Essential Shannon Security with Keys Smaller than the Encrypted Message

Gideon Samid
Technion - Israel Institute of Technology
AGS Encryptions, Ltd. Israel
www.agsencryptions.com
+ 972-54-200-400
gideon@agsencryptions.com

To a cryptographer the claim that "Shannon Security was achieved with keys smaller than the encrypted message" appears unworthy of attention, much as the claim of "perpetuum mobile" is to a physicist. Albeit, from an engineering point of view solar cells which power satellites exhibit an "essential perpetuum mobile" and are of great interest. Similarly for Shannon Security, as it is explored in this article. We discuss encryption schemes designed to confound a diligent cryptanalyst who works his way from a captured ciphertext to a disappointing endpoint where more than one otherwise plausible plaintexts are found to be associated with keys that encrypt them to that ciphertext. Unlike some previous researchers who explored this equivocation as a special case of existing schemes, this approach is aimed at devising a symmetric encryption for that purpose per se.

Introduction

The prevailing cryptographies feature committed ciphertexts. Their effectiveness is based on an expected computational difficulty facing an adversary. Once this computational distance has been crossed, the plaintext lays "naked" and non-repudiative. This state of affairs suffers from certain weaknesses. The most important one is the general lack of proof with respect to computational difficulty.(*) The *threat* of additional relevant mathematical insight is everpresent. An adversary may possess a combination of computing power and brain power that will invalidate the user's assumptions with respect to infeasibility of cryptanalysis. There may also be special case weaknesses. Certain keys, or ciphertexts may be "weak" in some fashion. An adversary might discover such special case weakness, while the user might not. Psychologically speaking, non-repudiation is troublesome. The user might feel uncomfortable about having his confidential matters at the mercy of an adversary who might eventually dig it out.

If H(**P**) is the entropy of the plaintext space Π, then cryptanalyzing the prevailing committed ciphertexts

∗ Menezes, Oorschot and Vanstone [Ref 5] assert (page 32): "No public-key scheme has been proven to be secure (the same can be said for block ciphers). The most effective public-key encryption schemes found to date have their security based on the presumed difficulty of a small set of number-theoretic problems."

is a process where the entropy generally begins with $H(\Pi)=\lg(n)$, (n is the size of Π), and ends with $H(\Pi)=0$ (all but a single plaintext have lost their candidacy as the pre-encrypted message).

This committed ciphertext situation is considered a necessary step-down from what has been defined by Claude Shannon as information-theoretical secrecy, ("Shannon Security"). Shannon [Ref 1] defined it as a case where knowledge of the ciphertext is not at all helpful with respect to arriving at the pre-encrypted plaintext. We may write:

(eq-1) $$\boldsymbol{H(P|C=?) = H(P|C=!)}$$

The entropy of Π is the same whether the ciphertext C is not known (C=?) or known (C=!).

We may now concern ourselves with a case where after as much cryptanalysis as desired:

(ineq-2) $$\boldsymbol{H(P|C=?) > H(P|C=!) > 0}$$

The knowledge of C diminishes the entropy of the plaintext space but it would never vanish. Such will happen if $P_i, P_j \in \Pi$ end up with probability 0.5 each, leading to H=1; or H=2 if four plaintext messages end up with probability p=0.25 each. Since $H(\Pi)=0$ is the definition of a successful cryptanalysis, it will be sufficient to insure H>0 to defeat an adversary. While Shannon Security (eq-1) is preferable, one may settle for some point in the range ***[H(P|C=?) ---- 0],*** to claim what we designate as "essential Shannon security" or say, "Meta Shannon Security".

Shannon proved [Ref 1] that in order to achieve what he referred to as "Perfect Security" (eq-1), the encryption key can not be shorter than the encrypted message. The familiar "One Time Pad" (OTP) cipher exhibits Shannon Security: any ciphertext, C, can be matched with same size plaintext, P, by XOR-ing it with a key, K, which in turn, is computed as: $K = C \veebar P$.

We consider the general case where all the members of the plaintext set are of equal likelihood to be the encrypted message: p=(1/n). In this case $H(\Pi)=\lg(n)$ and remains lg(n) even after capturing ciphertext C – as long as the key, K, which was used to encrypt P is of size n. What happens when the key is one bit shorter than the message? This would reduce the lowest entropy to lg(n)-1. (Assuming for convenience n is even). This is the highest security case for Meta Shannon Security:

(ineq-3) $$\boldsymbol{H(P|C=?) = lg(n) > H(P|C=!) = lg(n)-1}$$

If OTP is used then an (n-1) bits key may serve as a "seed" to generate an n-bit key to XOR P. The n-th bit can be computed as, say, the parity of the (n-1) bits key. The computed n-bits key will be XOR-ed with P to generate C. This will reduce the number of plaintext candidates in half (hence H(P)=lg(n)-1)). The more bits we drop from the key, the lower the final entropy of the plaintext space, but as long

that it is not zero, one can claim that one has prevented a successful cryptanalysis.

In a practical case the initial entropy of the plaintext space will be much lower than the lg(n) maximum. Let's consider the extreme case where some circumstantial information led to a state where $P_k \rightarrow 0$ for $k \neq i, j$ and $P_{i=} P_{j=} 0.5$. ($Pi,Pj,Pk \in \Pi$) In that case a pseudo-OTP with (n-1) bits key will eliminate half of the Π space, and while the encrypted message, say Pi, will not be eliminated, Pj, may, or may not be eliminated (probability 50%), so the cryptanalyst may or may not be able to identify Pi as the true message. If both Pi and Pj are included in the surviving (0.5n size) subset of Π, then this meta Shannon security case is equivalent to the full (Perfect Security) Shannon case.

Consider now a case where on account of circumstantial evidence only m messages ($\in \Pi$) end up above a given probability threshold, T. If by using a certain key K of size $s < n$, all of the m plausible messages fall in the subset of surviving plaintext candidates, then the cryptanalyst is facing the same difficulty that would have been posed to her by a full Shannon Security system.

The competing claimant to the rank of "Almost Shannon Security" is the notion of Semantic Security defined as a case where for all values of $H(\Pi)$, (eq.1) holds against an adversary limited to polynomial time. Much as with Meta Shannon Security, Semantic Security opens the door for keys smaller than the encrypted message.

All this leads us to the conclusion that in reality a key smaller than the message may be as effective as message-size key. And if not, its effectiveness may be at various levels above the committed ciphertext state. If only (m-1) messages end up in the surviving subset of Π, then the effective security is lower than the perfect Shannon security case – but not much lower (especially for large m).

All in all, the above defined meta Shannon security offers parts or whole of the venerated Perfect Shannon Security, using key of no predetermined minimum size.

Related Work:

Shannon security is a classic chapter in cryptography and as such enjoys an honorable mention in every serious, or semi-serious text book. The admitted theoretical security is quickly followed with statements that lament the impracticality of the method, owing to the size of the key. Since all the prevailing cryptographies feature committed ciphertexts, there was not much that was done in terms of tying that theoretical Shannon security with practical algorithms.

Ganetti, Dwork, Naor and Ostrovsky [Ref 2] present an important discussion of Meta Shannon Security. Their angle is "deniability": the ability of a sender, a receiver, or both, to credibly deny the fact that plaintext P_0 was communicated, and falsely maintain that a different message P_1 was the one which the sender dispatched to the receiver. The purpose they see in the various schemes they propose is mainly resistance to coercion, as in vote-buying, and in multi-party computation facing an adaptive adversary. That work lists the One Time Pad as a case in point, and then offers a technique in which

the true message P_0 is listed along n decoy messages P_1, P_2,…P_n. Each such message is encrypted with a respective key: K_0, K_1, K_2,…K_n, and the resulting ciphertexts: C_0, C_1, C_2, ….C_n are concatenated in some order into Ct. When coerced, the sender (or the receiver) might choose any of the n decoy messages, and claim that it was the "true" one on account of the corresponding portion of Ct. As the authors admit, this is a very tedious practice and rather impractical.

The more interesting, and rather ingenious part of the Ganetti et al article [Ref 2] is focused on non-deterministic cryptography. When a ciphertext bit c is generated from plaintext bit p with the aid of a random sequence r, the sender may be able to find a different, pseudorandom sequence, r*, such that c can be claimed to represent a fake plaintext bit p* . The authors envision a publicly known faking algorithm:

(eq…4) $r^* = \phi(c,r)$

r* should exhibit the property:

(eq…5) $c = E(p^*,r^*)$

Where E is the encryption algorithm. It must be necessary for the adversary to be fooled into determining that r* is truly random (like r). Because of the probabilistic nature of the scheme, the faking claim must be associated with a faking error $\delta(s)$, where s is the security parameter. The article describes sender-deniable public-key encryption for which the error, δ, can be made as small as desired at the expense of the ciphertext size which is linear with $1/\delta$, and thus bloats into super-polynomial size.

The various schemes discussed in [Ref 2] appear quite cumbersome for implementation (no doubt protocol improvements will follow suit), and are aimed at a single faked plaintext with which to establish deniability. The proposed schemes don't appear to utilize deniability as a security element. Security there remains vested in the expected computational distance. By contrast, Meta Shannon Security aims at preventing an adversary of achieving $H(\Pi)=0$ regardless of her resource abundance.

R. Ganetti, U. Feige, O. Goldreich and M. Naor [Ref 3] describe certain non-committing encryption. That construct does not generally allow a sender or a receiver to deny the true plaintext to an adversary with possession of the ciphertext. It is rather a simulated ciphertext that is constructed for the purpose of deniability.

Entropy Reduction Through Key Discrimination.

The original plaintext space entropy $H_0 = H(\Pi)=\lg(n)$ is generally reduced through assumptions of language redundancy, contents-expectations and suchlike. It comes down to $H_1= H(\Pi|C=?)$. If ciphertext C is generated through some Meta Shannon Security Encryption, then, even the most unrestricted attack on C will depress the entropy to $Hc= H(\Pi|C=!)>0$. Hc will be computed on

account of those elements of Π which were associated with non negligible probability before the capture of C, and which are related to C through some key:

(eq…6) $C = E(Pi,Ki)$

where i=1,2,…m ; m being the number of such plaintexts.

One must now mind the following question: To what extent is the new information (the identity of the m keys) valuable in terms of further diminishing the entropy of Π?

Elaboration: The objective of an adversary is to eliminate (m-1) plaintexts and identify the one plaintext which corresponds to the key which is kept by the receiver. Whatever information that was available with respect to the $P_1,P_2,…Pm$ plaintexts was already expressed in H1. The new information which is available only after a thorough cryptanalysis of C is the identity of the m keys. The question is what can be deduced from this new information. In other words, a cryptanalyst is now concerned with the entropy of $K=\{K_1, K_2, K_3, …K_m\}$. If $H(K)=0$ then $H(\Pi)=0$.

It is therefore necessary for any viable Meta Shannon Security system to mind the identity of these K_1, $K_2,…K_m$ keys (namely: the equivocation keys). Their value should not undo the residual entropy which gives this encryption its strength.

For example consider a stream cipher where a message Pi of size n bits, is encrypted with key Ki into ciphertext C of same size (n). And let Kj be a key which encrypts Pj into C:

(eq…7) $C = E(Ki,Pi) = E(Kj,Pj)$

Let Ki be of size s bits ($s << n$). And let Kj be of size n . In that case Pi will look much more plausible than Pj, since we know how easy it is to match a message-size key with a given message.

Constructing Meta Shannon Security Encryption

The following construction is inspired by the notion that the basic reason for the prevalence of committed ciphertexts is that they are generated by a rather complex algorithm. A simple encryption algorithm might be easier to work with for the purpose of constructing Meta Shannon Security. Alas, if the algorithms $P \leftrightarrow C$ are to be simple, then where would the necessary complexity reside? (to foil an adversary). Looking around, one's finger naturally points to the left-out element which has not undergone conceptual changes in decades: *the key*.

The cryptographic key, K, and the encryption algorithm E are the two elements that are necessary for P to be computed into C. While the intuitive notion of complexity has traditionally been divided between K and E, the two have seen a clear division with respect to exposure. Following Kerckhoff's principle E is fully in the open and K is fully secret. Moreover, a functional definition of K may be: The

cryptographic key K, is the sum total of what must be changed in order to fully reconstitute a compromised encryption system – regardless of the nature of the compromise. Accordingly, the more complexity and variability in K, the more secure the system. (Since a new K means a big change). For some reason, whenever the key needed more complexity, it was supplied in terms of additional bits in the sequence. When now we look for the key to shoulder more of that stuff, we may seek to broaden the sequential order into a network.

Specifically, consider a graph K, constructed with Kv vertices which are interconnected with Ke directional edges. Further consider the case where the plaintext, P, is represented via an alphabet Av comprised of Lv symbols (letters), and where the ciphertext C is represented via an alphabet Ae comprised of Le symbols (letters). Such representation, of course, does not diminish the generality of P and C. The graph K is constructed so that every vertex can be connected through a directional edge to any other vertex. We shall mark each vertex with a single letter from the Av alphabet, and mark each edge with a single letter from the Ae alphabet. The common key which is a simple binary sequence can be viewed as a special case for the K-map, (or K-graph). In this case Av:[0,1], and Ae:[R], (R for right pointing edge). Each vertex (except the rightmost), points to the next vertex (only).

We now iterate two (very mild) restrictions on this construction:

Restriction 1 (R-1): No two edges which emanate from the same vertex would be marked with the same symbol.

Restriction 2 (R-2): Also dubbed, the "full access condition" is informally described as follows: From every vertex in the graph there should be access to all other letters in the Av alphabet. Meaning: if a a given vertex marked by symbol X∈ Av, does not have an edge leading from it to any choice Y ∈ Av (Y X) marked vertex, then that given vertex will be edge-connected to another X-marked vertex which will either have an edge to a Y marked vertex, or would be connected to yet another X-marked vertex, and so on until one such X-marked vertex will be edge-connected to a Y-marked vertex.

Apart from these two restrictions, K can be of any size, complexity and interconnectivity.

The Encryption Algorithm, E:

Do:

E-1: Represent plaintext P through alphabet Av' which is constructed as alphabet Av minus the Lv^{th} letter. P is thereby represented as P'

E-2: Eliminate letter duplication in P' by interjecting the Lv^{th} letter between any two consecutive elements in P' which are marked by the same letter (symbols) in P'. This changes P' to P".

E-3: Mark a path on K such that it corresponds to the Av letters that constitute P". The path should connect vertices through existing edges. Restriction R-2 will insure that whatever the sequence in P", there would be at least one path that would reflect that order, either on a one to one basis (letter in P" matched with a vertex in K), or by substituting some letters in P" with a string of same letter. In case of

such substitution (resulting in P""), there would be no confusion as for transforming P"' → P" since P" was constructed to be free of any letter duplication. Thus all consecutive strings of same letter in P"' will be reduced to a single (same) letter, thereby reconstructing P" from P"'.
E-4: Read off the edges that capture the path that was marked in E-3. That sequence is the ciphertext. C.

To decrypt C into P with the help of K, one would need to know the starting vertex, and from there use the C sequence to identify the path in E-3 which instantly yields P"'. P"' as seen above, readily generates P" and so on: C → P"' → P" → P' → P.

An adversary capturing C and ignorant about K might try to guess it. In the process of trying out possible maps, the adversary will find various maps which decrypt C into several plausible plaintexts. Hence: Meta Shannon Security.

As described above the transformation C ←→ P is based on the notion that a sequence on the graph can be described either by identifying the sequence vertices, or by listing the respective edges. By letting the vertices reflect the plaintext and by allowing the edges to reflect the ciphertext, the transformation becomes a very quick process (both ways) provided the graph K is at hand. It remains an undecided question without that graph. The latter will be proven by construction.

Tailored Key Encryption (TaKE). The general concept was presented in [Ref 4]. One is posing the challenge to find a key K so that for a given encryption algorithm E, a known ciphertext C, and a chosen plaintext P, it will hold that:

(eq.8) $C = E(P,K)$

Applying to the above described encryption algorithm we proceed below:

Challenge: Given C as a sequence of letters of Ae, and given P" as a non-repeat sequence constructed from alphabet Av, find a map K that will satisfy C = E(P",K). P" is chosen to be of same or smaller size with respect to C.

A simple solution will proceed as follows:

If P" is smaller than C, then replace some letters in P" with a string of several consecutive letters (all of the same symbol). Do so until the size of the resulting P"' is equal to C. Let n be that size.

a1. Construct n vertices marked as the sequence in P"'.
a2. Connect the vertices in (a1) with edges marked according to the sequence in C
a3. Add additional vertices and edges, at will, to comply with restriction R-1, and R-2.
This creates a graph K which according to the above described procedure does satisfy

C=E(P",K) as required.

While this will work, it is not very encouraging since it has the flavor of One Time Pad. The key here will even be larger than the encrypted message.

We will show now a procedure to cut down the size of K (with no preset lower limit):

b1. Construct the full size K as in a1, a2 above.
b2. Perform as many back-pointing procedures as possible. A back-pointing procedure will be described here in a sample case which is easy to formalize. The advantage in the selected description is that it is rather intuitive.

Let X,Y,Z ∈ Av, and let P"' contain a sequence XYZXZ:

(eq…..9). P"' = ……..XYZXZ……. = á-X-Y-Z-X-Z-â

where á and â represent the leading and trailing strings of P"'.

Now remove the edge that connects the first Z with the second X in the P"' string, and substitute it with an edge that connects that Z with the first X. Then construct a vertex marked Z (for the second Z), and draw an edge from the first X to the second Z. There is a chance that the symbol corresponding to that edge (from C) will be the same as the one that leads from the first X to the Y vertex. In that case, this procedure will not work for this instance, since it violates restriction R-1. If that conflict does not happen then the construction of K will continue from the first X. An edge will point from it to a Z vertex (the 2nd Z), and from there to â. In case of a conflict as above, simply try with another instance of such back-pointing. A single such instance, will reduce the size of K. There are in general many opportunities for back-pointing. Each such occurrence cuts the K size by one vertex. Intuitively this procedure will reduce the key size dramatically. The exact odds depending on the size of Av, and Ae, and on P"', and C. Complete the process with step (a3).

Note that the above construction simulates a chosen plaintext attack. An adversary trying to build the key that connects a known plaintext and a matching ciphertext. This most advantageous attack option challenges the attacker with the full measure of key equivocation as is evident from the above procedure.

While demonstrating Meta Shannon Security, as shown above, this key-heavy, algorithm-light procedure offers additional interesting properties. One was mentioned already: the computational load for bona fide translation back and forth between ciphertext and plaintext (C ←→ P) is extremely fast, opening up some new vistas and applications. Such are (1) telecommunication (cellular devices), instant messaging, VPN, etc.; (2) Instant encryption for files as they are being saved on disk, and equally fast decryption as they are loaded to an application.

Another is the fact that in any given encryption, it is not clear to an adversary if the entire key material has been used or just a part (even a small part) thereof. Recall step (a3) above. This characteristics means that even if somehow an adversary was able to deduce the correct key that was used in the past, she might still be in the dark versus some future encryption where an unused portion of that key is being utilized.

Yet another important attribute is the one-to-many encryption characteristics, combined with many-to-one decryption. K can be constructed with a zone of interconnected vertices all marked with the same letter X ∈ Av. One (or more) of these vertices is also connected to a vertex marked Y (see R-2). Now consider: P" = á,X,Y,â, where á and â are leading and trailing strings in P". One could chart numerous paths in which the zone of X-marked vertices is being traversed in any which way, as long as it eventually emerges into a Y-marked vertex. If such traversals are comprised of ***t*** X-marked vertices then the corresponding P"' string will look as:

(eq.10) $$P''' = á,\underbrace{X,X,\ldots\ldots}_{\leftarrow\ t\ \text{times}\ \rightarrow},X,Y,â$$

And regardless of the value of t, the reverse transformation P"' → P" will be unequivocal.

This important degree of freedom (which can be exercised for any letter in P") can be used either in a deterministic fashion, or randomly. The latter will endow this encryption with the same advantages that are known for other probabilistic encryptions. A given plaintext will appear as different ciphertexts each time it is encrypted. The deterministic option offers a host of possibilities in the category of digital watermarks, and subliminal messages.

Wrapping up this limited presentation, it bears to mention that the very same procedure that is vying for the venerated rank of "almost a Shannon grade" is also applicable in fast and small setting. Consider: Av: {X,Y,Z,W}; Ae: {U,D,R,L}. One may construct a 3x3 matrix like:

$$K_{3x3} = \begin{matrix} X & X & Y \\ Z & W & Y \\ Z & Z & Y \end{matrix}$$

And define the edges as follows: (definition d.1) each vertex is connected to the vertex left to it (if it exists) with an edge marked L; connected to the vertex right to it (if it exists) with an edge marked R; connected to the vertex below it (if it exists) with an edge marked D, and connected to a vertex above it (if it exists) with an edge marked U.

This simple K_{3x3} key satisfies R-1,2, and is capable of encrypting and decrypting any size plaintext. For a sufficiently long P, this small K=K_{3x3} will not exhibit Meta Shannon Security since each vertex will be visited several times over, and thus the ciphertext here will become "committed". Yet, one could construct a key by concatenating two K_{3x3} keys (or some other extension), and only after a certain time

point, or encryption-length point, actually utilize the extended key part. Hence an adversary who will crack K_{3x3} will be surprised when the new key part comes into play.

Summary:

We have presented a key-heavy, algorithm-light encryption procedure which exhibits Meta Shannon Security defined as a situation where a cryptanalyst can not pinpoint a single member of the plaintext space (entropy zero), as the message that generated a captured ciphertext. There remain two or more viable plaintext candidates, and that equivocation can not be resolved without possession of the encryption key, regardless of any measure of computing power. This fundamental equivocation suggests some interest in exploring a proper role for this method within the heavy-duty end of the encryption spectrum. The simple and quick algorithm, in turn, renders this method into a prospective evaluation item for applications where speed is of the essence. It was also shown that this procedure lends itself to digital watermarking and related applications.

Reference:

1. C. E. Shannon "Communication Theory of Secrecy Systems", Bell Systems Tech. Jr. Vol 28, pages 656-715, 1949
2. Ran Gannetti, Cynthia Dwork, Moni Naor, Rafail Ostrovsky "Deniable Encryption" Crypto'97
3. R. Ganetti, U. Feige, O. Goldreich and M. Naor "Adaptively Secure Computation", 28th STOC, 1996
4. G. Samid "Tailored Key Encryption" IACR un refereed publication http://eprint.iacr.org/2000/011/ J2ooo.
5. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone "Handbook of Applied Cryptography" CRC Press 1997