Entropy Augmentation

The "Key" to PQC (Post Quantum Security) Encryption

Helder Figueira

Incrypteon, United Kingdom;

Abstract. Claude Shannon determined what it means to be "secure" in 1945, with his 2 foundation papers on cryptography, namely "A Mathematical Theory of Cryptography" (1945)[1] and "Communication Theory of Secrecy Systems"[2] (1949). He mathematically defined 2 "secure" conditions - "perfect secrecy" and "ideal secrecy" and these definitions are applicable to digital cryptography. In this paper, we revisit Shannon's cryptographic principles, and explain in simple terms, (1) what it means for a system to have "secrecy" and be "secure" and what conditions make it "Quantum-secure", (2) how the "secrecy" properties of any system can be objectively measured, plotted and visualized, (3) why One Time Pads are "Quantum -secure", (4) the difficult problem with using OTPs today in a digital context, and (5) the hidden critical "secrecy" problem (Entropy Depletion) that rendered almost all encryption systems "insecure" when they used a key that was shorter than the message, and (6) how the OTP problem and the Entropy Depletion problem were solved in 2016.

This paper then covers the solution to both the OTP and Entropy Depletion problems using a concept called Entropy Augmentation, where the "secrecy" in any system can be Augmented (increased) at a faster rate than the message length. We provide an example of a patented[3] Entropy Augmentation implementation called Archaios™ from Incrypteon™ and describe why Archaios™ is arguably the greatest advance in

[1] Shannon, Claude. "A Mathematical Theory of Cryptography", Memorandum MM 45-110-02, Sept. 1, 1945, Bell Laboratories.https://archivesspace.mit.edu/repositories/2/archival_objects/206272

[2] C. Shannon, "Communication Theory of secrecy systems." Bell Systems Technical Journal, 28(4), 656-715 (1949). https://en.wikipedia.org/wiki/Communication_Theory_of_Secrecy_Systems

[3] US Patent US20190363877A1 - Equivocation Augmentation Dynamic Secrecy System

cryptography since the One-Time Pad. We finish by listing several cryptographic properties of Archaios™, including it being: (1) a Chaos Security System – unpredictable and Polymorphic and Secure against all Brute Force and known cryptanalytic attacks, (2) Post Quantum-Secure and Quantum + AI Secure, (3) Crypto Agility - integration with NIST approved FIPS 140 and PQC FIPS 203/4/5 solutions, (4) integrated Zero Trust Architecture, (5) Defence in Design and in Development and (5) its ease of deployment to protect existing security implementations.

1 Introduction

At Incrypteon™ we have a very simple design philosophy for all our security products. If we are to build the most secure security products in the world ("Quantum + AI Secure"), we must build them according to the highest available scientific and mathematical cryptographic standards (Information-Theoretic Security). Cryptanalysis is by its very nature the implementation of a scientific-method approach to "break" an encryption system, not to decrypt any specific ciphertexts, but to define solutions which allow all or most encipherments of a specific secrecy system to be "broken".

We use the highest security design standards available – information-theoretic security – that was defined by Shannon in his 2 cryptographic papers on cryptography in 1945 and 1949. In keeping with Shannon's terminology, we will strictly use the word "secrecy" to mean "secure" or "security".

Shannon proposed the following simple principles to be adopted when designing or evaluating any secrecy system for "secrecy":

- **Assume the assailant has complete knowledge of the system**
- **Assume the assailant has infinite computing and time resources**
- **Measure "Secrecy" objectively using Conditional Entropy**
- **Visualise "Secrecy" by plotting Conditional Entropy and Unicity Points on a graph.**
- **Practical Systems that have no "secrecy", so are disqualified. They can never be "Quantum-secure", only "Quantum-safe".**
- **Determine the maximum theoretical weaknesses of the Secrecy system. All systems have weaknesses.**
- **There are no compromises to the above principles.**

In this paper we aim to also dispel several myths that are encountered in cryptographic papers and cryptographic security related literature.

- **Myth #1 – Unbreakable systems are impossible to create** - ANY encryption system which encrypts a message that is no longer than the "Unicity Point" is "unbreakable" and "secure" because that specific encipherment is "secret". It has the same characteristics as a One-Time Pad.
- **Myth #2 – The security of an encryption system depends on the internal mechanics of the system –** This is false (with one exception). Do not confuse the "secrecy" properties of the system with the "secrecy" properties of a specific encipherment. The secrecy of an encipherment depends only on the size of the key used (Key entropy), the redundancy of the language used in the message, and the length of the message to be encrypted. The mechanics of the secrecy system are completely irrelevant.

These and other myths will be dispelled as we cover the basics of cryptography as defined by Shannon.

2 Shannon's Cryptographic Principles

In this section we will briefly cover Shannon's cryptographic principles as described in his two foundation papers.

Claude Shannon[4] is considered the father of the Information Age and of Modern Cryptography. In 1945 and 1949 he published two academic papers on cryptography - "A Mathematical Theory of Cryptography" (1945) and "Communication Theory of Secrecy Systems" (1949) where he laid out the fundamental principles to modern cryptography.

2.1 Encryption / Decryption are Message Transformations

As can be seen below, an encryption is when a **message M** is transformed into a **ciphertext C** using a **key K** and a **transformation process T**. Decryption is the reverse transformation. If the same key is used it is called "symmetric" encryption, and if the keys are different but mathematically related, it is called "asymmetric" encryption.

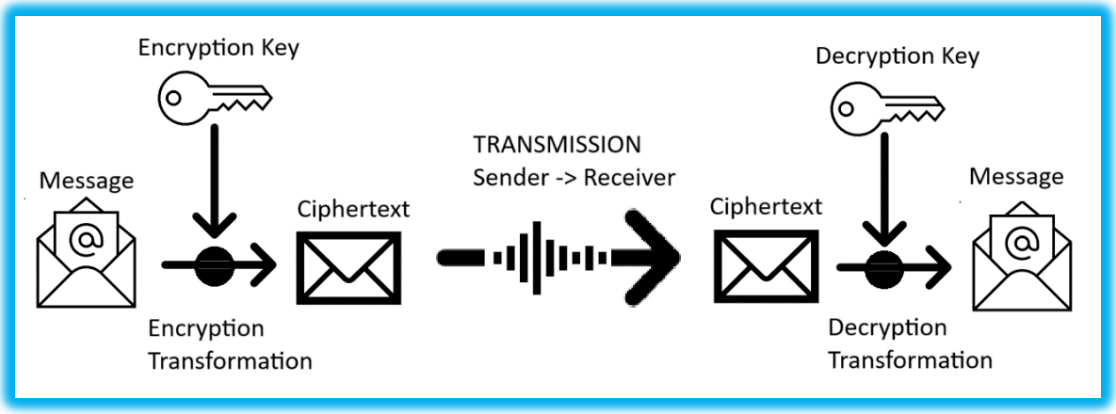

Fig. 1 Symmetric Encryption Example – Same Key

2.2 General Cryptographic Terms

The following cryptographic terms and definitions are used in this paper

- **Key K –** the key used for encryption
- **Message M –** the message used in encryption
- **Transformation TE & TD –** the specific cryptographic operation T with T**E** for encryption and T**D** for decryption

- **Key Entropy H(K)** - the logarithm of the set of all possible keys from which the key was chosen. A 256-bit key is selected from a set of 2^{256} possible keys. A certain length of key entropy can only protect a certain length of message entropy.
- **Message Entropy H(M)** – the logarithm of the set of all possible messages and language from which the message was selected. This is the amount of information in the message and is measured in bits.

[4] https://en.wikipedia.org/wiki/Claude_Shannon

- **Message Types –** there are generally 4 main message types we will consider. and they differ mainly in terms of the amount of information they have per word character (we will use 8 bit characters in our example (1 character is 1 Byte, 8 bits per byte).
    - **No Information - Known Plaintexts -** these messages have no information, since they are predictable, and so H(M) = 0 x message length in characters.
    - **Low Information - Normal Language Messages –** These have about 1.5 bits per character, so H(M) = 1.5 x message length.
    - **Medium Information – Small Alphabet Language Messages – Morse Code –** These have about 4 bits per character, so H(M) = 4 x message length.
    - **High Information – Random Messages – Entropy –** These have full 8 bits per character, so H(M) = 8 x message length.

- **Redundancy R** – Redundancy is the inverse of information. It is the entropic percentage of messages that have no information. To calculate the redundancy in a message we use the following equation:

$$\boldsymbol{R} = \frac{\text{total } H(M) - H(M)}{\text{total H(M)}} \times 100$$

where total H(M) is the total information per character, and H(M) is the information per character of the message. So, for an English message expressed in UTF-8

$$\boldsymbol{R} = \frac{8 - 1.5}{8} \times 100$$ = 81.25% and the information percentage per character is 18.75%

- **Conditional Entropy of Key HE(K) -** This is the residual key entropy (the logarithm of the keys which resulted in valid decryptions) after a brute-force search of all keys against the ciphertext.
- **Conditional Entropy of Message HE(M) –** This is the residual message entropy (the logarithm of the number of valid messages found) after a brute-force search of all keys against the ciphertext. If either HE(K) or HE(M) are reduced to zero, then the encipherment is insecure.
- **Unicity Point –** A point along the length of a ciphertext, where under a brute-force attack, an assailant can reduce the conditional entropy of keys. The length of ciphertext required, that will result in a single uniquely valid message if a comprehensive key search is performed on the ciphertext.

Myth #3 – The secrecy of a secrecy system is determined by the size of the key used only – This is false. The secrecy of a secrecy system depends on two factors, the size of the key and the redundancy of the language in the message used. Cryptographic secrecy uses the same principles as is found in efficiency scenarios where resources are limited.

- **Using a car analogy –** If we consider that key entropy is the "fuel" of encryption**,** the message length is the journey distance, redundancy is the difficulty of the trip, then we can calculate the range of the car. If the journey distance is within the range of the

car, then the journey is secure. It is not the car that should be secure, it's the journey. The same thing with cryptography.

2.3 Shannon's Secrecy Requirements – are the Requirements for "Quantum-secure" systems.

In his papers, Shannon specified several "requirements" which need to be observed when analysing and designing secrecy systems: To be "Quantum-secure" a system MUST strictly adhere to these requirements. We will cover these in a bit more detail before we visualise secrecy.

- **No Assumptions of Security are allowed -** Only assumptions which benefit the attacker are allowed.

- **Assume the assailant has gained complete knowledge of the system** – Do not underestimate the assailant's knowledge of the system. When the Germans changed their Enigma machines by secretly adding another rotor, it abruptly stopped codebreaking at Bletchley Park during World War 2. It took the British just 9 months to discover the change, and mass decrypt Enigma ciphertexts.

- **Assume the assailant has infinite computing and time resources** – Do not underestimate the assailant's abilities. Assume that an infinite number of maths problems are instantly solvable, and that comprehensive searches are instant. Security is measured under the maximum theoretical capabilities of the assailant. In World War 2, codebreaking led to the advances in computing we enjoy today, because they had to create a machine to break the encryptions. Today, this adversary is represented by Quantum Computing (infinite time and computing resources) and Cryptanalytical AI systems (Infinite Analytical resources). Assume that the assailant will attack all encipherments of the system using brute-force, to immediate completion.

- **Measure "Secrecy" objectively using Conditional Entropy** – Conditional entropy is the logarithm of the number of valid keys or messages found in a comprehensive search on a specific ciphertext. It is the primary indicator of the secrecy of a system. It is the additional valid messages found after a comprehensive search which give the encipherment any secrecy, in that if more than 2 valid messages are found, it is not possible to distinguish which is the correct one. An encipherment which can be reduced to a single valid message had no secrecy to start with.

    - If the assailant finds 4 valid keys/messages, then Conditional entropy is 2, since $4 = 2^2$,
    - If 256 valid keys/messages are found, then conditional entropy is 8, since $256 = 2^8$.

  The size of the search (the entropy of key) is irrelevant. Subjective measurements of security such as peer reviews and opinions are invalid – are you measuring system secrecy or the expert's security knowledge?

- **Display "Conditional Entropy" and "Unicity Points" Graphically** – Conditional Entropy and Secrecy can be graphed, plotted and visualised. Taking the size of the key and the redundancy of the language of the message in the ciphertext calculate the Unicity Point. When a ciphertext is attacked, the assailant will try all keys and look for valid decipherments. As he eliminates keys, the Conditional Entropy of Key will decrease, to the point where only 1 valid message will remain. Unicity Points indicate the point along the length of the ciphertext where conditional entropy of key will be reduced to zero. Specifically, Unicity Points indicate the length of ciphertext that can be secured by the amount of Key Entropy used. It is the point where the encipherment runs out of entropy "fuel".
    - Shannon defined the term "**Equivocation**" to indicate the length of Ciphertext up to the Unicity Point. It is the amount of ciphertext that is protected by the entropy of the key used.

- **Verify that Encipherments and Systems meet the Conditions for "Secrecy" -** Encipherments are "secret" under the following conditions ONLY.
    - **For Ciphertexts**
        - if the **Message Ciphertext is longer than the Equivocation (Unicity Point)**, the encipherment is "secret" and secure.
        - if the **Message Ciphertext is shorter than the Equivocation (Unicity Point)**, the encipherment ran out of protective key entropy and is insecure.
    - **For Secrecy Systems**
        - The secrecy of a secrecy system depends mainly on the length of the ciphertext.
            - For ciphertexts as long as the key used they are ALL secret (they are One-Time Pads) – "perfect secrecy"
            - For ciphertexts longer than the key, but shorter than the Equivocation they have "ideal secrecy".
            - For ciphertexts longer than the Equivocation, they are all insecure.

- **There are no exceptions to the above requirements –** Shannon also mentioned the use of "Practical Systems" – systems which have no "secrecy" and are insecure, and which rely on the difficulty of certain mathematical problems to encrypt messages. These include ALL the symmetric and asymmetric encryption systems used today. Shannon stated that these could be practically used if two caveats were followed:
    - They should only be used in cases where even if ciphertexts were "solved", by the time they were solved, the contents of the message were not "actionable".
    - They should only be used if it could be proven that a fast solution to the problems did not exist.

The above requirements were considered "impossible" to attain by the cryptographic community. So impossible that they created a (new) concept called "Practical" or "Semantic" Security was created. All current encryption technology used in the commercial security industry is based on "Practical" security. A "pseudo-secure" secrecy system can never attain "Quantum-secure" status. Neither can any system which uses "pseudo-secure" components. Shannon stated that a composite system is only

as “secure” as its weakest component. This is why we get “Quantum-safe” systems - they are not Quantum-secure.

2.4 Cryptographic Analysis of “Secrecy”, “Conditional Entropy” and “Equivocation”.

We will now take Shannon’s concepts and requirements mentioned above and apply them in a graphical context in what he called an “Equivocation Graph”.

In the Equivocation Graph below (which applies to ALL secrecy systems which use a 32bit key **H(K)** to encrypt messages of any length with 8-bit characters **M[0]…M[n]**) and plot Conditional Entropy of Keys **HE(K1).. HE(K4)**, and Conditional Entropy of Message **HE(M1).. HE(M4)** for 4 message types **M0, M1, M2, M3**. The message types are:

- **M0 - Known Plaintext** – 100% Redundancy
- **M1 - Normal Plaintext** – 81% Redundancy
- **M2 – Small Alphabet Plaintext** – 50% Redundancy
- **M3 – Random String** – 100% Redundancy

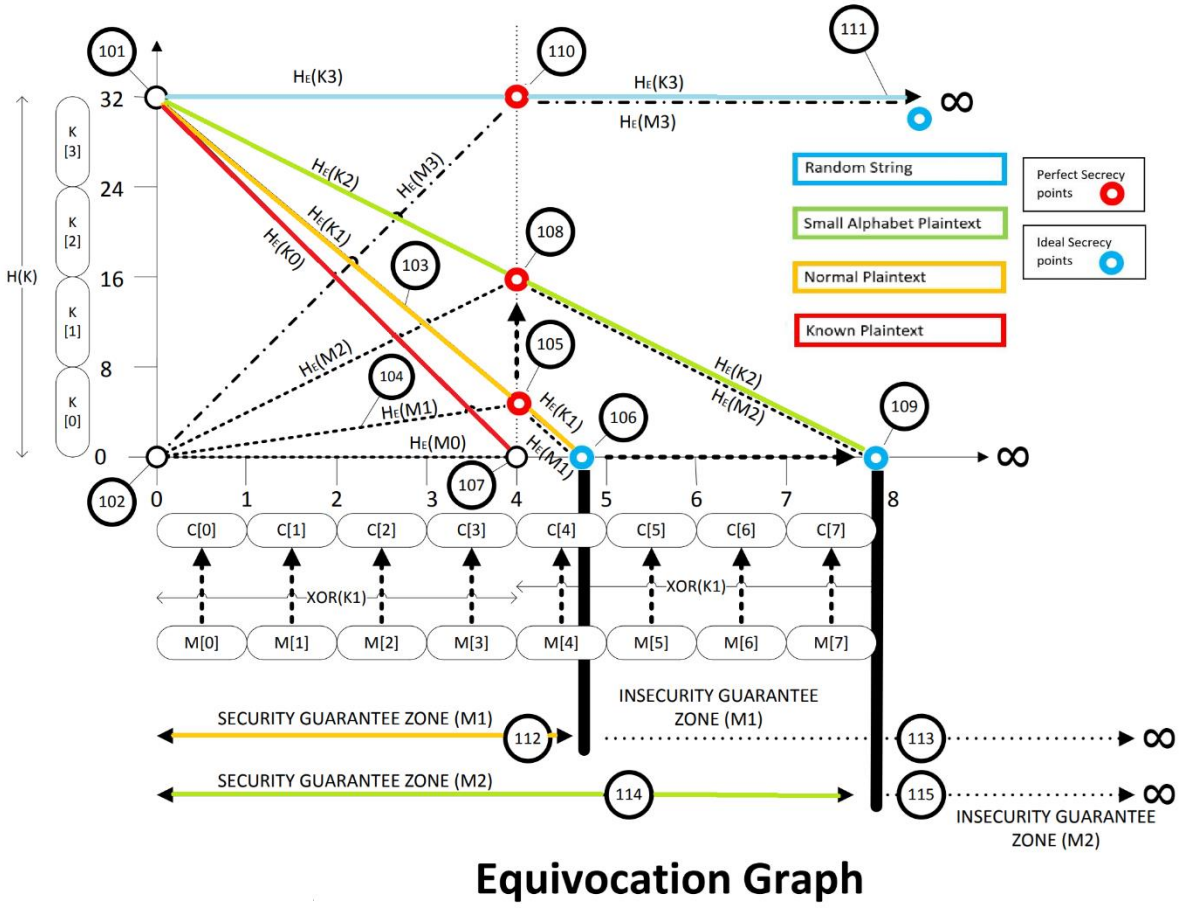

***Figure 2**: Equivocation Graph – Key (32bits), Characters (8bit)*

The graph can be analysed in the same manner as any other efficiency analysis graph which considers the impact of inefficiencies on a limited resource to perform a certain function.

Note that the graphical display of secrecy is explained in Shannon's papers. This is entirely Shannon's work. Absolutely brilliant stuff. All we are doing is applying his principles, which have been largely ignored by the cryptographic community since if you apply his principles, it becomes very difficult to explain exactly how AES-256 can be secure objectively against Quantum + Novel AI cryptanalysis if the ciphertext is longer than say, 40 characters. Let's proceed:

- **Key Entropy H(K)** - is shown on the left axis and starts with 32 bits **(101)** for all message types, and all encryption systems. Note that the key can only cover 4 characters of message and will be repeated every 4 characters.
- **Message Length M[0]…M[n] –** is shown on the bottom axis, in characters.
- **Conditional Entropy of Key HE(K)** - this is shown by the coloured lines which represent the rate or slope of "loss" of Key Entropy for each of the 4-message type ciphertexts (M0...M3), as keys are eliminated when the ciphertexts are attacked.
- **Conditional Entropy of Key HE(M)** - he dotted lines represent the associated increase in Conditional Entropy of Message, as the assailant finds valid messages. Note that at the 4th character mark (107) the HEK and HE(M) of the various message types meet. He(M) will decrease with HE(K) after the 4th character.
- Let's look at the 4 message types:
    - **M0 Known Plaintext** - with 0% information and 100% redundancy per character its Conditional Entropy of Key H(K0) hits the lower axis (Unicity point) at the 4th character mark (107).
        - It has no secrecy. It never had any to start with.
    - **M1 Known Plaintext** - with 18.7% information and 81.3% redundancy its Unicity point is just short of the 5th Character mark (106).
        - ciphertexts up to 4 characters have "Perfect secrecy" in that at point **105** all possible messages will be found. This is the basis for the proof of One-Time Pads.
        - ciphertexts between 4 and 5 characters long will have "ideal secrecy". Subset of all multiple messages will be found.
        - ciphertexts longer that 5 characters will be insecure.
    - **M2 Small Alphabet Plaintext** - with 50% information and 50% redundancy its Unicity point is at the 8th Character mark (109).
        - ciphertexts up to 4 characters have "Perfect secrecy" in that at point 108 all possible messages will be found.
        - ciphertexts between 4 and 8 characters long will have "ideal secrecy".
        - ciphertexts longer that 8 characters will be insecure.
    - **M3 Random String** - with 100% information and 0% redundancy (more like 99.9% info and 0.1% redundancy) its Unicity point will tend towards infinity.
        - ciphertexts up to 4 characters will have "Perfect secrecy" in that at point 105 all possible random messages will be found.
        - ciphertexts longer than 4 characters long will have "ideal secrecy".

What have we learnt:

- All One-Time Pad encipherments have "perfect secrecy". They are Quantum-secure.

- Any encipherment by any encryption system where keys are as long as the message have “perfect secrecy”. They are Quantum Secure.
- Any encipherment by any encryption system up to a certain length (determined by the redundancy in the language of the message) with have “ideal secrecy”. They are Quantum Secure.
- **Conditional Entropy Decay Problem** - Any encipherment by any encryption system beyond a certain length (determined by the redundancy in the language of the message) will be insecure. It cannot be Quantum Secure.
- Any system which aims to be “Quantum-secure” cannot use components which are not Quantum secure, since the graphic principles are applicable to when multiple systems are combined. It must use One-Time Pads.
- **OTP Digital Implementation Problem** – One-Time Pads are impossible to apply in a digital context.
    (a) Keys must be random and infinite in length
    (b) Keys must be securely generated, transmitted, secured and destroyed. Using what? Another One-Time Pad? And how is that secured?
- **There are two problems which must be solved for any Quantum-secure system to be viable:**
    - **You MUST solve the Conditional Entropy Decay Problem.**
    - **You MUST solve the OTP Digital Implementation Problem.**

2.5 Block Ciphers – Why they can never be “Quantum-Secure”.

These are the authors personal comments.

There have been comments in the cryptographic community that block ciphers, such as those based on Feistel networks like AES-256, are somehow “Quantum-secure”. As we have just noted in the previous section, they are not classically “secure” if their ciphertexts exceed a certain length (40 bytes for AES-256). They are too predictable (deterministic in operation and output) and have a “flawed” classical cryptographic structure – it’s exactly how an assailant would specify the design of an encryption algorithm so that it would make his job as easy as possible. All operations are predictable and deterministic, only M and K change to produce C, making block ciphers easy to attack if they are perceived as black boxes.

A block cipher is a 3-D construct that has inputs of a block of message M, and a block of key K and outputs a block of ciphertext C using a transformation T. Due to this, messages M, keys K and ciphertext C can be plotted in a 3-D cube with respect to T, having co-ordinate axes M, K and C. Where Key extensions are used (such as initialisation vectors) a 4^{th} dimension is added, or the Ciphertext dimension is extended.

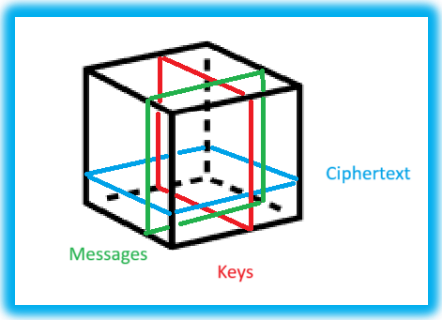

Fig 3: Block Cipher Analysis 3-D Cube

If the block cipher is “pure” (See Shannon’s definition) or not makes no difference, each individual point in the 3D matrix will have a unique set of M, K and C values. To begin with, there is no need to construct the entire cube for a specific encryption algorithm, by trying out all M, K and C permutations. For example, only valid messages need to be added. There is no need to try and “crack” the encryption by trying to study for weaknesses in the complicated internal mechanics – just treat the encryption system as a black box. All that needs to be done, is to compile the 3D cube of the M, K, and C results. And there are variations to cube construction, smaller cubes, and of course, database indexing allows for searches of cube elements.

After many combinations it is possible for a human to derive a pattern, not only in 3D, but also with specific regard to a specific Message, or Ciphertext or Key. There must be a pattern or else it would not work in practice. If you take the 2D plane for a specific message, you will see the distribution pattern for Keys and Ciphertext. It may look random to a human, but it is not. Finding patterns in complex data, is what AI systems excel at, and they do this flawlessly. This is what AI has been used for in military cryptanalysis for over 30 years now.

Once you have found the pattern, you can create the linear algebraic expression which allows you to determine the key with respect to a given ciphertext and message. So technically, for a 256bit encryption block system you only must look for around 4.2 billion message combinations if you know the message. If you have predictable elements in your message such as a “header” which is required by standard document formats, only about 1000 messages need to be searched. Document standards have caused more damage to security, since they make message content predictable – they destroy entropy.

3 The Solution – Next-Generation Encryption - Archaios.

Archaios ™ is a Next-Generation "Quantum-Secure" encryption system because:

- It is at its core, a solid "Quantum-Secure" point-to-point (P2P) security building block. It can be used as the foundational building block to create advanced Transport Layer, Application Layer and Distributed Security encryption solutions.
- It represents a revolution in basic cryptographic design – from being "2D, complicated, static, passive, predictable, analysis-friendly" to "3D+, simple, dynamic, active, chaotic, analysis-resistant".
- It complies with Shannon's "secrecy" requirements with respect to an assailant with infinite computing and time resources ("Quantum-secure"), to which we also add "infinite analytical resources" ("AI-secure")
- Its core operation extends Shannon's entropy to the Second Law of Thermodynamics, where we use "Entropy Augmentation" - entropy is continually inserted into an open cryptographic system.

The best way to describe Archaios ™, is to take the journey of cryptographic discovery that we experienced when we started creating an "unbreakable" encryption algorithm. This is described below in the following sections.

- The Origin Story - Initial Objectives.
- Cipher Selection - The OTP Problem and Solutions.
- The Breakthrough – Entropy Augmentation
- Pushing the boundaries of Cryptography.

3.1 ARCHAIOS – The Origin Story

The creation of Archaios ™ has spanned over 35 years, when the initial objective was to use the military cryptanalysis knowledge gained and used in battle (The Border Wars), to create an "unbreakable" encryption system according to Shannon's cryptographic principles, which could be used practically in a digital context. Whilst the One-Time Pad was "perfectly secret", it was impossible to apply in a digital context (The OTP Deployment Problem). The other issue we had is that there were no security solutions that were able to be "Shannon-secure" beyond a certain length of ciphertext. (The Entropy Depletion Problem)

The breakthrough came in 2014, in an Amsterdam (Holland) coffee shop on the Herengracht. A vision of an encryption system starting from a random fixed point, where multiple messages and multiple random sequences were mixed with each other and then encrypted. Conventional cryptography dictates that we take very predictable "paths" or "walks" from that initial random start point. As random sequences were added, they were being used to alter the very keys and operations that the system used, creating unpredictable cryptographic "random walks". Another party would be able to take and follow the same "random walks" if they had the ciphertext only. The problem was now, to put this "vision" into practice.

After developing a functional prototype, we patented the basic invention in the US in 2016, and subsequently patented the "Equivocation Augmentation" mechanism in several countries.

Since Archaios™ supports integrated TLS as a "Quantum-Secured" message stream, we can perform "Quantum-secure" TLS certificate exchanges without divulging certificates, session keys, authentication or identity verification exchanges.

Archaios ™ was created with the "vision" of protecting humanity from Quantum + AI risks. Any business application of Entropy Augmentation or use by an automated logic system, must discuss licencing terms with Incrypteon before use.

3.2 ARCHAIOS – Selecting the Ciphers and Operations

The first stage was deciding what we would use as cipher primitives.

3.2.1 OTP as the Base Cipher

The initial problem was that the One-Time Pad was the only viable solution that could be used in theory. Below is an example – simple and fast.

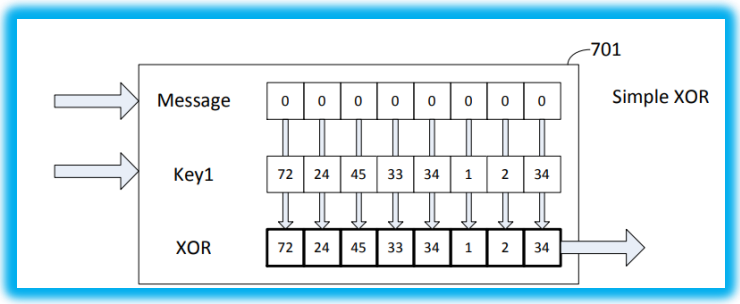

Fig 4: OTP Structure

Now, with any cryptographic assembly of ciphers, be it linear or composite in construction, we know that the assembly is only as strong as the weakest component.

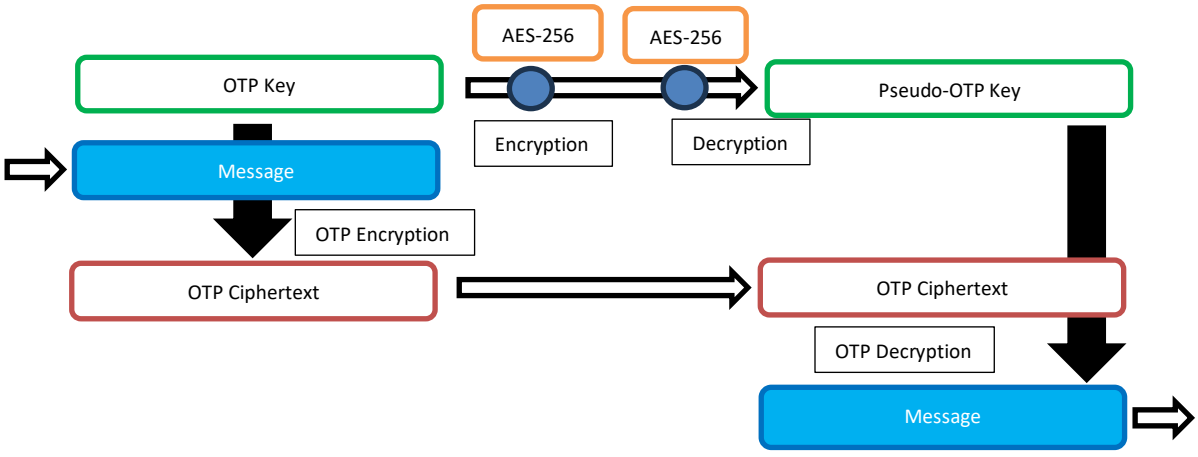

Fig 5: OTP Structure using AES-256

Taking the example above, where AES-256 is used to encrypt an OTP key. It is not possible to be "Quantum-Secure" if you use a "Quantum-insecure" symmetric or asymmetric cipher to encrypt and transport an OTP Key, and then use the "pseudo-OTP" key for encryption. You still have the OTP Deployment Problem for key storage and the solution is only as strong as the weakest link. In the example above the composite solution is only as strong as AES-256. It is not "Quantum-secure".

Only the One-Time Pad was viable, but it was fast.

Note: Any encryption algorithm that processes a message that is not longer than the key used, is a One-Time Pad.

3.2.2 OTPs + Transposition Required

Since we were limited to OTPs, we still had the problem that OTPs are unpredictable in their ciphertext values, but predictable in the position of the ciphertext values and length. An unbiased transposition cipher (A Shuffle) was added. Transpositions of OTP ciphertext is "Quantum Secure" in ciphertext values and location.

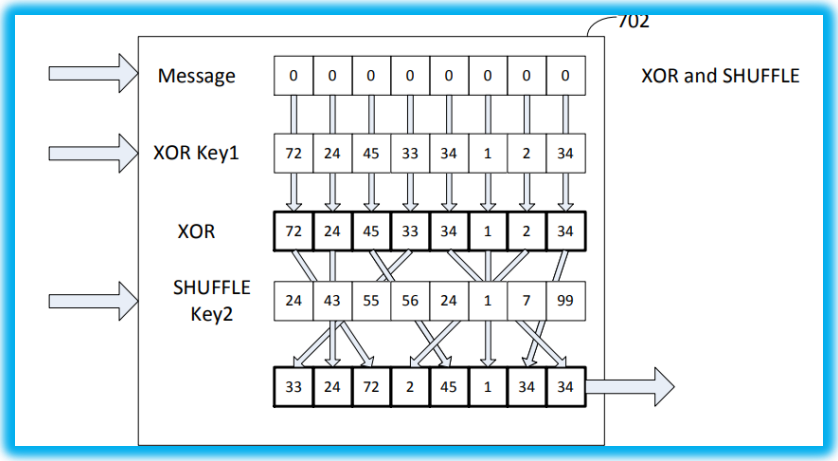

Fig 5: OTP with Shuffle

3.2.3 Dual OTPs Required – (XOR, TRANS, XOR)

We noted that no-one had ever tried to use OTP super-encryption, where two separate OTP keys are used, and the second OTP key is used to encrypt the first OTP ciphertext. They probably thought one OTP was the strongest solution. An important observation, and this was proven through experiment, was that even in the case of a known plaintext OTP encryption (which reveals the key), only the first OTP key conditional entropy was lost, but not from the second OTP key. This meant that no matter how many OTP keys one extracted from an entropy pool, it would only lose conditional entropy for the first OTP key. Thus, if one performs multiple encryptions on a plaintext, only the conditional entropy for the last encryption is lost, not for the preceding encryptions. This is trivial to prove.

In the example below, we have a message "00000000" and an encryption system using bytes (256 possible characters). Taking the 4th digit, we note that it becomes 33 in the ciphertext. Whilst the assailant may immediately know that OTP K1 and OTP K2 add up to 39, he does not know K1 and K2 values. There are 256 possible combinations – so only the conditional entropy for one OTP key was lost. Encryptions that are performed before an OTP, only lose conditional entropy for the first encryption.

Super-encrypted OTPs have the additional advantage that the OTP keys need not have "perfect randomness". The entropy of one OTP key effectively increases the overall entropy of both OTP keys.

AN example of an OTP(XOR)-Shuffle- OTP(XOR) encryption is shown below.

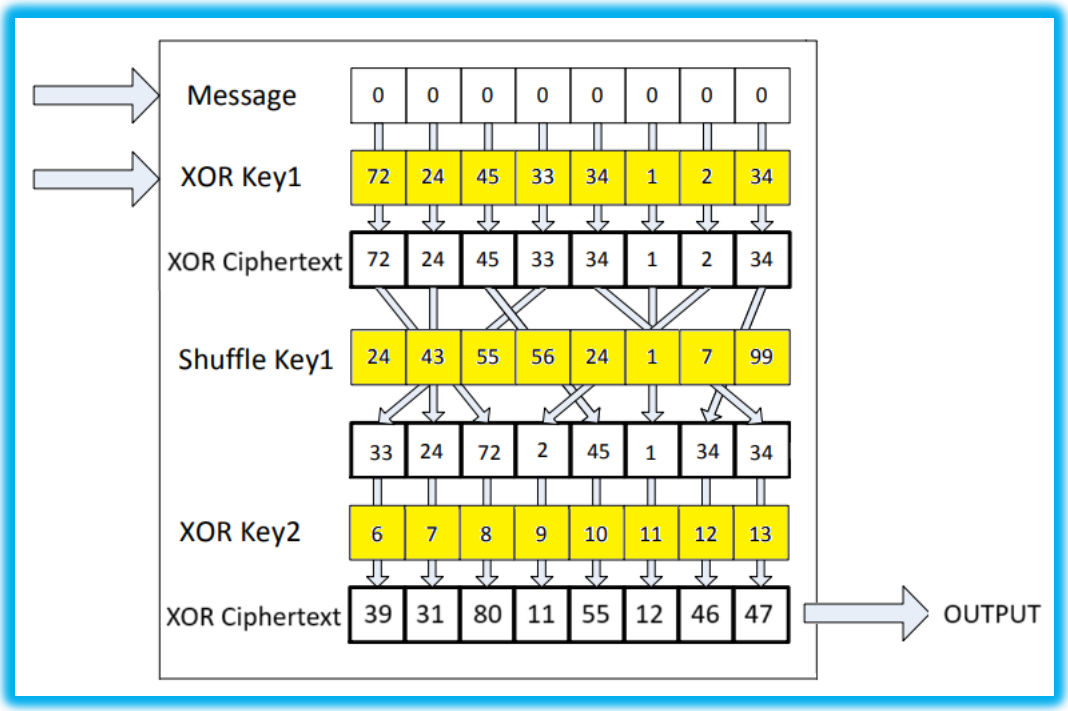

Fig 6: OTP -Shuffle- OTP

In the above example, conditional entropy is only lost for the first XOR Key, but none is lost for the shuffle key or the second XOR Key.

We now need to use a pool of entropy numbers as an RNG to get any OTP or Shuffle Keys. Most commonly available RNGs are predictable and cannot have their internal states changed on the fly.

3.2.4 Simplistic RNGs Required

There are a few problems with random number generation:

- All sequences must be possible.
- Entropy delivery must be unlimited.

The first problem with pseudo-random key generators, is that they are not capable of producing all possible keys, just keys which look like they are random. A true random number generator must be able to (in theory) output a sequence of zeros ”00000”s or non-random sequences of an arbitrary length. It must just not do it in a predictable manner. This is not possible with pseudorandom numbers.

The other problem was that hardware-based (such as Quantum) RNGs produce good randomness but are too slow at delivering the rates required for our solution. We required a random number generator that could have its entire state (entropy pool) directly altered from one state to another using a mathematical or logical operation. This will become apparent soon when we introduce entropy augmentation as the solution to OTP key generation. We had to create a simple RNG which used a string of variable length to store random numbers. Random output values are selected on the following basis (the specific operations vary – it can be made non-linear) of location and values. Starting at a point in the string, we would take the next value, that would point to a location, and we

would use values at the pointed location as output. Then advance to the next start point. In this manner, any number of required values can be generated for any number of OTP keys.

The above solution is simple, and fast, producing good entropy at the rate of entropy arrival. However, it requires at least 3 such entropy pools for proper true random number generation to work:

- An Origin RNG1 – creates entropy, transmits and passes it to the next RNG
- An Intermediate RNG – takes the entropy combines it with the existing state, and outputs to a third RNG – this is used for entropy encryption.
- An output RNG – takes RNG2 "overflow" and uses it to alter its internal state to produce multiple OTP keys (as many as required)
    - NB – NEVER uses more than 40-50% of the available values in any entropy pool when generating numbers. This protects the pool from reconstruction attempts by the assailant, who can at best only theoretically predict about 12% of the entire pool, before it changes completely.

3.3 The Breakthrough – Entropy Augmentation

The solution to both the OTP Deployment problem and the Entropy Depletion Problem, lies in the advantageous use of random strings.

In the same manner as any limited resource that can be "refuelled", at the end of an encryption step which consumes a key or series of keys, one can add the "fuel" of encryption, namely entropy (random numbers) to the system, by adding it to the residual Conditional Entropy of Key, thereby increasing overall key entropy, and allowing a new set of keys to be produced.

By referring to our graph, if entropy is added after each use of a key, this results in an increase in Conditional Entropy of Key, which then results in the Unicity Point being pushed further down the ciphertext, which in turn increases the Equivocation (or security) of the system – solving the Entropy Depletion problem.

In addition, if one can transmit encrypted entropy (not the actual OTP keys) and then combine the entropy with existing keys or random values to create new OTP keys, then OTP keys can be made at the point that they are need, with no need to store or protect them – this solves the OTP Implementation Problem.

In the following example, we show an OTP-SHUFFLE-OTP encryption where we have encrypted 4 Message characters with 4 random characters.

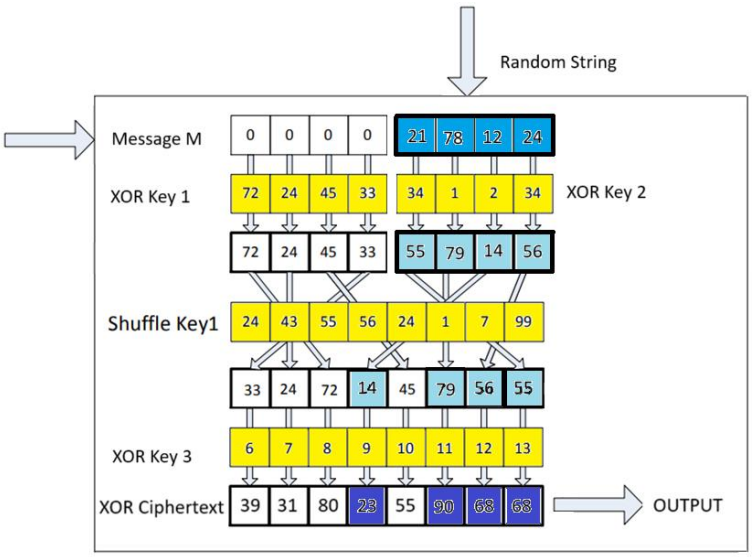

Fig 7: OTP -Shuffle- OTP (4 Message – 4 Random)

From the example above we see the following:

- 4 characters of Message are encrypted with OTP Key 1
- 4 characters of Random are encrypted with OTP Key 2
- The encrypted Messages and Random are mixed with a Shuffle Key 1
- The Shuffled sequence is encrypted with OTP Key 3 and sent to output.
- 24 characters of Key were used in total for 4 keys (OTP K1, K2 and K3, and Shuffle Key 1)

When we plot the Equivocation graph for the above encryption, we get the following interesting picture.

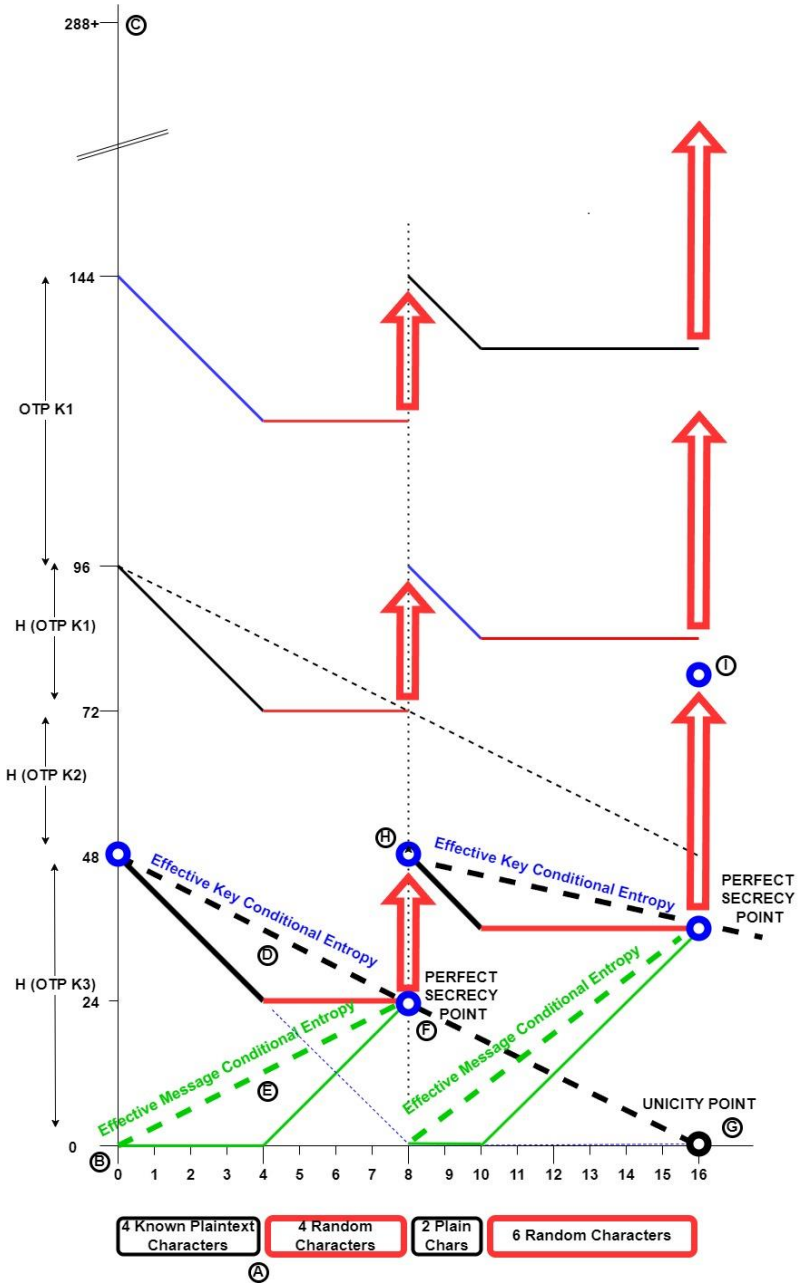

Fig 8: OTP -Shuffle- OTP (4 Message – 4 Random) – Equivocation Graph

The analysis of the graph above is as follows:

1. At point **A**, we encrypt the first 4 message bytes ***M[1-4]*** from the known plaintext message **M** using an XOR with ***OTP K1*** to create ***CM[1-4]*** , and we encrypt the first 4 random bytes ***R[1-4]*** from the random string ***R***, using an XOR with the **OTP *K2*** to create ***CR[1-4]***. This results in a "strongly ideal secrecy" encryption of ***R[1-4]***, since random strings have no redundancy. We then append ***CM[1-4]*** and ***CR[1-4]***, shuffle with ***Shuffle K1***, and encrypt the lot with ***OTP K3***.
2. At point **B**, before encryption and before applying message the system has no entropy. Applying the initial state, increases entropy to 288 bits as seen with point **C**. This is required for the current encryption set.

3. At point **D**, we see the effective **HE(K)** (Conditional entropy of key) over the two message types (Plaintext and random) – they are mixed together by the Shuffle encryption. At point **E**, we see the effective **HE(M)** (Conditional entropy of message) over the two message types. At point **F** we see where both **HE(K)** and **HE(M)** meet. Point **G** shows the unicity point for the joint encryption of ***M[1-4]*** and ***R[1-4]***.
4. Point ***H*** is where the "magic" happens. Since R[1-4] has 4 characters of entropy, these are added to the entropy pool used to generate keys, using a multiplicative operation. By multiplying the unique number sequence in the entropy poo by the new random number, the overall entropy of the system will be restored to its original value.
5. With point **I**, at the second encryption where we used 6 random characters, we can increase the entropy of the system beyond its original size. Note that the assailant does not know how large the entropy pool is, since it changes in size. The more entropy we add in encryption, the greater the overhead, so this aspect needs to be managed. We will intentionally lose conditional entropy, when we encrypt more messages than random.
6. The activity can be then conducted *ad nauseum*, until the message is fully transmitted.
7. All that is required, is the management process which tracks and ensures that conditional entropy never falls below 50% of the entropy in the entropy pool. **If conditional entropy can never be reduced to zero, the system cannot be broken using brute force. It will be "Quantum Secure".**

Further observations - the amount of entropy enhancement increases proportional to the arriving entropy updates, and the information theoretic secrecy attained above can be attained with at most 100 percent increase in message size. The benefit of the operation is increased with a decrease in message redundancy. Thus, better results can be obtained if the message is compressed prior to encryption.

The following diagram demonstrates the Entropy Augmentation capability in action over a series of encryption steps

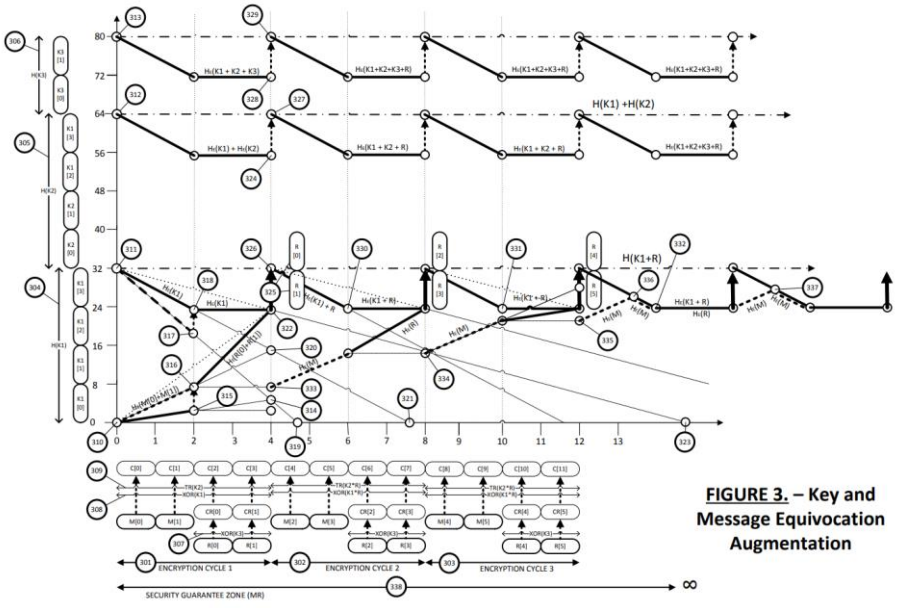

FIGURE 3. – Key and Message Equivocation Augmentation

Fig 9: OTP -Shuffle- OTP – Equivocation Graph over several encryptions

From an assailant's perspective – the Archaios ™ system cannot be breached by brute-force, since it is not possible to "try" all possible keys. There are technically no actual keys which can be verified. The assailant must reconstruct the dynamic entropy pool.

3.4 Archaios – The Core Engine Design

The Archaios ™ security system is an extensible, fast, modular, software-based, dynamic super-stream cipher and has been purposefully designed with information-theoretic security considerations in mind. Whilst composed of simple cryptographic modules, Archaios ™ is complex in its execution, effectively being dynamically polymorphic. By this we mean that every encryption operation is unique, since the state of the PE instance is randomly altered multiple times during each communication session.

Uniquely, given the same static key *K* and message *M*, every encryption of that message will result in a unique and different ciphertext C.

From a performance perspective, the Archaios ™ has demonstrated encryption and decryption speeds using a single thread, and a single 2.8 GHZ core of a quad core system, of 66 Mbytes / sec, using 12% random entropy overhead on a 65MB video file. This is on an unoptimized software implementation running 64-bit values in a 64-bit system.

The following is a basic conceptual diagram of the Archaios ™.

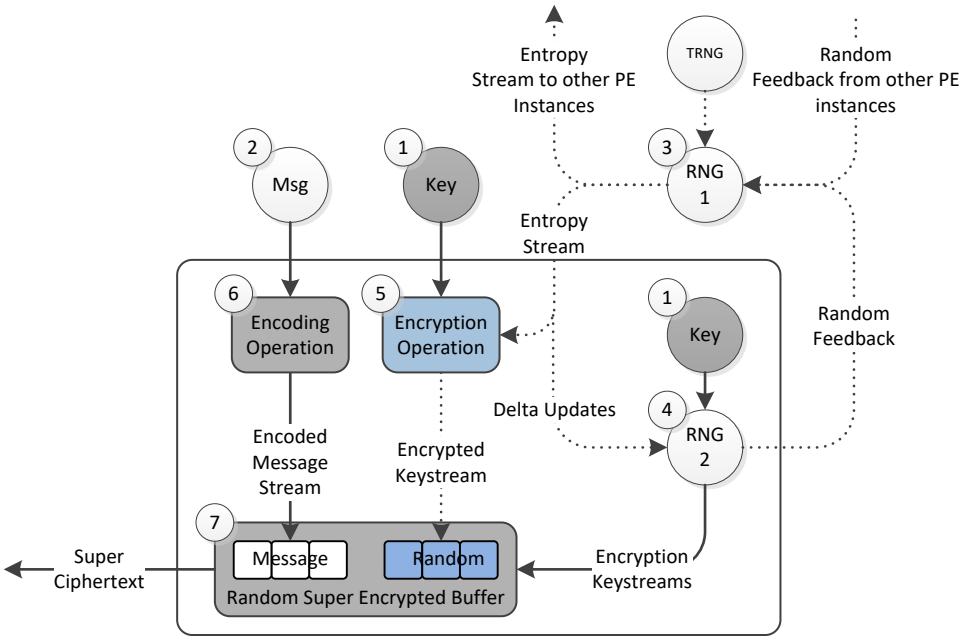

Figure 10. Archaios – Conceptual Design.

From the diagram above, we see that Archaios ™ is composed of a few simple components in a complex arrangement, namely:

- **A key K (1)** – used to initialise the state of the internal RNG2 which generates encryption keys encrypt the message and entropy inputs before transmission. Key can be of any size.

- **A message M (2)** – kept independent from the encrypted entropy keystream.

- **An independent random number generator - RNG1 (3)** – Any external RNG may be used, but preferable to use our bespoke stateful RNG as it is superior in speed, randomness and can have its entire state updated. The RNG is composed of an internal array of variable length (normally 256 bytes upwards) with random values as elements and can take any number of inputs. It does not have entropy decay issues as is found with hardware based RNGS.

- **A synchronised RNG - RNG2 (4)** – Provides the truly random encryption keystreams for the encryption buffer. Of similar construction to RNG1. With RNG2, we allow for the production of multiple keystreams, for multiple encryption operations and to secure message authentication processes.

- **A minor Key Encryption Operation (5)** - the entropy stream is encrypted with the key K before transmission to the receiver, producing a brute-force resistant "ideal secrecy" encrypted entropy cipher-stream and a new K with every encryption. This operation may be further augmented with an initialisation vector, or an asymmetric authentication solution. In such a case the asymmetric encryption would operate in a "protected and secured" area.

- **A minor Message Encoding Operation (6)** – all messages are pre-processed before final encryption and transmission, allowing various cryptanalytic countermeasures to be deployed. May include message re-encoding, message reconstruction, redundancy alteration, and message randomisation procedures. The Encoding Process aims to increase the rate of "false positives" already provided by the main encryption process, in that every random decryption has an increased probability of producing a viable message.

- **A main Random Superencryption Process** (7) - where 3 separate and simple encryption operations are applied to an Encryption Buffer of random and variable composition (containing the Encoded Message Stream, and the Encrypted Entropy Stream). This prevents attacks against the keystreams and internal state of RNG2. It also ensures that there are no message/key/ciphertext pairs to speak of, providing message indistinguishability. The block size and composition of the encryption buffer is changed with every separate encryption using the random OTP entropy stream.

- **A Random Feedback Process** - To ensure that RNG1 produces a truly random keystream, a feedback operation is used. This ensures that RNG1 never encounters an entropy depletion problem.

Decryption entails using a remote synchronised RNG2, to remove the super-encryption process, revealing the Encoded Message Stream and Encrypted Keystream. Completion of the Decoding operation presents the message, on completion of the simple key decryption process RNG2 is updated in synchronicity for the next encryption operation.

The Archaios ™ system was developed as a solution to the problems which prevented a one-time pad (OTP) from being practically possible. In essence, it allows for the "OTP pad" and the messages to be sent as a combined operation. Ordinarily, the OTP static key encryption mechanics are too restrictive to allow for a practical info-theoretic security solution. The use of an RNG allows for an interim

dynamic “entropy” repository, making simultaneous pad/message delivery possible. The OTP pad is therefore used to alter the RNG output (the true one-time pad). So “perfect secrecy” is attained indirectly. Indeed, controlling the amount of random entropy added to the message allows for the security of the system to be increased or decreased on demand.

Archaios ™ is a stochastic encryption delivery apparatus for messages and entropy augmentations, with info-theoretic security characteristics. This comes at a price, but message expansion by at most 100% is a relatively cheap price by any standards. Since the cipher security level can be determined by the user, this price only needs to be paid when the highest levels of security are required. From an attacker’s perspective, there is no way to know what amount of expansion has been used, so any amount (including over 100%) must be assumed and considered when calculating probabilities of messages.

Implementation Advantages. Archaios ™ is the first practical information-theoretically secure symmetric encryption stream cipher and is suitable for ultra-secure implementations. It is extensively configurable in terms of encryption operations and is suitable for all forms of encryption – encryption at rest and encryption in transit. The cipher can be initialised with any length of key (usually 288 bits plus), and its memory footprint can be configured to fit within a specific implementation requirement. With superior random number generation capabilities, it can be used to provide truly random numbers to other cryptosystems. It is suitable for modern super-scalar processors and allows for a high degree of parallelism. The cipher is also relatively fast, with encryption/decryption speeds of 66 MB/s using a software-only based solution, no parallelism and running with 8-bit elements on a 64-bit processor.

Security Advantages. Archaios ™ offers unprecedented levels of information theoretic security, and is therefore suitable for military, government, finance and strategic infrastructure implementations. We demonstrate that “perfect secrecy” is a limit and can be exceeded. We have thus had to define a new characteristic of information theoretic security, namely “perfectly ideal secrecy”. The cipher therefore easily attains security as defined by the lesser or reduced notions of security, such as semantic security, message indistinguishability, and “indistinguishability from random” as far as stream cipher keystreams are concerned. With regards to message indistinguishability, in the case of the Archaios™, an identical key and message will in the greatest of probability always result in a completely different ciphertext if the action is repeated. There are no plaintext/key/ciphertext triplets to speak of. This automatically qualifies the Archaios ™ as a semantically secure cryptosystem. Since the stream cipher keystreams are randomly altered, the deterministic number sequence outputs become probabilistic, evading several number generation issues such as periodicity, and we therefore easily satisfy the “indistinguishability from random requirement” with regards to keystreams. The Archaios ™ system is immune to most forms of cryptanalysis - brute-force, linear/differential cryptanalysis, known plaintext, man-in-the-middle, replay attacks, message alteration. It is one of very few systems able to transmit a null pad without compromising the key.

4 Conclusion

In this paper, we demonstrated the Entropy Augmentation method as a means of overcoming a 70-year-old cryptographic problem relating to the One-Time Pad and solving the “hidden” Entropy Depletion Problem. Our analysis indicates that it is only possible to solve these problems, if we step out of the confines of “Semantic Security” with its assumptions and static simple encryption systems and embrace Shannon’s information theoretic security paradigm. There is a world of cryptographic complexity to be explored.

Incrypteon has a solution to the One-Time Pad’s problems, allowing for a practical implementation of an “augmented” OTP solution. Thus, it seems that it is possible to have truly random keys, no key storage issues, just in time distribution, secure key destruction, and a means of generating keys at a faster rate than the message.

Statement: Please note that Incrypteon Archaios ™ and similar variants are covered by patent.