

The Lectrosonics 700 Series Encryption System

A White Paper

May 12, 2003

Introduction

Cryptography is an ancient art. For as long as there have been secrets, systems have been devised to lock them up... and to pry open the locks.

Anyone seeking to protect information is faced with trade-offs. A simple system is convenient and can be quite effective, but it might be easily defeated. A sophisticated system might be more intrinsically secure, but if it is too cumbersome to use, the temptation to bypass or shortcut its proper use might mean that a simpler system would be safer in reality. Where the happy medium lies depends on the nature of the user, the potential attackers, and the information being guarded.

In choosing a level of security, it is important to keep things in perspective. Once an encryption system is sufficiently strong that it makes it more difficult, time consuming and expensive to decrypt a message than it would be to obtain the information some other way (legitimately or otherwise), it has served its purpose.

The encryption system used in the 700 Series wireless is neither the weakest nor the strongest encryption available. Like all security systems, it represents a compromise between convenience and strength. This white paper introduces some basic cryptographic concepts, describing the 700 Series encryption system against that backdrop. We hope it will help you decide whether the 700 Series is right for you.

Algorithms, Encryption and Keys

An *algorithm* is a specific set of steps or instructions for doing something. For example, a shopper's supermarket check-out algorithm might read something like this: "If item count is less than ten, go to shortest express line. Otherwise go to shortest non-express line."

Encryption is the scrambling of a secret message to hide its meaning from unintended recipients. An *encryption algorithm*, then, is a set of steps or instructions for scrambling a secret message. Encryption algorithms are invariably accompanied by their counterpart *decryption algorithms*, so that intended recipients can receive the message.

A *key* is a shared secret that modifies an encryption algorithm's behavior, such that knowledge of the algorithm alone is not sufficient to decode a message. This is somewhat counterintuitive and it bears repeating, as it is a fundamental principle of cryptography. *In a well-designed cryptographic system, knowledge of the algorithm is of little use to an attacker.*

Key Management

If a good algorithm is used, there is no need to keep the algorithm secret. In fact, it is assumed that potential attackers will be able to obtain the algorithm. However, the security of the key is vital. *Key Management* refers to whatever steps are taken to ensure that keys are distributed securely to those who require them, and to no one else.

If parties can meet frequently to exchange keys in private, key management is trivial. However, in the real world safe, frequent meetings often prove inconvenient or impossible. Telephones can be tapped and postal mail and emails can be intercepted. For parties separated by some distance, key management becomes a thorny problem indeed.

One solution is to keep using the same key, but key reuse is a cryptographic no-no. An attacker with a large volume of messages encrypted with the same key has a substantial head start toward breaking the cipher. Once a key is compromised, further messages are transparent to the attacker until another key is chosen. Another solution is to use a *codebook*, with both parties agreeing on a list of keys and a protocol for how and when they are to change. This is better, but there is now a greater possibility that one or the other party will make a mistake, or that the code book might fall into the wrong hands.

Public Key Cryptography is a fascinating new technology that makes it possible to pass secret messages without a shared key. It is somewhat complicated to implement, but it overcomes one of cryptography's most vexing and persistent problems, the requirement for secure distribution of keys.

The 700 Series encryption system provides a number of features to assure good key management. Because of the ease with which the transmitter and receiver can be brought together, there is no need for key reuse, codebooks or public key cryptography.

Changing keys frequently is a highly recommended security practice, and the 700 Series makes that easy. At any time, a new key can be generated quickly and easily from the receiver's front panel controls. To get the key into the transmitter, a cable is connected from a special port on the receiver to the transmitter. The receiver then electronically whispers the key to the transmitter via the cable (*not* over the public airwaves!). The cable is then removed and normal wireless communications may proceed. The whole procedure takes just seconds.

Multiple transmitters can be taught the same key, but only if they are all programmed in the same session. This is a subtle but important security feature. An attacker with physical access to a receiver cannot send the key out the special port without starting a new session, which changes the receiver's key. The receiver will then have a different key than the legitimate transmitter, so the system won't work until a new key is set, thus obsoleting the stolen key.

The key management features of the 700 Series are carefully thought out to offer the best protection possible for your crucial secret keys. Nonetheless, it is still important to engage in good key management practice. It is a good idea to change keys frequently, and since the transmitters and receivers contain keys, treat them as you would keys, keeping them in a safe place when they are not in use.

Key Length

In researching cryptosystems, an important feature to consider is the key length, which is typically expressed as some number of bits. The number of bits in the key tells you how many possible keys exist. (For the mathematically inclined, the number is 2 to the power of the number of bits. For example there are sixteen possible 4-bit keys.) Anyone willing to try all possible keys can eventually crack a cipher. This is called a *brute-force attack*. Using a long key is the first defense against brute-force attacks.

The 700 Series encryption uses a 128-bit key. The number 128 might not sound like a very large number, but 2 to the 128 power is nothing short of staggering, weighing in at slightly more than three hundred trillion trillion trillion!

Number of 128-bit keys: 340,282,366,920,938,463,463,374,607,431,768,211,456

If a supercomputer were somehow able to try a trillion keys per second, it would still take ten billion *billion* years (yes, that's a billion squared!) to try them all.

Key Selection and Entropy

If the strength of a cipher rests on the key, then the selection of a good key is equally critical. When asked to select a password or combination, most people will choose something they can remember easily, like a word, a name, or their birthday. The people in black hats know this, and will try the most likely keys first.

The best defense is to ensure that all of the possible keys are equally likely to be chosen. This can be accomplished by basing key selection on a random process. The more chaotic and unpredictable the process, the better your key selection (and thus the security of your information) will be. The technical term for this chaos or randomness is *entropy*. A good *entropy source* means all keys are equally likely, and that means no shortcuts for the attacker.

The 700 Series encryption uses a clever but well-established entropy source: precise timings of user button presses. As the user navigates through the receiver's menu pages, a hardware timer is always spinning madly, at a rate so fast that it overflows several times a second. Every time the user presses a button, the instantaneous value of the timer is captured as a portion of the next key. The menu system guarantees that enough buttons will be pressed to generate wholly unique keys. This technique is generally recognized as a good entropy source. An attacker would not be able to save time by eliminating any "unlikely" key values.

A related security feature of the 700 Series is that there is no provision for manually entering a key, nor is a key ever displayed. Manual entry of a key would invite guessable keys (e.g. birthdays) or compromised keys (by nefarious types). With the 700 Series, the highly random secret key is known only to the transmitter and receiver. Even the legitimate operators of the system do not know the keys!

Data Compression

Repetition is the enemy of secrecy. If an attacker may make any assumptions at all about the key or the encrypted message, this information may be exploited to crack the cipher. In the ideal case, a message containing minimum information, with no redundancy, is encrypted using a random key. If the phrase "minimum information, with no redundancy" made you think of data compression, go to the head of the class.

The 700 Series wireless takes advantage of a very fortuitous coincidence. The way to get high audio quality at a low bit rate is to compress it, and the lowest distortion coincides with maximum entropy. The audio data is compressed in such a way that all transmitted numerical values occur with equal probability regardless of whether the audio signal is silent or 10 dB into limiting. The compression was optimized for best audio quality but a happy side effect is that the unencrypted data is highly chaotic and devoid of predictable patterns for an attacker to latch on to.

The Humble XOR

A very simple yet pervasive encryption technique is the *XOR (Exclusive OR)* operation. If message data is XORed with a particular *scrambling sequence* (usually the key or something derived from the key), it is mixed up, but not irretrievably. XORing again with the same scrambling sequence reverses the process and recovers the original message data.

XOR operations, combined with other techniques, can be found at the heart of many strong ciphers. In order for XOR to provide good protection, certain conditions are necessary which should sound familiar:

1. The scrambling sequence must be long (the less repetition, the better).
2. The scrambling sequence must be highly chaotic (high entropy).
3. The message data itself must be highly chaotic (high entropy).

If these conditions are met, the humble XOR can serve as the primary scrambling operation for fiendishly complex ciphers. The 700 Series uses XOR under the conditions listed above, offering quite a respectable level of security.

As a side note, it is actually possible to devise an unbreakable code based on XOR. This requires a *one-time pad*, a truly random scrambling sequence that is as long as the message being scrambled, and is destroyed after its first use. Such a system is practical for the President's red telephone but it would be too cumbersome for a digital wireless microphone.

Pseudo-Random Number Generators

It is not sufficient that the encryption key and the message being encrypted be as unpredictable, chaotic and devoid of patterns as possible. The encryption algorithm itself must scramble the message in as chaotic a way as possible. If the algorithm repeats itself or otherwise exhibits

nonrandom behavior, even if only for a few "weak" keys, it may provide all an attacker needs to crack the cipher.

Extremely useful, then, is some sort of very prolific entropy source, which can spit out large quantities of jumbled, yet reproducible, bits. Such a beast is called a *Pseudo-Random Number Generator* (or PRNG for short). Given a *seed* or starting point, a PRNG yields a very chaotic bit sequence. The pattern eventually repeats, but the sequence can be extremely long. Thus, for all intents and purposes, every unique seed generates a unique sequence, and any given seed generates the same sequence repeatably.

Modern systems typically use the key as the seed for the PRNG. That way, for every possible key, an arbitrarily long, highly chaotic sequence can be generated. These long, chaotic sequences are ideal for use as the scrambling sequence mentioned in the section on XOR, above.

Pseudo-Random Number Generators are a common building block used in a great many encryption algorithms, from simple to complex. The 700 Series encryption algorithm uses one PRNG in the transmitter and a matching one in the receiver. When the keys match, both generate the same scrambling sequence and the transmission can be successfully decrypted.

The 700 Series Algorithm

The 700 Series algorithm is fairly straightforward. The audio is aggressively compressed, yielding an unencrypted data stream with very high entropy. A 128-bit encryption key, generated with entropy harvested from button timings, is used to seed a pseudo-random number generator, which generates a continuous chaotic scrambling sequence. An XOR operation is used with the scrambling sequence to encrypt the packets and decrypt them again in the receiver. That is the algorithm in a nutshell, however the 700 Series offers three security levels, each of which introduces minor variations in the algorithm.

In level 1, the scrambling sequence is restarted approximately every 20,000 bits. This compromises the strength of the cipher somewhat but makes the system as easy to use as a traditional unencrypted system, once a key has been set.

In level 2, the scrambling sequence never repeats. This offers greatly enhanced security but imposes some minimal procedural requirements on the user, such as requiring that the receiver be powered on before the transmitter.

In level 3, the system enforces a strict policy that *no portion of any scrambling sequence shall ever be repeated*. This brings the strength of the encryption as close to cryptography's "Holy Grail", the one-time pad, as a commercial wireless microphone can probably ever hope to offer. The trade-off is that the system forces the user to do things

Mission Improbable: Top-Secret Guide for Would-Be Attackers

Assume you are Special Agent X from the other side. Your mission, should you choose to accept it, is to unlock secrets transmitted via a Lectrosonics 700 Series digital wireless microphone. This memo contains all the information you have so far. Good luck.

The 700 Series encryption uses 128-bit keys, so there are three hundred trillion trillion *trillion* possible keys. Thanks to a good entropy source (precise timing of button presses), no one, not even the engineers at Lectrosonics, knows of any shortcuts to finding the right key. Even if you could get your supercomputers to test a trillion keys per second, it would take ten billion *billion* years to try them all. You don't have that kind of time, Special Agent X!

It had occurred to you that you might not need the right key. Some weak cryptographic systems permit partial recovery of the data if the decryption key is merely "close" to the encryption key. Unfortunately your analysis of the 700 Series algorithm reveals that the pseudo-random sequence is too long and too chaotic to fall to this type of attack. All wrong keys do an equally poor job of decrypting the audio. So, you're back to square one.

Suppose you are able to gain physical access to one of the actual transmitters used to send the secrets you want. By dumping the contents of all memory devices, you might be able to extract the key without upsetting anything. You then quietly replace the transmitter, and rig a modified receiver to use the stolen key. This plan works only until the meeting starts. Because setting a new key is simple, your target sets up a new key at the start of each session. So, anything you managed to extract is now useless.

You then purchase a set of 50 transmitters and receivers [Hey, we can dream, can't we?] and send them to Headquarters, where a team of programmers and cryptanalytic experts take them apart. With a great deal of time and computing resources expended, your team might successfully reverse engineer the entire system: Lectrosonics uses a single pseudo-random number generator (PRNG), which you duplicate. A 128-bit key seeds the PRNG to generate a scrambling sequence, which is then XORed with the audio data.

What you hadn't counted on was that the audio is aggressively compressed, thus increasing entropy, before being encrypted. Adding to the bad news, one of the higher security levels is being used, so the scrambling sequence never repeats. Your experts haven't yet worked out Lectrosonics's proprietary audio compression algorithm, but you don't expect it will help much. Statistical analysis shows that it was optimized for maximum entropy. So far as you are able to tell, all numerical values are sent with roughly equal probability, and worse yet, the probability does not appear to be influenced by the audio at the microphone port.

You consider sending an agent over to Lectrosonics to work undercover with their engineers. No luck there. You already know how the system works. Unfortunately, *knowledge of the algorithm is of little use without the key!*

So, where does this leave you? You've decided that the Lectrosonics encryption is not the weakest point of attack. Passive remote reception of the transmission from the Lectrosonics wireless is just not going to be effective. It would be easier is to break in and plant a small surveillance device of your own, so you can steal secrets the old-fashioned way.

As you can see, trying to break both the encryption and audio algorithm code of the Lectrosonics transmission becomes a path of greatest resistance. The resources required to actually break these algorithms would be better spent pursuing other means of intelligence gathering. The Lectrosonics 700 Series encrypted wireless system remains a robust, secure wireless microphone system that provides the user a safe means of using a wireless microphone without the inherent

security risks found with any other analog or digital wireless system. On-site training without the restrictions of wired microphones and cables is again a reality even on a secured site.

Further Reading

Applied Cryptography: Protocols, Algorithms, and Source Code in C by Bruce Schneier. This is the definitive work on practical, modern cryptography.

Handbook of Applied Cryptography by Alfred J. Menezes, Paul C. Van Oorschot, Scott A. Vanstone. This is an excellent, rigorous textbook on the subject.

The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography by Simon Singh. This thrilling book brings to life the drama of real-life spy stories throughout history, and explains the codes in detail as well.