Deniable Liaisons

Abhinav Narain, Nick Feamster, Alex C. Snoeren*
Georgia Tech *UC San Diego
{nabhinav,feamster}@cc.gatech.edu, snoeren@cs.ucsd.edu

Abstract

People sometimes need to communicate directly with one another while concealing the communication itself. Existing systems can allow users to achieve this level of privacy in the wide-area Internet, but parties who are in close proximity (*e.g.*, a public square or coffee shop) may want a lightweight communications channel with similar properties. Today, covert exchanges in local settings typically require the exchange of physical media or involve other forms of direct communication (*e.g.*, conversations, blind drops); most, if not all, of these exchanges are observable: in other words, even if the message exchanges are confidential, they are not covert or deniable.

We construct a local communications channel that is unobservable to everyone except the parties exchanging messages. To do so, we take advantage of the ubiquitous phenomenon of packet corruption in wireless networks, which provide deniable cover for message exchange between parties within radio range. The communicating parties use a shared secret to differentiate truly corrupted frames from those that hide messages; to other parties, messages appear as corrupted wireless frames. We tackle the challenge of designing the observable corruption patterns to ensure that an observer can neither link sender and receiver of a hidden message (unlinkability), nor determine so much as the existence of any hidden message (deniability). We present the design and implementation of a prototype system that achieves these properties using off-the-shelf 802.11 hardware, evaluate its performance, and assess its resilience to various attacks.

Categories and Subject Descriptors: C.2.1 [Computer-Communication Networks] *Network Architecture and Design*: Network Communications, Wireless Communication

General Terms: Algorithms, Design, Experimentation, Security

Keywords: censorship; wireless; covert channels

1 Introduction

The need for private communications is perhaps greater than ever before. People have long needed to keep the communications among themselves private, but, increasingly, they may want to conceal not only the messages that they exchange, but also with whom they are communicating—or even the fact that they are communicating at all. This latter type of communication is said to be not only confidential and anonymous, but also deniable, in the sense that despite exchanging messages, participants can plausibly deny that any such exchanges ever took place.

In this paper, we consider scenarios where people congregate in common public spaces and want to communicate with others in that same space, yet wish to keep their communications both confidential and deniable. We call such a message exchange a *deniable liaison.* Moreover, it may be the case that one or more parties to the communication wish to remain anonymous. Consider, for example, a covert message exchange between a spy and her handler in a coffee shop, a whistleblower in an office environment, or a group of activists who wish to covertly organize a public protest. These scenarios require local communication that is confidential, anonymous, and deniable.

Covert agents have long employed a wide range of techniques in these scenarios, but they tend to be either limited in bandwidth (*e.g.*, a necessarily brief, clandestine conversation) or interactivity (*e.g.*, a "dead drop" of a physical message or storage device). Indeed, any real-world interaction bears some risk of observation, and most are not readily applicable to broadcast scenarios. Hence, a variety of anonymous electronic communications systems have emerged to provide important—and often widely used—communications channels, but most focus primarily on wide-area communications (*e.g.*, Tor [7], which supports communications between Internet-connected end hosts that are often separated by great distances) where deniability can sometimes be provided by hiding in a very large crowd of Internet citizens. In the circumstances we consider, a wide-area anonymous communication system not only introduces unnecessary complexity and latency, but exposes the parties to additional risk by requiring them to send their messages over the wide-area Internet. We argue that such schemes are not always available due to the widespread Internet arms race of blocking such services and instead a powerful local scheme can be used which leverages the broadcast nature of wireless communication. We argue that these settings call instead for an anonymous, confidential, deniable communications system for the local area that takes advantage of communications devices that users already own (*e.g.*, laptops, smartphones, tablets), without requiring that covert messages traverse the wide-area Internet.

We introduce DenaLi, a lightweight communications system to support deniable liaisons. DenaLi makes it possible for parties to exchange messages with one another in a local setting, without ever exposing with whom they are communicating, or even the fact that they communicated with a local party at all. Our system takes advantage of the ubiquitous deployment of 802.11 wireless communications networks and, in particular, the pervasive nature of corrupted frames on these networks. Frame corruption is a common phenomenon that inevitably results from a variety of factors, ranging from colliding transmissions to a noisy communications medium. Traditionally, corrupted frames are viewed as a source of inefficiency,

CCS'14, November 3–7, 2014, Scottsdale, AZ, USA.

http://dx.doi.org/10.1145/2660267.2660340

as they require the sender to retransmit the original frame; yet, in our case, they provide an opportunity to hide communications. DenaLi creates spurious corrupt frames by injecting covert messages into frames carrying cover traffic directed toward innocuous destinations. Since these frames are indeed corrupt, they will not be forwarded by the access point to their apparent destination. Instead, other nodes in the WiFi network that overhear the frame and posses the appropriate secret key can extract and decrypt the injected payload.

DenaLi is conceptually simple, and achieving anonymity and confidentiality is easy enough—any reasonable encryption technique will suffice. The challenges entail designing the communications channel so that the resulting stream of corrupted frames is deniable, which requires both understanding (and modeling) the properties of bit errors in an 802.11 wireless communications channel and appropriately modeling the attacker. To do so, we build on previous work that studies bit-error characteristics in the wireless medium, and perform our own measurements to understand these error characteristics in various settings and for different encodings. We develop a modified 802.11 wireless driver that modulates the covert message over a stream of cover traffic in such a way that the resulting sequence of corrupted frames mimics the existing pattern of corruption in the wireless channel. DenaLi traffic matches naturally occurring wireless corruption both in terms of the frequency of corrupted frames and the bit positions within the frames that are corrupted.

DenaLi provides deniability in a setting where an adversary can observe wireless communications in the local area, but cannot get very close to the suspected sender. An adversary who observes transmissions sufficiently close to the sender could infer the presence of a hidden message channel due to the (relatively) high level of packet corruption near the point of transmission. We envision that in typical cases an adversary would not be targeting an individual sender but would rather only be in a position to monitor a group of users (*e.g.*, in the midst of a larger group, perhaps close to the access point). In these cases, we demonstrate through empirical measurements that distinguishing DenaLi transmissions from naturally occurring corrupted wireless frames can be made arbitrarily difficult for message rates that can easily support the exchange of short covert messages. We show through extensive controlled experiments with real wireless chipsets that when we closely match the frame error rate and bit error distributions of the existing wireless channel, DenaLi achieves a bit error distribution pattern that is indistinguishable from naturally occurring errors. To achieve this level of deniability, throughput is quite low (sufficient for exchanging only small messages or "tweets"), but the sender can, of course, accept less deniability in exchange for higher throughput, a tradeoff that we explore in our evaluation. Traffic that the user is already sending as part of normal communication can provide the necessary cover traffic, which means that DenaLi does not need to create additional cover traffic but can rather hide its messages in the user's existing traffic.

Our work presents several contributions. First, we recognize that the increasing need for anonymous, deniable communications in settings where parties are physically close to one another calls for a new class of communications tools. Second, we observe that in these settings, the ubiquity of other WiFi communication (and the corresponding wireless frame corruption) can serve as useful cover to conceal communications. Third, we define the notion of deniability in this context and design a modulation scheme that achieves deniability by matching the corruption properties of the deniable messages to that of the cover traffic. Finally, we implement and evaluate a prototype system based on this design.

The rest of the paper proceeds as follows. Section 2 surveys related work in anonymous communication, detection of covert channels, and wireless errors and corruption. Section 3 defines our expected usage scenario and outlines our basic approach, threat model, and design goals. Section 4 describes the design of the DenaLi communication channel in detail. Section 5 describes our prototype implementations and explains the changes we made to the wireless driver to enable DenaLi. We evaluate DenaLi in Section 6, discuss limitations and future work in Section 7, and conclude in Section 8.

2 Related Work

We first survey related work on related anonymous and deniable communications systems. We then discuss various studies of wireless interference and channel properties to design DenaLi.

2.1 Anonymous Communications

DenaLi's design is inspired by Rivest's proposal for chaffing and winnowing, whereby a sender disguises the real message intended for the recipient by including additional "chaff" on the same channel [22]. With knowledge of a shared secret, the recipient can identify and discard the chaff, leaving only the message in question. Unlike Rivest, however, we further encrypt the message to make it easier to efficiently inject into the chaff without disturbing the statistical properties of the aggregate.

DenaLi is the first system to provide a point-to-point deniable communication channel in a WiFi network using commodity hardware. Previous work has sketched systems that use corrupted wireless frames to create a covert channel over 802.11 frames [19, 25] but no previous work has moved beyond paper designs. Calhoun *et al.* designed and simulated a covert channel based upon varying the link rate [2]. This work is purely simulation-based and develops neither a working prototype nor a communication protocol for exchanging messages. None of the previous work analyzes deniability in the presence of an adversary that can monitor channel quality.

Many existing anonymous communications systems aim to provide various levels of anonymity in the wide area. One of the most widely used anonymous communications systems is Tor [7], which allows communicating parties to establish anonymous communications channels via a layered encryption technique called onion routing [10]. Users of Tor establish circuits to communicate with each other anonymously in the wide-area. Tor provides anonymity but not deniability, in the sense that users of Tor can conceal who they are talking to, but not the fact that they are communicating using Tor (in fact, Tor is blocked in many countries outright). DenaLi's focus is different than Tor's: it aims to enable anonymous and deniable communication in settings where the communicating parties are physically close to one another.

DenaLi bears similarity to other censorship circumvention systems that aim to achieve deniability and covertness in addition to confidentiality and anonymity. Two such systems that operate from end systems are Infranet [9] and Collage [1]. These systems allow participants to establish communications under the cover of innocuous Web traffic: the censor only sees Web requests that are statistically indistinguishable from normal user behavior, thus providing the user with an important degree of deniability, in addition to confidentiality. Other recent systems such as Telex [28], Cirripede [13], and Decoy Routing [15] aim to achieve similar levels of deniability by deploying infrastructure in the core of the network

rather than at end systems. Briar [23] provides a secure, point-to-point anonymous encrypted communications channel between users' devices; like DenaLi, Briar enables point-to-point communication, but Briar does not provide deniability.

2.2 Detection

Others point out that covert channels must be detectable under some adversarial model, because the message channel introduces some statistical deviation to the underlying natural distribution. (In other words, there is no such thing as "perfect steganography" that is analogous to perfect security for encryption in practice.) Provos *et al.* demonstrate that statistical steganalysis is possible on JPEG images based on careful analysis of the entropy of the resulting compressed images [21]. Previous work evaluates two methods for detecting the anomalies in traffic distributions generated by covert channels (such as the deviation induced by DenaLi) by measuring the variation in the number of packets with invalid checksum value [19] (this analysis is based on simulation.) As in any mechanism that perturbs the underlying natural distribution of the cover traffic, DenaLi could be detected if the adversary both can observe the distribution generated by DenaLi, and is in a position to measure the deviation. We explain in Section 3 why we believe this to be generally difficult in the case of DenaLi's most likely deployment scenarios.

DenaLi provides a different type of deniability than deniable encryption [5], and therefore presents a different detection challenge. DenaLi aims to conceal the existence of communication, whereas deniable encryption explicitly admits he presence of communication: the adversary is provided with an alternate key to decode the cipher text to convert it into a different plaintext than the actual one.

2.3 Wireless Errors and Corruption

One of DenaLi's goals is to transmit corrupted wireless frames whose statistical properties match those that would occur in a normal wireless communications channel. The properties of a wireless medium can be both highly variable and difficult to model. In particular, the channel properties depend on a host of complex and time-varying factors ranging from radios to bitrate adaptation schemes. Our goal is not to develop a model for wireless channels. Rather, if the channel's error properties can be observed, we can develop a communications channel using corrupted wireless frames as cover traffic in a way that mimics the properties of the errors that naturally occur.

Since errors have plagued wireless communication since its inception, there are a number of studies that focus on the nature of such errors in wireless frames. The Maranello study [12] measures erroneous frames in an 802.11 LAN; their findings corroborate earlier work [14] that observes that bit errors in wireless frames occur in clusters, as the error process is not memoryless. Han *et al.* also observe that the probability of error increases with the bit position in the frame (*i.e.*, bits further into the frame are more likely to be corrupted) [11]. DenaLi applies these observations by injecting bit errors into frames in groups—as opposed to one bit at a time —at least 100 bytes into any frame.

3 Problem Description

We now explore the scenario where we believe that DenaLi is most likely to be used and the threat model, in terms of the capabilities of a typical adversary who might try to discover or thwart communication with DenaLi.

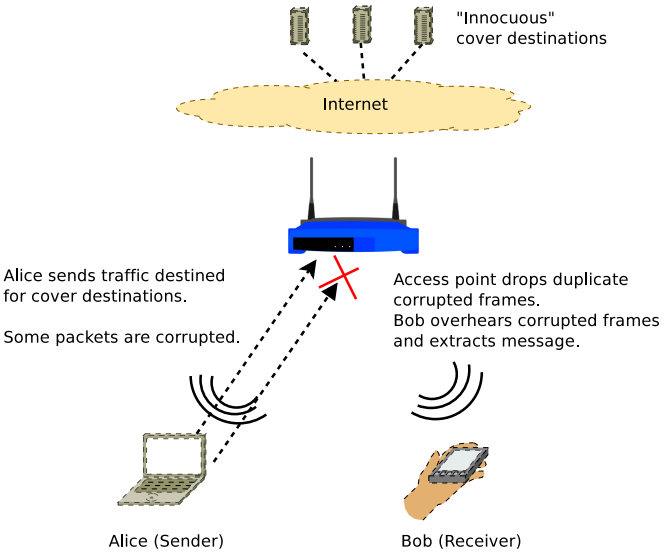


Figure 1: *Basic communication setup.*

3.1 Usage Scenario and Basic Approach

DenaLi is designed for settings where the communicating parties are within wireless range of one another and, hence, can hear one another's wireless transmissions to a local access point. We further presume that a DenaLi sender has some number of pre-existing connections to innocuous destinations on the Internet which will provide cover traffic for our covert communication channel. Figure 1 shows such a basic setup. An adversary may be positioned anywhere in the wireless network and is able to eavesdrop on any transmissions by the participants. DenaLi does not employ or require link-layer encryption schemes (like WEP or WPA) for its confidentiality guarantees.

In this scenario, the sender, Alice, sends traffic to her usual set of wide-area Internet destinations via the access point. Due to the nature of the wireless channel, some frames may experience corruption, and the access point will thus discard those frames. Alice will subsequently retransmit these frames until they are successfully received by the AP and forwarded on. But, if Alice and Bob share a secret, Alice can inject additional, deliberately corrupt frames, such that the frames corrupted by the wireless channel serve as chaff to conceal the fact that some of the corrupt frames contain a hidden message. If Alice and Bob share a secret, Bob can determine which corrupted frames are chaff and can retain only those corrupted frames that contain the hidden message.

Corrupt frames naturally result from various wireless effects, including low signal-to-noise ratio (SNR), broadband interference, hidden terminals, and multi-path fading, which depend on the relative position of the transmitting device and nearby wireless devices, materials of nearby objects, and other unknown factors. Because the causes of corruption are diverse and time-varying, detecting the hidden messages with certainty requires either knowledge of the secret, or the ability to monitor frame corruption rates and compare the measured distribution to the corruption rates that would be expected as a function of both space and time.

To construct a profile that closely matches that of a normal wireless channel, we exploit two important observations about the corruption of packets in a broadcast medium, particularly the 802.11 protocol. First, packet errors in packets occur in chunks of bytes [12], not as individual bits; most of the chunks of errors are about 400 bits, and occurrence of larger chunks of errors is not very usual. This phenomenon might occur as a result of interference, or the loss of synchronization. The second observation is that the bit errors inside wireless frames have specific patterns [11]; for example, bits that are farther from the start of the frame will experience an increasing probability of corruption.

3.2 Threat Model

The adversary's primary goal is to detect the presence of hidden communication on a shared wireless medium. If the adversary is able to further determine which transmitted frames contain hidden communication, it may be able to use existing techniques to determine the identity of the sender [8]. The adversary's main capability is to listen to wireless frames within its radio range.

We assume that the adversary has finite computation resources and a finite number of nodes that it can use to monitor the wireless channel. In our prototype, we assume the adversary has only one node with which it can monitor and has knowledge of at least one party which may be communicating using DenaLi. In a practical scenario, the adversary might know the identity of the sender but not his MAC address. He would still have to scan the channel and apply techniques to identify the sender's device, which might be hard in dense public places where signal strength of device varies considerably due to the commotion [27]. If the adversary has previous knowledge about which parties may be using DenaLi to communicate, it could position its radio(s) close to one of the senders and attempt to determine if the sender's wireless interface was sending corrupted frames at a rate that exceeds the typical rate at which a wireless radio emits corrupted frames. We assume that the adversary remains at a sufficient distance that it cannot conclusively determine that some frames are already corrupt when they are transmitted by the sender; rather, it can only monitor the frame corruption rate. As long as the adversary is sufficiently far away, the sender can always make his channel worse by staying far away from the public access point, thereby legitimately retransmitting at a higher rate than normal.

Even without knowledge of communicating parties, a stronger adversary can monitor and collect wireless transmissions from multiple independent locations in the network and run statistical analysis on the collection of captured traffic. In these cases, the adversary might be able to determine that the profile of bit-error corruption for certain nodes in the network does not match the corruption profile for other senders, or that the frame corruption profile does not change with increasing distance from the sender as one might expect. Such an adversary might be able to perform an analysis of error patterns with a tool such as Jigsaw [3], but even with the benefit of multiple observation points, if the distributions are matched appropriately, the perturbations that DenaLi introduces should still provide deniability for senders. Moreover, a global adversary, *i.e.*, one that can monitor at multiple locations in the wireless network—but not the sender or receiver—does not necessarily have a better chance at detecting the presence of hidden communication than a local adversary who only has one monitoring point. Although the ability to observe transmissions at multiple locations provides the opportunity to observe corruption patterns of the same packet at multiple locations, these observations still do not allow the adversary to ascertain what bit errors would look like at the exact sender and receiver locations [17]. Previous work suggests that bit error patterns within corrupted frames will differ depending on the adversary's location [18].

In our empirical evaluations and security analysis (Section 6), we assume an adversary who can observe all the corrupted frames from a single location in the network. We note that even if an adversary targets a particular sender (e.g., based on previous knowledge), the sender can always move away from a suspected adversary or maintain enough mobility to reduce the likelihood of being monitored at close range. (Indeed, previous work shows that simply rotating the communication device can dramatically impact the channel quality [27].) Therefore, we believe that it is extremely unlikely that an adversary could successfully target a sender *and* successfully monitor the sender at close enough range for an extended period of time without tipping off the sender.

3.3 Design Goals

We aim to develop a covert channel with a variety of properties, in addition to the standard properties of *confidentiality* and *covertness*. *Undetectability* says that the adversary cannot detect the presence of any messages. *Deniability* is a slightly weaker property that says that even if the channel is detectable, the adversary cannot determine with non-negligible probability that a particular user or group of users is exchanging messages. *Unlinkability* says that an adversary may be able to detect the presence of communications, but cannot link the sender of a message with its receiver. *Robustness* says that the adversary should not be able to disrupt the channel.

DenaLi technically does not achieve strict undetectability, since the process of sending a message does perturb the wireless channel from its original state. We design the resulting bit error profile to be statistically similar to a normal profile, however, making it difficult for an adversary to determine with certainty that the channel has been perturbed. Because frame corruption is a random process that is itself based on a non-stationary distribution (*e.g.*, it is affected by a variety of factors, ranging from the presence of other senders, to changes in obstructions such as people and doors, to the user's wireless radio, to physical properties of the air), we can perturb the corruption profile of the channel without allowing the adversary to determine that a sender is definitely sending a hidden message. In this way, we achieve deniability.

Independently, DenaLi achieves unlinkability because even if the adversary could detect the presence of additional corrupted frames, without having the key that Alice and Bob share, the adversary cannot determine that Bob is the intended recipient of the additional corrupted frames. In fact, by having multiple participants share a group key, DenaLi can be used to surreptitiously broadcast a hidden message.

Finally, DenaLi achieves practical robustness by virtue of the fact that an adversary cannot easily selectively disrupt the communication of the wireless frames containing the hidden message. An adversary could jam the entire wireless channel, but doing so would disrupt communication for legitimate traffic as well.

DenaLi does not rely on 802.11 encryption standards such as WEP and WPA to achieve confidentiality, as we assume that many adversaries may be powerful enough to either (1) join the channel with a known WEP or WPA2 keys (*e.g.*, in the case where the adversary is the network administrator, such as in a public square or a coffee shop); or (2) break the WEP encryption or WPA2 encryption

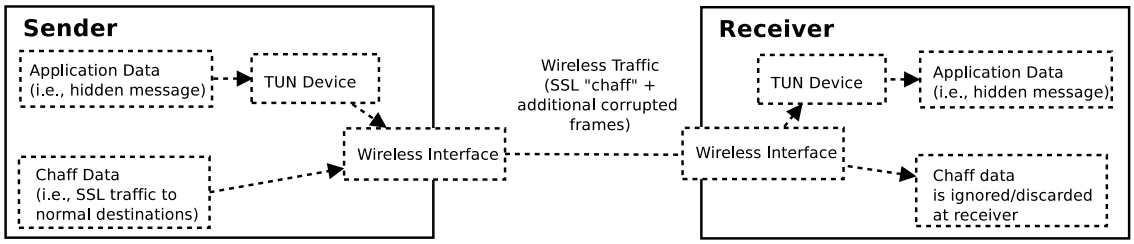


Figure 2: *Injection of additional corrupted frames via a virutal network interface (implemented as a Linux TUN device).*

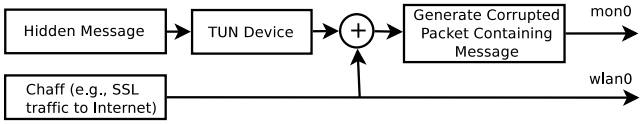


Figure 3: *Process of injecting corrupted frames at the sender; the receiver performs the reverse of this process.*

using known techniques. Instead, DenaLi provides confidentiality by encrypting the message contents before injecting them into the corrupted frames.

4 Communications Channel

This section describes the DenaLi design in more detail. The basic approach is for the sender to inject corrupted frames into an existing encrypted application traffic stream (the chaff), so that in the air, the adversary sees a single stream of encrypted application traffic with non-corrupted and corrupted frames. The goal is to make what is seen on the air appear as a plausible sequence of frames to the purported destination to anyone observing the traffic pattern. To do so, the sender occasionally duplicates existing frames and corrupts them by injecting a portion of the message to be communicated. The sender and receiver must also develop a common means to identify which corrupted frames contain hidden messages, and where (*i.e.*, at what byte offset) within a corrupted frame the hidden message lies.

4.1 Basic Mechanism: Frame Injection

DenaLi constructs corrupted frames and hides the corrupted frames among a larger stream of frames being transmitted to the access point. Some of these frames (perhaps including some of DenaLi's constructed frames) will be corrupted by the wireless channel. In order to make it more difficult to determine which corrupted frames contain embedded messages, DenaLi transmits hidden messages only in frames that otherwise are part of encrypted SSL connections (*e.g.*, to popular websites like Gmail). We chose to use SSL connections as the basis for DenaLi's cover traffic because the encrypted payload of these frames acts as a one-time pad into which we can embed similarly encrypted messages without obviously disturbing their statistical properties.

An SSL frame will necessarily have a TCP header, which DenaLi uses to compute the offset into the frame at which to place the embedded message. Because bits that are located further into a frame (*i.e.*, with a greater offset) have a greater chance of experiencing corruption [11], DenaLi skews the probability distributions on injecting message blocks to favor corrupting bits farther into the frame. Obviously, the message must be (substantially) smaller than the frame into which it is being injected. Our implementation exports a virtual network interface with a small MTU, which ensures that the covert channel is automatically broken into smaller message blocks. Figure 2 illustrates the communication tunnel between the sender and receiver, including how the hidden message is combined with chaff before being transmitted over the air; the receiver hears all of the wireless traffic but can discard the chaff before passing the message to the receiver.

Figure 3 shows the construction of the combined packet stream in more detail. The hidden message is passed through the virtual network interface (a Linux TUN device), whereupon it is combined with a copy of an existing frame from the chaff via bit-error injection. The corrupted frame is then transmitted very close in time to the unmodified chaff frame. To decode the hidden message, the receiver performs the reverse of this process. Ideally, the entire stream would be transmitted via the same outgoing interface, but limitations of current wireless chipsets prevented us from implementing the transmitter in this fashion; Section 5 discusses these limitations in more detail, and Section 7 explains how we conceal the presence of two separate transmitting interfaces.

4.2 Communication Protocol

Figure 4 shows the steps that are involved in exchanging messages in a two-party message exchange. We now explain these steps in detail.

Establishing a shared session key The sender and receiver use the DenaLi channel to establish a shared session key in a manner that is analogous to how session keys are established in many protocols. In case of DenaLi, the colluding parties should be aware of that they are in proximity of each other and then instantiate the key exchange process. The sender generates a session key and encrypts the key with the receiver's public key. It then sends the resulting ciphertext over the DenaLi channel, taking the resulting ciphertext and embedding it as corrupted bits in an outgoing sequence of frames. The receiver decodes the message from the corrupted frames to retrieve the session key. The session key is transmitted on the DenaLi channel just as any other message would be, except that the initial transmission and encoding is based on the receiver's public key, instead of the session key itself. All transmissions on the DenaLi channel involve a process of the sender encoding the hidden message and the receiver decoding it upon receipt, as described below.

Encoding and transmitting First, the sender obtains a cover frame by duplicating a frame that is about to go out of its wireless interface as part of an existing connection. It then corrupts this duplicate by injecting data from the covert channel. Before injecting the hidden message into a corrupted frame, the sender: (1) encrypts the hidden message with the shared session key (or, in the case of the initial key exchange, the receiver's public key) using CBC-AES 256-bit symmetric key encryption; (2) computes the offset into the

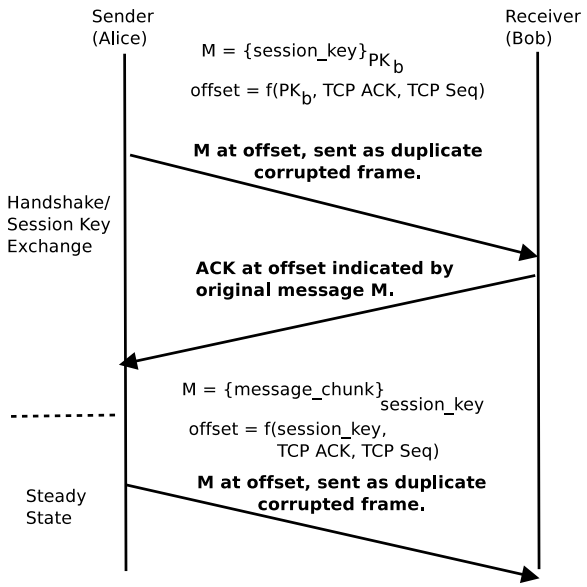


Figure 4: *Steps involved in exchanging messages using corrupted frames.*

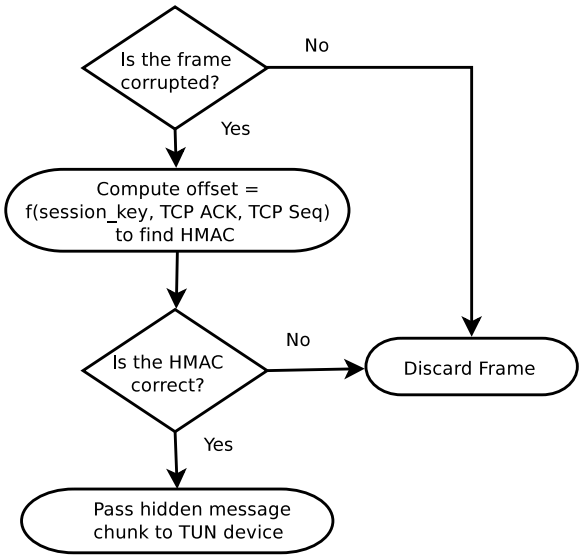


Figure 5: *Checking the integrity of received hidden messages.*

frame where the message should be inserted; and (3) computes an HMAC over the message ciphertext. The sender then inserts bits corresponding to the hidden message length, the HMAC, and the hidden message itself as a block into the corrupted frame. We describe the process of computing the frame offset and the HMAC below.

In addition to the session key, the sender uses the TCP sequence number and acknowledgment number as salts to compute the frame offset for the hidden message. Doing so helps randomize the offset, so that the inserted bits are not always in the same location in the corrupted frame; randomizing the offset makes it difficult for an adversary who is eavesdropping to ascertain the presence of a hidden message, since the location of the corrupted bits that contain the hidden message will be different for each packet. We considered using a pseudo-random number generator with an initial seed to allow the sender and receiver to compute this offset; the problem in doing so is that if any corrupted frame containing a hidden message is lost, reordered, or itself corrupted, the receiver and sender will lose synchronization. Instead, DenaLi uses the output of a public cryptographic hash function that uses the TCP sequence number, acknowledgment number, and shared secret (or, in the case of the initial key exchange, the receiver's public key) as the input for computing the offset. Thus, all of the information that the receiver needs to extract the hidden message from the frame is present in the frame itself. Unless the adversary has the shared secret, it cannot determine the offset of the artificially corrupted burst sequence.

Because the injected frame is corrupt (*i.e.*, its layer-two checksum is invalid), the receiver no longer has an inherent way to determine the integrity of the frame—or, more specifically, the embedded DenaLi message within—it receives. In lieu of the (now corrupted) frame checksum, a DenaLi sender also includes an HMAC computed over the hidden message contents that is keyed on the session key, the TCP sequence number, and the acknowledgment. The message's HMAC is prepended to the hidden message before the resulting bits are inserted into the frame.

The astute reader might observe two nuances about the way that the sender embeds the message into a corrupted frame. First, the message length is included "in the clear". Including the message length in the clear is necessary because the number of bits corresponding to the hidden message varies (both by design to make detection more difficult, and as a natural result of the original message sizes). Because both the value of the message length and the offset within the frame where the bits indicating the message length vary per-frame, recognizing a pattern would be difficult. A sender could, of course, introduce more entropy into the message length value by randomizing the block size for each block that it injects into a corrupted frame, making it essentially impossible to identify the presence of the message length value, at the expense of channel throughput.

Second, all of the corrupted bits are injected into the frame as a single block rather than interspersed at random bit locations throughput the packet. Previous work has established that wireless bit errors tend to occur as corrupted blocks [12], not as individual corrupted bits. Additionally, because the DenaLi sender injects ciphertext into other ciphertext (*i.e.*, the SSL stream that serves as the chaff), interspersing the block throughput the packet does not increase covertness: Because both the hidden message and the chaff are encrypted, the adversary can see that the frame is corrupted, but has no straightforward way of determining the bit positions corresponding to the corruption, unless he has the corresponding uncorrupted version of the frame. Injecting an encrypted message into SSL payload makes the likelihood of every bit to be corrupted to be ≈ 0.5.

Receiving and decoding To receive the hidden message, the receiver polls the wireless medium for all the corrupted frames and attempts to decode and decrypt the bits in each corrupted frame that are located at the appropriate offset, which is computed as a function of both the session key and the TCP sequence number and acknowledgment numbers in the packet header. The receiver can apply the same function to determine the appropriate offset of the message in the corrupted frame to extract the ciphertext and decrypt it to recover the session key, which will be used to encrypt future messages and as an input for computing the frame offsets.

Upon hearing a corrupted frame in the wireless medium, the receiver extracts the grain from the chaff by computing the offset where the hidden message is expected (as a function of the key and the TCP sequence and acknowledgment numbers contained in the frame) on every corrupted frame, extracting the bits that should

correspond to the hidden message, computing the HMAC on the decoded and decrypted message, and comparing it with the HMAC value present in the packet. The receiver computes the HMAC of the decoded message and compares it to the value of HMAC included in the packet, which (as mentioned above) is prepended to the transmitted message before being injected into the frame. If the HMAC is correct, the receiver then proceeds to decode and decrypt the hidden message. (It is extremely unlikely that the bits of the secret message and the HMAC will be corrupted simultaneously in such a way that the HMAC calculated over the corrupted frame will be the same as the corrupted value of the included HMAC.) Figure 5 illustrates this process.

5 Prototype Implementation

In this section, we describe a prototype implementation [6] of DenaLi using off-the-shelf wireless chipsets based on the design detailed in Section 4.

5.1 The TUN Interface

In the interest of simplicity, our prototype implementation of DenaLi provides a TUN interface that allows applications to use the covert channel just as any other network interface. It is a virtual interface in Linux, implemented as a TUNnel device, to exchange packets with user space. A user can determine how to design and implement applications that communicate over the channel, or just use existing ones.

Once a packet is transmitted on the TUN interface, DenaLi encrypts it (including the headers and checksums), calculates the HMAC of the encrypted message, computes the resulting message length, and concatenates them to arrive at the bit sequence that is ultimately inserted into a corrupt wireless frame. We compute the HMAC using SHA-256.

5.2 Dual Wireless Interfaces

Most existing wireless chipsets calculate the layer-two checksum, also known as the frame check sequence (FCS), in hardware. Hence, even the "corrupted" frames created by injecting the encrypted payload would normally be sent out with a correct FCS, meaning the destination of the encapsulating chaff frame (*i.e.*, the access point) would receive the packet and attempt to process it. While the IP checksum would still likely be incorrect, it is far less common for an IP checksum to be invalid on purportedly correctly received frames, destroying DenaLi's deniability.

Hence, we must ensure that the corrupted frame is transmitted with an invalid FCS. Unfortunately, the current architectures of most wireless chipsets do not expose an interface to manipulate the FCS. Instead, our prototype uses a wireless interface card with the Atheros AR9485 chipset, which exports a register that disables the calculation of the FCS (we are unaware of other vendors that provide this feature). Atheros *ath9k* and *ath5k* series of chipsets provide this feature available commercially for Linux [16]. The register setting is not selective, however: if enabled, all packets are transmitted without a proper FCS. Hence, in order to transmit the chaff traffic, our prototype employs two wireless interfaces: one to transmit the chaff SSL traffic, and one to transmit the additional corrupted frames that contain the hidden message with a corrupted FCS. We are using two wireless cards to facilitate usability of the prototype, as software defined radios are bulky and hard to carry for general purpose use by non-technical person. We use identical Acer Aspire One laptops with Intel Celeron processor running at 800 MHz, with Linux 3.2.0 and stable compat-wireless networking stack.

5.3 Driver Modifications and SoftMAC

Each wireless frame passes through multiple stages before it is transmitted, many of which occur in hardware by default (and, hence, are inherently challenging to modify), as shown in Figure 6. The specific stages depend on the architecture of the particular wireless chipset in use, although we provide a rough general outline that many chipsets follow. First, an application provides the payload to the operating system, which in turn copies the data to driver memory after adding 802.11 MAC header. The driver then encrypts the packet and transmits it; the encryption keys are retained in software, but the encryption process itself occurs in hardware. The transmission control unit manages the fine-grained timing of 802.11, including generating the frame checksum right before transmitting the frame.

Our DenaLi prototype makes two changes to the default processing pipeline: it (1) disables the FCS checksum; and (2) disables the retransmission of these frames, which obviously will never generate link-layer acknowledgments. Figure 6 illustrates where we made these modifications in the NIC processing pipeline.

To modify the behavior of the wireless interface, we use the SoftMAC 802.11 wireless MAC implementation [20], which offloads many functions of the wireless driver to the kernel subsystem, thus forming a clean interface with various vender drivers and allowing us to modify various parts of the process.

6 Security and Performance

In this section, we evaluate the security of DenaLi relative to the performance that it achieves. As discussed in Section 3, our primary goals for security are deniability and confidentiality, where deniability says that the resulting traffic is statistically indistinguishable from network traffic that does not contain any hidden message. We begin with a discussion of the characteristics of the resulting wireless traffic that should appear statistically indistinguishable to an adversary. We then formalize our definition of security in terms of the indistinguishability of the resulting DenaLi traffic from ordinary wireless frame corruption.

We conduct real-world experiments with our prototype implementation to explore the tradeoffs between deniability and throughput over the DenaLi channel. Across all of our experiments, our prototype consumes an average of 2% and maximum of 5% CPU time at both transmitter and receiver while it injects or decodes corrupted wireless frames. We confirm that no packets are dropped by kernel or socket buffers despite using the pcap library for packet reception and injection.

6.1 Traffic Characteristics

Detecting DenaLi communication requires the adversary to make observations about perturbations to the natural error patterns at one of two levels: packet errors in the medium, or patterns of bit errors within individual frames.

The *packet error rate* is the fraction of transmitted frames in the wireless medium that are corrupt. This rate depends on the characteristics and nature of the environment where the colluding parties

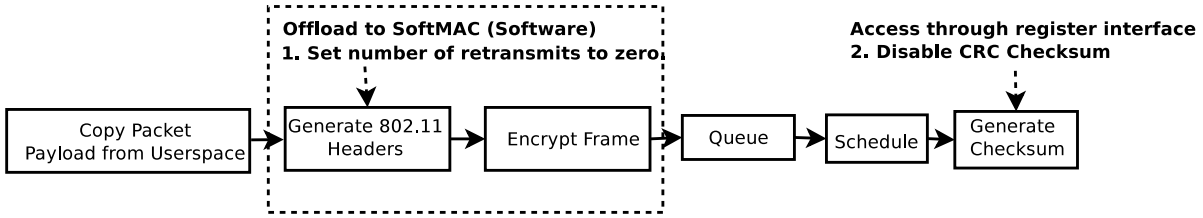


Figure 6: *Processing of an 802.11 wireless frame at the host, and the two modifications that we make to enable DenaLi: (1) setting the number of retransmissions to zero through the SoftMAC implementation; (2) disabling the frame checksum computation to allow the interface to transmit the corrupted frame.*

(and the adversary) are located. Although the instantaneous frame error rate cannot be modeled precisely because the type and frequency of events that cause interference or frame loss are inherently random, we can calculate the rate of corruption of the frames in a live capture of a collected packet trace and attempt to mimic that distribution. DenaLi users maintain statistics regarding the packet error rate of normal frames so that they can inject corrupted frames in a way that mimics the naturally occurring packet corruption in the current environment. In channels that are subject to corruption rates that are higher or more variable, DenaLi participants can inject hidden messages with higher frequency. We explore the relationship between the amount of noise in the channel and the throughput that we can achieve later in Section 6.3.

The *bit error distribution* is the distribution of the bit errors in specific positions *within a corrupted frame*. An adversary who captures the frames may analyze the corrupted frames to compare the error patterns. We modify the contents of the intentionally corrupted frame in such a manner that it is difficult to differentiate actually corrupted bits from the crafted corrupted frame. Our goal is to inject bit errors into packets in such a way that the resulting distribution of bit errors resembles a bit-error pattern that would result from the corruption of one or more symbols in an encoded wireless packet. The exact bit-error pattern is difficult to model because these patterns depend on how the sender modulates packets. In lieu of conducting additional experiments on bit error rates ourselves, we follow the assumptions from the Maranello study [12], which suggests that the bit errors in a frame occur in chunks, due to the loss of synchronization between the sender and receiver or the bursty nature of interference in the wireless channel, unlike uniform corruption of bits in the whole frame. In our evaluation, we use DenaLi to corrupt specific bit error patterns in such a way that mimics these observed distributions. We also note that the farther that the sender is from the adversary, the more likely that the adversary will observe naturally occurring frame corruption, which should make it more difficult to distinguish naturally occurring corruption from artificial corruption.

6.2 Security Goal

The security of DenaLi requires that: (1) sending a hidden message using DenaLi creates a perturbation of the wireless channel's packet error rate and bit error distribution that is statistically indistinguishable from if a DenaLi message had not been sent (*deniability*); (2) the adversary cannot recover the messages (*confidentiality*). As the confidentiality of DenaLi relies on the strength of existing encryption technologies, we focus on defining and evaluating DenaLi's deniability properties.

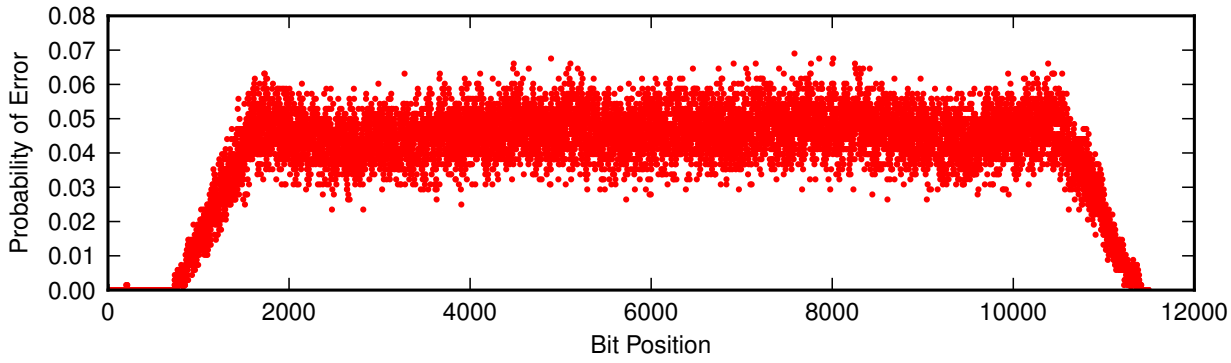
Consider an adversary who observes the properties of the wireless channel from a particular location. The adversary can empirically measure both the packet error rate for a sequence of frames, and the bit error distributions within each corrupted frame. Suppose that the adversary has two packet traces P and P', where P is a packet trace without DenaLi communication and P' is a trace with DenaLi communication. *Deniability* says that the adversary cannot determine which trace has DenaLi communication with probability greater than $1/2+\varepsilon$. If the adversary can correctly detect the presence of a covert channel with probability greater than $1/2+\varepsilon$, then the adversary wins.

Similarly, suppose also that the adversary runs a maximum likelihood detector based on observations of bit error distributions in corrupted frames to detect the presence of a DenaLi channel based on deviations in the respective distributions. According to the definition of deniability above, if ε is zero, the best threshold that an adversary could design would be unable to distinguish the two distributions of bit error patterns drawn from P and P'. The ε parameter measures the extent to which the two distributions do not overlap. We quantify the degree to which the two distributions do not overlap (which corresponds to the probability that the adversary succeeds) using the Pearson correlation coefficient between the two distributions [24]. ε is simply half times one minus the correlation coefficient. Formally, we denote the bit error distribution from packet trace P' as $f'(x)$, where x is the bit position in the packet; similarly, the normal bit error distribution from packet trace P is $f(x)$. For each of the distributions that are parameterized by frame error rate and bytes injected per frame, we compare the two distributions as follows:

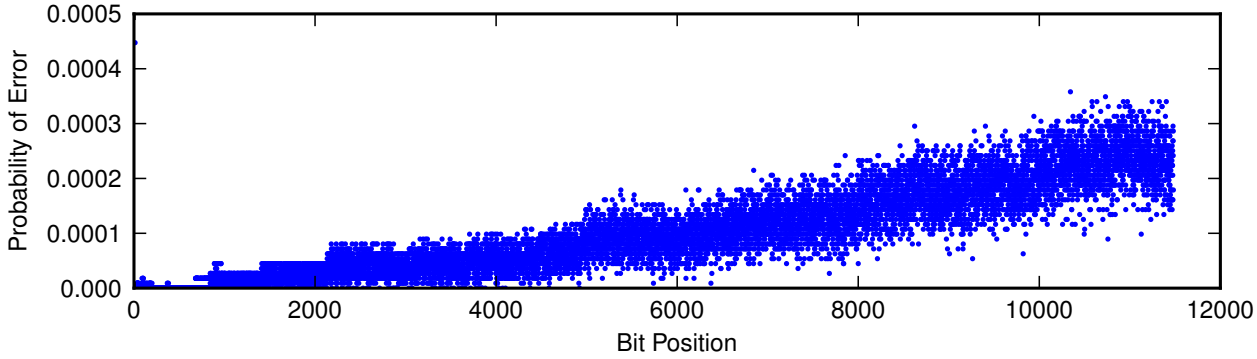
$$\varepsilon = 1/2 - \frac{cov(f(x), f'(x))}{2\sigma_{f(x)}\sigma_{f'(x)}}$$

Note that we can make ε arbitrarily small: If DenaLi injects no bits from the hidden message, the naturally occurring bit error distribution is unperturbed, and the two distributions are indistinguishable, both by definition and by construction. Such a channel, of course, is useless because its throughput is zero. Increasing the throughput of the hidden channel by injecting additional corrupted frames and introducing bit errors that deviate from the naturally occurring bit errors perturbs the underlying distribution. Thus, there is a tradeoff between the degree to which the bit error distribution is perturbed (*i.e.*, the number of bits from the hidden message that we inject into any corrupted frame) and the resulting throughput.

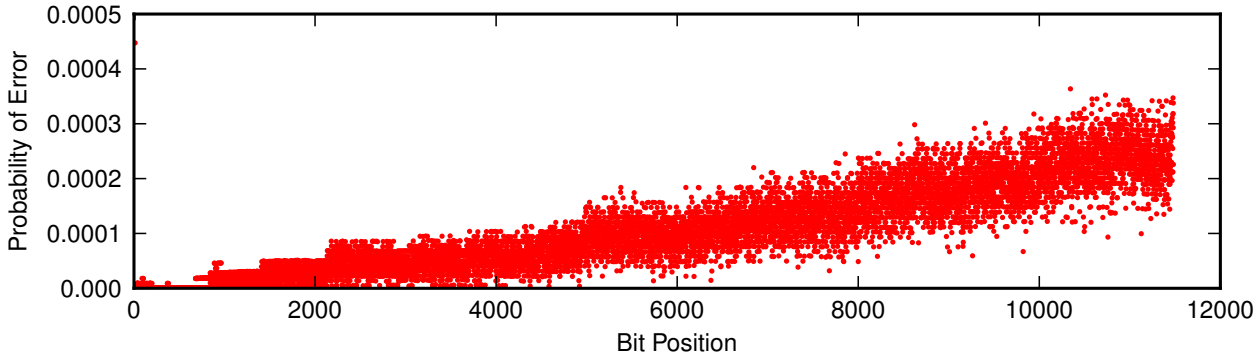
The packet error rate also has a naturally occurring value that varies over time. Suppose that for a given time interval i in packet trace P, the adversary observes a packet error rate f_i. Then, the adversary can observe a distribution $F = \{f_1, f_2, \ldots, f_n\}$ and a corresponding distribution F' for packet trace P'. We say that the packet error rate induced by running DenaLi achieves deniability if the adversary cannot succeed in distinguishing F and F' with a probability greater than $1/2+\varepsilon$. By defining ε according to the



(a) *The bit-error distribution from the perspective of the DenaLi sender, given a 23 KB message and a 70-byte TUN MTU.*



(b) *Natural bit error distribution.*



(c) *The bit error distribution after the DenaLi perturbation from (a) is added.*

Figure 7: *Bit-error distribution in an injected DenaLi frame at the sender, and bit error distributions as viewed at a monitor, with and without injected DenaLi frames.*

distance between these two distributions, we can determine the number of corrupted packets that a DenaLi sender can inject subject to an upper bound on ε. In principle, a DenaLi sender can detect the average packet error rate for some time interval and transmit corrupted packets in a way that tracks this packet error rate within some bound of ε. For the purposes of our evaluation, we have fixed the packet error rate, but in practice it might vary. Because packet corruption is a local phenomenon that is erratic and unpredictable, fine-grained control over this statistic may not be necessary or useful in practice.

6.3 Evaluating Deniability vs. Throughput

In this section, we evaluate the tradeoff between deniability and throughput of the DenaLi channel using our prototype implementation. We first describe the experimental setup and then present the results.

6.3.1 Experimental setup

We design an experiment with a sender, a receiver, and a single adversary. Each device is a laptop, where the sender and receiver are configured as described in Section 5. The sender generates cover traffic by browsing Gmail over a secure HTTP connection. The adversary is a third laptop with a wireless interface card configured

in monitor mode. We locate the adversary in close proximity to the receiver, which, as we described in Section 3, is the place where the adversary has the highest probability of detection. We assume that the adversary has only a single monitor. Each node in the setup collects packet traces and records the corresponding packet error rates and bit error distributions, allowing us to see these statistics at the sender, receiver, and the adversary.

6.3.2 Results

We first study the bit-error distributions that result from injecting chunks of hidden messages for a 70-byte MTU for the TUN device. Next, we study how this injected error distribution looks when viewed from the adversary, modeled as a monitor located near the receiver. Finally, to measure how throughput varies with deniability, we explore the relationship between the throughput of the DenaLi channel and the Pearson correlation coefficient between the normal bit error distribution and perturbed bit error distribution as seen at the adversary and the resulting throughput of the hidden message corresponding for the corresponding perturbation.

Figure 7a shows the bit-error distribution that results from injecting about 23 KB of a hidden message across a sequence of wireless frames, assuming a 70-byte MTU for the TUN device. We choose this size for the TUN MTU because previous studies [12] have shown that about 75% of corrupted packets have bit errors that are less than 400 bits, and a 70-byte MTU and 256-bit HMAC corrupts at most 100 bytes.

Figure 7b shows the original bit error distribution for chaff traffic, as viewed from the monitor; Figure 7c shows a similar distribution *after* DenaLi has injected a hidden message; as the figures show, the two distributions are essentially indistinguishable. For such a configuration, given "chaff" traffic throughput of about 2 Mbps and a packet error rate of every thousandth packet, DenaLi achieves a hidden message rate of about 6 bps. Although the two bit error distributions are not identical, they are reasonably close to one other. The Pearson correlation coefficient between these two distributions was 0.99801, yielding an ε value on the order of 10^{-4}. Part of the reason that the two distributions are so close is the relatively low throughput that we have chosen for the DenaLi channel. In the rest of this section, we further explore the tradeoff between the level of deniability that the DenaLi channel provides and the throughput that it achieves.

Our goal in the first experiment was to demonstrate DenaLi's ability to achieve deniability with respect to bit error distributions. We control the frequency and the extent of corruption so that the corruption is natural. Because the channel may further corrupt bits in the frame, we are conservative in how we corrupt bits in the frame, which naturally restricts throughput. We now explore how a sender can achieve higher throughput in exchange for less deniability (*i.e.*, a larger ε value). We inject one packet for every 10,000 frames of cover traffic This packet injection rate which clearly limits the maximum throughput we can achieve to (at most) 0.0001 of the throughput of the cover traffic, making the two distribution indistinguishable to the adversary. Figure 8 shows how ε varies as we increase the TUN MTU (i.e., throughput of the DenaLi channel). Naturally, ε increases with MTU. The throughput of the DenaLi channel is also directly proportional to both the throughput at which the chaff traffic is being sent and the packet error rate.

Finally, we study how the throughput of the DenaLi channel varies as we vary the packet error rate. To explore a range of packet error rates, we draw from the range of bit error rates reported in the

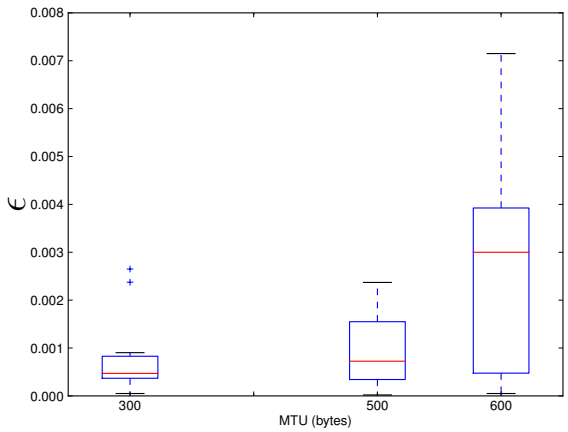


Figure 8: *ε vs. TUN MTU* (i.e., *injected frame size). We varied MTU sizes to achieve different throughput. Large TUN MTU values result in larger ε values and are less deniable.*

BER	PER	Throughput (bps)
10^{-4}	0.7	427.4
10^{-5}	0.1	103.6
10^{-6}	0.05	42.98

Table 1: *Bit error rates, approximate corresponding packet error rates assuming 1500-byte packets, and the resulting DenaLi throughput given a 70-byte TUN MTU. We test a range of bit error rates that are observed in practice [14].*

PPR study [14] and convert these observed rates to the corresponding packet error rates in this operating regime. For this experiment, we fix the MTU of the TUN interface to 70 bytes and send SSL chaff traffic by uploading a large file to a Gmail server while varying the packet injection rate (*i.e.*, the rate at which we inject corrupted frames containing hidden messages). Note that fixing the packet error rate and the MTU size is a rough mechanism for controlling ε, since the deviation is controlled by the size of the DenaLi block size (*i.e.*, the TUN MTU). We then measure the corresponding throughput (which is directly proportional to the throughput of the chaff traffic). Table 1 shows how the throughput of the DenaLi channel varies with packet error rates for a range of operating regimes. The channel efficiency is similar to previous experiments; as expected, the bitrate of the channel increases as the channel noise increases, as a noisier channel affords more opportunities to inject corrupted frames without deviating from "normal" packet corruption profiles. We caution that although traffic rates appear faster, the increase comes at the cost of deniability, as we showed in Figure 8.

7 Discussion

Here we discuss open issues, including both weaknesses with the current DenaLi design and avenues for future research.

Coping with limited wireless bandwidth Our experiments show that the cover traffic overhead for DenaLi is anywhere from about 10:1 (for high ε) to 100:1 (for low ε), depending on the burst of errors introduced and the frequency of they are injected. In any case, the amount of cover traffic required to achieve deniability is significant, and it may be prohibitive in settings where users bear

high data-usage costs or face usage caps. Although the overhead of cover traffic is inherently necessary for systems such as DenaLi, it may be more inconvenient for our use cases, where users may be communicating over DenaLi on wireless networks that are not that well-provisioned in the first place (*e.g.*, coffee shops, public squares). We intend to conduct more experiments in these types of settings to better understand the tradeoffs between the overhead that typical users would face and the deniability that they would need to achieve.

Analysis of bitrate adaptation algorithms In this paper, we have ignored the topic of bitrate selection. 802.11 devices have multiple bitrates to chose from, and some senders will decrease their bitrate when they encounter poor frame reception rates, hence corrupt frames may be transmitted at different rates than the eventually successfully received copy. In our prototype, we transmit corrupted frames at 1 Mbps. This is due to the limitation of commodity hardware as the chipset does not allow different transmission data rates. An adversary might profile the bitrates of the corrupted frames to discover anomalous bitrate adaptation patterns. The particular fallback rate(s) are determined by algorithm implemented by the driver at the sender, which might be vendor specific. For *softmac* drivers in the Linux distribution, the rate algorithm is Minstrel, and the fallback rates can be configured in the frame's transmit descriptor.

Timing attacks The adversary could perform more sophisticated timing attacks to discover a sender who is using DenaLi. Under ordinary circumstances, when a sender transmits a corrupted frame, the sender should follow that frame with a retransmission and ultimately receive a corresponding link-layer acknowledgment. Our implementation may not give rise to retransmissions within the appropriate time bounds; in particular, in the worst case, an adversary might see the corrupted frame and the retransmission within a very short time interval (possibly even simultaneously). This limitation results because of DenaLi's implementation on an off-the-shelf wireless chipset which constrain how we can modify the behavior of the wireless MAC. A software radio platform such as Sora [26] could be used to build a system that ensures that duplicate corrupted frames always precede the corresponding non-corrupted frame and link-layer acknowledgment, but such a prototype would not be as immediately deployable as DenaLi.

Transport Users can build two-way communication reliability using TCP or application-layer acknowledgements. Denali provides a decoupled virtual interface which gives DenaLi users freedom to choose. The attacker can mount DDoS attack by replaying corrupted packet traces, but we can see that DenaLi does not have a high overhead on commodity laptops. Also, the underlying encrypted hidden messages might arrive out of order, requring a transport-layer protocol like TCP.

Smartphones Our current prototype implementation of DenaLi was implemented on Linux laptops, but a likely deployment scenario for DenaLi might be on smartphones (*e.g.*, where citizens, operatives, or soldiers in a common area are coordinating and may only have small personal devices). In some of these areas, we might expect 802.11 WiFi deployments, in which case porting DenaLi to mobile devices might suffice. In some cases, 802.11 access points may not be deployed, in which case deniable communication might need to depend on some other wireless communication medium (*e.g.*, cellular, bluetooth). Current smartphones equipped with Snapdragon processors have clock cycles up to 1 GHz, which is powerful enough to process DenaLi packets.

Multi-hop wireless networks DenaLi currently operates only where the sender and recipient are within radio range of one another (*i.e.*, typically on the same wireless LAN). Although we believe that there are significant opportunities for using DenaLi in these settings, additional deployment opportunities exist in multi-hop wireless mesh networks, many of which are now explicitly being deployed for the express purpose of Internet freedom [4]. In these settings, DenaLi might still be used to deniably pass messages between each pair of participants (*i.e.*, it could form the "link layer" anonymous communication protocol), but applying DenaLi to a mesh network setting is less straightforward. First, doing so would involve constructing an overlay network of participants to relay the message, where the relays would be chosen both according to the level of trust for each participant, as well as their (rough) geographic location. Participants may also have to re-inject hidden messages into newly corrupted packets at each hop to avoid intersection attacks; doing so is not straightforward, since the intermediate hops may not possess the key to decode the hidden message.

8 Conclusion

Citizens of the world have an increasing need to achieve private communications in public spaces. Unfortunately, public meetings are observable, and users who are communicating with one another may need more covert means of exchanging messages when they are in close proximity. In many cases, users may wish to hide the fact that they are communicating in the first place. We suggest that parties who are near one another should take advantage of packet corruption in wireless networks to provide cover for their communications. To do so, we develop DenaLi, a lightweight deniable communications system that allows parties to exchange messages in a local setting, without exposing the fact that they are communicating. We take advantage of the ubiquitous nature of 802.11 "WiFi" networks to construct a covert communications channel, using corrupted packets as the "chaff" to hide communications between parties.

We have designed and implemented DenaLi using real end hosts and commodity wireless interface cards, demonstrating that such a system is practical. Our experiments explore the tradeoff between the deniability of the communications (*i.e.*, the extent to which the profile of packet corruption matches "normal" corruption characteristics) and the throughput that the user can achieve when sending hidden messages. Like many anonymous communications systems, DenaLi requires significant communications overhead in terms of the cover traffic that users must send to achieve deniability.

Acknowledgments

This research was supported by NSF awards CNS-1111723, CNS-1255274, and a Google Focused Research Award. We thank Geoff Voelker for suggesting the system name and paper title and Aaron Schulman for his feedback on the paper.

References

[1] S. Burnett, N. Feamster, and S. Vempala. Chipping Away at Censorship Firewalls with User-Generated Content. In *USENIX Security Symposium*, pages 463–468. Washington, DC, Aug. 2010.

[2] T. E. Calhoun, X. Cao, Y. Li, and R. Beyah. An 802.11 MAC Layer Covert Channel. *Wireless Communications and Mobile Computing*, 12(5):393–405, Apr. 2012.

[3] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: Solving the Puzzle of Enterprise 802.11 Analysis. In *ACM SIGCOMM*, volume 36, pages 39–50, Pisa, Italy, Aug. 2006.

[4] Commotion Wireless. `http://commotionwireless.net/`.

[5] R. C. Cynthia, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In *Advances in Cryptology (CRYPTO)*, pages 90–104, Aug. 1996.

[6] Denali codebase. `https://github.com/denalidenali/denali`.

[7] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, pages 303–320, San Diego, CA, 2004.

[8] D. B. Faria and D. R. Cheriton. Detecting identity-based attacks in wireless networks using signalprints. In *ACM Wireless Security Workshop (WiSE)*, Los Angeles, CA, Sept. 2006.

[9] N. Feamster, M. Balazinska, G. Harfst, H. Balakrishnan, and D. R. Karger. Infranet: Circumventing web censorship and surveillance. In *USENIX Security Symposium*, pages 247–262, San Francisco, CA, Aug. 2002.

[10] D. Goldschlag, M. Reed, and P. Syverson. Onion routing. *Communications of the ACM*, 42(2):39–41, Feb. 1999.

[11] B. Han, L. Ji, S. Lee, B. Bhattacharjee, and R. R. Miller. All Bits are Not Equal–A Study of IEEE 802.11 Communication Bit Errors. In *IEEE INFOCOM*, pages 1602–1610, Rio de Janeiro, Brazil, Apr. 2009.

[12] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. R. Miller. Maranello: Practical Partial Packet Recovery for 802.11. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 205–218, San Jose, CA, Apr. 2010.

[13] A. Houmansadr, G. T. Nguyen, M. Caesar, and N. Borisov. Cirripede: Circumvention Infrastructure Using Router Redirection with Plausible Deniability. In *ACM Conference on Computer and Communications Security (CCS)*, pages 187–200, Chicago, IL, Nov. 2011.

[14] K. Jamieson and H. Balakrishnan. PPR: Partial Packet Recovery for Wireless Networks. In *ACM SIGCOMM*, Kyoto, Japan, August 2007.

[15] J. Karlin, D. Ellard, A. W. Jackson, C. E. Jones, G. Lauer, D. P. Mankins, and W. T. Strayer. Decoy Routing: Toward Unblockable Internet Communication. In *USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, Apr. 2011.

[16] Linux Wireless Drivers for Atheros. `http://wireless.kernel.org/en/users/Drivers/ath9k`.

[17] S. Mathur, W. Trappe, N. B. Mandayam, C. Ye, and A. Reznik. Radio-Telepathy: Extracting a Secret Key from an Unauthenticated Wireless Channel. In *ACM MOBICOM*, pages 128–139, San Francisco, CA, Sept. 2008.

[18] A. K. Miu, H. Balakrishnan, and C. E. Koksal. Improving Loss Resilience with Multi-Radio Diversity in Wireless Networks. In *ACM MOBICOM*, Cologne, Germany, Sept. 2005.

[19] A. Najafizadeh, R. Liscano, M. V. Martin, P. Mason, and M. Salmanian. Challenges in the Implementation and Simulation for Wireless Side-Channel based on Intentionally Corrupted FCS. *Procedia Computer Science*, 5:165–172, 2011.

[20] M. Neufeld, J. Fifield, C. Doerr, A. Sheth, and D. Grunwald. Softmac-flexible wireless research platform. In *ACM SIGCOMM Workshop on Hot Topics in Networking (HotNets-IV)*, College Park, MD, Nov. 2005.

[21] N. Provos. Defending Against Statistical Steganalysis. In *USENIX Security Symposium*, volume 10, pages 323–336, Washington, DC, Aug. 2001.

[22] R. L. Rivest et al. Chaffing and Winnowing: Confidentiality Without Encryption. *CryptoBytes (RSA Laboratories)*, 4(1):12–17, 1998.

[23] M. Rogers and E. Saitta. Secure Communication over Diverse Transports. In *ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 75–80, Raleigh, NC, Oct. 2012.

[24] S. Rolewicz. *Functional Analysis and Control Theory: Linear Systems*, volume 29 of *East European Series*. Springer, 1987.

[25] K. Szczypiorski. HICCUPS: Hidden Communication System for Corrupted Networks. In *International Multi-Conference on Advanced Computer Systems*, pages 31–40, Oct. 2003.

[26] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wang, Y. Zhang, H. Wu, W. Wang, and G. M. Voelker. Sora: High Performance Software Radio Using General Purpose Multi-core Processors. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 75–90, Boston, MA, Apr. 2009.

[27] D. Turner, S. Savage, and A. C. Snoeren. On the Empirical Performance of Self-calibrating WiFi Location Systems. In *Proceedings of IEEE Conference on Local Computer Networks (LCN)*, Bonn, Germany, Oct. 2011.

[28] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman. Telex: Anticensorship in the Network Infrastructure. In *USENIX Security Symposium*, San Francisco, CA, Aug. 2011.