping [1, 16] achieve frequency diversity over the communication channel. Specific coding strategies [13] can additionally strengthen the jamming resistance of transmitted messages by increasing the fraction of the packet that the attacker needs to interfere with in order to prevent its successful decoding at the receiver.

Recently, anti-jamming strategies have been analyzed more intensely in the area of sensor networks; a common strategy taken in those approaches is to detect and localize the jammer in order to avoid the jammed region [12, 23, 25] or the jammed frequencies [24] during further communications. In [21] the authors note that wired sensor pairs as well as coordinated and uncoordinated (random) channel hopping can be used for the timely delivery of short warning message in sensor networks. The channel hopping solutions proposed in these previous works are, however, very specific to the considered sensor-networking applications and cannot be used to solve the key establishment problem observed in this paper.

Pre-establishing keys between devices in ad-hoc networks for subsequent frequency hopping communication suffers from scalability and network dynamics problems. Key-establishment approaches that rely on device proxim-

ity [19] [15] [10] [22] [9] can be used in this context, but require the nodes to be physically close to each other and to use communication channels that are not being jammed (e.g., infrared, wire, or visual). Unlike these approaches, the proposed UFH scheme enables key establishment over longer ranges using radio communication channels.

In a way, bootstrapping coordinated frequency hopping starting from uncoordinated hopping resembles the process of privacy amplification [5], where secret shared information is distilled from a larger body of shared information that is only partially secret.

To the best of our knowledge, there is no prior work that elaborates on the anti-jamming/key establishment circular dependency problem in the described setting, or that proposes a solution to this problem. Furthermore, we have not been able to identify any solutions prior to our UFH scheme that would enable the transfer of messages of arbitrary length across a jammed channel in the absence of a shared secret.

## 7. Conclusion

In this paper, we elaborated the problem of how two devices that do not share any secrets can establish a shared secret key over a wireless radio channel in the presence of a communication jammer. We addressed the cyclic dependency between anti-jamming spread-spectrum communication and key establishment that is inherent to this problem, and proposed a novel anti-jamming technique called Uncoordinated Frequency Hopping (UFH) as a solution to break this dependency. We further presented a UFH message transfer protocol and illustrated on the example of an authenticated Diffie-Hellman-protocol how it can be used to establish a shared secret key between two parties in the presence of a jammer. We performed a detailed analysis of our UFH scheme and showed its feasibility, both in terms of execution time and resource requirements. Moreover, our analysis showed that, although our UFH scheme has lower communication throughput and incurs higher storage and processing costs, it achieves the same level of anti-jamming protection as (coordinated) frequency hopping. Finally, we provided an example which illustrates that UFH key establishment can be executed using current technology with a running time well below one minute (with 80% jamming probability); we point out that this time is reasonably short, given that with known anti-jamming techniques the devices would not be able to communicate and therefore could not execute a key establishment protocol.

We hope that our work will encourage further research to improve the design of our proposed scheme and to provide an even more efficient and reliable solution under a potentially refined threat model.

## Acknowledgment

The authors would like to thank the anonymous reviewers for their thorough reviews and helpful suggestions. The work presented in this paper was supported (in part) by the Swiss National Science Foundation under Grant 200021-116444.

## References

[1] Specification of the bluetooth system (version 1.2), November 2003.

[2] D. Adamy. *A first course in electronic warfare*. Artech House, 2001.

[3] ANSI. X9.63-2001: Key agreement and key transport using elliptical curve cryptography. Technical report, American National Standards Institute, 2001.

[4] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid. Recommendation for key management. NIST special publication 800-57, revised, 2006.

[5] C. H. Bennett, G. Brassard, C. Crépeau, and U. Maurer. Generalized privacy amplification. *IEEE Transaction on Information Theory*, 41(6):1915–1923, Nov. 1995.

[6] D. Boneh and X. Boyen. Short signatures without random oracles. In *EUROCRYPT 2004, Springer LNCS 3027*.

[7] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[8] C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment*. Springer Verlag, 2003.

[9] C. Gehrmann, C. J. Mitchell, and K. Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1), 2004.

[10] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human-verifiable authentication based on audio. In *Proceedings of the IEEE International Conference on Distributed Computing Systems*, 2006.

[11] C. Karlof, N. Sastry, Y. Li, A. Perrig, and D. Tygar. Distillation codes and applications to DoS resistant multicast authentication. In *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*, 2004.

[12] M. Li, I. Koutsopoulos, and R. Poovendran. Optimal jamming attacks and network defense policies in wireless sensor networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[13] G. Lin and G. Noubir. On link layer denial of service in data wireless lans: Research articles. *Wireless Communications & Mobile Computing*, 5(3), 2005.

[14] M. Luby. LT codes. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, 2002.

[15] J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2005.

[16] R. A. Poisel. *Modern Communications Jamming Principles and Techniques*. Artech House Publishers, 2006.

[17] I. Riedel. Protocols and elliptic curve cryptography on an embedded platform. Master's thesis, Ruhr-Universität Bochum, Germany, 2003.

[18] L. Rizzo. Effective erasure codes for reliable computer communication protocols. *ACM SIGCOMM Comput. Commun. Rev.*, 27(2), 1997.

[19] F. Stajano and R. J. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of the 7th International Workshop on Security Protocols*, pages 172–194, London, UK, 2000. Springer-Verlag.

[20] Symantec. Securing enterprise wireless networks. White Paper, 2003.

[21] M. Čagalj, S. Čapkun, and J.-P. Hubaux. Wormhole-based antijamming techniques in sensor networks. *IEEE Transactions on Mobile Computing*, 2007.

[22] S. Čapkun and M. Čagalj. Integrity regions: authentication through presence in wireless networks. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, pages 1–10, New York, NY, USA, 2006. ACM.

[23] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *IEEE Computer*, 35(10):54–62, Oct. 2002.

[24] W. Xu, W. Trappe, and Y. Zhang. Channel surfing: defending wireless sensor networks from jamming and interference. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2006.

[25] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2005.