# Drone Targeted Cryptography

Swarms of Tiny Surveyors fly, stick, hide everywhere, securely communicating via solar powered new paradigm cryptography.

Gideon Samid

Department of Electrical Engineering and Computer Science
Case Western Reserve University, Cleveland, Ohio
BitMint, LLC
Gideon@BitMint.com

Abstract: As flying, camera-bearing drones get smaller and lighter, they increasingly choke on the common ciphers as they interpret their commands, and send back their footage. New paradigm cryptography allows for minimum power, adjustable randomness security to step in, and enable this emerging technology to spy, follow, track, and detect. E.g.: to find survivors in a collapsed structure. We describe here a cryptographic premise where intensive computation is avoided, and security is achieved via non-complex processing of at-will size keys. The proposed approach is to increase the role of randomness, and to build ciphers that can handle any size key without choking on computation. cryptography seeks to create a thorough mix between key bits and message bits, resulting in heavy-duty computation. Let's explore simple, fast ciphers that allow their user to adjust the security of the ciphertext by determining how much randomness to use. We present "Walk in the Park" cipher where the "walk" may be described through the series of visited spots (the plaintext), or, equivalently through a list of the traversed walkways (ciphertext). The "walking park" being the key, determines security by its size. Yet, the length of the "walk" is determined by the size of the plaintext, not the size of the "park". We describe a use scenario for the proposed cipher: a drone taking videos of variable sensitivity and hence variable required security – handled by the size of the "park".

Keywords—low-power encryption, randomness, Trans-Vernam Cipher, User-Controlled Security.

#### I. Introduction

Flying drones are inherently invasive; they see what was previously hidden. There are many laudable applications for such invasive devices, e.g. search and rescue operations, catching fugitives, the war on terror, etc. Yet, very often drones violate someone's privacy, or even endanger national security, and hence the visual vista exposed by them should be treated with proper sensitivity, namely encryption. Alas, as drones become smaller, power becomes an issue, and modern ciphers which churn and mix key bits and message bits tend to require too much power to function. This challenge is addressed herein.

We extend the introduction to discuss (i) the application environment, and (ii) the principles of the proposed solutions.

## A. Application Environment

Flying drones can network, communicate, and coordinate movements and activities in support of a surveillance goal. They need to be securely controlled, securely coordinated, and securely deliver their collected data to their customer. This implies fast, effective cryptography. Alas, the drones are mini or micro size, lightweight, and short on power, so most of the mainstay ciphers will not be practical for them. A team at Harvard just presented a "bee size" drone that easiy perches on any surface, sending sensitive videos to its operators [Ulanoff]. Some attributes of this new technology are discussed:

**Speed:** High speed, high-resolution cameras fitted on flying drones may be required to transmit to an operational center, to serve an important rescue operation, or other proper assignment. Similarly, an isolated device somewhere may be activated with a large stream of commands, most of them should be further transferred to devices down the line, exploiting directional microwave communication. All in all, a swarm of drones may need to accommodate high volume, high speed information exchange. The existing popular ciphers slow down that flow rate, and are not friendly to this requirement.

Maintenance: Quite a few flying drones will be placed in hard to access locations, and no physical maintenance will be feasible. They might use a solar power source and function indefintely. Hence the use of any specific cipher, which at any moment may be mathematically breached, is a risky practice. This applies to all algorithmic complexity ciphers. As Prof. Nigel Smith articulates in his book "Cryptography (an Introduction)": "At some point in the future we should expect our system to become broken, either through an improvement in computing power or an algorithmic breakthrough." Normally, cryptography gravitates towards very few ciphers considered 'secure'. If one of them is suddenly breached (e.g. GSM communication cipher), then all the "out of reach" nodes which rely on it, have lost their security, and physical attention is not practical.

**Magnetic Vulnerability:** Many flying drones are placed in very harsh environment, and are subject to lightening violence,

as well as man made electromagnetic impacts. Software based cipher may be at greater risk.

In summary, flying drones in particular and IOT nodes in general are vulnerable both to malicious attack, and to environmental punishment. These vulnerabilities may be remedied to a large extent if we come up with a new cryptographic approach: *Cryptography of Things* (CoT).

## B. Principles of the Proposed Solution

Modern cryptography erects security around data using two parameters: (i) algorithmic complexity, and (ii) randomness. It's generally believed that the more complex an algorithm the more secure the ciphertext, and also the more randomness that is being used (the larger the key), the more secure the ciphertext. Randomness is in a way dull, and of no much interest mathematically (except of course with respect to its definition and to metrics of quality). By contrast, algorithmic complexity is an exciting math dilemma. Academic cryptographers are attracted to this challenge and develop new and newer complex algorithms. Unfortunately in today's state of affairs, we only manage to compare complexities one to the other, not to ascertain their level in an objective mathematical way. And even if it turns out that  $P \neq$ NP as most complexity researchers believe, in cryptography complexity is used in combination with randomness, hence one is using a random key selected from a large key space. What is hard to know is how many specific keys when applied with specific plaintexts, offer some mathematical vulnerability, leading to effective extraction of the message. In other words, the de facto complexity, or security of algorithms cannot be ascertained. Worried about this, we come up with increasingly complex algorithms, which require more and more computational effort. They in turn require more and more power -- which many IOT nodes simply don't have.

Randomness, on the other hand, is passive memory, and even the smallest and most unsophisticated devices can be fitted with gigabytes of memory, serving as key. These realities lead one to aim to develop cryptography where the role of reliable, passive, manageable, secure randomness is enhanced, while the role of doubtful complex algorithms that are power hogs, is decreased.

This thinking brings to mind the famous Vernam cipher: the algorithm could not have been simpler, and the key could easily be as large as hundreds of gigabytes. So what? Memory is both cheap and light. It may be stored without requiring power. Too bad that Vernam is so impractical to use. Yet, can we re-analyze Vernam as a source of inspiration for security through more randomness and less algorithmic complexity?

Let's envision a Vernam Inspired Cipher (VIC) where at any stage the user can 'throw in a few more key bits' and by that achieve a large increase of cryptanalytic burden, together with a modest increase of nominal processing burden (encryption, and decryption). Let us further demand from the VIC the Vernam property of achieving mathematical secrecy at the minimum key size required by Shannon's proof of perfect secrecy.

To better analyze this vision let's regard any cryptographic key, k, as the natural number represented by binary interpretation of its bit sequence. Accordingly, the Vernam key space associated with n-bits long messages, will be: 1,2,....( $2^n$ -1) corresponding to  $\{00....0\}_n$  to  $\{11....1\}_n$ . We may further agree that any natural number  $N=K > 2^n$ -1 will be hashed to an n-bits size string. Once we agree on the hashing procedure we have managed to recast Vernam cipher as a cipher that accepts any positive integer as a key, with which to encrypt any message m comprised of n bits to a corresponding ciphertext. We regard this as natural number key representation (NNKR).

We can similarly recast any cipher according to NNKR. We consider a cipher for which the series  $n_1, n_2, \dots, n_{max}$ represents the allowable bit counts for the keys. E.g for DES the series has one member  $n_1=n_{max}=56$ ; for AES the series contains three members:  $n_1=128$ ,  $n_2=192$ ,  $n_3=n_{max}=256$ . For a cipher where the key is a prime number then the series is the series of primes. For ciphers defined over every bit string of length  $n_{max}$  all the natural numbers from 0 to  $2^{n}$ -1 qualify as a n<sub>max</sub> key. Larger keys will be hashed to a n<sub>max</sub> bits long hash. For ciphers where the series  $n_1, n_2, \dots n_{max}$  represents discrete possible keys, we may agree to hash any natural number to highest member of the list  $n_1, n_2,...$  which is lower than that natural number. For all natural numbers smaller than  $n_1$ , we will "hash" them to the null key (|K|=0), and we may formally agree that the case of K=NULL is the case of no encryption (the ciphertext is simply the plaintext).

With the above definition we have recast all ciphers as accepting every natural number as a key.

We define the concept of "normal cipher" i as a cipher for which any valid metric of security,  $s_i$ , is never lower for larger keys. Say, for two positive integers  $K_1$  and  $K_2$  used as keys, and where  $K_1 \le K_2$ , we may write:

$$s_i(K_1) \leq s_i(K_2)$$

In other words, with normal ciphers we "buy" security, and "pay" for it with a choice of a random number. Let  $s_i(K)$  be the security achieved by a user of cipher i, "investing" key K. The metric s, will reflect the average computational effort required of the cryptanalyst for extracting the message m from a captured ciphertext c, computed over the distribution of m  $\in$  M, where M is the message space from which m is selected. Let  $p_i(K)$  be the average combined processing effort (encryption plus decryption) required of a user of cipher i, while using key, K, over the distribution of message m  $\in$  M.

For any cipher i, using a natural number K as key, we may define the utility of the cipher at this point as the ratio between the cryptanalytic effort and the nominal processing effort:

$$(1).....U_i(K) = s_i(K)/p_i(K)$$

We can now define a Vernam Inspired Cipher as one where over some range of natural numbers  $K(K_1,...,K_2)$  as key, the utility of the cipher will be somewhat stable:

(2)..... 
$$U_1, U_{\kappa 1+1}, \dots U_{\kappa 2} \sim U$$

In that case a user encrypting with  $K_1$  will be able to increase the security he builds around the data, while still using the same cipher, by simply ratcheting up the key from  $K_1$  to  $K_2$ . She will then -- again, using the same cipher -- increase its associated security from  $s(K_1)$  to the higher value of  $s(K_2)$ 

(3)... 
$$S(K_2) = S(K_1) + \sum (U(K+1) * P(K+1) - U(K) * P(K))_{FOR K=K_1 TO}$$
  
=  $S(K_1) + (U(K_2) * P(K_2) - U(K_1) * P(K_1))$ 

which is reduced to:

(4)...... 
$$S(K_2) = S(K_1) + U*(P(K_2)-P(K_1))$$

Recasting cryptographic keys as natural numbers leads to redefinition of the key space, #K, as a subset of the natural numbers from 1 (or formally from zero) to the highest natural number to be considered as a key,  $\#K=K_{max}$ :

$$(5)$$
.....# $K \le K_{MAX}$ 

And hence, for messages comprised of n bits, a key max of value  $2^n$  ( $K_{max} = 2^n$ ) will allow for a cipher where the user could simply ratchet up the integer value used as key,  $K' < 2^n$ , to the point of achieving mathematical security. We can define a special case of a Vernam Inspired Cipher, as a **Trans Vernam Cipher (TVC)**, being a cipher where increase in the integer value used as key will eventually reach "Vernam Security Levels", or say, Shannon's security, for n-bits long messages:

(6).....
$$s_{max} = s(K_{max} = 2^n) = s(K') + U(K_{max}) * p(K_{max}) - U(K') * p(K')$$

**Existence:** It's readily clear that DES, AES and their like will not qualify as Vernam Inspired Ciphers. For DES:

$$_{S}(K > 2^{56}) = 0$$
  
 $_{S}(K > 2^{56}) = _{S}(K=2^{56})$ 

For AES:

$$(8)......_{S}(K < 2^{128}) = 0$$

$$s(2^{128} \le K < 2^{192}) = s(K=2^{128})$$

$$s(2^{192} \le K < 2^{256}) = s(K=2^{192})$$

$$s(K > 2^{256}) = s(K=2^{256})$$

The background 'philosophy' to casting key spaces onto the the natural numbers is discussed in reference: [Samid 2001, and Samid 2016 (b).]

#### II. "WALK-IN-THE-PARK" CIPHER

We present here a Trans-Vernam Cipher (TVC), that runs by the name *Walk-in-the-Park* because both encryption and decryption is taking place by "walking" – charting a path determined by the message, and then describing it through various entities in the "park" where the walk happens. It is based on the idea that a 'walk' can be described either via the places visited, or via the roads taken from one visited place to another. One needs the "park" (the key) to convert one description to the other.

The cipher is defined as follows:

We employ a four-letter alphabet: X, Y, Z, and W, expressed via 01,10,11,00 respectively. The key is a table (or matrix) of size u \* 2v bits, which houses some arrangement of the four alphabet letters (u\*v letters in total). We regard every letter as a node of a graph, and regard any two horizontally or vertically contiguous letters as connected with an edge. So every letter marked on the graph has between 2 to 4 edges connecting it to other letters on the graph. (4 edges for middle nodes, 3 edges for boundary nodes, and 2 edges for corner nodes).

We define a path on the graph as a sequence of marked letters such that any two contiguous letters on the path are connected via an edge.

Informally, the cipher works by mapping the plaintext into a sequence of X,Y,Z, and W; then using this sequence to mark a pathway on the graph. Given an agreed upon starting point, it is possible to describe the very same graph via denoting the edges traversed by the pathway. Each node, or vertex on the graph has up to four edges; let's mark them Up, Down, Right, Left: U,D,R,L, and assign the bit combinations 01,10,00,11 respectively to them. The translation of the pathway from a sequence of vertices to a sequence of edges amounts to encrypting the plaintext to the ciphertext. And respectively for the reverse (decryption).

Why is this a Trans Vernam Cipher? Because the graph may be large or small. The larger it is the more security it provides. It may be so large that it will be a Vernam equivalent, and it may be so small that brute force will extract it relatively easily. The processing effort is not affected by the size of the graph, only by the length of the pathway, which is

the size of the encrypted message. By analogy given a fixed walking speed, it takes the same time to walk, say, 10 miles on a straight stretch of a road, or zigzagging in a small backyard.

#### **Detailed Procedure:**

- **1. Alphabet Conversion:** Map a list of symbols to a three letters alphabet: X, Y, Z. By mapping every symbol to a string of 5 letters from the  $\{X,Y,Z\}$  alphabet. It is possible to map  $3^5$ =243 distinct symbols (a few less than the ASCII list of 256 symbols).
- 2. **Message conversion**: let  $m=m_0$  be the message to be encrypted, written in the symbols listed in the 243 symbols list (essentially the ASCII list). Using the alphabet conversion in (1) map  $m_0$  to  $m_3$  a sequence of the 3 letters alphabet: X, Y, Z
- 3. **DeRepeat the Message:** enter the letter W between every letter repletion in  $m_3$ , and so convert it to  $m_4$ .  $m_4$  is a norepeat sequence of the letters  $\{X,Y,Z,W\}$ . Add the letter W as the starting letter.
- 4. Construct a key: construct a  $u^*v$  matrix with the letters {X,Y,Z,W} as its elements. The matrix will include at least one element for each of the four letters. The letters marking will abide by the 'any sequence condition' defined as follows: Let  $i \neq j$  represent two different letters of the four  $\{X,Y,Z,W\}$ . At any given state let one of the  $u^*v$  elements of the matrix be "in focus". Focus can be shifted by moving one element horizontally (right or left), or one element vertically (up or down) – reminiscent of the Turing Machine. Such a focus shift from element to an adjacent element is called "a step". The 'any sequence condition' mandates that for any element of the matrix marked by letter i, it will be possible to shift the focus from it to another element marked by the letter j, by taking steps that pass only through elements marked by the letter i. The 'any sequence condition' applies to any element of the matrix, for any pair of letters (i,i).
- 5. **Select a starting point:** Mark any matrix element designated as "W" as the starting point (focus element).
- 6. Build a pathway on the matrix reflecting the message ( $m_4$ ): Use the {X,Y,Z,W} sequence defined by the  $m_4$  version of the message, to mark a pathway (a succession of focus elements) through the matrix. The "any sequence condition" guarantees that whatever the sequence of  $m_4$ , it would be possible to mark a pathway, if one allows for as much expansion as necessary, when an 'expansion' is defined as repeating a letter any number of times.

7. **Encrypt the pathway**: Describe the identified pathway as a sequence of edges, starting from the starting point. This will be listed as a sequence of up, down, right, left {U,D,R,L} to be referred to as the ciphertext, c.

The so generated ciphertext (expressed as 2 bits per edge) is released through an insecure channel to the intended recipient. That recipient is assumed to have in her possession the following: (i) the alphabet conversion tables, (ii) the matrix, (iii) the identity of the starting point, and (iv) the ciphertext c. The intended recipient will carry out the following actions:

- 8. **Reconstruct the Pathway:** Beginning with the starting element, one would use the sequence of edges identified in the ciphertext, as a guide to chart the pathway that the writer identified on the same matrix.
- 9. Convert the pathway to a sequence of vertices: Once the pathway is marked, it is to be read as a sequence of vertices (the matrix elements identified by the letters  $\{X,Y,Z,W\}$ ), resulting in an expanded version of the message,  $m_{4\text{exp}}$ . The expansion is expressed through any number of repetitions of the same letter in the sequence.
- 10. Reduce the Expanded Message (to  $m_4$ ): replace any repetition of any letter in  $m_{4\text{exp}}$  with a single same letter:  $m_{4\text{exp}} \rightarrow m_4$
- 11. Reduce  $m_4$  to  $m_3$ : eliminate all the W letters from  $m_4$ .
- 12. Convert  $m_3$  to  $m_0$ : use the alphabet conversion table to convert  $m_3$  to the original message  $m_0$ .

**Illustration:** . Let the message to be encrypted be:  $m=m_0=$ "love". Let the alphabet conversion table indicate the following:

Accordingly we map  $m_0$  to  $m_3 = XYZ ZYX XYZ ZYY$ .

We now convert  $m_3$  to  $m_4 = WXYZWZYXWXYZWZYWY$ .

We build a matrix that satisfies the 'any sequence condition':

Using m<sub>4</sub> as a guide we mark a pathway on the matrix:

The pathway may be read out through the traversed edges, regarded as the ciphertext, c:

#### c = URDDULDRUULDULDDLUDLULR.

In order to decrypt c, its recipient will have to use the matrix (the graph, the key, or say, "the walking park"), and interpret the sequence of edges in c to the visited vertices:

Pathway = 
$$5,2,3,6,9,6,5,8,9,6,3,2,5,2,3,6,9,8,5,8,9,6,5,6$$
.

This is the same pathway marked by the ciphertext writer. Once it is marked on the matrix it can be read as a sequence of the visited vertices:

$$m_{4\text{exp}} = WXYYZYWZZZYYXWXYYZZWZZYWY.$$

Which is reduced  $m_{4exp} \rightarrow m_4$ : WXYZWZYXWXYZWZYWY; Which, in turn, is reduced to the three letters alphabet:  $m_4 \rightarrow m_3 = xyz zyx xyz zyy$ , which is converted to m = ``love''

Walk-in-the-Park as a TVC: There are various procedures, which would translate the matrix (the key) into a natural number and vice versa. Here is a very simple one. Let k be a square matrix (key) as described above, comprised of  $u^2$ letters. Each letter is marked with two bits, so one can list the matrix row by row and construct a bit sequence comprised of 2u<sup>2</sup> bits. That sequence corresponds to a non-negative integer, k. k will be unambiguously interpreted as the matrix that generated it. To transform a generic positive integer to a matrix, one would do the following: let N be any positive integer. Find u such that  $2(u-1)^2 < N \le 2u^2$ . Write N in binary and pad with zeros to the left such that the total number of bits is  $2u^2$ . Map the  $2u^2$  bits onto a  $u^2$  matrix, comprised of 2 bits elements, which can readily be interpreted as u<sup>2</sup> letters  $\{X,Y,Z,W\}$ . If the resultant matrix complies with the 'any sequence' condition, this matrix is the one corresponding to N. If not, then increment the 2u<sup>2</sup> bit long string, and check again. Keep incrementing and checking until a compliant matrix is found, this is the corresponding matrix (key) to N.

A more convenient way to map an arbitrary integer to a "Park" is as follows: let N an arbitrary positive integer written as bit string of  $N_b$  bits. Find two integers  $u \le v$  such that:

$$18uv \ge N_b > 18u(v-1)$$

Pad N with leftmost zeros so that N is expressed via a bit string of 18uv bits. Map these 18uv bits into a rectangular matrix of (3u)\*(6v) bits. This matrix may be viewed as a tile

of *uv* "park units" (or "unit parks"), where each unit is comprised of 18 = 3\*6 bits, or say 3x3=9 letters: {X,Y,Z,W}.

There are 384 distinct arrangements of park units, when the bits are interpreted as letters from the  $\{X,Y,Z,W\}$  alphabet, and each unit is compliant with the 'any sequence condition'. This can be calculated as follows: We mark a "park unit" with numbers 0-8:

4 3 2 5 0 1 6 7 8

Let mark position 0 as W, positions 1,2,3 as X, positions 4,5 as Y, and positions 6,7,8 as Z. This configuration will be compliant with the 'any sequence condition'. We may rotate the markings on all letter place holders: 1-8, 8 times. We can also mark, 1 as X, 2,3,4 as Y, and 5,6,7,8 as Z and write another distinct 'any sequence compliant' configuration. This configuration we can rotate 4 times and remain compliant. Finally we may mark 1 as X, 2,3,4,4 as Y, and 6,7,8 as Z, and rotate this configuration also 4 times. This computes to 8+4+4=16 distinct configuration. Any such configuration stands for the 4! permutations of the four letters, which results in the quoted number 384 = 16\*4! We can mark these 384distinct configurations of "park units" from 0 to 383. We then evaluate the 'unit park integer' (N<sub>p</sub>) as the numeric value defined by stretching the 18 bits of the unit-park into a string. We then compute  $x = N_n$  mode 384, and choose configuration x (among the 384 distinct unit-park configurations), and write this configuration into this park unit. Since every 'park unit' is 'any sequence compliant' the entire matrix of (3u)\*(6v){X,Y,Z,W} letters is also 'any sequence' compliant. The resultant matrix of 18uv letters will challenge the cryptanalyst with a key space of: 384<sup>uv</sup> keys. Alas, the cryptanalyst is not aware of u and v, which are part of the key secret. This special subset of 'any sequence compliant' matrices is a factor of 683 smaller than the number of all matrices (compliant and noncompliant):  $683 = 2^{18}/384$ 

It is clear by construction that Walk-in-the-Park is a TVC: the key (the map) gets larger with larger integer keys, and for some given natural number  $k_{Vernam}$  a message m will result in a pathway free of any revisiting of any vertex. The resultant ciphertext can then be decrypted to any message of choice simply by constructing a matrix with the traversed vertices fitting that message.

**Cryptanalysis:** A 9-letters key as in the illustration above will be sufficient to encrypt any size of message m, simply because it is 'any sequence compliant'. A large m will simply zigzag many times within this single "park unit". A cryptanalyst who is aware of the size of the key will readily apply a successful brute force cryptanalysis (there are only 384 'any sequence' compliant configuration of a 3x3 key, as is computed ahead). Clearly, the larger the size of the key the

more daunting the cryptanalysis. Even if the pathway revisits just one vertex twice, the resultant cipher is not offering mathematical security, but for a sufficiently large map (key) the pathway may be drawn without revisitation of same vertices -- exhibiting Vernam, (or say, perfect) secrecy. **Proof:** let c be the captured ciphertext, comprised of |c| letters  $\{U.D.R.L\}$ . c marks a pathway on the matrix without revisiting any vertex, and hence, for every message  $m \in M$  (where M is the message space) such that  $|c| \ge |m|$ , we may write:

# $Pr[M=m \mid C=c] = 0.25^{|c|}$

That is because every visited vertex may be any of the four letters  $\{X,Y,Z,W\}$ . Namely the probability of any message m to be the one used depends only on the *size* of the ciphertext, not on its content, so we may write:  $Pr[M=m \mid C=c] = Pr[M=m]$ , which fits the Shannon definition of perfect secrecy. Clearly, if the path undergoes even one vertex revisitation, then it implies a constraint on the identity of the revisited vertex, and some possible messages are excluded. And the more re-visitation, the more constraints, until all the equivocation is washed away, entropy collapses, and only computational intractability remains as a cryptanalytic obstacle.

This "Walk in the Park" cipher, by construction, is likely using only parts of the key (the graph) to encrypt any given message, m. When a key K is used for t messages: m<sub>1</sub>, m<sub>2</sub>,...m<sub>t</sub>, then we designate the used parts as K<sub>t</sub>, and designate the unused parts as K<sub>-t</sub>. For all values of t=0,1,2,.... we have  $K_t + K_{-t} = K$ . And for  $t \rightarrow \infty$  Lim  $K_{-t} = 0$ . By using a procedure called "tiling" it is possible to remove from the t known ciphertexts:  $c_1, c_2, c_t$ , any clue as to the magnitude of  $K_{-t}$ . Tiling is a procedure whereby the key matrix is spread to planar infinity by placing copies of the matrix one next to each other. Thereby the ciphertext, expressed as a sequence of U,D,R,L will appear stretched and without repetition, regardless of how small the matrix is. The cryptanalyst will not be able to distinguish from the shape of the ciphertext whether the pathway is drawn on a tiled graph or on a truly large matrix. Mathematically tiling is handled via modular arithmetic: any address (x,y) on a tiled matrix is interpreted as  $x \mod u$ , and  $y \mod v$  over the u\*v matrix.

This tiling confusion may be exploited by a proper procedure for determining the starting point of the pathway.

Determining the Starting Point of the Pathway: In the simplest implementation, the starting point is fixed (must be a W element by construction of the pathway), for all messages. Alas, this quickly deteriorates the equivocation of the elements near the starting point. Alternatively the next starting point may be embedded in the previous encrypted message. Another alternative is to simply expose the starting point, and identify it alongside the ciphertext. This will allow

the user to choose a random W element each time. As long as t << uv the deterioration in security will be negligible.

A modification of the above, amounts to setting the address of the next starting point in the vicinity of the end point of the previous message. This will result in a configuration where consecutive pathways mark a more or less stretched out combined pathway. A cryptanalyst will be confounded as to whether this stretched combined pathway is marked on a large matrix, or on a tiled matrix.

And hence, regardless of how many messages were encrypted using the very same key, the cryptanalyst will face residual equivocation, and be denied the conclusive result as to the identity of the encrypted message.

Persistent Equivocation: A mistaken re-use of a Vernam key, totally destroys the full mathematical equivocation offered by a carefully encrypted message. Indeed, Vernam demands a fresh supply of random bits for each message used. By contrast, the "Walk in the Park" cipher exhibits residual equivocation despite re-use of the same key. Let us assume that the cryptanalyst knows the size of the key (3u\*3v letters), let us further assume that the cryptanalyst also knows that the 'any sequence condition' was achieved by using the "park unit" strategy. In that case the key space will be of size: 384<sup>uv</sup>. Let us also assume that the cryptanalyst knows the starting points for t encrypted messages. If by charting the t pathways, no re-visitation occurrence is found, then the cryptanalyst faces mathematical security. If there are h vertices which are visited by the t pathways at least twice, then even if we assume that the park units for all those h vertices suddenly become known, then the key space is reduced to 384<sup>uv-h</sup> which deteriorates very slowly with h.

This cipher targets drone as a primary application, but clearly it extends its utility way beyond. In the present state the "Walk in the Park" cipher is an evolution of the ciphers described in reference [Samid 2002, Samid 2004].

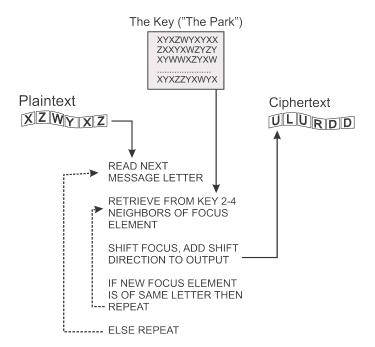


Drones Adjusting Encryption Level to match data sensitivity

#### III. USAGE SCENARIOS

We describe here a use case that is taken from a project under evaluation. It relates to swarms of tiny drones equipped with a versatile video camera. Each drone is extremely light, it has a small battery, and a solar cell. It is designed to land on flat or slanted objects like roofs. The camera streams to its operators a live video of the viewable vista. The drone requires encryption for interpretation of commands, communicating with other drones, and for transmitting videos. The high-powered multi mega pixel camera may be taping non sensitive areas like public roads; it may stream medium sensitive areas, like private back yards, and it may also stream down highly sensitive areas, like industrial and military zones. The micro drone may be dropped in the vicinity of operation, with no plans of retrieval. It should operate indefinitely.

Using *Walk-in-the-Park* the drone will be equipped with three keys (matrices, graphs): 1. a small hardware key comprised of square flash memory of 500x500 {X,Y,Z,W} letters. This will amount to a key comprised of 500,000 bits. 2. A flash memory holding 1000x1000 {X,Y,Z,W} letters, comprising 2,000,000 bits. 3. A flash memory holding 7500x7500 {X,Y,Z,W} letters comprising 112,500,000 bits. The latter key should provide perfect secrecy for about 6 gigabytes of data.



# "Walk in the Park" Encryption Algorithm

"Turing Machine" algorithmic simplicity. Security achieved via at-will randomness The determination of the security sensitivity of the photographed area (and the corresponding security level used) may be determined onboard the drone, or communicated from the reception center based on the transmitted pictures.

To achieve maximum speed the "Walk in the Park" cipher is written with "Turing Machine" simplicity: minimum number of operational registers, minimum operational memory; for every state (particular focus element in the matrix), the firmware reads the identity of the neighbors of the focus to decide where to shift the focus to, and output the direction of the shift as the next ciphertext letter. Decryption is symmetrically in the opposite direction.

#### IV. SUMMARY NOTES

We presented here a philosophy and a practice for Drone Cryptography, or more broadly: "Cryptography of Things" (CoT) geared towards Internet of Things applications.. The CoT is mindful of processing parsimony, maintenance issues, and security versatility. The basic idea is to shift the burden of security away from power-hungry complex algorithms to variable levels of randomness matching the security needs per transmission. This paper presents the notion of Trans-Vernam Ciphers, and one may expect a wave of ciphers compliant with the TVC paradigm. It's expected that the IoT will become an indispensable entity in our collective well being, and at the same time that it should attract the same level of malice and harmful activity experienced by the Internet of People, and so, despite its enumerated limitations, the IoT will require new horizons of robust encryption to remain a positive factor in modern civil life.

# REFERENCES

M. Hellman. 1977 "An extension of the Shannon theory approach to cryptography". IEEE Transactions on Information Theory, V. 23 , 3 1977 , pp. 289 - 294

Ma't'e Horva'th, 2015 "Survey on Cryptographic Obfuscation" 9 Oct 2015 International Association of Cryptology Research, ePrint Archive https://eprint.iacr.org/2015/412

Masanobu Katagi and Shiho Moriai "Lightweight Cryptography for the Internet of Things" Sony Corporation 2011 <a href="https://www.iab.org/wp-content/IAB-uploads/2011/03/Kaftan.pdf">https://www.iab.org/wp-content/IAB-uploads/2011/03/Kaftan.pdf</a>

Menezes, A. J., P. van Oorschot and S.A. Vanstone. The Handbook of Applied Cryptography. CRC Press, 1997.

Rein Canetti, Cynthia Dwork, Moni Naor, Rafail Ostrovsky "Deniable Encryption" CRYPTO '97Volume 1294 of the series Lecture Notes in Computer Science pp 90-104Date: 17 May 2006 S. Zhou ( ZTE Corporation ) Z. Xie ( ZTE Corporation) 2011 "On Cryptographic Approaches to Internet-Of-Things Security" http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity/paper s/ZhouSujing.pdf

Samid 2016 "To Increase the Role of Randomness" http://classexpress.com/IncreaseRandomness\_H6327.pdf

Samid, 2016 "Shannon's Proof of Vernam Unbreakability" https://www.youtube.com/watch?v=cVsLW1WddVI

Samid, G. "Re-dividing Complexity between Algorithms and Keys" Progress in Cryptology — INDOCRYPT 2001 Volume 2247 of the series Lecture Notes in Computer Science pp 330-338

Samid, G. 2001 "Anonymity Management: A Blue Print For Newfound Privacy" The Second International Workshop on Information Security Applications (WISA 2001), Seoul, Korea, September 13-14, 2001 (Best Paper Award).

Samid, G. 2001 "Encryption Sticks (Randomats)" ICICS 2001 Third International Conference on Information and Communications Security Xian, China 13-16 November, 2001

Samid, G. 2002 " At-Will Intractability Up to Plaintext Equivocation Achieved via a Cryptographic Key Made As Small, or As Large As Desired - Without Computational Penalty " 2002 International Workshop on CRYPTOLOGY AND NETWORK SECURITY San Francisco, California, USA September 26 -- 28, 2002

Samid, G. 2003 "Intractability Erosion: The Everpresent Threat for Secure Communication" The 7th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2003), July 2003.

Samid, G. 2003 "Non-Zero Entropy Ciphertexts (Stochastic Decryption): On The Possibility of One-Time-Pad Class Security With Shorter Keys" 2003 International Workshop on CRYPTOLOGY AND NETWORK SECURITY (CANS03) Miami, Florida, USA September 24 - - 26, 2003

Samid, 2004 "Denial Cryptography based on Graph Theory", US Patent #6,823,068

Samid, G. 2016 "Stupidity + Randomness = Smarts" https://www.youtube.com/watch?v=TYgNdoAAfkE

Shannon, Claude, 1949 "Communication Theory of Secrecy Systems" http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf

Smart, Nigel "Cryptography (an Introduction)" 3rd Edition http://www.cs.umd.edu/~waa/414-F11/IntroToCrypto.pdf

Stallings Williams, 2002 "Introduction to Cryptography" <a href="http://williamstallings.com/Extras/Security-Notes/lectures/classical.html">http://williamstallings.com/Extras/Security-Notes/lectures/classical.html</a>

Ulanoff Lance, 2016 "New Robot Bee May Soon Become a Spy's Secret Weapon" http://mashable.com/2016/05/19/perching-robot-bee/#YyrAVNEMmgqI

Vernam 1918; Gilbert S. Vernam, US Patent 1310719 Filed 13 September 1918.