

ENCRYPTION METHOD AND SYSTEM FOR USER-CENTRIC CRYPTOGRAPHY

The present invention relates to a method and system of cybersecurity; and particularly relates to an encryption method and system on the basis of cognitive computing for xenomorphic cryptography or unusual form of cryptography. In particular, the present method and system relates to an encryption method and system by adopting one-time-password and one-time-decryption key.

BACKGROUND TECHNOLOGY

The increased digitization of the cyber world combined with the greed and ingenuity of attackers has raised cybercrime to an unprecedented level that demands a strong security system and method which is beyond the hard-to-solve problems in number theory.

- 10 Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it. And cryptography in cybersecurity is the study of secure communications techniques that allow only the sender and the intended receiver of a message to view its contents. Nowadays, cryptography is one of the most crucial aspects for cybersecurity and an indispensable tool in the information age especially at the dawn of 5G.
- 15 Classic cryptography makes use of keys whether symmetric or asymmetric and a hash function for record, storage and retrieval. While this category of cryptography algorithms is mostly based on hard-to-solve problems in number theory, which is vulnerable to quantum computers attacks. Quantum algorithms, such as Shor's algorithm (IEEE 1994) poses an essential threat to the security of cryptosystems based on number theory difficulties such as RSA cryptosystem.
- 20 In mid-2020, after spending more than three years examining new approaches to encryption and data protection that could defeat an assault from a quantum computer, the National Institute of Standards and Technology (NIST) has winnowed 69 submissions it initially received down to a final group of 15. The latest details on the project appear in the Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process (NISTIR 8309), which was published on 22 July 2020. NIST is now
- 25 undergoing the third round of public review. This "selection round" will help NIST to decide on the small subset of these algorithms that will form the core of the first post-quantum cryptography standard. The new standard will specify one or more quantum-resistant algorithms each for digital signatures, public-key encryption and the generation of cryptographic keys, augmenting those in FIPS 186-4, Special Publication (SP) 800-56A Revision 3 and SP 800-56B Revision 2, respectively.
- 30 It is said that at the end of this third round, NIST will standardize one or two algorithms for encryption and key establishment, and one or two others for digital signatures. Also, by the time they are finished, the review process will have been going on for five or six years, and some new ideas may be proposed in the interim.

Because of delays due to the COVID-19 pandemic, the third round has a looser schedule than past rounds. It is said that the review period will last about a year, after which NIST will issue a deadline to return comments for a few months afterward. Following this roughly 18-month period, NIST will plan to release the initial standard for quantum-resistant cryptography in 2022. It is true at the moment to say that there is not yet a standard for post-quantum encryption algorithm for wide application.

Although there are no standards for post-quantum encryption algorithms yet, there are several common approaches which may become the standard and comprise lattice-based cryptography approach, multivariate cryptography approaches and some other approaches include hash-based cryptography, code-based cryptography and supersingular elliptic curve isogeny cryptography.

Lattice-based cryptographic constructions hold a great promise for post-quantum cryptography, as they enjoy extraordinarily strong security proofs based on worst-case hardness, relatively efficient implementations, as well as great simplicity. In addition, lattice-based cryptography is believed to be secure against quantum computers. Indeed, the main alternative forms of public-key cryptography are schemes based on the hardness of factoring and related problems and schemes based on the hardness of the discrete logarithm and related problems. However, both factoring and the discrete logarithm are known to be solvable in polynomial time on a quantum computer. Furthermore, algorithms for factorization tend to yield algorithms for discrete logarithm, and vice versa. This further motivates the study of constructions based on alternative assumptions, such as the hardness of lattice problems.

Many lattice-based cryptographic schemes are known to be secure assuming the worst-case hardness of certain lattice problems i.e., if there exists an algorithm that can efficiently break the cryptographic scheme with non-negligible probability, then there exists an efficient algorithm that solves a certain lattice problem on any input. In contrast, cryptographic schemes based on, e.g., factoring would be broken if factoring were easy "on an average input," even if factoring were in fact hard in the worst case. However, for the more efficient and practical lattice-based constructions (such as schemes based on NTRU and even schemes based on LWE with more aggressive parameters), such worst-case hardness results are not known. For some schemes, worst-case hardness results are known only for certain structured lattices or not at all.

Multivariate cryptography is the generic term for asymmetric cryptographic primitives based on multivariate polynomials over a finite field F . In certain cases those polynomials could be defined over both a ground and an extension field. If the polynomials have the degree two, multivariate quadratics is mentioned. Solving systems of multivariate polynomial equations is proven to be NP-complete. That is why those schemes are often considered to be good candidates for post-quantum cryptography. Multivariate cryptography has been very productive in terms of design and cryptanalysis. Overall, the situation is now more stable and the

strongest schemes have withstood the test of time. It is commonly admitted that Multivariate cryptography turned out to be more successful as an approach to build signature schemes primarily because multivariate schemes provide the shortest signature among post-quantum algorithms.

Post-quantum cryptography algorithms commonly require larger key sizes than commonly used "pre-quantum" public key algorithms. There are tradeoffs to be made in key size, computational efficiency and ciphertext or signature size. The table 1 lists some values for different schemes at a 128 bit post-quantum security level.

Algorithm	Type	Public Key	Private Key	Signature
NTRU Encrypt	Lattice	6130 B	6743 B	
Streamlined NTRU Prime	Lattice	1232 B		
Rainbow	Multivariate	124 KB	95 KB	
SPHINCS	Hash Signature	1 KB	1 KB	41 KB
SPHINCS+	Hash Signature	32 B	64 B	8 KB
BLISS-II	Lattice	7 KB	2 KB	5 KB
GLP-Variant GLYPH Signature	Ring-LWE	2 KB	0.4 KB	1.8 KB
New Hope	Ring-LWE	2 KB	2 KB	
Goppa-based McEliece	Code-based	1 MB	11.5 KB	
Random Linear Code based encryption	RLCE	115 KB	3 KB	
Quasi-cyclic MDPC-based McEliece	Code-based	1232 B	2464 B	
SIDH	Isogeny	751 B	48 B	
SIDH (compressed keys)	Isogeny	564 B	48 B	
3072-bit Discrete Log	not PQC	384 B	32 B	96 B
256-bit Elliptic Curve	not PQC	32 B	32 B	65 B

Table 1

One of the main challenges in post-quantum cryptography is on the implementation of potentially quantum safe algorithms into existing systems. There are tests done, for example, by Microsoft Research implementing PICNIC in a Public Key Infrastructure (PKI) using Hardware security modules.

Shi Jinjing and Chen Shuhui et al. (*Nature*, 2020) have proposed a continuous-variable quantum neural network (QNN) for designing a quantum cryptography algorithm that can set cryptography free from attacks on numbers. QNN was firstly proposed by Kak (1995) and it provided a potential solution to design a novel encryption and decryption mechanism. Progress has been made with quantum neural cryptography (QNC) and experiments on Strawberry fields are promising but this is far from being deployable.

With the development of QNN, there are new methods or devices coming in this area, such as in CN111563186 A, Baidu adopted QNN to develop a quantum data storage method and reading method and device and a computing device, related to the field of quantum computers, the storage method includes: making the first quantum neural network to act on the initial state to obtain the output quantum state; computing the loss function based on the target quantum state and the quantum state output by the first quantum neural network, where the dimension of the first quantum neural network relates to the dimensions of the target quantum, and the loss function corresponds to the distance between the target quantum state and the quantum states output by the first quantum neural network; adjusting the parameters of the first quantum neural network according to said loss function so as to conduct iteratively training to the first quantum neural network until it reaches the preset iteration stop condition; saving the parameters of the first quantum neural network after training into the hardware device.

In CN111582210A, Shenyang University of Technology developed a human behavior recognition method based on quantum neural network, comprising the following steps: collecting human body motion images and gray scaling each frame; storing the human body motion image in the quantum state to form the NEQR volume sub-image; detecting quantum image foreground: for NEQR quantum images, adopting the background difference method and the ViBe algorithm to respectively detect stable objects and moving objects; using Hadamard door edge detection algorithm to extract the edge information of the human body action from the "motion target" in the third step, and forming the edge quantum images; based on quantum black box convolution method, performing convolution operation on the "edge quantum image" in the fourth step, to extract the feature points of the action; and making a recognition sample set; reading the trained weights and constructing a quantum BP neural network to identify samples.

Password generators are simple daily life tools that encrypt the identity or factorize the authentication but have become burdensome. Password embodiment including account creation and further logins have become the main targets of cyber-attackers. The increased digitization combined with the greed and ingenuity of attackers has raised cybercrime to an unprecedented level that demands a strong technological response.

In CN103840936A, Shandong Institute of Quantum Science and Technology developed an encryption transmission method for quantum cryptographic networks, which uses classic encryption and quantum one-time encryption for business data to perform shunt transmission, the continuation of business data encryption transmission is increased and also the reliability of business data transmission can be improved; at the same time, it reduces the consumption of quantum key data, and reduces the quantum key generation load of the quantum cryptographic network, and reduces the cost of encrypted transmission of quantum cryptographic network business data.

Quantum cryptography is the science of exploiting quantum mechanical properties to perform cryptographic tasks. The best known example of quantum cryptography is quantum key distribution which offers an information-theoretically secure solution to the key exchange problem. The advantage of quantum

cryptography lies in the fact that it allows the completion of various cryptographic tasks that are proven or conjectured to be impossible using only classical (i.e., non-quantum) communication. For example, it is impossible to copy data encoded in a quantum state. If one attempts to read the encoded data, the quantum state will be changed (no-cloning theorem). This could be used to detect eavesdropping in quantum key distribution. The best-known and developed application of quantum cryptography is quantum key distribution (QKD), which is the process of using quantum communication to establish a shared key between two parties (Alice and Bob, for example) without a third party (Eve) learning anything about that key, even if Eve can eavesdrop on all communication between Alice and Bob. If Eve tries to learn information about the key being established, discrepancies will arise causing Alice and Bob to notice. Once the key is established, it is then typically used for encrypted communication using classical techniques.

While QKD is seemingly secure, its applications face the challenge of practicality. This is due to transmission distance and key generation rate limitations. Ongoing studies and growing technology have allowed further advancements in such limitations.

Artificial Intelligence modeling has been integrating research results from different fields of physics, neurobiology, biochemistry, and neuroscience to promote the adoption of inspired mechanisms that can contribute to user protection and cybersecurity in general.

BRIEF SUMMARY

The present invention relates to an encryption method and system on the basis of cognitive computing for xenomorphic cryptography or unusual form of cryptography. This form uses cognitive computing that combines incoming signal processing with neuromorphic artificial neural network for contextual password/passphrase and private key generation, encryption, and dynamic evolution by elongation and retraction over time. With the encryption method and system based on cognitive computing, the user cognitive variability itself becomes the password/passphrase or decryption key's originator so the generated password/passphrase/key can take unpredictable forms also called xenomorphic in this invention. In particular, the present method and system relates to an encryption method and system by adopting one-time decryption key.

The present invention provides an encryption method comprising: Generating a Functional Neural Network or KeyNode (KN) of the system by programming a chain of multiple nodes also called Artificial Mirror Neurons (AMN) based on captured information of reaction time and emotional response to simple tasks; one task per mode, racing the nodes in the Functional Neural Network or KeyNode (KN) and their many-valued output services as an encryption device or cipher for the time of use; Classifying the KeyNodes where each KeyNode comprises multiple nodes held in a chain which can be generated successively, elongated, and can also evolve by retraction of nodes or decay after a certain number of nodes has been generated; Encoding KeyNodes based on infinite proposition captured by interactive cognitive computing, forming an irregular

tetragon matrix, and allowing dynamic encoding of unusual forms of cryptography; Generating the boundary of the KeyNode, where the minimum size is defined as A, the maximum size is defined as B, and the time of maximum size of the KeyNode is defined as $C = B - (A - 1)$; Generating one-time-password or passphrase for decryption at the user's time of use based on the sum of intrinsic values of the nodes in the racing network at this time and adopting the generated password for authentication or adopting the generated passphrase to generate a one-time-pair of keys for encrypted communication to form a one-time-pair of keys for encrypted communication; Transmitting the one-time encryption key (public key) over any unsecured channel to selected senders for encrypting the plaintext to become a ciphertext; Decrypting the ciphertext of the senders at the receiver's end using the one-time decryption key (private key) and the passphrase to regenerate the plaintext. The passphrase serves as the ultimate stage of decryption before getting the plaintext.

In one aspect of the present invention, wherein the steps of programing Artificial Mirror Neuron or node, further comprising capturing reaction and emotional response where the reaction time is the time taken by the user to react to the cognitive task between input and output, which determines the motor core radius size; and the emotional response is the emotional response to the said cognitive task captured around 500ms from the input time, which determines the sensory core radius size; and determining the velocity which is inversely to the age information; and deriving the angular velocity ω from said velocity and the angle θ of both said motor core and sensory core; the steps of racing the chain of nodes in the Functional Neural Network or KeyNode (KN), further comprising programming multiple nodes, and the chain of nodes in the Functional Neural Network or KeyNode (KN) serves as an encryption device or a cipher; said KeyNodes comprise one first signup KeyNode or KN1 which contains at least 8 nodes; the steps of generating a passphrase, further comprising generating said passphrase for use at time 2 (T_2), on the base of each node angle θ changes constantly between time 1 (T_1) and time 2 (T_2), and the sum of angle θ of all nodes also called sum of angle θ_2 at time 2 depends on the position of the radiuses or the value of the velocities of the nodes at time 2; and the method further comprising defining each node by said motor core radius, said sensory core radius, two angle θ_1 and angle θ_2 , two velocity v_1 and velocity v_2 , and two angular velocity ω_1 and angular velocity ω_2 at the time T_1 and T_2 , and the time of each node creation in minute and each new node has a specific time of creation T_1+n that will refer to the initial nodes time of creation when made at signup or T_1 ; and when reaching the time T_2 or time of applying said passphrase, creating a new node by generation-elongation, and summing up said motor core and sensory core radius theta angles or angular velocities for all racing nodes in the KN arriving at T_2 , and adopting said passphrase for authentication.

The present invention provides another encryption method comprising: Generating a Functional Neural Network or KeyNode (KN) of the system by programming a chain of multiple nodes also called Artificial Mirror Neurons (AMN) based on captured information of reaction time and emotional response to a simple task; racing the nodes in the Functional Neural Network or KeyNode (KN) as an encryption device or cipher for the time of use; Classifying KeyNodes of multiple nodes where each KeyNode comprises multiple nodes held in a chain which can be generated successively, elongated, and can also evolve by retraction of nodes or decay after a certain number of nodes has been generated; Encoding KeyNodes based on infinite

proposition captured by interactive cognitive computing, forming an irregular tetragon matrix, and allowing dynamic encoding of unusual form of cryptography; Generating the boundary of the KeyNode, where the minimum size is defined as A, the maximum size is defined as B, and the time of maximum size of the KeyNode is defined as $C = B - (A - 1)$; Generating one-time-private key for decryption at the user's time of use based on the concatenation of intrinsic values of the nodes in the racing network at this time and adopting the generated private key to generate a one-time public key to form a one-time-pair of keys for encrypted communication; Transmitting the one-time public key over any unsecured channel to selected senders for encrypting the plaintext to become a ciphertext; Decrypting the ciphertext of the senders at the receiver's end using the one-time private key to regenerate the plaintext data; Alternatively using the one-time key for symmetric key encryption of user's personal data for secure storage.

In one aspect of another embodiment of the present invention, wherein said the public key and private key are the same by using said symmetric key encryption.

The present invention also provides corresponding system implementing the above encryption methods.

BRIEF DESCRIPTION OF SEVERAL VIEWS OF THE DRAWINGS

The drawings referenced herein form a part of the specification. Features shown in the drawings are meant as illustrative of only some embodiments of the invention, and not of all embodiments of the invention unless otherwise explicitly indicated.

Figure 1(a) and Figure 1(b) are the mechanisms of the first chain of nodes (the signup KeyNode, a KN of 8 nodes numbered N1 to N8) of the present invention.

Figure 2(a) indicates one example of the formation of chains of nodes (KeyNodes) of the present invention, with a linear decay mechanism starting with KN16.

Figure 2(b) indicates another example of the formation of chains of nodes (KeyNodes) of the present invention, with a random decay mechanism starting with KN16.

Figure 3 indicates one example of the encoding of chains of nodes (KeyNodes) in XenoBase of the present invention.

Figure 4(a) indicates KeyNode linear decay or simple retraction, i.e. type 1, in one example of the present invention.

Figure 4(b) indicates KeyNode random decay or disorganized retraction, i.e. type 2, in one example of the

present invention.

Figure 4(c) indicates KeyNode preset decay or organized retraction, i.e. type 3, in one example of the present invention.

5 Figure 4(d) indicates KeyNode binomial distributed decay or probabilistic retraction with expectation of 1 node removed per time, i.e. type 4, in one example of the present invention.

Figure 5 illustrates theta angle or angular velocity of the motor and sensory cores used in the present invention.

Figure 6 illustrates the many-valued theta angle of the motor core during a complete revolution of the present invention.

10 Figure 7 illustrates the changes of the angular velocity value during the course of the motor core of the present invention.

Figure 8 describes the workflow of the present invention.

Figure 9 refers to the Rijndael S-box of the step of substituting Bytes of the AES-128.

15 Figure 10 describes the organic workflow of one embodiment of the present invention using KeyNode with ECC.

Figure 11 describes the organic workflow of another embodiment of the present invention using KeyNode with RSA.

Table 1 indicates one example of the post-quantum cryptography approach in the prior art which lists some values for different schemes at a 128 bit post-quantum security level.

20 Table 2 shows the Question answer (“Qa”) types of the first encoding scheme in XenoBase;

Table 3 shows the Binary visual-choices (“Bv”) types of the second encoding scheme in XenoBase;

Table 4 shows the Binary Iconic-choices (“Ic”) types of the third encoding scheme in XenoBase;

Table 5 shows the Ternary visual-choices (“Tv”) types of the fourth encoding scheme in XenoBase;

25 Table 6 shows the comparison on the security levels between different keys’ sizes of RSA and ECC encryption keys.

Table 7 is a matrix example of the message which is broken down into 128-bits blocks in hexadecimal expressions.

Table 8 is a matrix example of the first 16 bytes of the encryption key of a 128-bit input block which is arranged in the form of a state array.

5 Table 9 refers to the require number of bits for the generation of RSA private keys.

Table 10 shows the necessary minimum size of the KeyNode for the integration of RSA and KeyNode.

DETAILED DESCRIPTION

10 It is understood that the components, as generally described and illustrated in the Figures herein, may be arranged and designed in a wide variety of configurations. Thus, the following detailed description of the embodiments of the apparatus, system, and method, as presented in the Figures, is not intended to limit the scope of the invention, as claimed, but is merely representative of selected embodiments.

The functional unit described in this specification with elements labeled as managers. A manager may be implemented in programmable hardware devices such as field programmable gate arrays, programmable array logic, programmable logic devices, or the like. The manager may also be implemented in software for
15 execution by various types of processors. An identified manager of executable code may, for instance, comprise one or more physical or logical blocks of computer instructions which may, for instance, be organized as an object, procedure, function, or other construct. Nevertheless, the executables of an identified manager need not be physically located together but may comprise disparate instructions stored in different locations which, when joined logically together, comprise the manager and achieve the stated purpose of the
20 manager.

Indeed, a manager of executable code could be a single instruction, or many instructions, and may even be distributed over several different code segments, among different applications, and across several memory devices. Similarly, operational data may be identified and illustrated herein within the manager and may be embodied in any suitable form and organized within any suitable type of data structure. The operational data
25 may be collected as a single data set or may be distributed over different locations including over different storage devices, and may exist, at least partially, as electronic signals on a system or network.

Reference throughout this specification to "a select embodiment," "one embodiment," or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment. Thus, appearances of the phrases "a select embodiment," "in one
30 embodiment," or "in an embodiment" in various places throughout this specification are not necessarily referring to the same embodiment.

Furthermore, the described features, structures, or characteristics may be combined in any suitable manner in one or more embodiments. In the following description, numerous specific details are provided, to provide a thorough understanding of embodiments. One skilled in the relevant art will recognize, however, that the invention can be practiced without one or more of the specific details, or with other methods, components, materials, etc. In other instances, well-known structures, materials, or operations are not shown or described in detail to avoid obscuring aspects of the invention.

The illustrated embodiments will be best understood by reference to the drawings, wherein like parts are designated by like numerals throughout. The following description is intended only by way of example, and simply illustrates certain selected embodiments of devices, systems, and processes that are consistent with the invention as claimed herein.

In the PCT/CN2019/124550, METHOD AND SYSTEM FOR INFORMED DECISION MAKING is provided. The present invention is based on the AMN disclosed in the PCT/CN2019/124550. The contents of the PCT patent application PCT/CN2019/124550 are all comprised into the present invention.

In the PCT/CN2020/127474, ENCRYPTION METHOD AND SYSTEM FOR XENOMORPHIC CRYPTOGRAPHY is provided. The present invention is also based on the KeyNode disclosed in the PCT/CN2020/127474. The contents of the PCT patent application PCT/CN2020/127474 are all comprised into the present invention.

Embodiment 1: Generating password or passphrase from KeyNode.

PROGRAMMING OF THE AMN(s), FNN, KEYNODE GENERATION

The programming structure of the Artificial Mirror Neuron (AMN) or Node is tri-dimensional, which means 3 dimensions (two spatial and one temporal) are drawn from 3 cognitive datapoints related to the performance of the user when exposed to a simple cognitive task, and repeatedly: the reaction time is captured in second, as a reaction to the cognitive task (between input and output) which determines the motor core radius size among 8 categories or integer 0-7; the emotional response is captured in emotional type, as a reaction to the same cognitive task which determines the sensory core radius size among 8 types or integer 0-7; the velocity is determined inversely to the age of the user in degree per minute. The angular velocity ω can be derived from the velocity and the theta angle of both motor and sensory cores.

When multiple nodes are programmed by the user, it forms a chain, and the nodes in the chain will start racing in the Functional Neural Network also called a **KeyNode** (KN) in this invention which has dynamics that serve as an encryption device or cipher. The first KN1 should contain a minimum of 8 nodes and is programmed by the user at the time of account creation also called signup. Figure 1(a) and (b) describes the

mechanism of the first KN of the present invention. Figure 1(b), describes the changes of the value of angle θ of each node between two different times arbitrarily called time 1 (T1) and time 2 (T2), indicating the sum of all nodes angle θ or $\Sigma\theta_2$ at time 2 which contributes to the generation of the password/passphrase to be used at time 2.

- 5 Each node is defined by the characteristics of a motor core radius, a sensory core radius, 2 theta angles, 2 velocities and 2 omega angular velocities.

Each node is defined by the time of its creation T in minute and each new node will have a specific time of creation $T1+n$ that will refer to the initial nodes time of creation when made at signup or T1.

- When a T2 or time of use of the password/passphrase, a new node will be created by generation-elongation and all racing nodes in the KN arriving at T2 will have their dual cores' radiuses position or their angular velocities summed up and the final result or password will be transmitted using a Json Web Token (JWT) to the preselected end users/accounts/backends where the password will be used for authentication.

ENCODING IN XENOBASE

- The present invention introduces a new base for encoding KeyNodes called **XenoBase** which contains the infinite proposition captured by interactive cognitive computing.

New encoding schemes such as question answer-types, binary visual-choices, binary iconic-choices, ternary visual-choices are replacing the usual set of numbers, letters, and symbols of traditional bases.

- Forming an irregular tetragon matrix, these nodes allow dynamic encoding of unusual form of KeyNodes so passwords/passphrases/keys can fit the preset security requirements in term of logic complexity, mutability, and timeliness.

Types of the first encoding scheme are shown in Table 2, wherein there are 12 Question answer-types or Qa if NS=A; RD=B; HA=C; and bold fonts are highlighting the Major types.

TYPE Qa1	A+C=A	TYPE Qa2	A+C=C
TYPE Qa3	A+B=A	TYPE Qa4	A+B=B
TYPE Qa5	B+C=B	TYPE Qa6	B+C=C
TYPE Qa7	B+A=B	TYPE Qa8	B+A=A
TYPE Qa9	C+A=C	TYPE Qa10	C+A=A
TYPE Qa11	C+B=C	TYPE Qa12	C+B=B

Table 2

- Types of the second encoding scheme are shown in Table 3, wherein there are 12 Binary visual-choices or Bv if NS=A; RD=B; HA=C; and bold fonts are highlighting the Major types.

TYPE Bv1	A+C=A	TYPE Bv2	A+C=C
TYPE Bv3	A+B=A	TYPE Bv4	A+B=B
TYPE Bv5	B+C=B	TYPE Bv6	B+C=C
TYPE Bv7	B+A=B	TYPE Bv8	B+A=A
TYPE Bv9	C+A=C	TYPE Bv10	C+A=A
TYPE Bv11	C+B=C	TYPE Bv12	C+B=B

Table 3

Types of the third encoding scheme are shown in Table 4, wherein there are 12 Binary Iconic-choices or Ic if NS=A; RD=B; HA=C; and bold fonts are highlighting the Major types.

5

TYPE Ic1	A+C=A	TYPE Ic2	A+C=C
TYPE Ic3	A+B=A	TYPE Ic4	A+B=B
TYPE Ic5	B+C=B	TYPE Ic6	B+C=C
TYPE Ic7	B+A=B	TYPE Ic8	B+A=A
TYPE Ic9	C+A=C	TYPE Ic10	C+A=A
TYPE Ic11	C+B=C	TYPE Ic12	C+B=B

Table 4

Types of the fourth encoding scheme are shown in Table 5, wherein there are 18 Ternary visual-choices or Tv if NS=A; RD=B; HA=C; Major/minor types cannot be separated.

TYPE Tv1	A+B+C=A
TYPE Tv2	A+B+C=B
TYPE Tv3	A+B+C=C
TYPE Tv4	A+C+B=A
TYPE Tv5	A+C+B=B
TYPE Tv6	A+C+B=C
TYPE Tv7	B+A+C=B
TYPE Tv8	B+A+C=A
TYPE Tv9	B+A+C=C
TYPE Tv10	B+C+A=B
TYPE Tv11	B+C+A=C
TYPE Tv12	B+C+A=A
TYPE Tv13	C+A+B=C
TYPE Tv14	C+A+A=A
TYPE Tv15	C+A+A=B
TYPE Tv16	C+B+A=C
TYPE Tv17	C+B+A=B
TYPE Tv18	C+B+A=A

Table 5

The combination forms an irregular quadrilateral polygon or tetragon matrix.

Figure 3 describes the encoding of KN in XenoBase, using 4 different colors to identify the components. Qa in red, Bv in green, Ic in yellow, and Tv in blue. In this example, KN1 has been preset such as

5 “QaBvBvQaQaTvQaIc” but can be made randomly.

The KN can also be programmed according to preset security requirements in term of length, and combinations can be set at first before the user starts programming the first KeyNode.

KEYNODE DECAY MECHANISMS

Figure 4a illustrates **linear decay** or simple retraction (type 1) taking place between KeyNode 16 and
10 KeyNode 21 as an example. Linear decay means that the earlier a node is generated, the earliest the node will be removed. Therefore, the removal of nodes starts from the first node, and then the second, etc., in linear order.

Figure 4b illustrates **random decay** or disorganized retraction (type 2) taking place between KeyNode 16 and KeyNode 21 as an example. Random decay or disorganized retraction is capable of selecting one node
15 from the list of existing nodes in the KeyNode every time and remove it. This can be expressed as a selection from the existing nodes each with a uniform probability $1/N$ to be selected if N is the number of existing nodes.

Figure 4c illustrates **preset decay** or organized retraction, ex: 1-3-6/2-4-7 (type 3) taking place between KeyNode 16 and KeyNode 21 as an example. Preset decay or organized retraction is similar to linear decay
20 that it can remove one node in a fixed order but differs in the sense that linear decay removes the nodes based on the time they are generated while preset decay removes the node based on a preset order in which the generation time is not important. It can be the 1st node removed, then the 3rd node and the 6th, etc.

Figure 4d illustrates the **binomial distributed decay** or probabilistic retraction with expectation of 1 node removed per time (type 4) taking place between KeyNode 16 and KeyNode 21 as an example. Binomial
25 distributed decay or probabilistic retraction sets a removal probability to each existing node, and the whole removal process follows a binomial distribution with expectation to be 1 node removed each time.

AUTHENTICATION MECHANISM FORMULATION:

- The KeyNode generation by the user is lower bounded by the minimum KN size, called A, and is upper bounded by a maximum size, called B for which keeping some flexibility. If T1 is the time of generation, or signup, then the time of maximum size, called C, will be $C = B - (A - 1)$.

5 For example, if $B = 20$ and $A = 8$, time of maximum size should be $C = 20 - (8 - 1) = 13$. Once the KeyNode reaches the maximum size, the decay mechanism should start to ensure that the KeyNode size remains stable around the maximum size.

- The formulation of the KeyNodes is based entirely on the formulation of the Functional Neural Network FNN. Even though it is known that FNN can have predictive power on decision making of the users, the current formulation will simply focus on the pure functionalities of the FNN.

10 Each Artificial Mirror Neuron (AMN) of the FNN has two cores, namely, the motor core M and sensory core S. The password/passphrase generation will combine the values of both cores and can either use the theta angle or the angular velocity of the motor and the sensory cores can be used, and the details are provided below in Figure 5.

Figure 5 illustrates the mirror structure of the AMN with the motor and sensory cores.

Figure 6 illustrates the many-valued theta angle of the motor core during a complete revolution.

20

1. Theta Angle per core

For the motor cores, the following steps are provided:

$$\text{Individual theta angle from M of each node at time } T = (v_m T) \% (2\pi r_1)$$

$$\text{Sum of theta angles from M of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i})$$

25

where v_m is the velocity of the motor core in an AMN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_1 is the radius of the motor core in an AMN, r_{1i} is the radius of the motor core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T .

Similarly, for the sensory cores, the following steps are provided:

$$\text{Individual theta angle from S of each node at time } T = (v_s T) \% (2\pi r_2)$$

$$\text{Sum of theta angles from S of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$$

where v_s is the velocity of the sensory core in an AMN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_2 is the radius of the sensory core in an AMN, r_{2i} is the radius of the sensory core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T .

Figure 7 illustrates the changes of the value of the angular velocity during the course of the motor core over time.

2. Angular Velocity per core

For the motor cores, the following steps are provided:

$$\text{Individual angular velocity from M of each node at time } T = \frac{v_m}{2\pi r_1}$$

$$\text{Sum of angular velocities from M of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}}$$

where v_m is the velocity of the motor core in an AMN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_1 is the radius of the motor core in an AMN, r_{1i} is the radius of the motor core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T .

Similarly, for the sensory cores, the following steps are provided:

$$\text{Individual angular velocity from S of each node at time } T = \frac{v_s}{2\pi r_2}$$

$$\text{Sum of angular velocities from S of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_s is the velocity of the sensory core in an AMN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_2 is the radius of the sensory core in an AMN, r_{2i} is the radius of the sensory core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T .

- There are four encoding schemes in the XenoBase, namely, the Question answer-types (Qa), the Binary visual-choices (Bv), the binary Iconic-choices (Ic) and the Ternary visual-choices (Tv). Each time the user will be randomly prompted to answer a question or make a choice from either one of these four categories, which will then generate a new node in the KeyNode.

5 Therefore, the cipher will combine the reactional nature of the response given by the user to a question/choice, each one at a time, with the timeliness of the response so each time a new question/choice is submitted, and an answer is captured, a new node appears and refreshes the FNN.

- 10 • In more details, KN elongation is made by addition of new nodes, i.e. 1 new node per each password/passphrase generation. The new node generated at each time step will not affect the password/passphrase at time it is generated, but only affect the future passwords/passphrases.

15 For example, if at time T_1 there are N_{T_1} nodes and if the user answers a question of one from XenoBase which starts a new node, then this $N_{T_1} + 1$ node will not be accounted for the password/passphrase at time T_1 but for future passwords/passphrases only, for example at time T_2 .

So, in mathematical formulation, have theta angles at time T_1 :

20 Sum of theta angles from M of the whole FNN at time $T_1 = \frac{1}{N_{T_1}} \sum_{i=0}^{N_{T_1}} (v_{mi} T_1) \% (2\pi r_{1i})$

and theta angles at time T_2 :

sum of theta angles from M of the whole FNN at time $T_2 = \frac{1}{N_{T_2}} \sum_{i=0}^{N_{T_2}} (v_{mi} T_2) \% (2\pi r_{1i})$

where $N_{T_2} = N_{T_1} + 1$.

25 Similarly, if considering angular velocities, having angular velocities of the motor core at time T_1 :

Sum of angular velocities from M of the whole FNN at time $T_1 = \frac{1}{N_{T_1}} \sum_{i=0}^{N_{T_1}} \frac{v_{mi}}{2\pi r_{1i}}$

and angular velocities at time T_2 :

$$\text{Sum of angular velocities from M of the whole FNN at time } T_2 = \frac{1}{N_{T_2}} \sum_{i=0}^{N_{T_2}} \frac{v_{mi}}{2\pi r_{1i}}$$

where $N_{T_2} = N_{T_1} + 1$.

- The decay mechanism of KeyNode is modelled to adopt different forms with different properties. They include linear decay (simple retraction), random decay (disorganized retraction), preset decay (organized retraction) and binomial distributed decay (probabilistic retraction).

1. Linear decay

Linear decay or simple retraction means that the earlier a node is generated, the earliest the node will be removed. Therefore, the removal of nodes starts from the first node, and then the second, etc., in linear order.

2. Random decay

Random decay or disorganized retraction is capable to select one node from the list of existing nodes in the KeyNode every time and remove it. This can be expressed as a selection from the existing nodes each with a uniform probability

$$\frac{1}{N}$$

to be selected if N is the number of existing nodes. So, selecting a number from [1, 2, 3, ..., N] each with probability $\frac{1}{N}$.

3. Preset decay

Preset decay or organized retraction is similar to linear decay that it can remove one node in a fixed order but differs in the sense that linear decay removes the nodes based on the time they are generated while preset decay removes the node based on a preset order in which the generation time is not important. It can be the 1st node removed, then the 3rd node and 6th, etc.

4. Binomial distributed decay

Binomial distributed decay or probabilistic retraction sets a removal probability to each existing node, and the whole removal process follows a binomial distribution with the expectation to be 1 node removed each time.

In details, the removal probability is set to be

$$5 \quad \frac{1}{N}$$

for each node, but unlike the random decay, it is not a selection problem from [1, 2, 3, ..., N]. Instead, each node can be removed independently with probability $\frac{1}{N}$. So, it is possible that 0 node is removed at this time and 2 nodes or more will be removed at the next time.

10 The number of nodes to be removed indeed follows a binomial distribution, which explains why it is called binomial distributed decay. Mathematically, if there are N existing nodes, the probability of removing K nodes (K can be 0, 1, ..., N) is

$$\text{Probability (k nodes to be removed)} = \binom{N}{k} \frac{1}{N}^k \left(1 - \frac{1}{N}\right)^{N-k}$$

One intriguing property of this binomial distributed decay is that although the number of removed nodes each time is not fixed to be 1, the *expectation* of number of nodes to be removed each time is 1.

$$15 \quad \text{Expectation (number of nodes to be removed)} = \sum_{k=0}^N k \binom{N}{k} \frac{1}{N}^k \left(1 - \frac{1}{N}\right)^{N-k} = 1$$

The expectation is a sophisticated statistical concept for measuring the central tendency of the randomness over a long period of time. In other words, if the password/passphrase generator runs long enough, which will if the user uses it long enough, the expected number of nodes removed each time is simply 1. Therefore, safely using this binomial distributed decay similar to other decay mechanism.

Lastly, the password/passphrase formulation will be the sum of values generated by the FNN (both the motor and sensory cores) and it can either be theta angles or angular velocities.

25 In mathematical formulations, having:

theta angles for one-time password/passphrase

$$\text{then, password/passphrase at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$$

or

angular velocities for one-time password/passphrase

$$\text{then, password/passphrase at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in FNN at time T .

In embodiments 1, 2 and 3 of the present invention, password/passphrase is generated by using KeyNode according to the following sequence:

10 at $T1+n \rightarrow (\text{KeyNode}1+n) \rightarrow \text{Password/passphrase} \rightarrow \text{Json Web Token ("JWT")} \rightarrow \text{Receiver}$

Json Web Token will carry the encapsulated password (of KeyNode1+n) which is the complex sum theta (degree) or omega (degree per minute) at $T2$, $T3$, etc., to the receiver when/where the usage takes place.

In embodiment 1, one application of the system with such password/passphrase can be used for offline authentication in-tandem with biometric facial recognition, for instance at the door of a secure building.

15 Another application of the same system with such password/passphrase can be used for online authentication where JWT will carry the password (of KeyNode1+n) to the receiver at the time of use either as a single factor (1FA) or combined with another factor (2FA) for example facial recognition.

In embodiment 2, one application of the system with the generated passphrase can be fed into any RSA or ECC encryption algorithms to generate the corresponding pair of encryption and decryption keys for encryption of communication.

20 In embodiment 3, instead of generating a pair of RSA or ECC asymmetric keys from the passphrase, the KeyNode mechanism can be used to generate a one-time key similarly to the one-time-passphrase and using AES for encryption of personal data for secure storage.

25 In embodiments 4 and 5 of the present invention, KeyNode is used to directly generate the one-time decryption key or private key.

The previous section has been confined to the use of KeyNode algorithm for one-time password generation. Although temporary one-time password or TOTP's generation is admittedly one of the most popular passwords algorithms in practice nowadays, temporary one-time password alone is not sufficient, especially when one has some data or information for which one would like to have additional security and protection.

In this case encryption and decryption of information or data by the authorized parties become necessary.

Encryption is the process of encoding information or data. This process converts the original representation of the information a.k.a. plaintext, into a ciphertext. So, the authorized party will be able to decrypt the ciphertext and return to the original plaintext i.e., access to the original information or data. The process of encrypting and decrypting messages involves keys. There are two main types of keys involved, namely, symmetric key and public key a.k.a. asymmetric key.

The following are embodiments of encryption and decryption algorithms and their corresponding types of key involved:

1. Symmetric Key

Encryption and decryption keys are the same. Communicating parties must have the same keys in order to encrypt or decrypt the data or information. However, because of having the encryption key means also having the decryption key, anyone who has access to the encryption key can decrypt and read the information or data even when not authorized to. For example, if A is the receiver of messages sent from 2 separated senders B and C, C could decrypt and read the data in the message sent by B to A and vice versa, showcasing the deficiency of such symmetric key algorithm. Another potential issue would be if C accidentally leaks the encryption key so anyone could decrypt and read the data or information sent to A because decryption key and encryption key are the same. For such reasons, asymmetric key a.k.a. public key was developed.

2. Public Key

At the beginning, the receiving party will design its own encryption key and decryption key. Thereafter the receiving party will broadcast the encryption key to anyone who wants to communicate with the receiving party can use this public key to encrypt the information or data before sending them. Because only the receiving party has access to the decryption key, it will be the only party who will be able to decrypt the information or data contained in the messages using the public key for encryption. Note that having the encryption key gives no information about the decryption key at all, so even if the encryption key is accidentally leaked, there will be no risk of leaking the encrypted data or information.

Public key algorithm is now the mainstream encryption algorithm in use particularly because of the extra protection offered by the asymmetry of the keys. RSA (Rivest–Shamir–Adleman) is the most coveted public key encryption algorithm because it is using number theory results in the generation of the encryption key from a user-selected or a randomly generated decryption key to make sure knowing the encryption key cannot give any information about the decryption key.

Another example of public key encryption algorithm is the elliptic-curve cryptography (ECC) which generates encryption keys from the user-selected or randomly generated decryption keys based on some algebraic results of elliptic curves.

3. Comparison of RSA with ECC – Public Key

The difference between RSA and ECC encryption keys is the length of key for the same security. The table below compares key's sizes for the same level of security by RSA and ECC, respectively. For example, a ECC key of 160 bit achieves the same level of security of an RSA key of 1024 bit. Table 6 provides details. Note that the relationship between the size of ECC keys and RSA keys is not linear. Because of this difference in key size, ECC key generation, encryption and decryption is substantially quicker and require less computer power and memory than for RSA. Compared to RSA, ECC allows smaller ciphertext, key, and signature, and faster generation of keys and signatures. Its decryption and encryption speeds are slightly faster.

RSA Key Length (bit)	ECC Key Length (bit)
1024	160
2048	224
3072	256
7680	384
15360	521

Table 6

4. Symmetric Key Algorithm

Although the above discussion on RSA and ECC was centered around asymmetric key encryption algorithm, it is also possible to apply KeyNode for symmetric key encryption algorithm. As the name suggests, the keys in such algorithm are symmetric. In other words, the public key and private key are the same. One major example of symmetric key encryption algorithm is the Advanced Encryption Standard (AES).

AES is being developed by The National Institute of Standards and Technology (NIST) of the United States of America since 1997. AES became effective as a federal government standard in 2002. It was also included by the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) as the 18033-3 standard, which specifies: block ciphers for the purpose of data confidentiality.

AES includes three block ciphers: AES-128, AES-192 and AES-256. The numbers 128, 192 and 256 are describing the number of bits of key, the larger is the number the more secure is the encryption. For example, AES-128 is using a 128-bit key length to encrypt and decrypt a block of messages while AES-256 is using a 256-bit key to do the same. Each cipher encrypts and decrypts data in blocks of 128 bits using cryptographic keys of 128, 192 and 256 bits, respectively. Note that keys of 192 and 256 bits are encryption standards reserved for top secret encryption of the US government.

In details, there are 10 rounds of processing for the 128-bit keys, 12 rounds of processing for 192-bit keys and 14 rounds of processing for 256-bit keys, respectively. A round consists of several processing steps that include substitution, transposition and mixing of the input plaintext to transform it into the final output ciphertext.

Key length (bit)	Round of processing
128	10
192	12
256	14

5. Comparison of AES with RSA

AES is used widely for protecting data at rest, passive data, while RSA is more widely used for data in transfer, communications data. Applications for AES include self-encrypting disk drives, database encryption and storage encryption, which only involves one endpoint.

5 On the other hand, RSA is often used in web browsers to connect to websites, in virtual private network (VPN) connections and communication of secret messages between different parties. Typically, RSA keys are employed when there are two separate endpoints, for example, for protecting the transfer of data across geographic boundaries.

6. AES algorithm details

10 The following sections describe the processing steps of AES in details:

Firstly, the message from the sender to receiver is broken down into 128-bits' blocks. The bytes (16 in all) in a given block are then organized as a 4x4 matrix.

An example in hexadecimal expression is shown in Table 7.

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

Table 7

15 Then the block is passed through the following sequence of steps a total of x times, whereas x depends on the size of the cipher key.

Step 1: substitute Bytes. In this step, each element in the matrix is mapped to the corresponding byte in the Rijndael S-box.

Figure 9 is the Rijndael S-box of the step of substituting Bytes of the AES-128.

Step 2: shift Rows. In this step, rotating each element x elements (bytes) to the left, whereas x is the index of the row. For row 0, shift left 0 bytes (i.e., do not shift), for row 1, shift left 1 byte, for row 2, shift left 2 bytes, for row 3, shift left 3 bytes, etc.

Step 3: mix Columns. Multiplying every column by a predefined matrix for which the mathematical details are the followings: if any term is greater than 2 to the power of 8, dividing the polynomial by the Galois irreducible polynomial as follows

$$x^8+x^4+x^3+x+1$$

Step 4: add Round Key. In this step, performing a bitwise XOR operation between the columns of the matrix obtained in the preceding step and the Round Key. However in the first iteration, the Round Key is the first 128 bits of the cipher key. The preceding matrix is used as the input to the next round, and the process itself is repeated for another x rounds. However, the final round excludes the Mix Columns step. And then, computing a new key for the following rounds. Such process is called Key Schedule. The number of rounds depends on the length of the initial cipher key. A 128-bit key will lead to 10 rounds. A 192-bit key will lead to 12 rounds. A 256-bit key will lead to 14 rounds.

In the same manner as the 128-bit input block is arranged in the form of a state array, the algorithm arranges the first 16 bytes of the encryption key in the form of a 4×4 matrix of bytes, which is shown in Table 8.

k_0	k_4	k_8	k_12
k_1	k_5	k_9	k_13
k_2	k_6	k_10	k_14
k_3	k_7	k_11	k_15

Table 8

The four words of the original 128-bit key being expanded into a key schedule consisting of $4 \times 11 = 44$ words. The first four bytes $w_0 = (k_0, k_1, k_2, k_3)$ of the encryption key constitute the word w_0 , the next four bytes $w_1 = (k_4, k_5, k_6, k_7)$ the word w_1 , and so on up to w_3 . The four words of the round key for the i -th round is given:

$$w_i \ w_{i+1} \ w_{i+2} \ w_{i+3}$$

Hereinafter are how the words that will be used in the next round can be determined:

$$w_{i+4} \ w_{i+5} \ w_{i+6} \ w_{i+7}$$

The first word in this sequence w_{i+4} is computed as follows

$$5 \qquad w_{i+4} = w_i + g(w_{i+3})$$

where the function g consists of the following three steps:

1-Perform a one-byte circular rotation on the 4-byte word;

2-Substitute each byte in the word using the 16×16 lookup table;

3-XOR the bytes obtained from the previous step with what is known as a round constant.

10 And then, proceed to compute the remaining words in the Round Key.

$$w_{i+5} = w_{i+4} + w_{i+1}$$

$$w_{i+6} = w_{i+5} + w_{i+2}$$

$$w_{i+7} = w_{i+6} + w_{i+3}$$

The new 4x4 matrix (Round Key), i.e., w_{i+4} , w_{i+5} , w_{i+6} , w_{i+7} , will then be used for the

15 Add Key step of the proceeding round. The process is repeated for each of the 10 rounds.

Using passphrase for encryption

The present invention applies KeyNode in scenarios where encryption and decryption are necessary for guaranteeing the confidentiality of data and information. Although KeyNode is used for generation of one-time passphrase through a mechanism that involves the response of the user to questions from the list of

20 Question answer-types (Qa) or to single choices from the list of choices whether Binary visual-choices (Bv), Binary Iconic-choices (Ic), or Ternary visual-choices (Tv). The chain of nodes or KeyNode will increase in

size by one node each time the user gives a response to a new question or make a choice, and a decay mechanism will be put in motion at a certain time to maintain the size of the KeyNode within certain range. Each time a temporary one-time passphrase will be generated according to the theta angles or angular velocities of the nodes in the KeyNode, similarly to the one-time password.

5 Theta angles for passphrase

$$\begin{aligned} \text{one-time passphrase at time } T &= \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \\ &\quad \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i}) \end{aligned}$$

Angular velocities for passphrase

$$10 \quad \text{one-time passphrase at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the FNN at time T .

15 In order to make use of the KeyNode system in encryption and decryption algorithm, KeyNode one-time passphrase can be used as a one-time passphrase for generation of decryption key and encryption key. In other words, each time a user wants to set up a new decryption key, the user can answer a new question/choice that sets a new passphrase as the one-time passphrase which will be used to create keys.

Embodiment 2: Using passphrase with RSA and ECC for encryption

20 The generated passphrase can be fed into any RSA or ECC encryption algorithm to generate the corresponding pair of one-time encryption key and decryption key together. As said already, the encryption key will not leak any information about the decryption key at all. The user can then broadcast the one-time encryption key to any parties who want to encrypt data or information and send the encrypted ciphertext to the user.

25 The Figure 8 describes the workflow of generation and use of the one-time decryption/encryption pair of keys of the present invention; especially describes the workflow of the one-time encryption decryption algorithm using passphrase with RSA or ECC.

In step 201, at the beginning, user A initiates the KeyNode mechanism. In Step 202, after answering a question/choice and giving the response/choice, the KeyNode mechanism will generate a one-time passphrase for user A in the following way:

If theta angles are used for one-time passphrase

$$5 \quad \text{One-time passphrase at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$$

If angular velocities are used for one-time passphrase

$$\text{One-time passphrase at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in at time T .

In Step 203, the generated one-time passphrase will pass through either RSA or ECC systems to generate simultaneously a pair of keys i.e., one-time decryption key in step 206 and one-time encryption key in step 204.

If theta angles are used for passphrase, then one-time encryption key and decryption key are as follows:

15 One-time encryption key and decryption key at time T for RSA

$$= \text{RSA} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i}) \right)$$

or

One-time encryption key and decryption key at time T for ECC

$$20 \quad = \text{ECC} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i}) \right)$$

If angular velocities are used for passphrase, then one-time encryption key and decryption key are as follows:

$$\text{One-time encryption key and decryption key at time } T \text{ for RSA} = \text{RSA} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}} \right)$$

or

$$25 \quad \text{One-time encryption key and decryption key at time } T \text{ for ECC} = \text{ECC} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}} \right)$$

where $\text{RSA}()$ is the RSA algorithmic function for generating encryption key and decryption key with the passphrase, and $\text{ECC}()$ is the ECC algorithmic function for generating encryption key and decryption key

with the passphrase, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the FNN at time T .

- 5 In step 205 this one-time encryption key is then broadcasted to any parties who want to send an encrypted message over to user A.

In step 200, selected parties, such as party B, can start encryption of the data or information with the one-time encryption key and send them to user A who will be the only person to be able to decrypt the encrypted data or information.

- 10 After the step 206, the one-time decryption key will be sent to the User A for step 207.

Each time user A wants someone to send over encrypted data or information again, user A can repeat the same process from step 201 to step 207 for utmost security protection. Although it is entirely possible to use a same pair for an entire communication thread and dispose the pair of keys only at the end.

Embodiment 3: Using passphrase and concatenation with AES for encryption

- 15 Instead of having a different key for encryption and decryption, the same key can be used for both functions. The steps are similar to the case of ECC and RSA but there will be fewer steps since we have to generate only a single key that can be used to encrypt and decrypt.

After having answered to the question/choice of the new node, the KeyNode mechanism will generate a one-time key for user A (similarly to the passphrase).

- 20 If theta angles are used in summation for one-time key

$$\text{One-time key at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$$

If angular velocities are used in summation for one-time key

$$\text{One-time key at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

- 25 where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the at time T .

Since AES keys must have a length of either 128, 192 or 256 bits, there are ways to extend it to the desired length if the average theta angle or angular velocity does not provide enough length for the key. One of the

possible ways would be to concatenate the individual theta angle or angular velocity of each node in the KeyNode instead of using the average theta angle and average angular velocity of all nodes. For example, if the KeyNode has a total of nodes N_T at the moment, a key with a size of N_T times longer more than if just using the average theta angle or angular velocity is generated. Similarly, theta angle and angular velocity of the sensory and the motor cores of each node separately are also concatenated in order to double the length of the key.

The formulation for the above example is

If theta angles are used in concatenation for one-time key

$$\text{One-time key at time } T = \text{Concatenate} [(v_{m0}T)\%(2\pi r_{10}), \dots, (v_{mN_T}T)\%(2\pi r_{1N_T}), \dots, (v_{s0}T)\%(2\pi r_{20}), \dots, (v_{sN_T}T)\%(2\pi r_{2N_T})]$$

If angular velocities are used in concatenation for one-time key

$$\text{One-time key at time } T = \text{Concatenate} [\frac{v_{m0}}{2\pi r_{10}}, \dots, \frac{v_{mN_T}}{2\pi r_{1N_T}}, \dots, \frac{v_{s0}}{2\pi r_{20}}, \dots, \frac{v_{sN_T}}{2\pi r_{2N_T}}]$$

Another way to increase the length of the key would be to concatenate separately all data such as motor core radius, motor velocity, sensory core radius, sensory velocity, time, age etc. into the key.

Note that the main benefits offered by KeyNode itself are extended to the private key generation, the user no longer has to memorize his own private key in order to decrypt encrypted messages. All the user A has to do is to answer a new question/make a new choice to generate a new node for the KeyNode each and every time the user A wants to generate a decryption key (and the corresponding encryption key).

The user is now free from any burden of memorizing passphrases and decryption keys.

In comparison with the disclosure in the PCT/CN2020/127474, the advantageous developments on the present case are that the present invention offers a better description of this user-centric technology with significant new features such as using a passphrase to protect directly the private key though using the commercial systems in place or offering the opportunity to secure storage of personal data all using the similar one-time item generated by KeyNode. Therefore, the present embodiments 2 and 3 constitute an optimized solution.

Embodiment 4: integration of KeyNode with ECC for encryption:

In this implementation, ECC and KeyNode are integrated together in a more organic manner. Here KeyNode is no longer used as the generator of one-time passphrase but as the generator of the one-time private key for the ECC algorithm. In other words, the encryption key is constructed based on the thinking process of the human brain of the user modelled by the KeyNode mechanism.

Now, the ECC key length takes either 160, 224, 256, 384 or 521 bits.

ECC Key Length (bit)
160
224
256
384
521

In terms of bytes, the ECC key length takes around either 20, 28, 32, 48 or 65 bytes.

ECC Key Length (byte)
20
28
32
48
65

To make keys from KeyNode with such length, the following data will be considered:

1. the radius of motor core,
2. the velocity of motor core,
3. the angle theta of the motor core,
4. the angular velocity of the motor core,
5. the radius of sensory core,
6. the velocity of sensory core,
7. the angle theta of the sensory core,

8. the angular velocity of the sensory core,

as possible candidates for concatenation to make the KeyNode ECC private key.

Normally, a decimal number has double precision of 64 bits. As relying on Python for implementation, each of the numbers that listed above, for example the radius of motor core, can take 24 bits. In order to make up a sufficient length for the ECC key, the number of values needed for concatenation to make the key would be:

Number of values for concatenation (ECC)
2.5
3.5
4
6
9

In order to formulate a satisfying concatenation method for the maximum bit size of ECC key, i.e., 521-bit, we would need at least 9 values.

In consideration of which the theta angle should be our first choice since the radius, velocity and angular velocity of motor or sensory cores of nodes do not change from time to time. Their values are determined at the time when the node is initiated, making it not random enough for a good candidate for private key.

Therefore, as the minimum number of nodes in KeyNode is 8, the following formulas are suggested:

Theta_m1 Theta_m2 Theta_m3 Theta_m4 Theta_m5 Theta_m6 Theta_m7 Theta_m8 Theta_m_avg

or

Theta_s1 Theta_s2 Theta_s3 Theta_s4 Theta_s5 Theta_s6 Theta_s7 Theta_s8 Theta_s_avg

In mathematical expression, the following two formulas are using respectively whether the motor cores or the sensory cores for private key for ECC:

$$\text{Concatenate} [(v_{m1}T)\%(2\pi r_{11}), (v_{m2}T)\%(2\pi r_{12}), \dots, (v_{m8}T)\%(2\pi r_{18}), \frac{1}{8} \sum_{i=1}^8 (v_{mi}T)\%(2\pi r_{1i})]$$

or

$$\text{Concatenate} [(v_{s1}T)\%(2\pi r_{21}), (v_{s2}T)\%(2\pi r_{22}), \dots, (v_{s8}T)\%(2\pi r_{28}), \frac{1}{8} \sum_{i=1}^8 (v_{si}T)\%(2\pi r_{2i})]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN.

As there are 9 values in each of the two formulas, both of them is a good candidate for the private key.

When there will be more than 8 nodes in KeyNode, randomly selecting 8 nodes out of the KeyNode and their average to formulate the private key as described by the above formulas.

However more variations are possible, for example:

Theta_s1 Theta_m1 Theta_s2 Theta_s3 Theta_m2 Theta_s4 Theta_m3 Theta_m4 Theta_m_s_avg

where randomly selecting 4 nodes and take the theta angle of their motor cores and randomly select 4 other nodes and take the theta angle of their sensory cores and also the average of the theta angles of all these motor and sensory cores.

In mathematical expression, the following formula is combining the motor cores with the sensory cores for private key for ECC:

$$\text{Concatenate} [(v_{s1}T)\%(2\pi r_{21}), (v_{m1}T)\%(2\pi r_{11}), (v_{s2}T)\%(2\pi r_{22}), (v_{s3}T)\%(2\pi r_{23}), (v_{m2}T)\%(2\pi r_{12}), (v_{s4}T)\%(2\pi r_{24}), (v_{m3}T)\%(2\pi r_{13}), (v_{m4}T)\%(2\pi r_{14}), \frac{1}{16} \sum_{i=1}^8 \{ (v_{si}T)\%(2\pi r_{2i}) + (v_{mi}T)\%(2\pi r_{1i}) \}]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN.

For private keys with size lower than 521-bit, still making use of the above formulas, but only taking the first x bits of the formulas to formulate the private keys.

Figure 10 describes the organic workflow of one embodiment of the present invention using KeyNode with ECC.

In step 301, at the beginning, user A initiates the KeyNode mechanism. In Step 302, after answering a question/choice and giving the response/choice, the KeyNode mechanism will generate a one-time decryption key for user A.

In Step 303, the generated one-time decryption key will pass through ECC systems to generate one time encryption key in step 304 and then broadcast one-time encryption key to the party B who wants to send message to user A in step 305. In step 300, Party B will send encrypted message to user A.

In step 306, this one-time decryption key is then broadcasted to user A for decrypting the sent messages.

Embodiment 5: integration of KeyNode with RSA for encryption:

In this part, RSA is also integrated with KeyNode together in an organic manner. Similarly to the case of ECC, KeyNode is no longer used as the generator of a one-time passphrase but as the generator of the one-time private key for the RSA algorithm. In other words, the encryption key is constructed based on the thinking process of the human brain of the user modelled by the KeyNode mechanism.

It is not suggested directly taking the same approach of ECC's private key generation by concatenation of values in KeyNode to integrate RSA and KeyNode, because RSA would require a much bigger size of key to achieve a similar level of security. Table 9 is referred to for the require number of bits for the generation of private keys.

RSA Key Length (bit)
1024
2048
3072
7680
15360

Table 9

As each decimal take double precision of 64 bits, the number of values from KeyNode needed for concatenation to generate the private keys would be:

Number of values for concatenation (RSA)
16
32
48
120
240

- 5 As having at most a total of 22 nodes, it is hardly possible to generate a private key with RSA by making use of the theta angles alone. So, it is not recommended to integrate RSA with KeyNode in the same way as how to integrate ECC with KeyNode.

Instead, the following two alternative approaches are suggested:

1. The most direct solution to integrate KeyNode with RSA would be to expand the KeyNode up to the adequate size of desired security strength. For example, if user needs an RSA key of size 3072 bits, then the KeyNode's minimum size should be expanded to 24 nodes (64bits*24*2) instead of 8, so that the private key can be formulated as follows:

15 Theta_m1 Theta_m2 Theta_m3 Theta_m4 Theta_m5 Theta_m6 Theta_m7 Theta_m8 Theta_m9
Theta_m10 Theta_m11 Theta_m12 Theta_m13 Theta_m14 Theta_m15 Theta_m16 Theta_m17
Theta_m18 Theta_m19 Theta_m20 Theta_m21 Theta_m22 Theta_m23 Theta_m24 Theta_s1
Theta_s2 Theta_s3 Theta_s4 Theta_s5 Theta_s6 Theta_s7 Theta_s8 Theta_s9 Theta_s10 Theta_s11
Theta_s12 Theta_s13 Theta_s14 Theta_s15 Theta_s16 Theta_s17 Theta_s18 Theta_s19 Theta_s20
Theta_s21 Theta_s22 Theta_s23 Theta_s24

- 20 In mathematical expression, the following formula is combining the motor cores with the sensory cores for private key for RSA:

Concatenate [$(v_{m1}T)\%(2\pi r_{11})$, ..., $(v_{m24}T)\%(2\pi r_{1\ 24})$, $(v_{s1}T)\%(2\pi r_{21})$, ..., $(v_{s24}T)\%(2\pi r_{2\ 24})$]

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN.

5 Table 10 shows the necessary minimum size of the KeyNode for the integration of RSA with KeyNode.

Minimum KeyNode Size
8
16
24
60
120

Table 10

10 The alternative way would be to concatenate the ECC key with random numbers so that the key size becomes larger enough to meet the requirements of an RSA key. For example, if user needs an RSA key of size 3072 bits, but only having KeyNode of 8 nodes, the private key can be generated as follows:

15 Theta_m1 Theta_m2 Theta_m3 Theta_m4 Theta_m5 Theta_m6 Theta_m7 Theta_m8 Theta_m_avg
 Theta_s1 Theta_s2 Theta_s3 Theta_s4 Theta_s5 Theta_s6 Theta_s7 Theta_s8 Theta_s_avg
 random_1920_bits

In this formula, the last term “random_1920_bits” means to generate 1920 bits from a random seed generator. This formula would have 3072 bits in total, because each Theta_m and Theta_s takes 64 bits and the random_1920_bits take 384 bits, making a total of $64 \times 18 + 1920 = 3072$ bits.

20 In mathematical expression, the following formula is combining the motor cores with the sensory cores and a random generator for private key for RSA:

$$\text{Concatenate} \left[(v_{m1}T)\%(2\pi r_{11}), \dots, (v_{m8}T)\%(2\pi r_{18}), \frac{1}{8} \sum_{i=1}^8 (v_{mi}T)\%(2\pi r_{1i}), \right. \\ \left. (v_{s1}T)\%(2\pi r_{21}), \dots, (v_{s8}T)\%(2\pi r_{28}), \frac{1}{8} \sum_{i=1}^8 (v_{si}T)\%(2\pi r_{2i}), \text{random_generator}(1920 \text{ bits}) \right]$$

where the "random_generator(bits)" is a function for generating 1920 bits of digits randomly, v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN.

Both of the two approaches are possible solutions to extend KeyNode for integration with RSA.

Figure 11 describes the organic workflow of another embodiment of the present invention using KeyNode with RSA.

In step 401, at the beginning, user A initiates the KeyNode mechanism. In Step 402, after answering a question/choice and giving the response/choice, the KeyNode mechanism will generate a one-time decryption key for user A.

In Step 403, the generated one-time decryption key will pass through RSA systems to generate one time encryption key in step 404 and then broadcast one-time encryption key to the party B who wants to send message to user A in step 405. In step 400, Party B will send encrypted message to user A.

In step 406, this one-time decryption key is then broadcasted to user A for describing the sent messages.

As will be appreciated by one skilled in the art, aspects of the present invention may be embodied as a system, method or computer program product. Accordingly, aspects of the present invention may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a "circuit," "module" or "system." Furthermore, aspects of the present invention may take the form of a computer program product embodied in one or more computer readable medium(s) having computer readable program code embodied thereon.

Any combination of one or more computer readable medium(s) may be utilized. The computer readable medium may be a computer readable signal medium or a computer readable storage medium. A computer readable storage medium may be, for example, but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. More specific examples (a non-exhaustive list) of the computer readable storage medium would include the following: an electrical connection having one or more wires, a portable computer diskette, a

hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing. In the context of this document, a computer readable storage medium may be any tangible medium that can contain or store a program for use by or in connection with an instruction execution system, apparatus, or device.

A computer readable signal medium may include a propagated data signal with computer readable program code embodied therein, for example, in baseband or as part of a carrier wave. Such a propagated signal may take any of a variety of forms, including, but not limited to, electro-magnetic, optical, or any suitable combination thereof. A computer readable signal medium may be any computer readable medium that is not a computer readable storage medium and that can communicate, propagate, or transport a program for use by or in connection with an instruction execution system, apparatus, or device.

Program code embodied on a computer readable medium may be transmitted using any appropriate medium, including but not limited to wireless, wireline, optical fiber cable, RF, etc., or any suitable combination of the foregoing.

Computer program code for carrying out operations for aspects of the present invention may be written in any combination of one or more programming languages, including an object oriented programming language such as Java, Python, C++ or the like and conventional procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider).

Aspects of the present invention are described above with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer program products according to embodiments of the invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These computer program instructions may be provided to a processor of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

The computer program instructions may also be loaded onto a computer, other programmable data

processing apparatus, or other devices to cause a series of operational steps to be performed on the computer, other programmable apparatus or other devices to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

- 5 The computer program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other devices to cause a series of operational steps to be performed on the computer, other programmable apparatus or other devices to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.
- 10 The flowcharts and block diagrams in the Figures illustrate the architecture, functionality, and operation of possible implementations of systems, methods and computer program products according to various embodiments of the present invention. In this regard, each block in the flowcharts or block diagrams may represent a module, segment, or portion of code, which comprises one or more executable instructions for implementing the specified logical function(s). It should also be noted that, in some alternative
- 15 implementations, the functions noted in the block may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in the block diagrams and/or flowchart illustration, can be implemented by special purpose hardware-based
- 20 systems that perform the specified functions or acts, or combinations of special purpose hardware and computer instructions.

The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the invention. As used herein, the singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the

25 terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

The corresponding structures, materials, acts, and equivalents of all means or step plus function elements in

30 the claims below are intended to include any structure, material, or act for performing the function in combination with other claimed elements as specifically claimed. The description of the present invention has been presented for purposes of illustration and description but is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the invention. The embodiment was

35 chosen and described in order to best explain the principles of the invention and the practical application,

and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated. Accordingly, the enhanced assessment module supports cognitive and behavioral assessment of a participant subject in the field, and at the same time provides a unique employment of test and associated test batteries for the assessment.

- 5 As used herein, "one embodiment," "an embodiment," or "one or more embodiments" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present invention. Also, please note that the examples of the words "in one embodiment" do not completely refer to the same embodiment.

- 10 It will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without departing from the spirit and scope of the invention. Accordingly, the scope of protection of this invention is limited only by the following claims and their equivalents.

- 15 The present invention can be applied to secure i.e. prevent hacking of personal data or theft of corporate information/secrets in online and mobile communication especially at the dawn of 5G with generalization of open API lifestyle platforms, to encrypt smart city identification such as smartID for smart citizen, to protect personal health data, to enable more data transfer for smart banking and smart finance, to fight industrial espionage, to allow real-time identification for digital cryptocurrency payments and other public distributed ledger technology (DLT) mechanisms.

CLAIMS

1. An encryption method comprising:

Generating a Functional Neural Network or KeyNode (KN) of the system by programming a chain of multiple nodes also called Artificial Mirror Neurons (AMN) based on captured information of reaction time and emotional response to simple tasks; one task per mode, racing the nodes in the Functional Neural Network or KeyNode (KN) and their many-valued output services as an encryption device or cipher for the time of use;

Classifying the KeyNodes where each KeyNode comprises multiple nodes held in a chain which can be generated successively, elongated, and can also evolve by retraction of nodes or decay after a certain number of nodes has been generated;

Encoding KeyNodes based on infinite proposition captured by interactive cognitive computing, forming an irregular tetragon matrix, and allowing dynamic encoding of unusual forms of cryptography;

Generating the boundary of the KeyNode, where the minimum size is defined as A, the maximum size is defined as B, and the time of maximum size of the KeyNode is defined as $C = B - (A - 1)$;

Generating one-time-password or passphrase for decryption at the user's time of use based on the sum of intrinsic values of the nodes in the racing network at this time and adopting the generated password for authentication or adopting the generated passphrase to generate a one-time-pair of keys for encrypted communication to form a one-time-pair of keys for encrypted communication;

Transmitting the one-time encryption key (public key) over any unsecured channel to selected senders for encrypting the plaintext to become a ciphertext;

Decrypting the ciphertext of the senders at the receiver's end using the one-time decryption key (private key) and the passphrase to regenerate the plaintext.

2. The method as claimed in claim 1, wherein, the steps of programing Artificial Mirror Neuron or node, further comprising capturing reaction and emotional response where the reaction time is the time taken by the user to react to the cognitive task between input and output, which determines the motor core radius size; and the emotional response is the emotional response to the said cognitive task captured around 500ms from the input time, which determines the sensory core radius size; and determining the velocity which is inversely to the age information; and deriving the angular velocity ω from said velocity and the angle θ of both said motor core and sensory core; the steps of racing the chain of nodes in the Functional Neural Network or

KeyNode (KN), further comprising programming multiple nodes, and the chain of nodes in the Functional Neural Network or KeyNode (KN) serves as an encryption device or a cipher; said KeyNodes comprise one first signup KeyNode or KN1 which contains at least 8 nodes;

5 the steps of generating a passphrase, further comprising generating said passphrase for use at time 2 (T2), on the base of each node angle θ changes constantly between time 1 (T1) and time 2 (T2), and the sum of angle θ of all nodes also called sum of angle θ_2 at time 2 depends on the position of the radiuses or the value of the velocities of the nodes at time 2; and

10 the method further comprising defining each node by said motor core radius, said sensory core radius, two angle θ_1 and angle θ_2 , two velocity v_1 and velocity v_2 , and two angular velocity ω_1 and angular velocity ω_2 at the time T1 and T2, and the time of each node creation in minute and each new node has a specific time of creation T1+n that will refer to the initial nodes time of creation when made at signup or T1; and

when reaching the time T2 or time of applying said passphrase, creating a new node by generation-elongation, and summing up said motor core and sensory core radius theta angles or angular velocities for all racing nodes in the KN arriving at T2, and adopting said passphrase for authentication.

15 3. The method as claimed in claim 2,

using the angle θ for one time passphrase:

$$\text{One-time passphrase at time T} = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$$

using the angular velocities ω for one time passphrase is:

$$\text{One-time Passphrase at time T} = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

20 where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in FNN at time T.

4. The method as claimed in claim 2, wherein for the motor cores, the following steps are provided:

$$\text{individual angle theta from M of each node at time T} = (v_m T) \% (2\pi r_1)$$

25 $\text{sum of angles theta from M of the whole FNN at time T} = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i})$

where v_m is the velocity of the motor core in an AMN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_1 is the radius of the motor core in an AMN, r_{1i} is the radius of the motor core of the i -th

node in the FNN, and N_T is the number of nodes in FNN at time T .

wherein for the sensory cores, the following steps are provided:

$$\text{individual angle theta from S of each node at time } T = (v_s T) \% (2\pi r_2)$$

$$\text{sum of angles theta from S of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$$

- 5 where v_s is the velocity of the sensory core in an AMN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_2 is the radius of the sensory core in an AMN, r_{2i} is the radius of the sensory core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T .

wherein for the Angular Velocity per core, the following steps are provided:

$$\text{individual angular velocity from M of each node at time } T = \frac{v_m}{2\pi r_1}$$

$$10 \quad \text{sum of angular velocities from M of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}}$$

where v_m is the velocity of the motor core in an AMN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_1 is the radius of the motor core in an AMN, r_{1i} is the radius of the motor core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T ;

Similarly, for the sensory cores, the following steps are provided:

$$15 \quad \text{individual angular velocity from S of each node at time } T = \frac{v_s}{2\pi r_2}$$

$$\text{sum of angular velocities from S of the whole FNN at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_s is the velocity of the sensory core in an AMN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_2 is the radius of the sensory core in an AMN, r_{2i} is the radius of the sensory core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T .

- 20 5. The method as claimed in claim 1, wherein the steps of encoding KeyNodes based on infinite proposition captured by interactive cognitive computing, there are four encoding schemes such as Question answer-types (Qa), Binary visual-choices (Bv), binary Iconic-choices (Ic) and Ternary visual-choices (Tv) to form the irregular tetragon matrix; generating a new node in the KeyNode according to the output from one of these four types, producing a cipher combining both the reactionary nature and the timeliness of nodes and
- 25 refreshing the FNN with a new node.

6. The method as claimed in claim 2, wherein said one time passphrase is fed into Rivest–Shamir–Adleman (“RSA”) or elliptic-curve cryptography (“ECC”) encryption algorithms to generate a pair of one time encryption key and decryption key.

7. The method as claimed in claim 6. if theta angles are used for one-time encryption key and decryption

key, then

One-time encryption key and decryption key at time T for RSA

$$= \text{RSA} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T) \% (2\pi r_{2i}) \right)$$

or

5

One-time encryption key and decryption key at time T for ECC

$$= \text{ECC} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T) \% (2\pi r_{2i}) \right)$$

if angular velocities are used for one-time encryption key and decryption key, then

$$\text{One-time encryption key and decryption key at time T for RSA} = \text{RSA} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \right.$$

$$\left. \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}} \right)$$

And/or

$$\text{One-time encryption key and decryption key at time T for ECC} = \text{ECC} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \right.$$

$$\left. \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}} \right)$$

where $\text{RSA}()$ is the RSA algorithmic function for generating encryption key and decryption key with the passphrase, and $\text{ECC}()$ is the ECC algorithmic function for generating encryption key and decryption key with the passphrase, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the FNN at time T;

8. The method as claimed in claim 7, wherein using the motor cores or the sensory cores respectively for generating a private key compatible with ECC:

$$\text{Concatenate} [(v_{m1}T) \% (2\pi r_{11}), (v_{m2}T) \% (2\pi r_{12}), \dots, (v_{m8}T) \% (2\pi r_{18}), \frac{1}{8} \sum_{i=1}^8 (v_{mi}T) \% (2\pi r_{1i})]$$

or

$$\text{Concatenate} [(v_{s1}T)\%(2\pi r_{21}), (v_{s2}T)\%(2\pi r_{22}), \dots, (v_{s8}T)\%(2\pi r_{28}), \frac{1}{8} \sum_{i=1}^8 (v_{si}T)\%(2\pi r_{2i})]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN; or

Combining the motor cores with the sensory cores for generating a private key compatible with ECC:

$$\text{Concatenate} [(v_{s1}T)\%(2\pi r_{21}), (v_{m1}T)\%(2\pi r_{11}), (v_{s2}T)\%(2\pi r_{22}), (v_{s3}T)\%(2\pi r_{23}), (v_{m2}T)\%(2\pi r_{12}), (v_{s4}T)\%(2\pi r_{24}), (v_{m3}T)\%(2\pi r_{13}), (v_{m4}T)\%(2\pi r_{14}), \frac{1}{16} \sum_{i=1}^8 \{ (v_{si}T)\%(2\pi r_{2i}) + (v_{mi}T)\%(2\pi r_{1i}) \}]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN; and

combining the motor cores with the sensory cores for generating a private key compatible with RSA:

$$\text{Concatenate} [(v_{m1}T)\%(2\pi r_{11}), \dots, (v_{m24}T)\%(2\pi r_{124}), (v_{s1}T)\%(2\pi r_{21}), \dots, (v_{s24}T)\%(2\pi r_{224})]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN; or

combining the motor cores with the sensory cores and a random generator for generating a private key compatible with RSA:

$$\text{Concatenate} [(v_{m1}T)\%(2\pi r_{11}), \dots, (v_{m8}T)\%(2\pi r_{18}), \frac{1}{8} \sum_{i=1}^8 (v_{mi}T)\%(2\pi r_{1i}),$$

$$(v_{s1}T)\%(2\pi r_{21}), \dots, (v_{s8}T)\%(2\pi r_{28}), \frac{1}{8} \sum_{i=1}^8 (v_{si}T)\%(2\pi r_{2i}), \text{random_generator}(1920 \text{ bits})]$$

where the “random_generator(bits)” is a function for generating 1920 bits of digits randomly, v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in

the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN.

5 9. An encryption system comprising:

KeyNodes Generation module, where the Functional Neural Networks or KeyNodes (KN) of the system are generated by programming a chain of multiple nodes also called Artificial Mirror Neurons (AMN) based on captured information of reaction time and emotional response to simple tasks, one task per node; the nodes in the Functional Neural Network or KeyNode (KN) are racing and their many-valued output serves
10 as encryption device or cipher for the time of use;

KeyNodes classification module, where each KeyNode comprises multiple nodes held in a chain which can be generated successively, elongated, and can also evolve by retraction of nodes or decay after a certain number of nodes has been generated;

KeyNodes encoding module, where said KeyNodes are encoded based on infinite proposition captured
15 by interactive cognitive computing, forming an irregular tetragon matrix, and allowing dynamic encoding of unusual forms of cryptography;

Boundary generation module, where the boundary of the KeyNode is generated, and where the minimum size is defined as A , the maximum size is defined as B , and the time of maximum size of the KeyNode is defined as $C = B - (A - 1)$;

20 Pair of keys generation module, where the one-time-password or passphrase is generated for decryption at the user's time of use based on the sum of intrinsic values of the nodes in the racing network at this time, and adopting the generated password for authentication or adopting the generated passphrase to generate a one-time-pair of keys for encrypted communication to form a one-time-pair of keys for encrypted communication;

25 Key transmission module, where the one-time encryption key (public key) is transmitted over any unsecured channel to selected senders for encrypting the plaintext to become a ciphertext;

Decryption module, where the ciphertext of the senders at the receiver's end is decrypted using the one-time decryption key (private key) and the passphrase to regenerate the plaintext.

10. The system as claimed in claim 9, wherein, in the KeyNode Generation module, further comprising
30 reaction and emotional response capturing where the reaction time is the time taken by the user to react

to the cognitive task and captured between input and output, which determines the motor core radius size; and the emotional response is the emotional response to the said cognitive task captured around 500ms from the input time, which determines the sensory core radius size; and the velocity which is inversely to the age information is determined; and the angular velocity ω from said velocity and the angle θ of both said motor core and sensory core are derived;

further comprising programming multiple nodes, and the chain of nodes in the Functional Neural Network or KeyNode (KN) serves as an encryption device or a cipher; said KeyNodes comprise one first signup KeyNode or KN1 which contains at least 8 nodes;

further comprising said passphrase for use at time 2 (T2) is generated, on the base of each node angle θ changes constantly between time 1 (T1) and time 2 (T2), and the sum of angle θ of all nodes also called sum of angle θ_2 at time 2 depends on the position of the radiuses or the value of the velocities of the nodes at time 2; and

the system further comprising a definition module, where each node by said motor core radius is defined by, said sensory core radius, two angle θ_1 and angle θ_2 , two velocity v_1 and velocity v_2 , and two angular velocity ω_1 and angular velocity ω_2 at the time T1 and T2, and the time of each node creation in minute and each new node has a specific time of creation T1+n that will refer to the initial nodes time of creation when made at signup or T1; and

when the time T2 or time of applying said passphrase is reached, a new node by generation-elongation is created, and said motor core and sensory core radius theta angles or angular velocities for all racing nodes in the KN arriving at T2 is summed up and said passphrase for authentication is adopted.

11. The system as claimed in claim 10, if the angle θ for one time passphrase is used:

$$\text{one time Passphrase at time T} = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T) \% (2\pi r_{2i})$$

if the angular velocities ω for one time passphrase is used:

$$\text{one time Passphrase at time T} = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in FNN at time T.

12. The system as claimed in claim 10, wherein for the motor cores:

individual angle theta from M of each node at time T = $(v_m T) \% (2\pi r_1)$

sum of angles theta from M of the whole FNN at time T = $\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi} T) \% (2\pi r_{1i})$

where v_m is the velocity of the motor core in an AMN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_1 is the radius of the motor core in an AMN, r_{1i} is the radius of the motor core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T.

wherein for the sensory cores:

individual angle theta from S of each node at time T = $(v_s T) \% (2\pi r_2)$

sum of angles theta from S of the whole FNN at time T = $\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si} T) \% (2\pi r_{2i})$

where v_s is the velocity of the sensory core in an AMN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_2 is the radius of the sensory core in an AMN, r_{2i} is the radius of the sensory core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T.

wherein for the Angular Velocity per core:

individual angular velocity from M of each node at time T = $\frac{v_m}{2\pi r_1}$

sum of angular velocities from M of the whole FNN at time T = $\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}}$

where v_m is the velocity of the motor core in an AMN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_1 is the radius of the motor core in an AMN, r_{1i} is the radius of the motor core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T;

Similarly, for the sensory cores:

individual angular velocity from S of each node at time T = $\frac{v_s}{2\pi r_2}$

sum of angular velocities from S of the whole FNN at time T = $\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$

where v_s is the velocity of the sensory core in an AMN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_2 is the radius of the sensory core in an AMN, r_{2i} is the radius of the sensory core of the i -th node in the FNN, and N_T is the number of nodes in FNN at time T.

13. The system as claimed in claim 9, wherein, in the KeyNodes encoding module, KeyNodes are encoded based on infinite proposition captured by interactive cognitive computing, there are four encoding schemes such as Question answer-types (Qa), Binary visual-choices (Bv), binary Iconic-choices (Ic) and Ternary visual-choices (Tv) to form the irregular tetragon matrix; generating a new node in the KeyNode according to the output from one of these four types, producing a cipher combining both the reactional nature and the timeliness of nodes and refreshing the FNN with a new node..

14. The system as claimed in claim 10, wherein said one time passphrase is fed into Rivest–Shamir–Adleman (“RSA”) or elliptic-curve cryptography (“ECC”) encryption algorithms to generate a pair of one time encryption key and decryption key.

5 15. The system as claimed in claim 14, wherein, if theta angles are used for one-time encryption key and decryption key, then:

One-time encryption key and decryption key at time T for RSA

$$= \text{RSA} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T) \% (2\pi r_{2i}) \right)$$

or

One-time encryption key and decryption key at time T for ECC

$$10 \quad = \text{ECC} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T) \% (2\pi r_{2i}) \right)$$

If angular velocities are used for one-time encryption key and decryption key, then:

$$\text{One-time encryption key and decryption key at time T for RSA} = \text{RSA} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}} \right)$$

And/or

$$15 \quad \text{One-time encryption key and decryption key at time T for ECC} = \text{ECC} \left(\frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}} \right)$$

where $\text{RSA}()$ is the RSA function for generating encryption key and decryption key with the passphrase, and $\text{ECC}()$ is the ECC function for generating encryption key and decryption key with the passphrase, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the FNN at time T.

20 16. The system as claimed in claim 15, wherein, the motor cores or the sensory cores are used respectively for generating a private key compatible with ECC:

$$\text{Concatenate} [(v_{m1}T) \% (2\pi r_{11}), (v_{m2}T) \% (2\pi r_{12}), \dots, (v_{m8}T) \% (2\pi r_{18}), \frac{1}{8} \sum_{i=1}^8 (v_{mi}T) \% (2\pi r_{1i})]$$

or

$$\text{Concatenate} [(v_{s1}T)\%(2\pi r_{21}), (v_{s2}T)\%(2\pi r_{22}), \dots, (v_{s8}T)\%(2\pi r_{28}), \frac{1}{8} \sum_{i=1}^8 (v_{si}T)\%(2\pi r_{2i})]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN; or

Combining the motor cores with the sensory cores for generating a private key compatible with ECC:

$$\text{Concatenate} [(v_{s1}T)\%(2\pi r_{21}), (v_{m1}T)\%(2\pi r_{11}), (v_{s2}T)\%(2\pi r_{22}), (v_{s3}T)\%(2\pi r_{23}), (v_{m2}T)\%(2\pi r_{12}), (v_{s4}T)\%(2\pi r_{24}), (v_{m3}T)\%(2\pi r_{13}), (v_{m4}T)\%(2\pi r_{14}), \frac{1}{16} \sum_{i=1}^8 \{ (v_{si}T)\%(2\pi r_{2i}) + (v_{mi}T)\%(2\pi r_{1i}) \}]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN; and

combining the motor cores with the sensory cores for generating a private key compatible with RSA:

$$\text{Concatenate} [(v_{m1}T)\%(2\pi r_{11}), \dots, (v_{m24}T)\%(2\pi r_{124}), (v_{s1}T)\%(2\pi r_{21}), \dots, (v_{s24}T)\%(2\pi r_{224})]$$

where v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN; or

combining the motor cores with the sensory cores and a random generator for generating a private key compatible with RSA:

$$\text{Concatenate} [(v_{m1}T)\%(2\pi r_{11}), \dots, (v_{m8}T)\%(2\pi r_{18}), \frac{1}{8} \sum_{i=1}^8 (v_{mi}T)\%(2\pi r_{1i}), (v_{s1}T)\%(2\pi r_{21}), \dots, (v_{s8}T)\%(2\pi r_{28}), \frac{1}{8} \sum_{i=1}^8 (v_{si}T)\%(2\pi r_{2i}), \text{random_generator}(1920 \text{ bits})]$$

where the “random_generator(bits)” is a function for generating 1920 bits of digits randomly, v_m are the velocities of the motor core of the nodes in the FNN, r_1 are the radiuses of the motor core of the nodes in the FNN, v_s are the velocities of the sensory core of the nodes in the FNN, r_2 are the radiuses of the sensory core of the nodes in the FNN, v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the

radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN.

17. An encryption method comprising:

5 Generating a Functional Neural Network or KeyNode (KN) of the system by programming a chain of multiple nodes also called Artificial Mirror Neurons (AMN) based on captured information of reaction time and emotional response to a simple task; racing the nodes in the Functional Neural Network or KeyNode (KN) as an encryption device or cipher for the time of use;

10 Classifying KeyNodes of multiple nodes where each KeyNode comprises multiple nodes held in a chain which can be generated successively, elongated, and can also evolve by retraction of nodes or decay after a certain number of nodes has been generated;

 Encoding KeyNodes based on infinite proposition captured by interactive cognitive computing, forming an irregular tetragon matrix, and allowing dynamic encoding of unusual form of cryptography;

 Generating the boundary of the KeyNode, where the minimum size is defined as A , the maximum size is defined as B , and the time of maximum size of the KeyNode is defined as $C = B - (A - 1)$;

15 Generating one-time-private key for decryption at the user's time of use based on the concatenation of intrinsic values of the nodes in the racing network at this time and adopting the generated private key to generate a one-time public key to form a one-time-pair of keys for encrypted communication;

 Transmitting the one-time public key over any unsecured channel to selected senders for encrypting the plaintext to become a ciphertext;

20 Decrypting the ciphertext of the senders at the receiver's end using the one-time key to regenerate the plaintext data; or using the one-time key for symmetric key encryption of user's personal data for secure storage.

18. The method as claimed in claim 17, wherein said the public key and private key are the same by using said symmetric key encryption.

25

19. The method as claimed in claim 17, wherein said symmetric key encryption is Advanced Encryption Standard ("AES").

20. The method as claimed in claim 19, wherein said AES comprises AES-128, AES-192, and AES-256.

30

21. The method as claimed in claim 20, if theta angles are used in summation for one-time key

$$\text{One-time key at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T) \% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T) \% (2\pi r_{2i})$$

5 If angular velocities are used in summation for one-time key

$$\text{One-time key at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the at time T .

22. The method as claimed in claim 19, if theta angles are used in concatenation for one-time key, then:

One-time key at time $T =$

$$\text{Concatenate} [(v_{m0}T) \% (2\pi r_{10}), \dots, (v_{mN_T}T) \% (2\pi r_{1N_T}), \dots, (v_{s0}T) \% (2\pi r_{20}), \dots, (v_{sN_T}T) \% (2\pi r_{2N_T})]$$

If angular velocities are used in concatenation for one-time key, then:

$$\text{One-time key at time } T = \text{Concatenate} \left[\frac{v_{m0}}{2\pi r_{10}}, \dots, \frac{v_{mN_T}}{2\pi r_{1N_T}}, \dots, \frac{v_{s0}}{2\pi r_{20}}, \dots, \frac{v_{sN_T}}{2\pi r_{2N_T}} \right]$$

23. The method as claimed in claim 20, wherein AES-128 using a 128 bit key, AES-192 using a 192 bit key, and AES 256 using a 256 bit key respectively to encrypt and decrypt a block of messages, wherein there are 10 rounds of processing 128 bit keys, 12 rounds of processing 192 bit key, 14 rounds of processing for 256-bit keys, respectively.

24. The method as claimed in claim 21, wherein said AES-128 comprises the following steps:

Breaking the message from the sender to the receiver into 128-bits' blocks;

Organizing the bytes in a given block as a 4×4 matrix;

Passing the block through the following sub-steps for a number of times depending on the size of the cipher key, i.e. 10 times for AES-128:

Substituting the bytes by mapping each element in the matrix to the corresponding byte in the Rijndael S-box;

Shifting rows by rotating each element x bytes to the left, whereas x is the index of the row;

5 Mixing columns by multiplying every column by a predefined matrix according to the following algorithm:

if any term is greater than 2 to the power of 8, dividing the polynomial by the Galois irreducible polynomial as

$$x^8+x^4+x^3+x+1$$

10 Adding a round key by performing a bitwise XOR operation between the columns of the matrix and the Round Key; although in the first round, the Round Key which is de facto the one-time-key generated by KeyNode, is the first 128 bits of the cipher key;

Inputting the preceding matrix to the next round and repeating the sub-steps, minimum 10 times for an AES-128;

15 Excluding the final step of mixing columns for the final round.

25. An encryption system comprising:

KeyNode Generation module, a Functional Neural Network or KeyNode (KN) of the system by programming a chain of multiple nodes also called Artificial Mirror Neurons (AMN) based on captured information of reaction time and emotional response to a simple task are generated; the nodes in the
20 Functional Neural Network or KeyNode (KN) are racing and their many-valued output serves as encryption device or cipher for the time of use;

KeyNodes classification module, where each KeyNode comprises multiple nodes held in a chain which can be generated successively, elongated, and can also evolve by retraction of nodes or decay after a certain number of nodes has been generated;

25 KeyNodes encoding module, KeyNodes based on infinite proposition captured by interactive cognitive computing, forming an irregular tetragon matrix, and allowing dynamic encoding of unusual form of cryptography are encoded;

Boundary generation module, the boundary of the KeyNode, where the minimum size is defined as A, the maximum size is defined as B, and the time of maximum size of the KeyNode is defined as $C = B - (A - 1)$ is generated;
30

Pair of keys generation module, one-time-private key for decryption at the user's time of use based on

the concatenation of intrinsic values of the nodes in the racing network at this time and adopting the generated private key to generate a one-time public key to form a one-time-pair of keys for encrypted communication are generated;

5 Key transmission module, the one-time public key over any unsecured channel to selected senders for encrypting the plaintext to become a ciphertext and using symmetric key encryption for encryption of user's personal data for secure storage is transmitted;

Decryption module, the ciphertext of the senders at the receiver's end using the one-time private key to regenerate the plaintext data is decrypted or the one-time key for symmetric key encryption of user's personal data for secure storage is used.

10 26. The system as claimed in claim 25, wherein said the public key and private key are the same by using said symmetric key encryption.

27. The system as claimed in claim 25, wherein said symmetric key encryption is Advanced Encryption Standard ("AES").

15

28. The system as claimed in claim 27, wherein said AES comprises AES-128, AES-192, and AES-256.

29. The system as claimed in claim 28, if theta angles are used in summation for one-time key, then:

$$\text{One-time key at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{mi}T)\% (2\pi r_{1i}) + \frac{1}{N_T} \sum_{i=0}^{N_T} (v_{si}T)\% (2\pi r_{2i})$$

20

If angular velocities are used in summation for one-time key, then:

$$\text{One-time key at time } T = \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{mi}}{2\pi r_{1i}} + \frac{1}{N_T} \sum_{i=0}^{N_T} \frac{v_{si}}{2\pi r_{2i}}$$

25

where v_{mi} is the velocities of the motor core of the i -th node in the FNN, r_{1i} is the radius of the motor core of the i -th node in the FNN, v_{si} is the velocities of the sensory core of the i -th node in the FNN, r_{2i} is the radius of the sensory core of the i -th node in the FNN and N_T is the number of nodes in the at time T .

30. The system as claimed in claim 27, if theta angles are used in concatenation for one-time key, then:

$$\text{One-time key at time } T =$$

$Concatenate [(v_{m0}T)\% (2\pi r_{10}), \dots, (v_{mN_T}T)\% (2\pi r_{1N_T}), \dots, (v_{s0}T)\% (2\pi r_{20}), \dots, (v_{sN_T}T)\% (2\pi r_{2N_T})]$

If angular velocities are used in concatenation for one-time key, then:

$$5 \quad \text{One-time key at time } T = Concatenate [\frac{v_{m0}}{2\pi r_{10}}, \dots, \frac{v_{mN_T}}{2\pi r_{1N_T}}, \dots, \frac{v_{s0}}{2\pi r_{20}}, \dots, \frac{v_{sN_T}}{2\pi r_{2N_T}}]$$

31. The system as claimed in claim 28, wherein AES-128 using a 128 bit key, AES-192 using a 192 bit key, and AES 256 using a 256 bit key respectively to encrypt and decrypt a block of messages, wherein there are 10 rounds of processing 128 bit keys, 12 rounds of processing 192 bit key, 14 rounds of processing for 256-bit keys, respectively.

32. The system as claimed in claim 29, wherein said AES-128 comprises the following modules:

Message breaking module, the message from the sender to the receiver into 128-bits' blocks is broken;

Bytes organization module, the bytes in a given block are then organized as a 4×4 matrix;

15 Block pass module, the block through the following sub-steps for a number of times depending on the size of the cipher key, i.e. 10 times for AES-128:

Substitution module, the bytes by mapping each element in the matrix to the corresponding byte in the Rijndael S-box are substituted;

20 Row shifting module, rows are shifted by rotating each element x bytes to the left, whereas x is the index of the row;

Column Mix module, columns by multiplying every column by a predefined matrix according to the following algorithm are mixed:

if any term is greater than 2 to the power of 8, dividing the polynomial by the Galois irreducible polynomial as

$$25 \quad x^8 + x^4 + x^3 + x + 1$$

Key Addition module, a round key is added by performing a bitwise XOR operation between the columns of the matrix and the Round Key; although in the first round, the Round Key which is de facto the one-time-key generated by KeyNode is the first 128 bits of the cipher key;

Repeat module, the preceding matrix is input to the next round and the sub-steps, minimum 10 times for an AES-128 are repeated; and Excluding the final step of mixing columns for the final round.