HAL
archives-ouvertes.fr

Attainable Unconditional Security for Shared-Key Cryptosystems

Fabrizio Biondi, Thomas Given-Wilson, Axel Legay

To cite this version:

Fabrizio Biondi, Thomas Given-Wilson, Axel Legay. Attainable Unconditional Security for Shared-Key Cryptosystems. The 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-15), Aug 2015, Helsinki, Finland. hal-01192859

HAL Id: hal-01192859

https://hal.inria.fr/hal-01192859

Submitted on 3 Sep 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Attainable Unconditional Security for Shared-Key Cryptosystems

Fabrizio Biondi*, Thomas Given-Wilson*, Axel Legay*
*Inria

Email: {fabrizio.biondi,thomas.given-wilson,axel.legay}@inria.fr

***Abstract*—Preserving the privacy of private communication is a fundamental concern of computing addressed by encryption. Information-theoretic reasoning models unconditional security where the strength of the results is not moderated by computational hardness or unproven results. Perfect secrecy is often considered the ideal result for a cryptosystem, where knowledge of the ciphertext reveals no information about the message or key, however often this is impossible to achieve in practice. An alternative measure is the equivocation, intuitively the average number of message/key pairs that could have produced a given ciphertext. We show a theoretical bound on equivocation called *max-equivocation* and show that this generalizes perfect secrecy when achievable, and provides an alternative measure when perfect secrecy is not. We derive bounds for max-equivocation, and show that max-equivocation is achieved when the entropy of the ciphertext is minimized. We consider encryption functions under this new perspective, and show that in general the theoretical best is unachievable, and that some popular approaches such as Latin squares or Quasigroups are also not optimal. We present some algorithms for generating encryption functions that are practical and achieve $90 - 95\%$ of the theoretical best, improving with larger message spaces.**

I. Introduction

Preserving privacy of private communication is a fundamental concern of computer science. Modern efforts can be divided into two categories: computational and unconditional security. Computational security of the privacy of a message depends on the assumed superpolynomial lower bound on complexity of some particular functions, e.g. factorization. Such lower-bound results are currently unproven, and may be weakened by technological progress in engineering, algorithmic theory, and quantum computing. Unconditional security is based on information-theoretic reasoning and proven independently of computational hardness. For this reason, unconditional security results are more general and solid than computational security results. However, the strict requirements to obtain unconditionally-secure cryptographic algorithms can make them generally impractical.

The first formal results on unconditional security were published by Shannon [1] using results from the formal theory of communication that he had just invented [2]. Shannon considers the transmission of an encrypted message on a public channel between two agents A and B that share a cryptographic key. Shannon investigates how long the key has to be to transmit the message with *perfect secrecy*, meaning that a third agent intercepting the encrypted message obtains no information about the original message. Shannon proves that to achieve perfect secrecy the key must be as long as the message and can be used only once, and that the one-time pad algorithm achieves this under uniform message distribution. This means that, to transmit each n-bit long message, the sender and receiver have to have previously agreed on a fresh n-bit long key, which is often impractical. Perfect secrecy is proven to be unachievable with a shared key shorter than the message to be transmitted.

In this work we consider the same scenario in which a key with a different size to the message is shared between A and B, and we ask how many bits of message can be sent while respecting a suitable definition of unconditional security. However, we want to define a security condition that is also attainable also when the key is smaller than the message, since this is the most common case in practice.

Since perfect secrecy cannot be achieved in this scenario, we define the maximum achievable security and how to attain it. Secrecy, as defined by Shannon, is a measure based on mutual information, and measures how much information is leaked by the communication, but not how hard it is for the attacker to decrypt the message given this leaked information. Instead of measuring secrecy, we measure the message *equivocation* of the encryption, measured as the conditional entropy of the message given the ciphertext. Equivocation, introduced by Shannon, measures how difficult it is for the attacker to decrypt the message after intercepting the ciphertext. Intuitively, *equivocation measures the average number of message/key pairs that could have produced a given ciphertext*.

We show a theoretical upper bound on equivocation, and we call this condition *max-equivocation*. We show that *max-equivocation generalizes perfect secrecy*, corresponding to perfect secrecy when it can be achieved and providing a characterization of maximum possible security when perfect secrecy cannot be achieved.

We derive upper and lower bounds to equivocation. We show that max-equivocation is achieved when the entropy of the ciphertext is minimized, contrarily to intuition. That is, the best theoretical encryption scheme is one that minimizes the entropy of the ciphertext.

We then consider encryption functions under this new perspective and show that in general the theoretical best is not achievable. Further, we show that popular approaches to encryption functions, such as Latin squares or Quasigroups [3], [4], [5], are also not practically optimal.

We present some algorithms for generating encryption functions from the message and key information that are

practical and achieve reasonably good results. In general these give solutions with a quality of $90 - 95\%$ of the theoretical best when the message space is less than 2^{11}, and the quality improves as the message space increases in size.

The structure of the paper is as follows. Section II recalls key concepts. Section III introduces max-equivocation and theoretic bounds. Section IV presents results on when the theoretic bounds can and cannot be achieved. Section V introduces some algorithms for finding encryption functions. Section VI analyzes the experimental results of the algorithms. Section VII briefly discusses alternative measures of entropy. Section VIII draws conclusions and discusses related work.

II. Background

We denote with $|\mathcal{S}|$ the size of set $\mathcal{S}$. Given sets $\mathcal{A}$ and $\mathcal{B}$ we say that a function $f : \mathcal{A} \to \mathcal{B}$ is *injective* if $\forall a_1, a_2 \in \mathcal{A}$ it holds that $f(a_1) = f(a_2) \Rightarrow a_1 = a_2$.

We refer to literature [6] for the definitions of sample space $\mathcal{X}$, probability $P(E)$ of an event $E \subseteq \mathcal{X}$, random variable X on $\mathcal{X}$, entropy $H(X)$ of a random variable, mutual information $I(X;Y)$ of X and Y, and so on. We will write $\rho_{\mathcal{X}}(X)$ for a probability distribution on the random variable X on the sample space $\mathcal{X}$, abbreviated in $\rho(X)$ when the sample space is unambiguous.

Recall that if X and Y are independent random variables on sample spaces $\mathcal{X}$ and $\mathcal{Y}$ respectively, then $H(X,Y) = H(X) + H(Y)$.

A. Shared-Key Cryptosystems

A shared-key cryptosystem can be defined as follows.

Definition 1: A *(shared-key) cryptosystem* is a 3-tuple $(\mathcal{M}, \mathcal{K}, \mathit{enc})$ where:

- the *message space* $\mathcal{M}$ is a finite set of possible messages;
- the *key space* $\mathcal{K}$ is a finite set of possible keys;
- the *encoder enc* is a function $\mathcal{M} \times \mathcal{K} \to \mathcal{C}$ to some space $\mathcal{C}$ such that $\forall m \in \mathcal{M}.\ \mathit{enc}(m, \cdot)$ is injective, and $\forall k \in \mathcal{K}.\ \mathit{enc}(\cdot, k)$ is injective.

A shared-key cryptosystem induces the set of its possible ciphertexts and a decoder function.

Definition 2: Let $\mathcal{C} = \{c \mid \exists m \in \mathcal{M}.\ \exists k \in \mathcal{K}.\ c = \mathit{enc}(m,k)\}$ be the *ciphertext space*, i.e. the set of possible ciphertexts c given $\mathcal{M}$, $\mathcal{K}$ and *enc*.

Definition 3: Let the *decoder dec* be the function $\mathcal{C} \times \mathcal{K} \to \mathcal{M}$ such that $\mathit{dec}(\mathit{enc}(m,k),k) = m$.

The existence and uniqueness of such a decoder function is ensured by the requirement that $\mathit{enc}(\cdot, k)$ is injective. Note that injectivity of $\mathit{enc}(m, \cdot)$ is not strictly necessary, but is chosen here to preserve symmetry of the results.

The channel model of a cryptosystem was introduced by Shannon [1]. In this model, agent A wants to send a message $m \in \mathcal{M}$ to agent B on a public channel that is eavesdropped by agent E. Initially, A and B share a secret *key* $k \in \mathcal{K}$.

A encodes the message m with key k using an appropriate encoder function enc, obtaining the *ciphertext* c. A sends c to B via a public channel, where c is also read by the eavesdropper E. Knowing the key k, B decodes the message m using the inverse of the encoding function, dec. Not knowing the key k, E tries to infer m from their knowledge of enc and c and using unlimited computational power.

Consider the information available to agent E. Agent E knows the encoder function enc, and ciphertext c, but not the message m or key k. We will use random variables to model E's knowledge about the communication before and after E intercepts the ciphertext c. Let M (resp. K, C) be a random variable on the support set $\mathcal{M}$ (resp. $\mathcal{K}, \mathcal{C}$) representing the *a priori* value of the message m (resp. key k, ciphertext c) according to E, i.e. the value before the ciphertext is intercepted.

Random variables M and K are assumed to be independent, so

$$H(M,K) = H(M) + H(K)\ .$$

We call $H(M)$ the *prior entropy* of M, modeling the amount of uncertainty that E has on the message before observing the ciphertext. Similarly, we call $H(M|C)$ the *posterior entropy* of M given C, modeling the uncertainty left on the message after observing the ciphertext. The prior entropy gives a measure of how hard it is for E to guess the message before observing the ciphertext, while the posterior entropy measures the same after observing the ciphertext.

Shannon defined *perfect secrecy* as the highest possible security condition attainable on a cryptosystem [1]. Perfect secrecy is attained when the mutual information between the message and the ciphertext is zero; when knowledge of the ciphertext gives no information about the message. That is, the prior and posterior entropies coincide, meaning that observing the ciphertext did not reduce the uncertainty of E on the message in any way.

Definition 4 (Perfect Secrecy): A cryptosystem attains *perfect secrecy* iff

$$H(M) = H(M|C)$$

It can be shown that

$$H(M) = H(M|C) \Leftrightarrow I(M;C) = 0$$

thus perfect secrecy is attained when the mutual information between the message and the ciphertext is 0.

Shannon also proved that for a cryptosystem to be perfectly secure it is necessary that the key space is at least as large as the message space, i.e.

$$|\mathcal{K}| \geq |\mathcal{M}|$$

making perfect secrecy hard to achieve in practice. Indeed, Shannon proved that perfect secrecy can be achieved when $|K| = |M|$ as long as the keys are uniformly distributed [1]. Observe that such requirements induce the relation that

$$H(K) \geq H(M)\ .$$

However, this requires A and B to have exchanged such a key k before sending the message m, and k can be used only once.

B. Latin Squares and Rectangles

A *Latin rectangle* is a $a \times b$ matrix with elements in $\{1, \ldots, b\}$ such that all of its rows and columns are distinct. It is a *Latin square* iff $a = b$.

An encoder function can be represented as a Latin rectangle if $|C| = \max(|M|, |K|)$, in the form of a $|K| \times |M|$ matrix with elements in $\{c_1, \ldots, c_{|C|}\}$ [1], [7]. However, we will show in Section IV that the optimal encoder function is in general not a Latin rectangle.

C. Unicity

Informally, the *unicity* of a cryptosystem is the average length of the ciphertext such that only one pair of message and key could have produced the ciphertext. This is only an issue when not all bit strings are messages. It is possible to compute the ciphertext length for which the number of messages per ciphertext reduces to 1. This is called the *unicity point*.

Let M_n (resp. C_n) be a random variable on messages (resp. ciphertexts) of length n. Then the unicity point u is defined as the least positive value of n such that $H(M_n) + H(K) - H(C_n) = 0$.

III. Max-Equivocation Generalizes Perfect Secrecy

We want to provide a measure of the security of a shared-key cryptosystem that is meaningful whether or not the the key used is smaller than the message to be sent.

Consider the entropy $H(M)$ of the message m. It measures the lack of information that E has about the message, so a higher $H(M)$ corresponds to a greater difficulty for E to guess the message. In our scenario, A and B share a secret key k, and the lack of information that E has about k is measured by $H(K)$. The amount of entropy left in the message after the attacker has intercepted the ciphertext is measured by the message equivocation $H(M|C)$, so we focus upon this.

Shannon proved that $H(K) \geq H(M)$ is a necessary but not sufficient condition to find an encoder function *enc* such that $I(M;C) = 0$. Since $I(M;C) = H(M) - H(M|C)$, this corresponds to $H(M|C) = H(M)$, thus the message equivocation is equivalent to the entropy of the message. Since $H(K) \geq H(M)$, it is also true that $H(M|C) \leq \min(H(M), H(K))$.

Consider the complementary case in which $H(K) < H(M)$. Since E knows every other detail of the communication except for k, we conjecture that it is not possible to encrypt m as a ciphertext $c \in C$ in a way such that the message equivocation $H(M|C)$ is greater than the entropy $H(K)$, i.e. $H(M|C) \leq H(K)$. Since $H(K) < H(M)$, this again corresponds to $H(M|C) \leq \min(H(M), H(K))$.

We conclude that irrespective of the relative sizes of the entropies of the message space and key space, the maximum equivocation is always achieved when it corresponds to their minimum. We say that a cryptosystem respects *max-equivocation* when it achieves this upper bound.

Definition 5: A cryptosystem achieves *max-equivocation* iff

$$H(M|C) = \min(H(M), H(K)) \ .$$

Key equivocation can similarly be defined as $H(K|C)$, measuring the average number of keys that could have produced a given ciphertext.

We will now introduce semi-injectivity, the core property of an encoder function. We will show that as a consequence of the encoder function being semi-injective on both message and key, message equivocation and key equivocation actually coincide, so we can just call them equivocation.

A. Semi-injectivity

We introduce the concept of *semi-injectivity* of a function, that will be used throughout the paper.

Definition 6: A function $f : \mathcal{A} \times \mathcal{B} \to \mathcal{C}$ is *semi-injective* on $\mathcal{B}$ iff it holds that

$$\forall a \in \mathcal{A},\ \forall b_i, b_j \in \mathcal{B}.\ f(a, b_i) = f(a, b_j) \Leftrightarrow b_i = b_j \ .$$

Observe that a function on a tuple can be semi-injective on any element of the tuple, and that a semi-injective function on a single argument is injective. Semi-injectivity is equivalent to the property that the function can be Curried and after being applied to one argument the remaining function is injective, i.e. $f : \mathcal{A} \times \mathcal{B} \to \mathcal{C}$ is semi-injective on $\mathcal{B}$ when $f' : \mathcal{A} \to \mathcal{B} \to \mathcal{C}$ and $f'(a) : \mathcal{B} \to \mathcal{C}$ is injective.

Lemma 1: Let a function $f : \mathcal{A} \times \mathcal{B} \to \mathcal{C}$ be semi-injective on $\mathcal{A}$. The image $f[\mathcal{A}, \mathcal{B}]$ of f in $\mathcal{C}$ has at least the size of $\mathcal{A}$:

$$|f[\mathcal{A}, \mathcal{B}]| \geq |\mathcal{A}| \ .$$

Consequently, if f is semi-injective in both $\mathcal{A}$ and $\mathcal{B}$ then

$$|f[\mathcal{A}, \mathcal{B}]| \geq \max(|\mathcal{A}|, |\mathcal{B}|) \ .$$

Lemma 2: Let $f : \mathcal{A} \times \mathcal{B} \to \mathcal{C}$ be semi-injective on $\mathcal{A}$, A be a random variable on $\mathcal{A}$, $\rho_{\mathcal{A}}(A)$ a probability distribution on A, and $H(A)$ the entropy of $\rho_{\mathcal{A}}(A)$. Similarly, let C be a random variable on $f[\mathcal{A}, \mathcal{B}]$, $\rho_{\mathcal{C}}(C)$ a probability distribution on C, and $H(C)$ the entropy of $\rho_{\mathcal{C}}(C)$. Then

$$H(A) \leq H(C) \ .$$

Proof: Recall that $|f[\mathcal{A}, \mathcal{B}]| \geq |\mathcal{A}|$ by Lemma 1. If $|f[\mathcal{A}, \mathcal{B}]| = |\mathcal{A}|$ then there must be one $c \in f[\mathcal{A}, \mathcal{B}]$ that corresponds to each $a \in \mathcal{A}$ and thus $H(A) \leq H(C)$. If $|f[\mathcal{A}, \mathcal{B}]| > |\mathcal{A}|$ then there must be at least two c_i and c_j such that $f(a, b_x) = c_i$ and $f(a, b_y) = c_j$. This increases the entropy, due to either $H(B)$ (when $b_x \neq b_y$), or via f itself otherwise. In either case, $H(A) \leq H(C)$. ■

Definition 7: A function $enc : \mathcal{M} \times \mathcal{K} \to \mathbb{N}$ is an *encoder function* iff it is semi-injective on $\mathcal{M}$ and $\mathcal{K}$.

Since the encoder function is semi-injective on $\mathcal{M}$, then the message is uniquely determined by a given ciphertext c and key k, thus $H(M|K, C) = 0$. Analogous reasoning on the encoder being semi-injective on $\mathcal{K}$ shows that $H(K|M, C) = 0$. We use these results to show that message equivocation and key equivocation coincide.

Theorem 1: $H(M|C) = H(K|C)$.

Proof: The proof adapts the similar proof of Theorem 1 of [8, sec. 7.3]:

$$\begin{aligned} H(M|C) &= H(M,C) - H(C) \\ &= H(M,K,C) - H(K|M,C) - H(C) \\ &= H(M,K,C) - H(M|K,C) - H(C) \\ &\qquad \text{(since } H(M|K,C) = H(K|M,C) = 0) \\ &= H(K,C) - H(C) = H(K|C) \ . \end{aligned}$$

■

Since message equivocation and key equivocation coincide by Theorem 1, henceforth we will refer to them just as equivocation, denoted by $H(M|C)$.

B. Bounds on Equivocation

This section is dedicated to studying upper and lower bounds for equivocation, and to find necessary and sufficient conditions for a cryptosystem to achieve max-equivocation.

Consider after agent E intercepts the ciphertext c, their *a posteriori* information about the message is measured by the equivocation $H(M|C)$.

Lemma 3: The entropy of the ciphertext has: lower bound of the greater of the entropy of the message or the key; and upper bound of the entropy of the message plus the entropy of the key. That is,

$$\max(H(M), H(K)) \leq H(C) \leq H(M) + H(K) \ .$$

Proof: By definition of enc and Lemma 1. ■

Lemma 4: The entropy of the ciphertext given the message is that of the key. That is, $H(C|M) = H(K)$.

Proof: By definition of enc being semi-injective on K and Lemma 2. ■

Theorem 2: Equivocation ranges from 0 to the minimum of the entropy of the message and the entropy of the key, i.e.

$$0 \leq H(M|C) \leq \min(H(M), H(K)) \ .$$

Proof:

$$\begin{aligned} H(M|C) &= H(C|M) + H(M) - H(C) \\ &= H(K) + H(M) - H(C) \qquad \text{(by Lemma 4)} \end{aligned}$$

with the lower bound being

$$\begin{aligned} H(M|C) &\geq H(K) + H(M) - (H(M) + H(K)) \\ &\qquad \text{(by Lemma 3)} \\ &= 0 \end{aligned}$$

and upper bound

$$\begin{aligned} H(M|C) &\leq H(K) + H(M) - (\max(H(M), H(K))) \\ &\qquad \text{(by Lemma 3)} \\ &= \min(H(K) + H(M) - H(M), \\ &\qquad H(K) + H(M) - H(K))) \\ &= \min(H(K), H(M)) \ . \end{aligned}$$

■

Corollary 1: A necessary and sufficient condition for max-equivocation is that the entropy of the ciphertext is equivalent to the maximum between the entropy of the key and the entropy of the message:

$$\begin{gathered} H(C) = \max(H(K), H(M)) \\ \Updownarrow \\ H(M|C) = \min(H(K), H(M)) \ . \end{gathered}$$

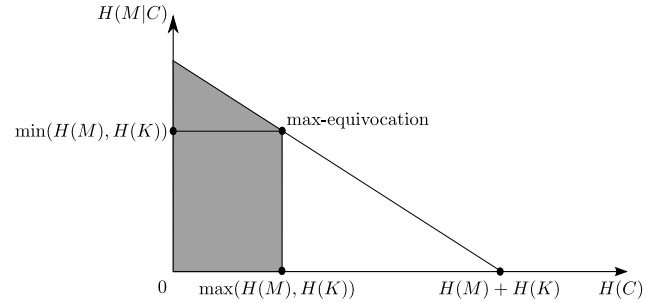

Fig. 1. Graph showing equivocation $H(M|C)$ as a function of the entropy of the ciphertext $H(C)$. The gray area is unachievable, since $H(C) \geq \max(H(M), H(K))$ by Lemma 3, thus the maximum possible equivocation is $H(M|C) = \min(H(M), H(K))$.

Since by Lemma 3 we know that $H(C) \geq \max(H(K), H(M))$, this means that *max-equivocation is achieved when the entropy of the ciphertext is minimal*. The relation between equivocation and entropy of the ciphertext is depicted in Figure 1.

Note that the idea that the entropy of the ciphertext has to be *minimized* goes against common intuition in the field of unconditional security, and claims of the opposite can be found e.g. in [8, p. 117]. In the next Sections we will study how to construct an encoder function minimizing the entropy of the ciphertext, and we will show that the theoretical minimum of $H(C) = \max(H(K), H(M))$ cannot always be achieved.

IV. Uniform Key, Non-Uniform Message

A. Motivation

Since we have assumed that the eavesdropper E has unlimited computational power, we must address the problem of unicity. If, in the language in which the message m is encoded, all sequences of symbols are messages, this is not a problem. Otherwise, there will be a certain ciphertext length after which there is only a single message that can be decoded from the ciphertext. Thus, a viable form of attack for E is to decrypt the ciphertext with all possible keys and keep the only possible message. We refer to [1], [8] for a more thorough treatment of unicity.

Let $\mathscr{L}$ be the language used for the message, $|\mathscr{L}|$ the number of its symbols, and $H_{\mathscr{L}}$ its entropy per symbol. Then the average amount of ciphertext symbols that E needs to have a single message behaves as $H(K)/R_{\mathscr{L}}$, where $R_{\mathscr{L}} = 1 - \frac{H_{\mathscr{L}}}{\log|\mathscr{L}|}$ is the redundancy of the language.

Redundancy of a language can typically be significantly reduced by using compression methods upon the language and then using the compressed result as the message. While this reduces redundancy, it still creates some structure within the language and does not achieve uniform message distribution.

Even languages that have a fixed or bounded message length (such that redundancy does not ensure a unique message from a given ciphertext) do not in general have uniform distributions over the messages [9].

Thus we shall consider max-equivocation where the message distribution is non-uniform. We assume the key distribution is uniform for simplicity, although no results implicitly rely upon this assumption. From Corollary 1 and Lemma 3 we have that the max-equivocation is ensured when the entropy of the ciphertext is minimized. Unfortunately max-equivocation (and thus perfect secrecy) cannot be achieved in all scenarios.

B. Achieving Max-Equivocation

Max-equivocation can be achieved for any distribution over the message space when the key space is of equal size and the key distribution is uniform. The solution for max-equivocation in this case is any encryption function that can be represented as a Latin square.

Observe from Corollary 1 that $H(C) = \max(H(K), H(M))$ and that since $|\mathcal{K}| = |\mathcal{M}|$ and $\mathcal{K}$ has uniform distribution that $H(K) \geq H(M)$. From this we require that $H(C) = H(K)$ for max-equivocation.

The easiest way to achieve $H(C) = H(K)$ is for $|\mathcal{C}| = |\mathcal{K}|$ and for the distribution of $\mathcal{C}$ to be uniform. This is achieved when the encryption function corresponds to a Latin square.

Theorem 3: Given $|\mathcal{K}| = |\mathcal{M}|$ and uniform distribution over $\mathcal{K}$, any encryption function enc that can be represented as a Latin square achieves max-equivocation.

Proof: As enc can be represented as a Latin square it follows that every $c \in \mathcal{C}$ must appear in each row and each column exactly once. Since the key distribution is uniform, the probability for each c is $\sum_{m \in \mathcal{M}} \frac{\rho(m)}{|\mathcal{K}|}$ and since $\sum_{m \in \mathcal{M}} \rho(m) = 1$ it follows that $\rho(c) = \frac{1}{|\mathcal{K}|}$. Conclude via the uniform distribution over $\mathcal{C}$ and $|\mathcal{C}| = |\mathcal{K}|$ that $H(C) = H(K)$ and Corollary 1. ■

There are other heuristics for achieving max-equivocation that can exploit much smaller key spaces. For example, when the message distribution can be partitioned into subsets $\mathcal{M}_p$ where $\forall m_i, m_j \in \mathcal{M}_p,\ \rho(m_i) = \rho(m_j)$ holds for all p, and where $\forall p,\ |\mathcal{M}_p| \geq |\mathcal{K}|$. Then by using uniform distribution of keys and having every $c \in \mathcal{C}$ abide by $\forall m_i, m_j.\ enc(m_i, k_x) = c = enc(m_j, k_y) \Leftrightarrow m_i, m_j \in \mathcal{M}_p$ for some k_x, k_y and p.

C. Unachievable Max-Equivocation

In general max-equivocation cannot be achieved. Indeed, there are small examples where $H(C)$ cannot be made to match $\max(H(K), H(M))$. For example, consider when there are three messages with probabilities $\rho(m_1) = 0.2$ and $\rho(m_2) = 0.3$ and $\rho(m_3) = 0.5$ and the key space is of size 2 with uniform distribution. It turns out that there is no encoder function such that $H(C) = H(M) \approxeq 1.4854$. The best possible is a Latin rectangle where the symbols must have the probabilities $\rho(c_1) = 0.25$ and $\rho(c_2) = 0.35$ and $\rho(c_3) = 0.4$ which yields $H(C) \approxeq 1.5588$.

D. Non-Optimality of Latin Rectangles

When $|\mathcal{M}| \neq |\mathcal{K}|$, each Latin rectangle of size $|\mathcal{K}| \times |\mathcal{M}|$ represents an encoder function with $|\mathcal{C}| = \max(|\mathcal{M}|, |\mathcal{K}|)$. We compute the minimum and maximum entropy of the ciphertext space generated by such an encoder function:

Lemma 5: The entropy of the ciphertext when the encoder function can be represented as a Latin rectangle has: lower bound of the greater of the entropy of the message and the entropy of the key; and upper bound of the logarithm of the size of the maximum between the sizes of the message space and the key space. That is,

$$\max(H(M), H(K)) \leq H(C) \leq \log(\max(|\mathcal{M}|, |\mathcal{K}|)) \ .$$

Proof: By definition of enc and Lemma 1 we have that $\max(H(M), H(K)) \leq H(C)$ and the upper bound is the maximum entropy for the ciphertext. ■

When the distribution on the larger side of the rectangle is uniform then any Latin rectangle achieves max-equivocation, as shown in Section IV-B.

When the distribution on the larger side of the rectangle is not uniform, it is possible that the optimal encoder function is not a Latin rectangle. For instance, let $\mathcal{M} = \{m_1, m_2, m_3\}$ with distribution $\rho(M) = \{m_1 \mapsto 0.02, m_2 \mapsto 0.49, m_3 \mapsto 0.49\}$ and let the key be uniform on $\mathcal{K} = \{k_1, k_2\}$. The best Latin rectangle encoding shown in Table I(a) has $\mathcal{C} = \{c_1, c_2, c_3\}$ and $H(C) \approxeq 1.5097$, while the optimal encoding shown in Table I(b) has $\mathcal{C} = \{c_1, c_2, c_3, c_4\}$ and $H(C) \approxeq 1.1414$. Note that neither reaches the theoretical lower bound of $H(C) = H(M) \approxeq 1.1214$.

TABLE I. OPTIMAL ENCODINGS FOR A) 3 SYMBOLS (BEST LATIN RECTANGLE) B) 4 SYMBOLS (OPTIMAL)

(a)

		Message		
		m_1	m_2	m_3
Key	k_1	c_1	c_2	c_3
	k_2	c_2	c_3	c_1

(b)

		Message		
		m_1	m_2	m_3
Key	k_1	c_1	c_2	c_3
	k_2	c_4	c_3	c_2

Due to the inherent complexity of finding an encoder function that minimizes the entropy of the ciphertext, computing an optimal encoder function for a given key and message distribution is not trivial. In the next Section we experimentally compare different heuristics.

V. ENCODER GENERATOR HEURISTICS

We propose 4 different heuristic algorithms that produce an encoder function for a given message space $\mathcal{M} = \{m_1, m_2, \ldots, m_{|\mathcal{M}|}\}$ with a given probability distribution $\rho(M)$ and a given key space $\mathcal{K} = \{k_1, k_2, \ldots, k_{|\mathcal{K}|}\}$ with uniform distribution. We will assume that $|\mathcal{K}| < |\mathcal{M}|$ and $\rho(m_1) \leq \rho(m_2) \leq \cdots \leq \rho(m_{|\mathcal{M}|})$.

The algorithms we propose are similar as they all consider the encoder function as a $|\mathcal{K}| \times |\mathcal{M}|$ matrix $\mathbf{C}$ where the

```
Data: message space M = {m_1, m_2, ..., m_|M|},
      probability distribution ρ(M), key space
      K = {k_1, k_2, ..., k_|K|}.
Result: matrix form C of an encoder enc.
1  Set C_{i,j} = c_j, ∀1 ≤ i ≤ |K|, 1 ≤ j ≤ |M|;
2  for j ← 1 to |M| do
3      for i ← 1 to |K| do
4          if c = C_{i,j} is a conflict then
5              Find the best swap C̄[c ↔ c'] with any
               element c' on row i;
6              Find the best transformation C̄[c ← c_ν];
7              if H(C̄[c ← c_ν]) < H(C̄[c ↔ c']) then
8                  C ← C̄[c ← c_ν]
9              else
10                 C ← C̄[c ↔ c'])
11             end
12         end
13     end
14 end
15 Return C;
```

Algorithm 1: INCROW

rows correspond to values of the key and the columns to the values of the message, as in Table I. The cell $\mathbf{C}_{i,j}$ contains the ciphertext symbol produced when the key is k_i and message is m_j. For brevity we denote by $H(\mathbf{C})$ the entropy of the ciphertext induced by the encoder function represented by $\mathbf{C}$.

The algorithms initialize $\mathbf{C}$ by assigning to each value in the column m_j the symbol c_j, for $1 \leq j \leq |\mathcal{M}|$. The resulting ciphertext would have exactly $H(C) = H(M)$ and thus achieve max-equivocation, however it is not a valid encoder function since the same symbol is repeated more than once in each column. The algorithms then transform it into a valid encoder function while greedily trying to increase the entropy of the ciphertext by the smallest possible amount.

For $\mathbf{C}$ to be an encoder function, it must be that in each row and column of $\mathbf{C}$ each symbol appears at most once. We call a symbol $c = \mathbf{C}_{i,j}$ a *conflict* if it violates this property, i.e. if c appears more than once in row i or column j.

The algorithms proceed by reducing the number of conflicts in $\mathbf{C}$, and returning $\mathbf{C}$ when there are no conflicts left and consequently $\mathbf{C}$ represents an encoder function. A conflict can be replaced in two different ways: by *swapping* it with a symbol in a different position in $\mathbf{C}$ or by *transforming* it to another symbol.

Swapping $c = \mathbf{C}_{i,j}$ and $c' = \mathbf{C}_{i',j'}$ means that we simultaneously set $\mathbf{C}_{i,j} \leftarrow c'$ and $\mathbf{C}_{i',j'} \leftarrow c$. A swap is considered valid only if c is not a conflict in position (i', j') and c' is not a conflict in position (i, j). The algorithms perform only valid swaps, so each swap strictly reduces the number of conflicts in $\mathbf{C}$. We will denote the result of the swap with $\mathbf{C}[c \leftrightarrow c']$. A *best swap* operation $\overline{\mathbf{C}}[c \leftrightarrow c']$ for a given conflict c is a valid swap such that $\forall c'' \in \mathbf{C}$. $H(\overline{\mathbf{C}}[c \leftrightarrow c']) \leq H(\mathbf{C}[c \leftrightarrow c''])$.

Transforming $c = \mathbf{C}_{i,j}$ means finding a symbol c_ν that does not appear in row i nor in column j and setting $\mathbf{C}_{i,j} \leftarrow c_\nu$. A transformation reduces the number of conflicts by one. We will denote the result of the transformation with

```
Data: message space M = {m_1, m_2, ..., m_|M|},
      probability distribution ρ(M), key space
      K = {k_1, k_2, ..., k_|K|}.
Result: matrix form C of an encoder enc.
1  Set C_{i,j} = c_j, ∀1 ≤ i ≤ |K|, 1 ≤ j ≤ |M|;
2  while C is not an encoder function do
3      Let minC ← C, minH ← ∞;
4      foreach conflict c do
5          Find the best swap C̄[c ↔ c'] with any element
           c' ∈ C;
6          Find the best transformation C̄[c ← c_ν];
7          if H(C̄[c ← c_ν]) < H(C̄[c ↔ c']) then
8              Let candidateC ← C̄[c ← c_ν];
9          else
10             Let candidateC ← C̄[c ↔ c'];
11         end
12         if H(candidateC) < minH then
13             minC ← candidateC;
14             minH ← H(candidateC);
15     end
16     C ← minC;
17 end
18 Return C;
```

Algorithm 2: EXTENSIVE

$\mathbf{C}[c \leftarrow c_\nu]$. Note that no encoder function has more than $|\mathcal{M} \times \mathcal{K}|$ different symbols. A *best transformation* operation $\overline{\mathbf{C}}[c \leftarrow c_\nu]$ for a given conflict c is a transformation such that $\forall c' \in \{c_1, c_2, ..., c_{|\mathcal{M}\times\mathcal{K}|}\}$. $H(\overline{\mathbf{C}}[c \leftarrow c_\nu]) \leq H(\mathbf{C}[c \leftarrow c'])$.

All algorithms always terminate. We introduce them and explain how they differ.

The INCROW algorithm scans $\mathbf{C}$'s columns in increasing order of probability, and each column from top to bottom. When it finds a conflict c, it considers the best swap $\overline{\mathbf{C}}[c \leftrightarrow c']$ of c with elements on the same row and the best transformation of c with a different symbol $\overline{\mathbf{C}}[c \leftarrow c_\nu]$. If $H(\overline{\mathbf{C}}[c \leftarrow c_\nu]) < H(\overline{\mathbf{C}}[c \leftrightarrow c'])$ it performs the best transformation, otherwise the best swap. The pseudocode of INCROW is described in Algorithm 1.

The DECROW algorithm is equivalent to INCROW except that the columns are scanned in decreasing order of probability. The pseudocode of DECROW is the same as the one described in Algorithm 1 except that the order of the **for** on line 2 is $j \leftarrow |\mathcal{M}|$ *to* 1.

The RANDROW algorithm is equivalent to INCROW except that the columns are scanned in a random order. The pseudocode of RANDROW is the same as the one described in Algorithm 1 except that the order of the columns of $\mathbf{C}$ is randomized after line 1.

The EXTENSIVE algorithm considers all conflicts in $\mathbf{C}$ and for each one the best swap and the best transformation, and then it performs the swap or transformation operation that is the best among all possible conflict resolutions (ties in favor of swaps). It repeats this step until there are no conflicts left. The pseudocode of EXTENSIVE is described in Algorithm 2.

Lemma 6: Algorithms INCROW, DECROW, RANDROW and EXTENSIVE terminate.

Proof: Since each iteration of the main cycle of each algorithm reduces the number of conflicts. ■

In the next section we perform experiments on the proposed heuristics to determine which one is the most effective in producing an encoder function approaching max-equivocation.

VI. Experimental Results

A. Methodology

Unless otherwise specified, the results are from running tests on randomly generated message spaces and probability distributions. In general the message space sizes are randomly chosen between 3 and 50, and two algorithms (ProbA and ProbB) are used to generate the probability distributions. (Message spaces with $H(M) < 1$ are discarded and new distribution found.) The key space size is randomly chosen to be between 2 and the maximum $|K|$ such that $H(K) < H(M)$.

The ProbA algorithm splits the probability available x (initially 1) randomly into two parts x_1 and x_2, and then recursively calls itself with these parts of the two halves of the message (sub-)space. This continues until the subspace is a single message that is assigned the whole probability.

The ProbB algorithm splits the probability available into two parts, assigns the first part to the current message (initially the first), and then continues with the remaining probability and remaining messages. This tends to generate more skewed probability distributions than ProbA.

Once the message space size, message distribution, and key space size have been chosen, each algorithm is run in turn on the same initial state, with the data recorded after each encoder function has been created.

The code to perform these test is written in Java 1.8 and run on Linux 3.13 64-bit kernel on an Intel Core i7-3720QM 2.60GHz CPU with 8GB of RAM. The code used to run these experiments is available upon request.

B. Comparing Algorithms

The tests generate random message spaces, message distributions, and key sizes; determining the domain for an encryption function. The different algorithms for generating encryption functions are then all run on the same random examples and their outputs compared, in particular:

Win% The percentage of times each algorithm generates the lowest entropy (may sum to over 100% since more than one algorithm can generate the same lowest entropy).

Δ The average percentage of the message entropy that is lost; that is the average $\frac{(H(C)-H(M))\times 100}{H(M)}$.

Runtime The average runtime for the algorithm in milliseconds.

The results for comparing IncRow, DecRow, RandRow, and Extensive over 450,000 tests (equally split across the two probability generation algorithms) are shown in Table II. The results indicate that the DecRow algorithm performs the best in all categories, followed by RandRow, then IncRow and finally Extensive. In particular the results show that Extensive performs by far the worst, with much higher runtimes yielding the lowest win percentage and only negligibly improved loss of message entropy over the next worse IncRow. Although IncRow is not significantly slower, the low win percentage and highest entropy loss suggest that despite greediness not being optimal, beginning with the smallest probabilities is worse than a random selection. Thus, although DecRow and RandRow are not optimal, or even always better[1] than IncRow and Extensive, they are worthy of further consideration.

TABLE II. Comparison of IncRow, DecRow, RandRow, and Extensive

	Win%	Δ	Runtime
IncRow	9.3462	11.04	0.3978
DecRow	80.5007	9.00	0.2909
RandRow	21.5667	10.31	0.3608
Extensive	8.4744	11.00	172.4039

We now consider a more refined analysis of the DecRow and RandRow algorithms to contrast their results and study the impacts of other factors.

The following results consider the DecRow and RandRow algorithms run while comparing message sizes. Table III shows the results for 100,000 tests (equally split across ProbA and ProbB) with fixed message size (and random key space size such that $1 \leq H(K) < H(M)$).

The win percentage appears to be slightly dropping for both algorithms, however this is due to them both achieving the same result less often. The indication here is that DecRow provides a better solution approximately 80% of the time, and RandRow approximately 20%. The loss of entropy is also decreasing as the message size increases, suggesting that with more message space to play with the algorithms can find encoder functions that approach the optimal and theoretical best (i.e. $H(C) = H(M)$). These are graphed in Figure 2 along with other experiments.

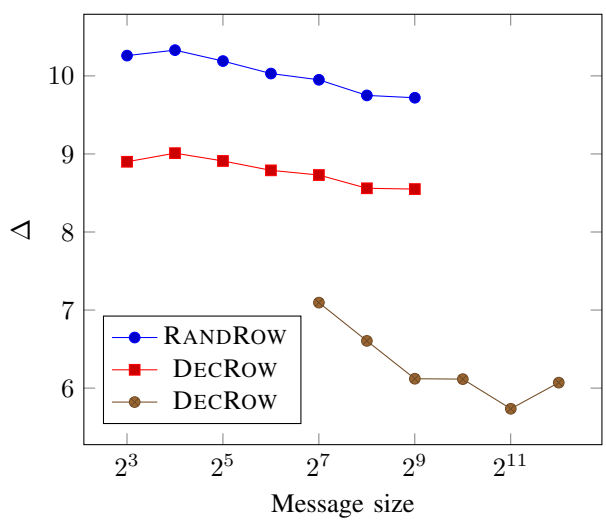

Fig. 2. Comparison of Δ's for DecRow and RandRow versus message size

The other line in Figure 2 shows the DecRow algorithm with only 200 tests using only the ProbA generation algorithm

[1] Although not clear in Table II there are experiments where either of IncRow or Extensive have been the sole winning algorithm.

TABLE III. COMPARISON OF DECROW AND RANDROW BY MESSAGE SIZE

	DECROW Win%	DECROW Δ	RANDROW Win%	RANDROW Δ
8	82.041	8.90	35.402	10.26
16	81.741	9.01	20.836	10.33
32	81.715	8.91	18.462	10.19
64	81.112	8.79	18.914	10.03
128	80.700	8.73	19.325	9.95
256	81.033	8.56	18.986	9.75
512	80.455	8.55	19.569	9.72

(PROBB gives very skewed results that are unrepresentative for large message spaces). Note that no win percentage is calculated as only the DECROW algorithm was run.

VII. DIFFERENT ENTROPY MEASURES

Recently it has been shown that Shannon entropy measures the effort for decrypting the ciphertext by asking arbitrary binary questions, which is not a general security scenario [10]. Other measures based on different entropies have been introduced. Smith's *min-entropy* [11] has been shown to measure the resistance of the cryptosystem against one-time attacks in which the attacker has a single chance to guess the message correctly. Other entropy measures include g-leakage, that parametrizes min-entropy with a given gain function [12], and guessing entropy, that quantifies the effort required to decrypt the ciphertext with equality tests [10].

The results presented in this paper can be re-derived using any other entropy measure, generating alternate definitions of max-equivocation that consider different types of attackers. Modification of the heuristics to use a different type of entropy is trivial. While in general we do not expect that the optimal encoder function for a type of entropy to be optimal also for other measures that are not refinements of the first, we leave this as an open question.

VIII. CONCLUSIONS

We have presented max-equivocation, a generalization of Shannon's perfect secrecy condition that can also be established when the entropy of the key is smaller of the entropy of the message. Message equivocation measures the number of different messages that can be decrypted from the ciphertext, and max-equivocation holds when this number is maximized. Max-equivocation is achieved when the encoder function minimizes the entropy of the ciphertext.

We have considered the problem of finding an encoder function when the distribution over the message space is non-uniform and the distribution over the key space is uniform.

We have shown that having a ciphertext space larger than the message space can increase the effectiveness of encoder functions, proving the general non-optimality of encoder functions that correspond to Latin rectangles. Also, we have shown that in general it is impossible to give an encoder function that achieves max-equivocation.

We have compared four heuristics to efficiently derive an effective encoder function. We established that the DECROW algorithm works better than the others under our experimental parameters, consistently providing encoder functions that are only a few percentage points worst than the theoretical optimum and in less time than the other heuristics. Finally, we have shown that the effectiveness of the encoder functions constructed by our heuristics improves with the size of the message space, suggesting that they could be used on real instances of the problem.

A. Related Work

Other works have considered variations or alternatives to perfect secrecy in shared-key cryptosystems. Csiszár and Körner [13] introduce *ϵ-secrecy* as a relaxing of the requirement of perfect secrecy to $I(M;C) \leq \epsilon$ for a given ϵ. Ho et al. [14] consider the bounds required for perfect secrecy in the sense of Shannon, while considering key reuse, and parameterise by the entropy of the message space. Russell and Wang [15] consider *entropic security* as an alternative, that considers min-entropy bounded statistical security measures and smaller key spaces.

The design of encryption functions, or their components such as **S**-boxes, has been widely studied [3], [4], [5], particularly when designing optimal systems. The results here suggest that when the message space (or inputs to an **S**-box) are non-uniformly distributed then the result cannot ensure max-equivocation. Further, finding optimal functions should not be limited merely to Latin squares/rectangles or Quasigroups, since higher max-equivocation can be achieved without such tight restrictions.

REFERENCES

[1] C. E. Shannon, "Communication Theory of Secrecy Systems," *Bell System Technical Journal*, vol. 28, 1949.

[2] ——, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, pp. 379–423, Jul. 1948.

[3] Y. Wu, J. Noonan, and S. Agaian, "Dynamic and implicit latin square doubly stochastic s-boxes with reversibility," in *SMC 2011*, Oct 2011, pp. 3358–3364.

[4] A. Mileva and S. Markovski, "Quasigroup representation of some feistel and generalized feistel ciphers," in *ICT Innovations 2012*, 2013, vol. 207, pp. 161–171.

[5] Y. Wu, Y. Zhou, J. P. Noonan, and S. Agaian, "Design of image cipher using latin squares," *Information Sciences*, vol. 264, no. 0, pp. 317 – 339, 2014.

[6] T. Cover and J. Thomas, *Elements of Information Theory*, ser. A Wiley-Interscience publication. Wiley, 2006.

[7] A. Bruen and M. Forcinito, *Cryptography, Information Theory, and Error-Correction: A Handbook for the 21st Century*. Wiley, 2011.

[8] D. Welsh, *Codes and Cryptography*. New York, NY, USA: Clarendon Press, 1988.

[9] F. Reza, *An Introduction to Information Theory*, ser. Dover Books on Mathematics Series. Dover, 1961.

[10] B. Köpf and D. A. Basin, "Automatically deriving information-theoretic bounds for adaptive side-channel attacks," *Journal of Computer Security*, vol. 19, no. 1, pp. 1–31, 2011.

[11] G. Smith, "On the foundations of quantitative information flow," in *FOSSACS 2009*, ser. LNCS, L. de Alfaro, Ed., vol. 5504. Springer, 2009, pp. 288–302.

[12] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith, "Measuring information leakage using generalized gain functions," in *CSF 2012*, S. Chong, Ed. IEEE, 2012, pp. 265–279.

[13] I. Csiszár and J. Körner, "Broadcast channels with confidential messages," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 339–348, 1978.

[14] S.-W. Ho, T. Chan, and C. Uduwerelle, "Error-free perfect-secrecy systems," in *ISIT 2011*, July 2011, pp. 1613–1617.

[15] A. Russell and H. Wang, "How to fool an unbounded adversary with a short key," in *EUROCRYPT*, 2002, pp. 133–148.