Family Key Cryptography

INTERCHANGEABLE SYMMETRIC KEYS: A DIFFERENT CRYPTOGRAPHIC PARADIGM

Keywords: paradigm, Vernam, Shannon, Mathematical Secrecy

Gideon Samid[1]

Abstract: In the current crypto paradigm a single secret key transforms a plaintext into a ciphertext and vice versa, or at most a different key is doing the reverse action. Attackers exposed to the ciphertext are hammering it to extract that single key and the plaintext. This paradigm may be challenged with an alternate setup: using a particular crypto algorithm, there is an infinite number of keys that are perfectly interchangeable -- each has the same effect. Nonetheless they are hard to find. And unlike regular cryptography, the best an attacker can hope for using this new "Family Key Cryptography", is to identify the entire infinitely large family of keys, not the actual key that executed the cryptographic action. This very fact is a cornerstone for a host of applications, mostly still to be unfolded. E.g.: (1) *Community Cryptography*, where each member has a different key, but the community will encrypt and decrypt as if sharing the same key; (2) *'Forever Key Cryptography':* crashing the Shannon's limit, the Forever Key strategy will allow a single finite key to last indefinitely. The shared secret key will be used to derive a succession of operating keys, which will be replaced before they are being compromised. Since any cryptanalysis of usage will end up with an infinite list of key candidates, there will be equal number of candidates for the shared "Forever Key", and thus there will be no erosion in the secrecy of the Forever Key regardless of its level of use. The very idea of infinite number of interchangeable keys is so fundamentally different, that most of its applications are still unknown.

I. INTRODUCTION

In the early 70s of the last century, a thousand years old paradigm was blown away -- one key turned to two. They were closely related, they were each other reverse, but they were different, and knowing only one made it difficult to deduce the other. This simple departure from an age old established paradigm has catapulted cryptography to what it is today, the foundation of commercial cyber space.

If going from single key to two made such a big change, then what about changing from one or two to infinity?

It sounds weird, a key is something one tries to hide. It is harder to hide an infinite number of keys then to hide one, or two. How does one hide a single key? By drawing it from a large pool of keys -- "the key space". Similarly an infinite number of keys can be diluted to any desired degree in a corresponding infinite key space.

But why would someone wish to list infinite number of keys? One reason: interchangeability.

Let a cryptographic function *fam* convert a cryptographic input I to a cryptographic output O, by using a cryptographic key, K: $O = fam(I, K)$ with a corresponding function fam^{-1} reversing the process: $I = fam^{-1}(O, K)$

Further suppose that *fam*, and fam^{-1} are such that there are infinite number of keys K, K', K".... that would yield the same results over the same input I and output

$$O = fam(I, K) = fam(I,K') = fam(I, K'')=....$$
$$I = fam^{-1}(O, K) = fam^{-1}(O, K')= fam^{-1}(O, K'')= ....$$

The next question is what good does it offer?

Typically new paradigms open up new vista, which unfolds over time. Right off the bat two benefits come to mind: (i) community cryptography (ii) forever key cryptography.

The first one relates to a community of communication partners who shares a secret cryptographic instrument while not sharing actual keys. The second relates to breaking through the Shannon boundary that asserts that no finite key can extend mathematical secrecy to an infinite amount of processed data..

These two applications are discussed ahead. Later on we bring a particular family key cryptography function, *fam,* which is based on a special kind of cryptography called 'ordinal cryptography'. It is based on ordinal properties of data. Ordinal properties are shared by an infinite number of keys. So encryption, decryption or any other algorithm which relies only on the ordinal properties of a key, cannot be cryptanalyzed to extract the actual key used. In other words, ordinal cryptography is an embodiment of equivocation key cryptography, namely *'family key cryptography'.*

---

(1) Dpt of Elect' Eng. and Comp. Science, Case Western Reserve Univ, Cleveland, OH * BitMint, LLC Gideon.Samid@Case.Edu

A. *Vernam, Vernam+, Vernam++*

Vernam cipher offers mathematical secrecy based on bit-wise refreshment of the key. Every next bit of message is encrypted via an unused key bit. As a result Vernam consumes 1 bit of key for every bit of encrypted message.

We define a 'Vernam+' cipher as one where the key, K, is larger than the message, M, and hence the key may be refreshed after being used for m > 1 bits of the plaintext message, while maintaining mathematical secrecy. Case in point: the Unary Cipher [2]

We define 'Vernam++' cipher as one where for every triset of, M, K, C (plain message, key, and ciphertext), there are infinitely more keys that match the same M and C, and hence cryptanalysis of such a cipher will limit the key options to an infinite list. If the same key is used for sufficient amount of plaintext then cryptanalysis will nail it down and use this knowledge of K to compromise the next use of same key, K. Alas, if K is derived from a master key K_m using an open derivation algorithm, D, then the infinity of the K series will translate to infinity of options for K_m. When K_m now is used as a source to service the next key, K', then the prior knowledge regarding K does not transfer to knowledge of K' because of the 'infinity barrier' -- there are infinite number of options for K_m from the point of view of the attacker. Since the same cipher is used, we then have an infinite number of keys that could replace K' for the same pair of M and C. So while with use the identity of K' may be gradually brute force compromised, all this knowledge becomes useless, when K' is replaced with K" the same way that K' was replacing K. This procedure can repeat itself indefinitely and thereby achieves a *practical* breach through the Shannon limitation: practicing infinite use of a finite key while maintaining secrecy as nearly close to mathematical secrecy as desired. The replacement of keys: K, K', K" can be done after very small use of each key.

B. *Security*

In the abstract family key cryptography is mathematically secure. However, in practice the picture looks differently owing to the fact that no finite computing system can operate with infinitely large numbers. So while the use of family key cryptography will insure “at will” equivocation as to the key identity, an attacker is assumed to have a good guess as to the users’ computational power, speed and storage, and thereby bound the cryptographic equivocation so as to allow them to practice brute force, or more efficient cryptanalysis.

II. COMMUNITY KEY CRYPTOGRAPHY

Family Key cryptography enables a situation where n communication partners use n unique keys K_1, K_2, ...K_n which all belong to the same cryptographic family and hence these partners can encrypt and decrypt messages among themselves, using a family cryptography cipher, as if they shared the same key as in normal cryptography.

The actual key, K_i, held by partner i, identifies it as partner i. This identification can be put to good use. Three options presented:

- percolation tracing
- revocation management
- Intersecting Sub-Communities Management

Percolation Tracing relates to 'percolation' where a message percolates from one peer to the next with no central distribution. In between carrier nodes without the family key will simply pass the message on. Every message writer will add a hash of their key. The recipient of the message will add a hash of their key. The running message will carry its chain of custody, to be analyzed at any stage as to who received it.

Revocation Management: We consider a communication manager, CM, managing a conversation among n members m_1, m_2, ... m_n. Each member is given a unique member key k_1, k_2, ... k_n, passed in a secure mode, preferably off line. The communication manager, CM, is setting up n family keys. Member m_i holding k_i is assigned family key k^1_i.

Then the CM communicates to each member key adjusting data a_1, a_2, ... a_n such that:

$$k^1_i = k_i + a_i$$

Instead of addition, many other relations will do. Only member i in possession of k_i will be able to construct k^1_i. By so doing all n members share a key for family key cryptography which allows them to communicate freely as if they shared the same key. In fact they operate n different keys.

At any time the communication manager, CM, may switch from K^1 to K^2, and repeat the protocol done over K^1. The members will then use k^2_i , i=1,2,...n and communicate as if they share the same key while in fact they each have a different key.

This switch of keys to K^3, K^4, etc. can continue indefinitely. Any cryptanalysis done over the communication of the n members conversation will be lost when the keys are switched.

It is easy to add members to the group. Each new member receives its own personal key k_i and is treated like the former members.

It is easy to exclude members from the conversation. The CM simply switches to the next key and excludes the terminated members from the update. Their knowledge of the prior keys is not helpful in guessing and extracting the current key.

This at-will termination may take place for any reason, including suspicion of compromise. Should there be a suspicion that member i compromised its k_i, this member will be excluded until he or she comes forth to get a new personal key.

Intersecting Sub-Communities Management: A community of m communication partners may define s sub-communities, S_1, S_2, ... S_s, which may or may not intersect.

Namely a member i of the community may belong to none, one, or as many as s sub-communities. Each sub-community will wish to run private confidential conversations.

The Family Key solution to this challenge is to give member i (i=1,2,...n) t_i +1 family keys for the full community conversation, and for each of the t_i sub-community member i belongs to.

Member i will belong to sub-committee S_j (j=1,2,...s), which has m_j members. All these m_j members will have distinct keys which nonetheless operate as a family key setting, namely the members of S_j have distinct keys but can communicate freely owing to the Family Key setting.

In summary, every member i of the community will have distinct keys, unshared by any other members, but the community will be conversing in full community mode or in any sub-community mode with full privacy and confidentiality.

A CM will be able to update the keys as described above, as frequently as desired.

One operational setting for this communication mode is to assign each member of the community (s+1) secret keys that can be delivered off line. This will allow the CM to send key modification recipes to each member to put them on the full community conversation or on one, any or all the sub-communities conversations.

III. Forever Key Cryptography

Discussing a theoretical and Practical Framework to Extend the Service of a Cryptographic Key Indefinitely.

The longer one uses a cryptographic key, the higher the accumulated probability for a successful cryptanalysis thereof. A simple way to counter this erosion is for the users to share a master key from which to derive a succession of use-keys that are replaced before they are being compromised. While this measure does slow the attack, the accumulated cryptanalytic gains eventually chip away the secrecy of the master key. Family Key Cryptography could alleviate this threat.

Considering a set comprising a cryptographic input, I, a cryptographic output, O, and a key K that operates on either I, or O to generate the other. We show that Family Key Cryptography, will shield the master key from cryptanalytic erosion indefinitely. This technology is critical for instances where one communication station is beyond easy or possible access for key replacement, including: far off climate sensors, orbiting satellites, implanted medical devices, etc.

Consider two parties sharing a master key K_0. Party one then sends party two a recipe R_1 to convert K_0 to use-key K_1 ($K_1 = R_1(K_0)$. The parties then use K_1 for some measure, and then either party sends the other another recipe R_2 to convert the master key to a second use-key, $K_2 = R_2(K_0)$. K_2 is used for some measure, and then either party switches the communication to the next key K_3, K_4,.... indefinitely.

In normal cryptography, assuming the recipes R_1, R_2,..... are in the open, the on going switch of keys will eventually erode the secrecy of the master key, complying with Shannon's limitation on mathematical security. But with family cryptography, every time a key is switched, the previous cryptanalytic effort is rendered null and void. The most that a cryptanalyst will deduce from use of key K_i, for i=1,2,.... is the infinite series K_i, K'_i, K''_i that operate indistinguishably on the input that was used with K_i. Now given R_1, the cryptanalyst will face an infinite list of candidates for the master key,

$$K_0: K'_0 = R^{-1}{}_1(K'_1),\ K''_0 = R^{-1}{}_1(K''_1) = ...$$

Where R^{-1} is the reverse function for R.

After using t keys K_1, K_2, ... K_t, the cryptanalyst will remain with an infinite series for the master key. There are infinite key candidates that satisfy a finite number of limitation, t, levied on t infinite series. Say then: family key cryptography can be applied indefinitely with stable robust security based on finite shared key.

IV. Ordinal Cryptography

We now present an embodiment of family key cryptography: ordinal cryptography. It is based on ordinal properties of sets, which will be presented shortly. We will show that by limiting the use of a key so that only its ordinal properties are used, one establishes a robust family key cryptography because there are infinitely many keys that share ordinal properties and an attacker cannot pin them down.

A. Ordinal Properties of Sets

We define a numeric set as a set of elements comprising identity and numeric value. The identity of each element is set unique (namely no other element in the set bears the same identity mark), and the numbers are limited to real numbers.

We define a mathematical construct regarded as "ordinal property". Two numeric sets will be considered "ordinally equal" (or “ordinally equivalent”) if each pair of elements in one set may be matched with a corresponding pair of elements in the other set such that the pair-wise ordinal function of the respective values will be the same.

We define an ordinal function "o" over a pair of ordered real numbers a, and b: o(a,b) as follows:
If a < b then o(a,b) = -1
If a = b then o(a,b) = 0
If a > b then o(a,b) = 1

Let X and X' be two proper numeric sets comprised of n elements each. Every element $x \in X$ is matched with an element $x' \in X'$. X and X' will be regarded as "ordinally equivalent" if, and only if:

$$o(x_i, x_j) = o(x'_i, x'_j) ..... \text{for all } i,j = 1,2,...n$$

where x_i represents the numeric value of x_i; x_j represents the numeric value of x_j; x'_i represents the numeric value of x'_i; x'_j represents the numeric value of x'_j.

Illustration: Let X = $\{x_1, x_2, x_3, x_4\}$ with corresponding values: -1 12 100 6; and let: X' = $\{x'_1, x'_2, x'_3, x'_4\}$ with corresponding values: 30 32 33 31;

In this case X and X' will be ordinally equal because the ordinal value of each pair is the same in both sets. For example $o(x_2, x_4) = 1$, and $o(x'_2, x'_4) = 1$.

We examine the following case: X = $\{x_1, x_2, x_3, x_4\}$ with corresponding values: 1, 2, 2 300; and also: X' = $\{x'_1, x'_2, x'_3, x'_4\}$ with corresponding values: 1, 2, 3, 4;

In this case X and X' will be ordinally non-equivalent because $o(x_2, x_3) = 0$ while $o(x'_2, x'_3) = -1$

The fundamental Theorem of Ordinal Properties: Every numeric set has an infinite number of ordinally equivalent sets.

Proof: Let $x'_i = x_i + \delta$, for i=1,2,...n where δ is any real number. Accordingly:

$$x'_i - x'_j = (x_i + \delta) - (x_j + \delta) = x_i - x_j,$$

and hence:

$$o(x_i, x_j) = o(x'_i, x'_j) \text{..... for } i,j = 1,2,...n$$

And since: $-\infty < \delta < \infty$, the lemma is proven.

More generally: Let the n items of a numeric set X be organized according to their numeric values: $x_1, x_2, ... x_n$ such that for any $i < j$ $x_i \leq x_j$ for i,j=1,2,...n. Let Y be a numeric set of n items. Let us match the n items in both sets and mark them such that y_i is the match for x_i. We can then line up the n y items to match the order of the n x items. The numeric values will also be lined up: $y_1, y_2, ... y_n$.

In order for the sets X and Y to be ordinally equivalent we require:

$$y_i = (x_i + \rho_i(x_{i+1} - x_i))$$

for $0 < \rho_i < 1$ for all i=1,2,....n

Thereby given any set element x_i, the corresponding y_{i-1}, y_{i-2}... will be lower in value, and the corresponding y_i, y_{i+1}, y_{i+2}... will be higher in value, unless $x_i = x_{i+1}$, in which case $y_i = x_i$. And since x_i by construction is larger than x_i, x_{i-1}, ... and smaller than x_{i+1}, x_{i+2},... we have ordinal equivalence between the two sets. This proof of ordinal equivalence allows for an infinite value variety of $\rho_1, \rho_2, ...\rho_{i,...}$ which implies that there are infinite number of n y values that will be ordinally equivalent to the n x values.

We can also prove that for the case where $y_i = s * x_i$, where s is any positive real number. We can write:

$$y_j - y_i = s(x_j - x_i)$$

and hence: $o(y_j, y_i) = s * o(x_j - x_i)$

and since $s > 0$ we conclude that $o(y_j, y_i) = o(x_j, x_i)$ for all values of i,j=1,2....n

Conclusion: there is infinite number of choices for the values of s, and $\rho_1, \rho_2, ... \rho_n$, hence any ordinal set has infinite sets ordinally equal to it.

Matrix Elements as Numeric Sets: We consider a square matrix n x n listing m elements [1], [2],...[n] as columns and as rows with the n^2 cells of the matrix representing an assigned measure of distance between every one of the n elements to every other. The distance measure written as d_{ij} is representing the distance between elements i and j. These n^2 matrix cells may be viewed as a numeric set. Each cell has a unique identity designated by the value of i and j, and a unique value -- the positive distance between i and j.

Two such matrices M_1 and M_2 over a set of n elements [1], [2],....[n] may be numerically different but ordinally equivalent.

Complete Ordinal dis-Equality: Two numeric sets X, and X' with matching n elements will be regarded as completely dis-equal if and only if:

$$o(x_i, x_j) \neq o(x'_i, x'_j) \text{..... for every combination of } i,j = 1,2,...n$$

Illustration: Let X = $\{x_1, x_2, x_3, x_4\}$ with corresponding values: 1, 2,3, 4 ; and let: X' = $\{x'_1, x'_2, x'_3, x'_4\}$ with corresponding values: 4, 3, 2, 1

In this case X and X' will be ordinally completely dis-equivalent because the ordinal value of each pair is different in both sets.

Ordinal Similarity: Two numeric sets of matching n elements for which e pairs have the same ordinal value, and *0.5n(n-1)-e* pairs have different ordinal values between the sets, will be regarded as having ordinal similarity σ defined as

$$\sigma = e / 0.5n(n-1)$$

Ordinal similarity may range from zero (state of complete dis-equality) to one (state of equality):

$$0 \leq \sigma \leq 1$$

Ordinal Proprties of a Set: *Any property of a set which is the same for all the sets ordinally equal to it, will be regarded as an ordinal property of the set.*

The Ordinal Equivocation Premise: Let information quantity Z be generated by operating on a numeric set X such that Z is generated by an ordinal property of X. In that case no analysis of Z will be able to determine whether it was generated by X or by any of the infinite number of sets, which are ordinally equal to X. This is on account of the definition of an ordinal property of a set.

Ordinal Disruption Operation: Let D_o be an operation applied on a numeric set X, with n elements, such that the result is another numeric set, X* with the same number of elements, but such that the ordinal similarity, or say, ordinal similarity index, σ, between X, and X* is zero or close "enough" to it:

$$\sigma(X, X^*) \rightarrow 0$$

Such an operation, D_o, will be regarded as ordinal-disruption operator on X.

The Ordinal Matrix of a Numeric Set: A numeric set X with n elements may be associated with an n x n matrix constructed with the n elements both columns and rows, and where o_{ij} fitted for row i and column j, represents the ordinal value of elements x_i and x_j in the given order, $o_{ij} = o(x_i, x_j)$.

This matrix will be regarded as the "Ordinal Image" of the numeric set.

Ilustration. Let X = {x_1, x_2, x_3} or respective values 3, 5, 5, The corresponding ordinal matrix will be:

	x_1	x_2	x_3
x_1	0	-1	-1
x_2	1	0	0
x_3	1	0	0

.

Flatness of an Ordinal Matrix: Flatness is a measure of how many zeros there are in the ordinal matrix. Written in a matrix form, we formally account also for the main diagonal representing the 'fake' ordinal values of an element relative to itself, which is always zero. We therefore don't count these diagonal placed zeros. In the ordinal matrix representing a numeric set with n elements, there are z zeros, and the other 0.5n(n-1)-z matrix values are not zeros. If all the values of the ordinal matrix are zeros, then the represented set is regarded as 'completely flat set'. If there are no zeros (z=0), then the represented set is regarded a 'completely non-flat'. And for in between values of z, the represented set is regarded as partially flat. We compute:

$$\varphi = z/0.5n(n-1)$$

Ordinal Disruption Options: An efficient way to disrupt ordinal status of a numeric state is to apply some procedure involving modular arithmetic. Modular arithmetic will switch large values to small values. Combined with an enlargement option like squaring or raising to some power, p, a numeric set X may be turned to a numeric set X* where the similarity index between them σ(X, X*) is very low.

For example a set X with elements x_1, x_2, ... x_n will be operated on, element by element as:

$$x^*_i = x_i^2 \ MOD \ R$$

Where one may choose R to at least 2 or 3 times n,

Illustration: Let X = {x_1, x_2, x_3, x_4} with respective values 4, 6, 2, 0. Using R=10 we have:

$x^*_1 = 4^2$ MOD 10 = 6

$x^*_2 = 6^2$ MOD 10 = 6

$x^*_3 = 2^2$ MOD 10 = 4

$x^*_4 = 0^2$ MOD 10 = 0

The ordinal matrix for X:

	x_1	x_2	x_3	x_4
x_1	0	-1	+1	+1
x_2	+1	0	+1	+1
x_3	-1	-1	0	+1
x_4	-1	-1	-1	0

The ordinal matrix for X*:

	x_1	x_2	x_3	x_4
x_1	0	0	+1	+1
x_2	0	0	+1	+1
x_3	-1	-1	0	+1
x_4	-1	-1	-1	0

We record similarity of σ(X, X*) = 10/12 = 83%

Trying with another disruptive algorithm:

$$x^*_i = (x_i + 1)^2 \ MOD \ 11$$

We get X* = {x_1, x_2, x_3, x_4) = 11, 3, 5, 1

with ordinal matrix:

	x_1	x_2	x_3	x_4
x_1	0	+1	+1	+1
x_2	-1	0	-1	+1
x_3	-1	+1	0	+1
x_4	+1	-1	-1	0

with a lower similarity: σ(X, X*) = 7/12 = 58%

Given a reference numeric set one could explore disruption algorithms, D, that would process an input numeric set to one with very poor similarity to it. Alternatively one could use 'brute force' and with trial and error look for a D that would transport an input set to a very dissimilar output set.

B. Ordinal Ciphers

We describe ciphers that limit the use of a cryptographic key such that only ordinal properties will be applied, in order to establish unbreachable equivocation as to which of the possible keys that share the same ordinal properties was actually used.

At some point the communicating parties can apply an ordinal disruptor operation on their key, and then use the resultant disrupted key, K* for further applications in the crypto protocol. To the extent that every possible key (from the infinite list of possible keys) is disrupted such that it holds little similarity with the result of applying the disruption algorithm on the ordinal equivalent keys then the adversary will face the persistent equivocation of all possible disrupted keys. Thereby the life of the original (pre-disruption) key is extended. Alternatively, the parties can share a never-used master key, and keep disrupting it to successively used keys, guarding thereby the identity of the master key indefinitely.

We first make the foundational theoretical case for Breaching the Shannon Limit for mathematical secrecy, then we describe the general procedure for ordinal cryptography.

Generic Ordinal Cryptography: Generic Ordinal cryptography serves two parties sharing a key K_0.

They do:

1. Extract from K_0 a use key, K_i for i=1
2. Use K_i by relying only on its ordinal properties.
3. Repeat steps 1 and 2 after incrementing i → i+1.

By relying only on ordinal properties, (step 2), the users deny an attacker the option to extract from the use of K_i the numeric identity of K_i, from which to extract the identity of K_0 by reversing the extraction process which was: $K_0 \rightarrow K_1$. The users may hide this derivation process, and keep their attacker further in the dark, but per the analysis herein, even if the derivation (key transformation) algorithm is in the open, the master key K_0 cannot be deduced from cryptanalysis of K_1 because of ordinal equivocation.

Proof: Let the ordinal procedure above be practiced for i=t rounds. t keys have been used by the parties: K_1, K_2, ... K_t. The users aware of their level of exposure will stop using a key in favor of the next one "early", but even if they fail to do so, and use all keys for so long that the attacker figures out their ordinal structure, this will leave the attacker with t infinite series of possible keys that would have the same ordinal properties and as such will be indistinguishable for the attacker.

The attacker does not know the size of K_0. There are infinite number of options for K_0 that given the extraction procedures to K_1, K_2, .. K_t would have a key in each of the infinite series (t infinite series). Say then, that despite infinite use, $t \rightarrow \infty$, brute force cryptanalysis of the secret K_0 is doomed to fail.

For ordinal encryption there is another security feature.

Ordinal Encryption: When ordinal cryptography is used for encryption then the transformation $K_0 \rightarrow K_i$ may not necessarily be released in the open. Each encrypted message, C, may include the secret parameters how to transform K_0 to the next key. This will add difficulties for the attacker. When ordinal cryptography (OC) is used for hashing, for authentication, etc., this option does not present itself.

Implementing Generic Ordinal Cryptography. The shared secret K_0 may be constructed via a strong randomness source. It may be shared online with standard security measure -- with all the vulnerability thereto, or shared off line, with greater security. It may be written in software, firmware, or hardware. The transformation of K_0 to K_i may be carried out through a transformation algorithm D that maps K_0 to K_i based on some g transformation parameters h_1, h_2, h_g:

$$K_i = D(K_0, h_1, h_2, ...h_g)$$

where the g transformation parameters $\{h\}_g$ influence the output K_i.

While the algorithm D is arbitrary the values of the g parameters may be randomly picked for a trial and error application, to insure that the growing list of K1, K2, ... keys is sufficiently diverse (no great ordinal similarity).

4.3 SpaceFlip

SpaceFlip refers to cryptographic procedures where the key is a distance matrix between letters of an alphabet. An alphabet featuring l letters defines $q=0.5(l)(l-1)$ mutual distances, d_{ij} between letter L_i and letter L_j of the used alphabet. Each distance is identified by the two letters it connects and its numeric value. So the q distances represent a numeric set, Q.

The numeric set Q is used per its ordinal properties only for all the protocols and procedures presented. Therefore even a very smart and very capable cryptanalyst will only identify, at most, the infinite number of ordinally equivalent sets that will give the same outcome as the set that was actually used.

The users would share a master key K_0. They will derive from it a series of use keys K_1, K_2, ..., use each derived key for a prescribed measure, and then switch to the next key. Because the keys are used only per their ordinal properties, then any degree of cryptanalysis accomplished by an attacker on the usage of keys K_1, K_2, ... K_t, will be of no use, and of no value when the users switch to key K_{t+1}. And thereby the master key K_0 will serve the users indefinitely.

SpaceFlip defines the following functions over the alphabet distance matrix: NEXT, LINE, FIGURE.

NEXT: maps a letter L_i to a letter L_j for $i \neq j$: L_j = NEXT(L_i). The mapping is based on the distances from letter L_i to all other (n-1) letters. Letter L_j is the one that is the closest to L_i. If two or more letters share the same smallest distance with respect to L_i, then a well defined 'next equivocation resolution' kicks in to specify the identity of L_j. See [1] for details.

LINE: maps a letter L_i to a letter L_j for $i \neq j$: L_j = LINE(L_i, r). where:

$$L_j = LINE(L_i, r) = NEXT(NEXT,,,,(NEXT(L_i).....$$

applying NEXT r times. Where each time the NEXT letter is selected among the letters that have not been selected before. A LINE is a sequence of letters that don't repeat twice in the sequence.

FIGURE: maps a letter L_i to a letter L_j for $i \neq j$:

$$L_j = FIGURE(L_i, r_1, r_2, ... r_f)$$

where one draws a LINE from L_i to letter L_k = LINE(L_i, r_1), and then draws a second line form letter L_k to letter L_k' = LINE(L_k, r_2), and so on until L_f = LINE($L_k^{(f-1)}$, r_f).

Family Key Compliance: Any cryptographic operation based on the three functions: FIGURE, LINE, and NEXT is limited to making use of only ordinal properties of the key, which is the distance matrix. FIGURE is based on LINE, and LINE is based on NEXT, and NEXT is based on the ordinal order of distances from the pre-NEXT letter, x, to the post-NEXT letter, y. The actual values of the (n-1) distances from x to all the other letters play no role in evaluating NEXT. Only the ordinal order counts, which puts SpaceFlip in strict compliance with the Family Equivocation paradigm.

SpaceFlip Authentication: The environment: two mutually remote stations sharing a secret key K_0 are every so often reaching out to each other to establish secure communication. They wish to convince one another that they are not talking to an identity thief. They do so via an identity-proving dialogue which hides the identity of K_0 so that even after practicing such an identity-proving dialogue as many as desired times, no attacker will be able to extract the identity of K_0.

Procedure:

1. A Key derivation operator D is applied to K_0 to extract K_i for i=1

2. K_i is used an arbitrary t_i times via the SpaceFlip dialogue for the two stations to mutually prove themselves to the other.

3. Incrementing i to i+1, and repeating steps 1 and 2 for as long as needed.

Schematically, for $r \rightarrow \infty$:

$$[K_0 \rightarrow K_1][t_1 \text{ times use of } K_1]\ [K_0 \rightarrow K_2][t_2 \text{ times use of } K_2]\ \ldots\ldots\ [K_0 \rightarrow K_r][t_r \text{ times use of } K_r]$$

The two communicating stations are each equipped with a good source of randomness.

Illustration: Alice and Bob use a five letters alphabet L1, L2, L3, L4, L5 in conjunction with a SpaceFlip cipher. They share the following 5*5 master key (K_0):

0	8	11	5	15
8	0	7	12	6
11	7	0	5	10
5	12	5	0	9
15	6	10	9	0

They decide to derive from K_0 the first use key, K_1 based on a generic key transformation formula where distance between any two letters of the used alphabet, d_{ij} (distance between letters L_i and letter L_j) is derived via a formula of three parameters:

$$d_{ij} = D(m_{ij}, u, v, w)$$

where m_{ij} is the distance corresponding to d_{ij} in the master key, and u, v, and w are arbitrary numeric values. D -- is the disruption operator that disrupts (transforms) the mater key.

Alice and Bob further decide to implement the disruptor function D as follows:

$$d_{ij} = (m_{ij} + u)^v \ MOD\ w + 1$$

For K_1 Alice and Bob use u=4, v=1, w = 14 and hence

(K_1):

	L1	L2	L3	L4	L5
L1	0	13	2	10	6
L2	13	0	12	3	11
L3	2	12	0	10	1
L4	10	3	10	0	14
L5	6	11	1	14	0

Alice approaches Bob claiming to be Alice. Bob then wishes to authenticate Alice, so he challenges her to identify the end of a figure (y) defined as:

$$y = FIGURE(x, r_1, r_2) = FIGURE(L3, 2, 1)$$

Alice then applies the NEXT function to L3: L5 = NEXT(L3). And again L1 = NEXT(L5). She then identifies the next element to L1: L3 = NEXT(L1). Alice communicates to Bob: y=L3. Bob then concludes that he is talking to Alice since it appears she is in possession of K_1.

Since the key is small Alice and Bob decide to switch to K_2 right away. Alice selects random values to u, v, and w. *u=1, v=2, w=17:*

$$d_{ij} = (m_{ij} + 1)^2 \ MOD\ 17 + 1$$

So (K_2):

	L1	L2	L3	L4	L5
L1	0	10	10	3	16
L2	10	0	17	17	8
L3	10	17	0	3	5
L4	3	17	3	0	5
L5	16	8	5	5	0

Now it is Bob who approaches Alice to prove his identity, and Alice randomly defines a figure: y = FIGURE(x, r_1, r_2, r_3) =FIGURE(L2, 1, 2, 1).

Now Bob computes L_5 = NEXT(L2). When it comes to finding next to L5 both L3 and L4 compete. Bob then resorts to the agreed-upon equivocation resolution procedure. Accordingly Bob checks the NEXT distance for both. It turns out that L4 = NEXT(L3) and L3=NEXT(L4), so according to the agreed-upon NEXT equivocation resolution (NER) both are disqualified and L2 becomes L2=NEXT(L5). Then L5 = NEXT(L2), but L5 is disqualified since it is part of the LINE so the NEXT letter of the 2nd line is L1. Bob then computes L4 = NEXT(L1), and reports to Alice: y=L4. This answer convinces Alice that Bob is who he says he is because he appears to be in possession of K_2.

An attacker who somehow figures out K_1, will interpret Alice challenge as: L3 (L2 → L4; L4 → L2 → L5; L5 → L3), and will not pass the test.

In practice the size of the alphabet may be much larger than 5, and the confidence in the test is greater. For small alphabets the test can be carried out several times.

SpaceFlip Encryption: The overall procedure for SpaceFlip encryption is similar to SpaceFlip authentication. Keys are used and serially replaced to keep the degree of secrecy at the desired level. The difference is in the fact that with encryption the working key is used more extensively, and each time it is being used it provides raw material for cryptanalytic attack, it may be that for long communication sessions, keys will have to be switched to the next in the middle of a long transmission.

On the other hand Ordinal Encryption allows the users to encrypt the derivation parameters (the ordinalization disruption algorithm) for key derivation algorithm D_{i+1} in the encrypted message where key K_i is used. Thereby

denying the cryptanalyst the identity of the derivation algorithms.

Randomized FIGURE Encryption: Given an alphabet A comprising n letters, L1, L2, ... Ln, two communicating partners agree on a "space" for the alphabet, namely on a set of q=0.5n(n-1) distances between each letter to each other, where the distance between a letter to itself is zero, and all distances are positive real numbers.

A partner will transmit to the other partner a letter X ∈ A as follows:

The transmitter will:

1. Randomly select a letter Y ∈ A.
2. Arbitrarily select two integers g_l, and g_h -- the minimum and maximum number of LINES to build a FIGURE
3. Randomly select a value l between g_l and g_h
4. Randomly select *l*-1 values in the range *1* to (n-1): t_1, t_2, ... t_{l-1}
5. Define a FIGURE, FIG1, with Y as a starting letter, and (l -1) LINEs defined with t_1, t_2, ... t_{l-1}.
6. Compute Y' = FIG1(Y, t_1, t_2, ... t_{l-1})
7. Build a LINE starting with letter Y'. Do: repeatedly apply NEXT to Y', until, after t_l applications of NEXT on Y' the result is X: X = LINE(Y', t_l).
8. Define FIGURE FIG2 as X = FIG2(Y, t_1, t_2, ... t_l)
9. Communicate FIG2 to the receiving partner as follows: (i) send letter Y; (ii) for i=1, 2, ... *l* , randomly select an integer k in the range 0 to some arbitrary limit, then compute: $w_i = k*n + t_i$,

(iii) send to the recipient by order w_1, w_2, ... w_l

The receiving partner will use the shared space (the shared key) to compute FIG2 and identify X as the plaintext letter sent to them encrypted via FIG2. Aware of the value of n, the recipient will extract t_i from the respective w_i: $t_i = w_i$ MOD n.

The transmitting partner will send any size message letter by letter to the receiving partner. The partners can reverse roles and thereby conduct a conversation.

With sufficiently large values for n and g_l and g_h, the randomized selection of FIG1 will insure that for the life of the key, there will be no repetition of FIG2. However, the parties may maintain a log of FIG2 used before, and if the randomized selection generates a duplicate, then, this duplicate is dropped and the randomized selection is tried again.

The higher the values of g_l and g_h the greater the cryptanalytic burden, but the larger the size of the ciphertext relative to the plaintext. The user will determine the values of g_l and g_h according to the sensitivity of the encrypted material and the threat thereto. So when passing super sensitive cryptographic keys or text, the values of g_l and g_h will be high. When encrypting say audio or video stream it may be made vary low, say $g_l = 1$, $g_h = 3$.

One could also use small FIGURES as a baseline with occasional large FIGURES, and less frequently very large FIGURES, according to counter-cryptanalytic analysis.

Flatness management: Given an alphabet A comprised of n letters, the respective ordinal key will be comprised of q = 0.5n(n-1) numbers. The ordinal key is associated with a respective ordinal image. If the ordinal key is with zero flatness, then its ordinal image has no zeroes. φ = z/q = 0. This means that all the numbers in the ordinal key are different from each other. This in turn means that there are q! different keys. For alaphabet A being Base64 we have q=0.5*64*63 = 2016. And 2016! = 2.325849581 E+5788 is the size of the key space. This very large number may be further enlarged by increasing flatness to optimal levels.

Now we consider flatness. Let w numbers be featured with repetition: r_1, r_2, ... r_w. The remaining (non duplicating numbers) will be ordered in $(q- \Sigma\ r_i)!$ ways. The i-th duplicate will have $C^{r_i}_{q'i}$ ways to be assigned to the remaining locations, where

$$q'_i = q - \Sigma\ r_j \text{ .....for } j=1,2,...(i-1),$$

leading to a much larger hurdle for brute force cryptanalysis:

$$[\text{Partial Flatness Key Space}] = (q- \Sigma\ r_i)! * \pi\ (C^{r}_{q'i})$$

In summary some flatness increases the key variability. Although too much flatness will choke it. In the extreme, for $w=1$, and $r_w=q$, the key space collapses: {K} = 1

V. OUTLOOK

The cited applications for family key cryptography represent a range of applications from authentication to encryption, further applying to multi party cryptography and to extending the life of a finite shared secret. Albeit, the more interesting aspect are the yet unidentified applications. We have seen the dramatic impact of public key cryptography over the traditional symmetric practice. One should expect a hefty impact on account of extending one or two keys to an infinite series of keys.

VI. REFERENCE:

[1] G. Samid "SpaceFlip: Unbound Geometry Cryptography, https://eprint.iacr.org/2019/285"

[2] G. Samid "A Unary Cipher with Advantages over the Vernam Cipher" https://eprint.iacr.org/2020/389

[3] Shannon, Claude. "Communication Theory of Secrecy Systems", Bell System Technical Journal, vol. 28(4), page 656–715, 1949.