Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 663–671
DOI: 10.15439/2022F71
ISSN 2300-5963 ACSIS, Vol. 30

Universal Key to Authentication Authority with Human-Computable OTP Generator

Sławomir Matelski
Research and Development
INTELCO LLC
Lodz, Poland
s.matelski@intelco.pl

***Abstract*—The subject of this paper is an enhanced alternative to the Multi-Factor Authentication (MFA) methods. The improvement lies in the elimination of any supplementary gadgets/devices or theft-sensitive biometric data, by substituting it with direct human-computer authentication optionally supplemented by cognitive biometric. This approach remains secure also in untrusted systems or environments. It allows only one secret as a universal private key for all obtainable online accounts. However, the features of this new solution pretend it to be used by the Authentication Authority with the Single-Sign-On (SSO) method of identity and access management, rather than for individual services. This secret key is used by our innovative challenge-response protocol for human-generated One-Time Passwords (OTP) based on a hard lattice problem with noise introduced by our new method which we call Learning with Options (LWO). This secret has the form of an outline like a kind of handwritten autograph, designed in invisible ink. The password generation process requires following such an invisible contour, similar to a manual autograph, and it can also be done offline on paper documents with an acceptable level of security and usability meeting the requirements for post-quantum symmetric cyphers and commercial implementation also in the field of IoT.**

***Index Terms*—authentication, lattice, OTP, secret key.**

I. Introduction

DUE TO the growing threat of cyber attacks, multi-factor authentication (MFA) or the two-step verification has recently become a cybersecurity standard.

Step 1 - comprises entering a user ID and static password. For security reasons, it is recommended to use different passwords for each online account. As a result, users often adopt insecure password practices (e.g., reuse or weak password) or they have to frequently reset their passwords. Blocki et al. introduced in [7] an innovative Human-Computable Passwords (HCP) scheme, which ensures that even if an adversary has seen one-hundred of the user's passwords still has high uncertainty about remaining passwords. The disadvantage of their scheme is the need to memorize dozens of pictures, mapping to numbers with the help of associated mnemonics.

In such an HCP scheme the user reconstructs each of his passwords by computing the response to a public challenge, by performing simple mathematical operations i.e. addition modulo 10. A similar approach to the idea of password computing is used by our iChip protocol inspired by the topography of electronic microchips and handwriting (Fig. 1).

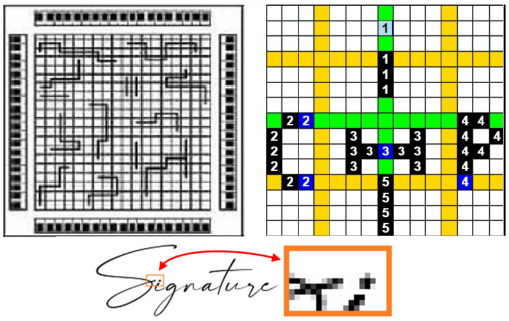


Fig. 1. Topography of: microchip, iChip, handwriting.

It requires much less effort to remember the secret in the form of only one (but detailed) picture, and only half the time for authentication. As we show in Section VI, it guarantees a safe generation of many thousands of such passwords. What follows, it can be used as an OTP generator as well.

Step 2 - requires usually an additional electronic device (using the same device in both steps may not be safe), that uses an embedded one-time password (OTP) generator or biometrics. In such cases, the OTP is entered into the verification system automatically, e.g., from a smartcard or an IoT device, or by the user after being read from the screen of a token or personal smartphone via SMS or special application. Unfortunately, this solution does not ensure that the device is being used by its owner; it must be always available, and can be stolen, lost, damaged or cloned. Biometric methods are an alternative, but these can be relatively easily cheated by *replay attack* using snooped biometric data, and with help of machine learning or AI algorithm if necessary [18].

The MFA obviously requires more time than entering a regular static password. Therefore, a human generated OTP protocol with comparable authentication time and sufficient security, eliminating the long list of drawbacks mentioned above, stands a chance of mass user acceptance.

Many attempts to achieve this goal have been made for over 30 years since Matsumoto's first publication [1] in 1991, but only two protocols have been commercially implemented: HB presented by Hopper and Blum in [2] and GrIDsure (GS) presented in [19]. We will show further that our iChip scheme has security properties better than HB and usability close to GrIDsure, while eliminating their drawbacks.

- The HB is based on the Learning Parity with Noise (LPN) method, which ensures a high level of security, but the time of ca. 668 sec. needed for authentication by a human is too long to be acceptable. Nevertheless, the properties of this protocol or later improved variants (HB+, HB#) are well suited to applicate in resource-constrained devices, such as Internet of Things (IoT) devices or RFID.
- The GrIDsure scheme has exactly the opposite properties confirmed in [19]: the high usability level and very low level of safety, as only 3 samples of challenge-response pairs are sufficient to reveal the secret. In addition, the entropy of this scheme is also low, as detailed research has shown, that users choose secret patterns that are easy to remember and frequently reused, so its scheme is highly vulnerable to dictionary attacks, as the choice is very limited due to the small grid and the small number of secret objects. The only effective improvement proposed in [20] is the use of a few secrets switched by the Out-of-Band (OOB) channel, but that requires the employment of an additional device that we intend to eliminate.

The iChip has similar usability properties to GrIDSure as the secret pattern of cells in the grid is employed by both schemes. However, the similarity is noticeable only in the so-called generator block. The most significant difference lies in the extraordinary mapping method used in iChip, which makes a huge difference in the key space (3e+5 vs 3e+154), and provides many thousands of times greater resistance against peeping attacks than GS. The conclusions about low practical entropy of GS do not apply to the iChip as getting all the easy-to-remember keys from such a huge key space is a task with a difficulty near to brute-force, which is not feasible for current supercomputers.

As mentioned above, the iChip is applicable also in step 1 of the MFA as one universal secret key to the creation of multiple original static passwords for each online account. However, the first step of MFA is redundant in this case as it relates to the same secret as the OTP generator. On the other hand, instead of the 1st step, we propose a discreet introduction of the 2nd factor in the form of cognitive memory using proposed in Section III-E, and Single Sign-On (SSO) method based on the OAuth2.0 protocol [23] under control of authentication server (as an Authenticating Authority) that owns the user identities and credentials, including the iChip secret key or container with multiple pairs of challenges and hashed OTPs.

The contributions of this work are: the challenge-response cryptographic protocol, based on lattice problem with noise, introduced by our Learning with Options (LWO) method as a more effective new variant of the LPN method of easy OTP computation by a human; a graphical interface for the implementation of that protocol, which allows the user to create his secret in the form of an easy-to-remember image, and a special wizard to compose it; both well-proven in usability and security study discussed after the presentation of mathematical rules, illustrated by examples of the iChip core and its TurboChip overlays; further protocol enhancement against active attacks and by cognitive memory usage.

The completed implementation can be tested in an interactive demo or viewed in a short film, either as a professional tutorial or an alternative version made by children participating in the research process; both available online [25]. It is much more effective to understand than a mathematical description.

II. Learning with Options method as the main strength of the protocol

An important element of the iChip scheme is the implementation of the Learning with Rounding (LWR) method, which is an LPN variant of worst-case hard lattice problem included in the lattice-based cryptography. The implementation of core LPN or Learning with Errors (LWE) methods increase the security of any protocol; however, the degree of usability is reduced, and authentication requires much more time, as the user has to perform additional protocol rounds to compensate for rounds lost to incorrect responses due to reduced resistance to random attacks. In contrast, the LWR and described below LWO methods requires only correct responses. The iChip uses Equation 1 in Section III-A, as its base function which satisfies the criteria of the LWR method of deterministic rounding by $x \mod p$, where $p = 10$ is admittedly too small to effectively introduce noise, but convenient for human computation. This function is a node for the various protocol variants and for our proposed LWO method of introducing noise, which is far more efficient.

III. The iChip as an OTP generator

The iChip is a challenge-response protocol to authenticate the user to the verifier using the shared secret, where the user has to answer the challenge generated by the verifier (server). The way the iChip scheme worked was inspired by the image of the photolithographic mask used to create conductive paths on the surface of PCBs (Printed Circuit Board) or ICs (Integrated Circuits) like shown in Fig.1. The user composes his secret by designing such a layout in a special wizard by drawing a map of blocks B of masking elements as paths conducting the digital signal from input to output; provided from the generator block. These paths will determine the change in value from V_{inp} at the input to V_{out} at the output and define their properties and mutual logical relations. This layer consists of $n \times n$ fields and is represented by the C matrix, containing $n \times n$ cells.

The user specifies his secret key S by specifying a list of b blocks that occupy the fields selected by him from the C matrix, and specifies the block elements that act as input or output. For a short and easy explanation, we will use the example of the secret key illustrated in Fig. 2 or Fig. 4 as an iChip layout and the matrix coordinates of the input and output elements encoded hexadecimal in the associated table, while for the description of the protocol, we will use the Python convention. The C matrix is a set of n^2 random values generated by verifier as $C = [[V_{1,1}, V_{1,2}, ..., V_{1,n}], \dots [V_{n,1}, V_{n,2}, ..., V_{n,n}]]$. Each i-th element of block $B[i] = [y_i, x_i, z_i]$ is defined by 3 parameters: row y and column x as field coordinates (y, x) in matrix C, and parameter z defining its state: $z = \{I, O\}$, where: $I = Input, O = Output$.

We use also an alternative compact notation of block elements as: $B^z_j[i] = B^z_j[(y_i, x_i)]$. Each block B_j is a list of such elements, divided into two segments for inputs and outputs: $B = [B^I, B^O] = [B[1], B[2], ..., B[k]]$. A list of b blocks B_j is included in the secret $S = [B_0, B_1, \dots B_{b-1}]$, where $0 \leqslant j < b$. The parameters of the algorithm are denoted by four positive integers $N, L, b, k \in \mathbb{N}$, where:

- chip size (the matrices describing both private part of the key and the challenge matrix have size $N = n \times n$);
- parameter describing OTP length, $L \leqslant 10$;
- maximal number of blocks, for the sake of clarity and memorability we restrict $1 \leqslant b \leqslant 10$;
- maximal block length $k \leqslant 10$;

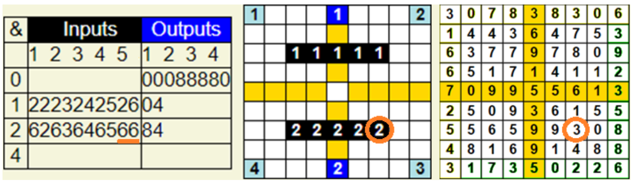
A. Generating OTP

$G = B_0$ is the first of these blocks in key S, and it is called a generator because it does not contain inputs and the values $V_G = C[G]$ from all its $L = |B_0|$ output elements are mapped by the remaining blocks. The user has to remember the position of all blocks and their order in the S. The verifier generates a challenge matrix C of N random digits. To generate the OTP, the user has to collate the C matrix with the secret key S and calculate all OTP digits, one at each i-th of $L = |\text{OTP}|$ rounds of the protocol in the following 3 steps:

1) Read the V^i_{inp} value of the $G[i]$ element in C at position (y_i, x_i): $V^i_{inp} = C[G[(y_i, x_i)]]$
2) Starting from j-th block (where $j = 1$ in the 1st round), search input elements ($z = Input$) of j-th block for the coordinates (y_i, x_i) such that $V^i_{inp} = C[B^I_j[(y_i, x_i)]]$. If no such coordinates are found in the j-th block, move to the subsequent block. By $j = \phi$ denote the index of the *current block* (ϕ) in which the searched so-called *target input* (ψ) has been found first and let
$$V^i_{out} = C[B_\phi[y, x, z = Output]].$$
If the search fails for all $j < b$: let $V^i_{out} = V^i_{inp}$.
3) The i-th digit of the OTP you will get as
$$OTP[i] = (V^i_{inp} + V^i_{out}) \mod 10 \tag{1}$$

To avoid overloading the first blocks, it is recommended to resume the search for V^i_{inp} from the block next to the last searched. For additional security, the following three exceptions/rules (*I, *O, *Θ) have been added to the 2nd step of the algorithm; these significantly increase the resistance of the iChip protocol against passive attacks with a statistical algorithm or Gaussian Elimination. For their consideration let (y_i, x_i) be the coordinates on which the *target input* ψ in B such that $C[B_\phi[\psi]] = V^i_{inp}$ was found first in the challenge matrix.

However, first, we will present a simple example illustrated in Fig. 2 to explain the principle of calculating the OTP without the exceptions mentioned above. Alternatively, it is recommended to watch the short video tutorial [25].



&	Inputs	Outputs
	1 2 3 4 5	1 2 3 4
0		00088880
1	2223242526	04
2	6263646566	84
4		

1				1				2
		1	1	1	1	1		
		2	2	2	2	2		
4				2				3

3	0	7	8	3	8	3	0	6
1	4	4	3	6	4	7	5	3
6	3	7	7	9	7	8	0	9
6	5	1	7	1	4	1	1	2
7	0	9	9	5	5	6	1	3
2	5	0	9	3	6	1	5	5
5	5	6	5	9	9	3	0	8
4	8	1	6	9	1	4	8	8
3	1	7	3	5	0	2	2	6

Fig. 2. An example of a secret: block input elements given as black fields and output as blue or light blue fields. The positions of all input and output elements are hexadecimal encoded in the associated table. The first column (&) contains the index of each block. On the right: The challenge matrix.

There are generator block 0 containing 4 light blue cells in the matrix corners and two mapping blocks labeled by their index (1 or 2) in the example above (Fig. 2). In the 1st round, we read the value $V^1_{inp} = 3$ from the 1st element of generator block at position (0, 0).

We look for this value sequentially in all mapping blocks from 1 to 2. The first occurrence of this value is in the last element of block 2, i.e. $B^I_2[5]$ in cell (6, 6), which is the *target input* $\psi = 5$ in the *current block* $\phi = 2$. Now, we read a value of the output element of this block, which is in cell (8, 4), hence $V^1_{out} = C[8, 4] = 5$. The 1st round ends with a calculation of the 1st OTP digit according to Equation 1 as:
OTP[1] = $(V^1_{inp} + V^1_{out}) \mod 10 =$
$= (C[G[0]] + C[B^O_2[1]]) \mod 10 =$
$= (C[0, 0] + C[8, 4]) \mod 10 =$
$= (3 + 5) \mod 10 = 8.$

EXTRA RULES / EXCEPTIONS

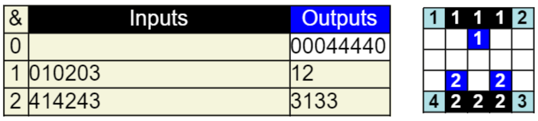
*I) Let V^i_{inp} be the sum of all input elements of the current block, from ψ to $\psi + n$, where $\psi + n \leqslant |B^I_\phi|$ and $n \leqslant 2$:

$$V^i_{inp} = (\sum_{k=0}^{k \leqslant n} C[B_\phi[\psi + k]]) \mod q \tag{2}$$

This introduces non-linearity to cryptanalysis and protection against Gaussian Elimination, as the number of arguments in Equation 2 varies randomly in each challenge. Depending on the variant of *I, the q modulus can be 10 or omitted as default.

*O) If the *current block* B_ϕ contains more than one output element $|B^O_\phi| > 1$, then randomly choose one of them as $V^i_{out} = C[B^O_\phi[randrange(1, |B^O_\phi|)]]$. This is the case of using the LWO method illustrated by Fig. 3: Block 2 with fields labeled by 2 has two outputs/options at positions (3, 1) and (3, 3). If the value searched for is found in this block, then the user has to choose one of these two options at random.

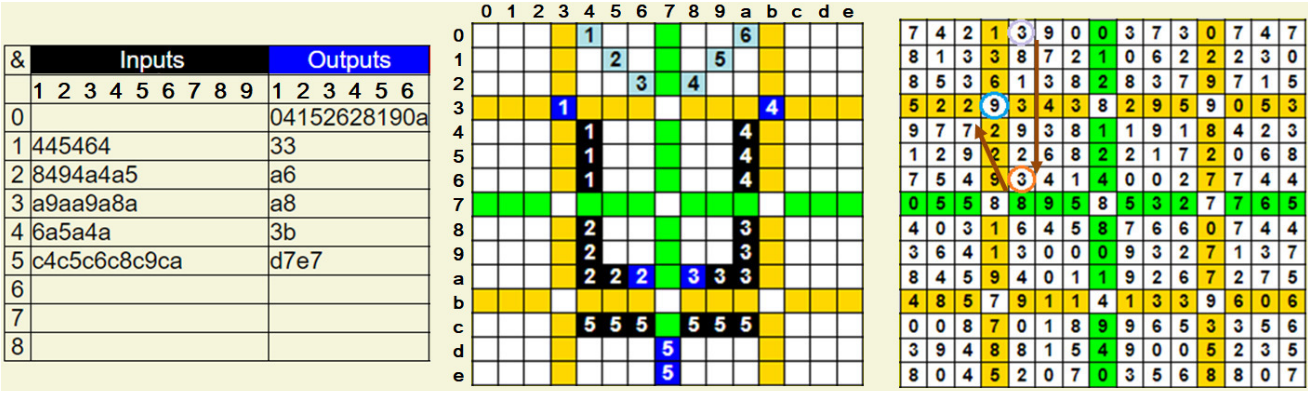
*Θ) If the *current block* B_ϕ has no output $|B^O_\phi| = 0$, then we use the next input instead: $V^i_{out} = C[B_\phi[(\psi + 1) \mod |B_\phi|]]$.



&	Inputs	Outputs
0		00044440
1	010203	12
2	414243	3133

1	1	1	1	2
		1		
	2		2	
4	2	2	2	3

Fig. 3. An example of secret with exception *O.



&	Inputs 1 2 3 4 5 6 7 8 9	Outputs 1 2 3 4 5 6
0		04152628190a
1	445464	33
2	8494a4a5	a6
3	a9aa9a8a	a8
4	6a5a4a	3b
5	c4c5c6c8c9ca	d7e7
6		
7		
8		

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e
0					1						6				
1						2				5					
2							3		4						
3				1								4			
4					1						4				
5					1						4				
6					1						4				
7															
8					2						3				
9					2						3				
a					2	2	2		3	3	3				
b															
c					5	5	5		5	5	5				
d								5							
e								5							

7	4	2	1	3	9	0	0	3	7	3	0	7	4	7
8	1	3	3	8	7	2	1	0	6	2	2	2	3	0
8	5	3	6	1	3	8	2	8	3	7	9	7	1	5
5	2	2	9	3	4	3	8	2	9	5	9	0	5	3
9	7	7	2	9	3	8	1	1	9	1	8	4	2	3
1	2	9	2	2	6	8	2	2	1	7	2	0	6	8
7	5	4	9	3	4	1	4	0	0	2	7	7	4	4
0	5	5	8	8	9	5	8	5	3	2	7	7	6	5
4	0	3	1	6	4	5	8	7	6	6	0	7	4	4
3	6	4	1	3	0	0	0	9	3	2	7	1	3	7
8	4	5	9	4	0	1	1	9	2	6	7	2	7	5
4	8	5	7	9	1	1	4	1	3	3	9	6	0	6
0	0	8	7	0	1	8	9	9	6	5	3	3	5	6
3	9	4	8	8	1	5	4	9	0	0	5	2	3	5
8	0	4	5	2	0	7	0	3	5	6	8	8	0	7

Fig. 4. An example of a secret defined by the user and challenge matrix with a schema for determining the 1st digit of OTP.

B. Advanced Example

Based on Fig. 4 we will compute the 6-digit OTP as follows: The generator block contains V_G = C[G] = [3, 7, 8, 8, 6, 3]. The 1st element $V_G[1]$ at position (0, 4) has a value of 3. When looking for it sequentially in blocks 1 to 6, it can be found in the 3rd input element of the 1st block $B_1^I[3]$ at position (6, 4), (marked in the red ring as a target input ψ); the output element $B_1^O[1]$ of this block is in cell (3, 3) with a value of 9. The 1st digit of OTP is calculated according to Eq. 1 as OTP[1] = (3 + 9) mod 10 = 2.

$V_G[2]$ at position (1, 5) has a value of 7, which is also in $B_4^I[2]$ at position (5, a) and the output element $B_4^O[1]$ has a value of 9. Since $\psi = 2 < |B_4^I| = 3$, then due to rule *I: $V_{inp}^2 = C[B_4^I[2]] + C[B_4^I[3]] = 7 + 1 = 8$. Now, according to Eq. 1 we can calculate: OTP$[2] = (8 + 9) \mod 10 = 7$.

$V_G[3]$ = 8 appears in $B_5^I[3]$ at position (c, 6), but this block has 2 output elements $B_5^O[1]$ in cell (d, 7) and $B_5^O[2]$ in cell (e, 7). Therefore due to exception *O, we can choose any of them; assuming we choose the first with value of 4,
OTP[3] = (8 + 4) mod 10 = 2.

$V_G[4]$ = 8, hence this round is similar to the previous one, but now, we use the second output $B_5^O[2]$ in cell (e, 7) for our calculations: OTP[4] = (8 + 0) mod 10 = 8.

$V_G[5]$ = 6 appears in $B_I^2[1]$, but this block has 4 inputs, therefore we add three of them to $V_{out}^5 = 1$, hence: OTP$[5] = 6 + 3 + 4 + 1 \mod 10 = 4$. Entire OTP = [2, 7, 2, 8, 4, 2].

C. TurboChip overlays for the iChip protocol

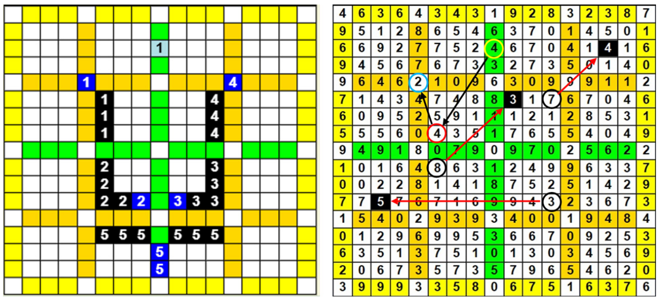
For a radical reduction of an authentication time, we have developed two variants of TurboChip overlays for the iChip scheme. They only uses 1 round of the base iChip protocol and only needs 1 element in the generator block. In the example below illustrated in Fig.5: Since $V_G = [4]$, then OTP$[1] = (4 + 2) \mod 10 = 6$. However, we keep this value a secrete as $V = 6$ and use it according to FlexiChip or ClickChip. For FlexiChip, we calculate each i-th OTP number as OTP$[i] = (V + C[B_\phi^I[\psi + i]]) \mod 10$; If $|B_\phi^I| < \psi + i$ then continue in the next block. In this example: $\phi = 1; \psi = 3$, but $|B_1^I| <$ 3+1, hence OTP$[1] = V + C[B_2^I[1]] \mod 10 = 6 + 8 \mod 10 = 4$; OTP$[2] = 6 + C[B_2^I[2]] \mod 10 = 6 + 1 \mod 10 = 7$; e.t.c.

The ClickChip generates response as matrix coordinates instead of digits. This approach requires a bit of proficiency from the user, however, it allows cutting the number of protocol rounds in half. It stands as a good trade-off for it and is explained in the example illustrated in Fig. 5. Determining of 3 matrix coordinates in the range of (-8, -8) to (8, 8), one at each i-th of 3 rounds of ClickChip protocol is as follows:

We go to the i-th element in $j = \phi + i$ block. Now, get the value of this element to calculate the distance of d_i fields, where $d_i = (V + C[B_j^I[i]]) \mod 10$; to move the virtual pointer on a horizontal, vertical or diagonal line towards the centre of the grid. Important notes: The order of choosing these directions is random, but 1 diagonal and 1 reverse direction must be used if d_i is lower than the distance from the edge of the grid. For long blocks: If $|B_j^I| > 3$ then the counting of d_i cells starts from the element whose value is closest to d_i.

To quickly find the endpoint, move the pointer in jumps a'4 fields, with a help of the coloured background lines.

In our example, the current block $\phi = 1$, therefore, in the 1st round we go to block 2, and get the value of the 1st element in it, hence $d_1 = (6 + 8) \mod 10 = 4$. Now, we move the pointer e.g., diagonal from a position (-1, -3) to position (3, 1). In the 2nd round $d_2 = (6 + 3) \mod 10 = 9$, so we click the position (-3, -6). In the 3th round $d_3 = (6 + 7) \mod 10 = 3$, but to satisfy the protocol rules, we alter the direction to the opposite $d_3 = -3$ and then click the cell (6, 6).



								1								
				1								4				
					1						4					
					1						4					
					1						4					
					2						3					
					2						3					
					2	2	2		3	3	3					
					5	5	5		5	5	5					
								5								
								5								

4	6	3	6	4	3	4	3	1	9	2	8	3	2	3	8	7
9	5	1	2	8	6	5	4	6	3	7	0	1	4	5	0	1
6	6	9	2	7	7	5	2	4	6	7	0	4	1	4	1	6
9	4	5	6	7	6	7	3	3	2	7	3	5	6	1	4	0
9	6	4	6	2	1	0	9	6	3	0	9	9	9	1	1	2
7	1	4	3	4	7	4	8	8	3	1	7	6	7	0	4	6
6	0	9	5	2	5	9	1	1	1	2	1	2	8	5	3	1
5	5	5	6	0	4	3	5	1	7	6	5	5	4	0	4	9
9	4	9	1	8	0	7	9	0	9	7	0	2	5	6	2	2
1	0	1	6	4	8	6	3	1	2	4	9	9	6	3	3	7
0	0	2	2	8	1	4	1	8	7	5	2	5	1	4	2	9
7	7	5	7	6	7	1	6	9	9	4	3	2	3	6	7	3
1	5	4	0	2	9	3	9	3	4	0	0	1	9	4	8	4
0	1	2	9	6	9	9	5	3	6	6	7	0	9	2	5	3
6	3	5	1	3	7	5	1	0	1	3	0	3	4	5	6	9
2	0	6	7	3	5	7	3	5	7	9	6	5	4	6	2	0
3	9	9	9	3	3	5	8	0	6	7	5	1	6	3	7	6

Fig. 5. Secret key and challenge matrix with the schema of the ClickChip for OTP determining protocol.

The variant with an 18x18 grid shown in Fig. 6 is more convenient for moving the pointer in jumps a'3 fields. The worst-case probability of randomly hitting the correct OTP is here $p = 3/207 \cdot 2/207 \cdot 1/207 = 7e^{-7}$.

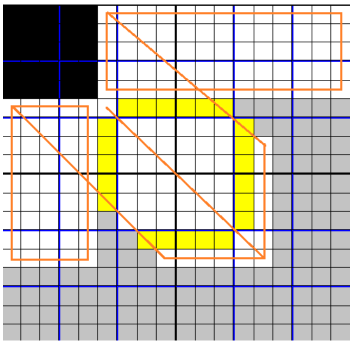


Fig. 6. ClickChip's grid variant of 18x18 = 324 cells with 25 secret inputs marked in black and the non-clickable area marked in grey.

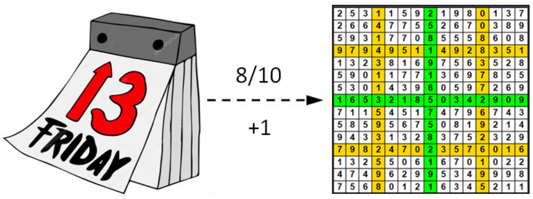
D. Preliminary stage against active attacks

In this stage, User U and Verifier V swap their roles, so V responds to U's challenge. V initiates the authentication process by sending a challenge to U. Then, U randomly clicks on any field in C, and the element of the block closest to that field is used as ψ for the calculation of OTP[0]. U remembers it and sends such a challenge to V. In response, V has to calculate the OTP[0] in this same way, and then generate a new challenge, but the value of $V_G[0]$ is set to OTP[0], which is hidden. The next 3 rounds run as usual according to ClickChip.

E. Biometrics, captcha and clock in the iChip scheme

To further strengthen the iChip protocol it is beneficial to increase the entropy of the random option selection in the LWO case by aid of a pseudorandom number generator, the result of which is entered via a cognitive biometric interface working like an OOB channel. This interface (emOTP) is based on the stimulation of emotional states by recalling the knowledge already acquired in the past and preserved in long-term memory. The great advantage of this approach is that the user is not required to remember a secret specially built for this purpose, so they can without much effort, insert a lot of such items into the user's account profile resources in the form of catchwords or pictures, associated with its evaluation in points: +1 as positive, -1 as negative and 0 as neutral; referring to a universal question, e.g., *Do you like it?* with possible answers: *Yes, No, Neutral*. The response can be effortlessly applied to choose/address 1 of the 2 or 3 options when using the LWO method.

Increasing the range of ratings to 10 points requires changing the above question to *How much do you like it?*. Now, the response ranging from 0 to 9 allows the generator block to be completely replaced. Unfortunately, the limitation of such a generator is its inaccuracy, occurring from poor behavioural repeatability. Nevertheless, due to the LWO, the unintended incorrect response to the emOTP challenge does not affect the response correctness. Hence, an additional protocol round to compensate for this mistake is not needed. In other cases than the LWO, such a 3 stage emOTP trigger/generator in the iChip protocol can be used as a biometric factor in the MFA. The emOTP criteria should be quick to evaluate and be personal, rather than popular and predictable. The challenge in the form of an image (as an example in Fig. 7) can work well as a captcha at the same time.



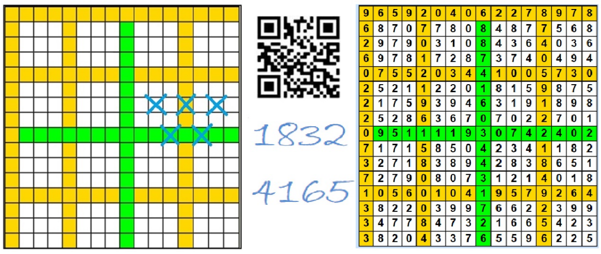
2	5	3	1	1	5	9	2	1	9	8	0	1	3	7
2	6	6	4	7	7	5	5	2	6	7	0	3	8	9
5	9	3	1	7	7	0	8	5	5	5	8	6	0	8
9	7	9	4	9	5	1	1	4	9	2	8	3	5	1
1	3	2	3	8	1	6	9	7	5	6	3	5	2	8
5	9	0	1	1	7	7	1	3	6	9	7	8	5	5
5	3	0	1	4	3	9	6	0	5	9	7	2	6	9
1	6	5	3	2	1	8	5	0	3	4	2	9	0	9
7	1	1	5	4	5	1	7	4	7	9	6	7	4	3
5	8	5	9	5	6	7	5	0	8	1	9	2	1	4
9	4	3	3	1	3	2	8	3	7	5	2	3	2	9
7	9	8	2	4	7	0	2	3	5	7	6	0	1	6
1	3	2	5	5	0	6	1	6	7	0	1	0	2	2
4	7	4	9	6	2	9	9	5	3	4	9	9	9	8
7	5	6	8	0	1	2	1	6	3	4	5	2	1	1

Fig. 7. An example of the emOTP challenge and its evaluation depending on the user and not on a statistical basis.

As an alternative and much simpler source of entropy for the LWO method, a random moment of reading seconds from the system clock or user's watch can be used.

IV. iChip for a Hash-Based Signature

To ensure that the user authorizes the correct message M (e.g. transaction conditions), and not falsified by active adversary, we propose the Human-Hashed variant of (hash-based) message authentication code (MAC). Such a HHMAC can be used offline, i.e. the previously computed SHA-256 function from M and written here as $h()$ is hashed by iChip's OTP. The iChip-256 variant with N=256 fields layout is the most optimal here. The challenge matrix C created by the RNG is modified by adding one bit of the hashing result H to each of the $|C| = 256$ elements, where $H = h(M, C)$ is the hash value of a message M and C according to the formula $C'[i] = (C[i] + H_2[i]) \mod 10$, where $0 \leqslant i \leqslant 255$. The user performs the signature by entering the OTP on the keyboard or writing OTP digits on the document containing: a blank iChip grid for global UID (optional), the challenge matrix C', a QR code specifying the document identifier in the repository for automatic scanning and HHMAC verification.



9	6	5	9	2	0	4	0	6	2	2	7	8	9	7	8
6	8	7	0	7	7	8	0	8	4	8	7	7	5	6	8
2	9	7	9	0	3	1	0	8	4	3	6	4	0	3	6
6	9	7	8	1	7	2	8	7	3	7	4	0	4	9	4
0	7	5	5	2	0	3	4	4	1	0	0	5	7	3	0
2	5	2	1	1	2	2	0	1	8	1	5	9	8	7	5
2	1	7	5	9	3	9	4	6	3	1	9	1	8	9	8
2	5	2	8	6	3	6	7	0	7	0	2	2	7	0	1
0	9	5	1	1	1	1	9	3	0	7	4	2	4	0	2
7	1	7	1	5	8	5	0	4	2	3	4	1	1	8	2
3	2	7	1	8	3	8	9	4	2	8	3	8	6	5	1
7	2	7	9	0	8	0	7	3	1	2	1	4	0	1	8
1	0	5	6	0	1	0	4	1	9	5	7	9	2	6	4
3	8	2	2	0	3	9	9	7	6	6	2	2	3	9	9
3	4	7	7	8	4	7	3	2	1	6	6	5	4	2	3
3	8	2	0	4	3	3	7	6	5	5	9	6	2	2	5

Fig. 8. A sample signature on a paper document includes:
Global UID, Transaction ID, HHMAC, Challenge C'.

V. Brief analysis of usability

- Intelligibility

Our time-limited study only focused on a small group of children aged 8-10, assuming that the adult performance should be better, because modular addition and abstract thinking are required, which develops with age [16]. For this group, the iChip protocol was compared with that of a board game, more especially the well-known Monopoly or Jumanji, where the throws of the dice symbolize the operation of the generator block, and all the fields on the board forming the track constitute the iChip blocks, which user have to go to achieve the target field/input and finally make a decision according to the rules of the game protocol The children took 1 standard lesson unit (45') to learn the protocol and the special wizard to design their own microchip.

- Memorizing and Rehearsing

The appropriate distribution of block elements is of major importance for entropy level and easy memorization of the entire structure of the secret. To obtain the maximum practical entropy and to make it easier to remember the secret, a suitable background image is very helpful, which can be built individually by the user or proposed by the wizard as a random structure. It is profitable to draw the secret contours in a single sequence like a short piece of text (e.g., Fig. 1) or a simple shape (e.g., Fig. 5). Additionally, since all key elements are used each time, the whole secret image can be easily remembered after 30-45 minutes of repeated authentication training attempts and frequently refreshed at the use stage.

- Authentication Time

The authentication time is proportional to the user's cognitive workload - ranges from 4 to 8 seconds (≈ 6) in each round of response, depending on the composition of the secret and the user's skill. After several searches, visual perception adapts a parallel analysis approach, i.e. the search for an ψ element with V_{inp} is not performed element-by-element, but in blocks, just like reading a text, with whole words being interpreted, rather than individual letters. Each modular addition and block search require ca. 1 sec. For the user who has to look at the keyboard to enter OTP, it will be easier, and faster, to use voice input, which is also a good source of biometric data and a 3-rd authentication factor. After introducing of TurboChip overlay, the authentication time is significantly reduced up to $\approx$15 seconds if the user becomes an experience in using it.

VI. Brief analysis of security

The resistance to a random attack depends on the number of OTP digits calculated by the user. Their number L is arbitrary and depends on the needs of the authentication system, e.g., $L = 6$ like OTP in most e-banking systems. Using the ClickChip overlay slightly weakens the protocol if the attacker records user responses, as it allows to reduce the number of possible click locations from $N = 324$ down to 207 (in the worst case - Fig. 6), however, the probability of p=7e-7 of randomly hitting the correct OTP is still lower than for 6 decimal digits, i.e. p=1e-6.

The iChip's resistance to active attacks is ensured in the preliminary stage (see Section 3.4) or by the hashing and signing the authenticated message, as the HHMAC is valid only for the signed message (see Section IV).

As the challenge in the iChip protocol is generated full at random, it is fully immune to frequency analysis.

The iChip's entropy is of course lower in practical use than its key size of 512 bits, but much higher than a text password due to large number of possible fonts and their positioning on the large grid or cell order. A good example is the word *iCHIP* used in Fig 1. The number of possibilities for designing this contour is enormous, despite the use of many symmetries compared to the number of combinations that the use of lowercase and uppercase letters offer.

The resistance to brute-force and Grover's quantum algorithm is provided by NP-hard lattice problem and huge keys space (see Table 1), estimated as follows:

$$\frac{N!}{(N-L)!}\cdot\sum_{i=B}^{B+b_0}(\sum_{d=1}^{k}\binom{N-L}{d}\cdot\sum_{j=E}^{E+e_0}\binom{N-L}{j}+ \\ +\sum_{j=E}^{E+e_0}\frac{(N-L)!}{(N-j-L)!})^i \tag{3}$$

where:
$N = n \times n$ is the size of Chip's matrix, default 16 x 16
L is the number of OTP digits, default 6
$[B, B + b_0] =$ number of blocks, in the range 3 to 7
$[E, E + e_0] =$ number of input elements in block: 3 to 9
$[0, k] =$ number of output elements in block: 0 to 2 or max. 3

Estimating the resistance of an authentication protocol to peeping attacks is very important, but also highly-complex, especially in the case of iChip, as it can simultaneously use many protocol variants, which interfere with each other and further increase their effectiveness (see Appendix). Therefore, we considered them separately, based on the results of related works: [2], [7], [6].

By using the LPN method or its variants, it is possible to efficiently reduce the amount of information leaked about the secret by random injection of erroneous information in the LWE or deterministic rounding in the LWR. Such leakage can also be reduced by modular reduction $x \mod p$, or by randomly selecting one of several correct LWO options.

It also works similar to introducing an error in the expected, acceptable narrow range. However, introducing noise too much here increases vulnerability to random attacks as well.

As shown in [7]: The k number of arguments used in the function $f(x_1, x_2, ..., x_k) = x_1 + x_2 + ... + x_k \mod p$ depends on the safety function for the statistical algorithm $r(f) = k/2$, however, f cannot be linear, because then the security for Gaussian Elimination is $g(f) = 0$ and the Equation 1 for LWR implementation in iChip takes only 2 arguments (k = 2). Therefore, in this case the secret could be recovered even from $O(n)$ challenge-response samples: $m = n^s, s = min(g, r)$.

TABLE I
COMPARISON OF THE MOST IMPORTANT AND OPTIMAL PARAMETERS.

HGPP [Ref.]	k secret objects	n objects pool	Window size	Key size (bits)	Password space	$s(f)$	Guess Rate /round	No of rounds	≈Time/Auth. (sec)
HB [2]	15	200	200	70	1.5e+22	2	0.5	20	668
APW [5]	16	200	200	79	8.4e+24	1	0.1	6	348
CAS Hi [4]	60	240	20	187	2.4e+57	1	0.5	20	221
CAS Lo [4]	60	80	80	70	8.9e+21	1	0.25	10	122
Foxtail [3]	14	140	30	60	6.5e+18	1	0.5	20	213
CHC [13]	5	112	83	24	1.4e+8	1	0.22	10	93
HCP [7]	50	50	14	164	1.0e+50	1.5	0.1	6	42
GrIDsure [19]	6	25	25	28	2.4e+8	1	0.1	6	4
iChip256	36	256	256	512	3.2e+154	2	0.1	6	36
ClickChip	31	289	289	500	1.7e+150	2	≈ 0.01	3	21
FlexiChip	31	256	256	490	1.0e+146	2	0.1	6	15

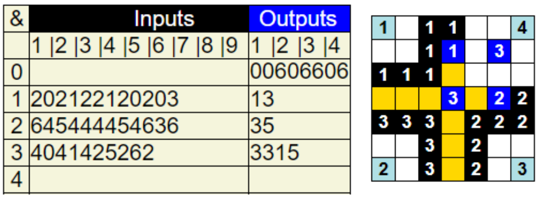
Fortunately, the introduction of noise by the LWO method in iChip brings the same effect as LPN in the HB [2], which does not allow the simple use of Gaussian Elimination, and the adversary needs to see $O(n^2)$ samples to reveal the secret, also in the case of secret's low entropy [14].

On the other hand, introducing an exception *I gives up to 2 additional arguments by Eq. 2 to this base function, hence $2 \leqslant k \leqslant 4$. The number of these arguments is not constant but varies randomly in each challenge, so the Eq. 1 becomes highly nonlinear, especially since V_{out} is the result of a previously used mapping,

Any statistical adversary needs approximately $m = n^{r(f)/2}$ samples to recover the secret, where n is the key size (512 bits for iChip), therefore, if both exceptions (*I and *O) are used then estimated safety function is limited by: $s = min(g, r) = min(2, 2) = 2$, hence $m \approx 262,144$ challenge-response samples are needed to reveal the secret.

We tested the resistance of the protocol against finding the secret key with an advanced Genetic Algorithm, which ran for m=1,000, m=10,000 and m=20,000 samples over several days on a computer with an 18-core CPU (Intel i9). The secret created in the default grid size of $N = 256$ but without exceptions (*I, *O) was found after approx. 2 hours of operation. After introducing the LWO, the cracker found a secret key only for microparameters i.e. $N = 25$ (Fig. 3).

With the simultaneous inclusion of *I and *O exceptions, the 2-days search did not give a correct result even for $N = 49$, represented by the microkey in Fig. 9.



&	Inputs	Outputs
	1 |2 |3 |4 |5 |6 |7 |8 |9	1 |2 |3 |4
0		00606606
1	202122120203	13
2	645444454636	35
3	4041425262	3315
4		

1		1	1			4
		1	1		3	
1	1	1				
			3		2	2
3	3	3		2	2	2
		3		2		
2		3		2		3

Fig. 9. Microkey 7x7 with exceptions *I & *O.

The tests conditions and results are available online [25].

VII. RELATED WORK

Referring to the data in Table 1 of the article from 13th NDSS [12] and the latest publications until today, we have compiled in Table 1, the parameters of the best Human Generated Passwords Protocols, that were created in the years 1991-2017 (there is no significant contribution after 2017), as a comparison with iChip. As we can see, the iChip's parameters have a significant advantage over all others, both in terms of security (key size, keyspace, $s(f)$) and usability (secret's memorizing and authentication time closest to grIDsure). Only HB, HCP, and iChip are protected against linearization [9], where $m = O(n^s)$ strongly depends on the key size n.

The enhanced versions of HB+, Foxtail+ also offer protection against active attacks, but only ClickChip and HHMAC are suitable for the user due to the required authentication time.

VIII. CONCLUSIONS

The result of our work are the iChip protocol and two TurboChip overlays, which significantly accelerate the OTP generation process and all these variants meet the safety and usability criteria required for commercial implementation.

- The FlexiChip is our favorite due to the smaller workload for the user and flexibility in generating passwords of any length L from 1 to $k = |S|$. As the video tutorial [25] shows, the 6-digit OTP computation time easily reaches 15 seconds. The protection against active adversary attacks is provided by the HHMAC with use of standard hash algorithm, preferable SHA-256 or SHA-512. Such signature can be also performed by the user offline on paper documents without any gadgets and automatically scanned and loaded into the system, where this signature is verified.

- The ClickChip overlay has the advantage of using the preliminary round to immediately detect an active adversary attacks and is more glamorous, but requires more proficiency in determining matrix coordinates. At the level of quasi automatic distance evaluation, the authentication time could be reduced to even 15 seconds, similar to the FlexiChip.

The iChip protocol parameters are well-suited to applicate also in resource-constrained devices, like IoT or RFID.

References

[1] T. Matsumoto, H. Imai. *Human Identification Through Insecure Channel*. EUROCRYPT 1991. https://doi.org/10.1007/3-540-46416-6_35

[2] N. Hopper and M. Blum. *A Secure Human-Computer Authentication Scheme*. Lecture Notes in Computer Science, 2248, 2000.

[3] S. Li, H.-Y. Shum. *Secure Human-Computer Identification (Interface) Systems against Peeping Attacks: SecHCI.* IACR's Cryptology ePrint Archive: Report 2005/268.

[4] D. Weinshall. *Cognitive authentication schemes safe against spyware.* IEEE Symposium on Security and Privacy (S&P), 2006.

[5] H. J. Asghar, J. Pieprzyk, H. Wang. *A New Human Identification Protocol and Coppersmith's Baby-Step Giant-Step Algorithm*. Applied Cryptography and Network Security, 349-366, 2010.

[6] M. Monteiro, K. Kahatapitiya, H. J. Asghar, K. Thilakarathna, T. Rakotoarivelo, D. Kaafar, S. Li, R. Steinfeld, J. Pieprzyk. *Foxtail+: A Learning with Errors-based Authentication Protocol for Resource-Constrained Devices.* IACR's Cryptology ePrint Archive, Report 2020/261.

[7] J. Blocki, M. Blum, A. Datta., S. Vempala. *Toward human computable passwords.*. ITCS 2017. https://doi.org/10.4230/LIPIcs.ITCS.2017.10

[8] M. Blum, S. Vempala. *Publishable humanly usable secure password creation schemas.* AAAI Conference on Human Computation and Crowdsourcing, HCOMP, 32–41, 2015.

[9] H. J. Asghar, R. Steinfeld, S. Li, M. A. Kaafar, J. Pieprzyk. *On the Linearization of Human Identification Protocols: Attacks Based on Linear Algebra, Coding Theory, and Lattices.* IEEE Transactions on Information Forensics and Security, 10(8), 1643–1655, 2015.

[10] S. Samadi, S. Vempala, A. T. Kalai. *Usability of humanly computable passwords.* In arXiv preprint arXiv:1712.03650, 2017.

[11] A. Juels and S. Weis. *Authenticating Pervasive Devices with Human Protocols*, Advances in Cryptology - CRYPTO 2005, vol 3621.

[12] Q. Yan , J. Han , Y. Li , R. H. Deng. *On Limitations of Designing Usable Leakage Resilient Password Systems: Attacks, Principles and Usability.*19th Network and Distributed System Security Symposium (NDSS), 2012.

[13] S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget. *Design and evaluation of a shoulder-surfing resistant graphical password scheme.* In Proceedings of the working conference on Advanced visual interfaces, pages 177–184, 2006. https://doi.org/10.1145/1133265.1133303

[14] J. Alwen, S. Krenn, K. Pietrzak, D. Wichs. *Learning with Rounding, Revisited.* Advances in Cryptology - CRYPTO 2013.

[15] A. Bogdanov, S. Guo, D. Masny, S. Richelson, A. Rosen. *On the Hardness of Learning with Rounding over Small Modulus*, Cryptology ePrint Archive, Report 2015/769.

[16] I. Dumontheila. *Development of abstract thinking during childhood and adolescence: The role of rostrolateral prefrontal cortex*. Developmental Cognitive Neuroscience, 57–76, 2014.

[17] S. Patil, S. Mercy, N. Ramaiah. *A brief survey on password authentication.* International Journal of Advance Research, Ideas and Innovations in Technology, 4(3), 943-946, 2018.

[18] F. Wang, L. Leng, A. Teoh, J. Chu. *Palmprint False Acceptance Attack with a Generative Adversarial Network (GAN).* Applied Sciences, 10. 8547, 2020. https://doi.org/10.3390/app10238547

[19] S. Brostoff, P. Inglesant, A. Sasse. *Evaluating the usability and security of a graphical one-time PIN system*, Proceedings of the BCS-HCI 2010, Dundee, United Kingdom, 2010.

[20] R. Jhawar, P. Inglesant, N. Courtois and M. A. Sasse. *Strengthening the security of graphical one-time PIN authentication*. 5th International Conference on Network and System Security, 2011.

[21] Z. Golebiewski, K. Majcher, F. Zagorski, M. Zawada. *Practical Attacks on HB/HB+ Protocols.* ePrint Archive, Report 2008/241.

[22] K. Sadeghi, A. Banerjee, J. Sohankar and S. K. S. Gupta. *Geometrical Analysis of Machine Learning Security in Biometric Authentication Systems*, 16th IEEE International Conference on Machine Learning and Applications (ICMLA), 309-314, 2017.

[23] Y. Sadqi, Y. Belfaik, S. Safi. *Web OAuth-based SSO Systems Security.*, Proceedings of the 3rd International Conference on Networking, Information Systems & Security. NISS 2020.

[24] A. F. Baig, S. Eskeland. *Security, Privacy, and Usability in Continuous Authentication. A Survey.* Sensors 21, 5967, 2021.

[25] *"Project lab for i-Chip authentication"*. (July 3, 2022). [Online]: https://www.researchgate.net/profile/i-Chip-Authentication

Appendix A
Increasing the iChip entropy

In the sample of 920 students of our university, no secret pattern was repeated. However, the entropy of iChip can be effectively increased by further increasing the key size as shown in subsections A and B, or by using of a suitable background image, which can be built individually by the user or proposed by the wizard as a random structure. The visual structure of such a background image (e.g., Fig. 10) provides reference points for easy remembering the image of the secret and thus allows building a secret key with much higher practical entropy.

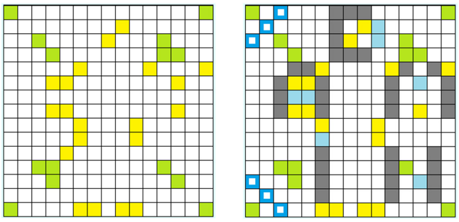
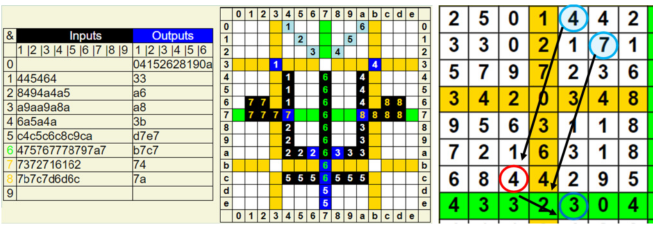


Fig. 10. An example of a secret key depicting the word *admin* on an individually designed background.

A. The iChip as multi-layout interface with 3D key

The iChip interface allows the user to expand the secret not only in 2 dimensions (row x and column y), but also use the 3rd dimension, i.e., the z parameter used here as a layout index (default $z = 1$), marked in colour of the secret elements and is indicated in the i-th round by $z = V_G[i+1]$. In this simple way, the key size can even exceed a thousand bits, without increasing the C size, where $N_{|z|=1} = x \cdot y = |C| \approx 256$.

Fig. 11 shows an example of a 3D secret that has been adapted from the secret key in Fig. 4, where $|z| = 2, N_2 = 450$, by adding 2 blocks on 4 additional layers resulting in $|z| = 6$ and $N_6 = 3 \cdot N_2 = 1350$.



&	Inputs 1 |2 |3 |4 |5 |6 |7 |8 |9	Outputs 1 |2 |3 |4 |5 |6
0		04152628190a
1	445464	33
2	8494a4a5	a6
3	a9aa9a8a	a8
4	6a5a4a	3b
5	c4c5c6c8c9ca	d7e7
6	475767778797a7	b7c7
7	7372716162	74
8	7b7c7d6d6c	7a
9		

2	5	0	1	4	4	2
3	3	0	2	1	7	1
5	7	9	7	2	3	6
3	4	2	0	3	4	8
9	5	6	3	1	1	8
7	2	1	6	3	1	8
6	8	4	4	2	9	5
4	3	3	2	3	0	4

Fig. 11. An example of a 3D secret on multi-layout iChip.

The zoomed fragment of the challenge image in Fig. 11 shows the first round ($i = 1$) of the OTP calculation: Since $z = V_G[i+1] = 7$, the search for a value of $V_G[1] = 4$ must start in block 7. It appears in the last element of this block $B_7^I[5]$ at position (6, 2), so we read the value of 3 at the associated output in cell (7, 4) and calculate: OTP$[2] = (V_G[1] + C[(7,4)]) \mod 10 =$ $= (4+3) \mod 10 = 7$.

B. The iChip as multi-protocol platform

A solution with a large number of protocol variants that can be combined with each other using a simple settings manager is beneficial for increasing its resistance, and makes the scheme more user-friendly, who only needs to know the variants he choose to create his secret. For example, choosing 5 out of 50 variants, the cracker has to check 2,118,760 additional combinations, which must first be analyzed and coded in such a cracker. This number can still expand as each subtle change in protocol represents a new variant that can be created not only by the scientist, but also by the creative user.

The iChip enables the implementation of other Human-Computable Password Protocols, and acts as an open platform for them. We invite other researchers to use it to compose their own licensed variant of HCPP or an adaptation of a previously developed one.

As an example implementation, we used the Foxtail scheme proposed in 2005 by Li and Shum in [3], adapted in 2020 for the needs of IoT in [6]. There are 4 pass-objects in the challenge given in Fig. 12, hence the response R = 4 mod 2 = 0. For a standard 6-digit OTP, this procedure must be repeated for 20 rounds, each for 1 bit.

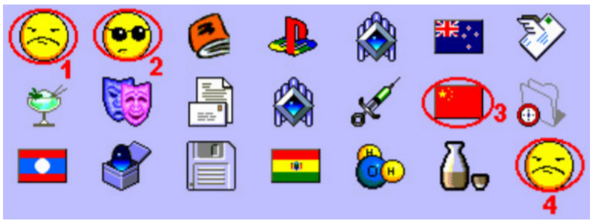
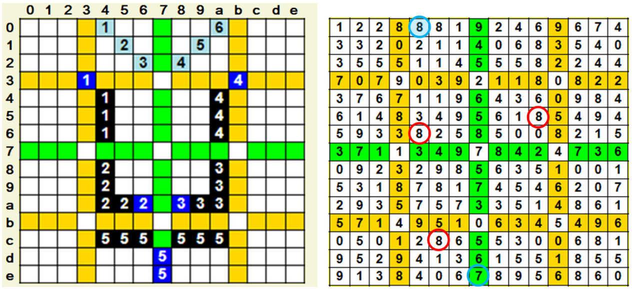


Fig. 12. An example of an original Foxtail challenge for 1 bit response marked by the four red rings.

For implementing the Foxtail protocol on the iChip platform, selected input elements are used as a hidden chain of challenge window in the Foxtail schema. To redirect the binary response of Foxtail, we use two output elements appropriate for the expected binary response (0 or 1) and any 3rd output as an option for the LWO method. Therefore, instead of four protocol rounds for each OTP digit, only one round is needed. To define a trigger for switching between Foxtail and iChip subprotocols, we assume that if the value searched for V_G[i] is found in the first block, then all input elements are treated as a challenge for the Foxtail scheme. Otherwise, for the iChip rules. As counted pass-objects, we assume the value of V_{inp}^{i}. After adapting the example in Fig. 4 for the Foxtail implementation illustraded in Fig. 13: The first element V_G[1] at position (0, 4) has a value of 8, which appears in the first block; therefore, all input elements in S are treated as a Foxtail challenge. As there are 3 entries with the value of 8, in cells [(6, 4), (5, a), (c, 5)], the response is 3 mod 2 = 1.

However, according to the redirection rules, we use this response for binary addressing of the 2nd element from 2 locations: (d, 7) for 0 and (e, 7) for 1. In cell (e, 7) is a value of 7, therefore, finally:
OTP[1] = (8 + 7) mod 10 = 5. In the 2nd round the $V_G[2] = 1$, which appear at 2 locations only: [(6, 4), (a, a)]. Since,
2 mod 10 = 0, we go to the cell (d, 7) with a value of 6,
and calculate OTP[2] = (1 + 6) mod 10 = 7. The next values in the generator block no longer appear in the first block, so iChip rules apply to them.



	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e
0					1						6				
1						2				5					
2							3		4						
3				1								4			
4					1						4				
5					1						4				
6					1						4				
7															
8					2						3				
9					2						3				
a					2	2	2		3	3	3				
b															
c					5	5	5		5	5	5				
d								5							
e								5							

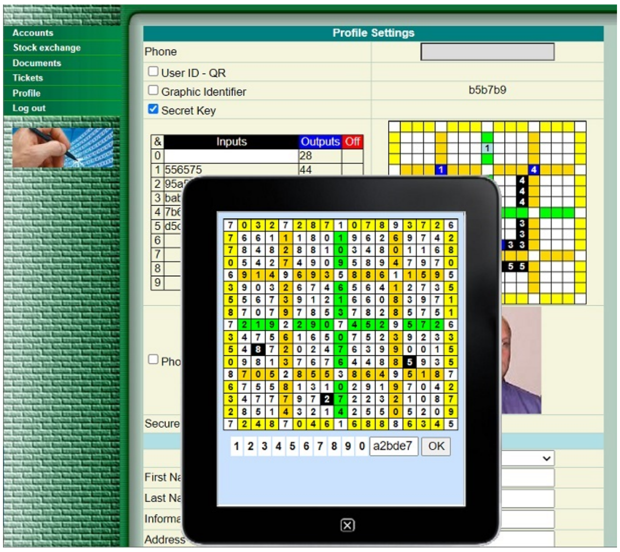
1	2	2	8	8	8	1	9	2	4	6	9	6	7	4
3	3	2	0	2	1	1	4	0	6	8	3	5	4	0
3	5	5	5	1	1	4	5	5	5	8	2	2	4	4
7	0	7	9	0	3	9	2	1	1	8	0	8	2	2
3	7	6	7	1	1	9	6	4	3	6	0	9	8	4
6	1	4	8	3	4	9	5	6	1	8	5	4	9	4
5	9	3	3	8	2	5	8	5	0	0	8	2	1	5
3	7	1	1	3	4	9	7	8	4	2	4	7	3	6
0	9	2	3	2	9	8	5	6	3	5	1	0	0	1
5	3	1	8	7	8	1	7	4	5	4	6	2	0	7
2	9	3	5	7	5	7	3	3	5	1	4	8	6	1
5	7	1	4	9	5	1	0	6	3	4	5	4	9	6
0	5	0	1	2	8	6	5	5	3	0	0	6	8	1
9	5	2	9	4	1	3	6	1	5	5	1	8	5	5
9	1	3	8	4	0	6	7	8	9	5	6	8	6	0

Fig. 13. Image of secret key and Foxtail's pass-objects marked by red rings in the challenge.

Appendix B
The iChip implementation in the e-banking system

We have implemented the iChip protocol in the e-banking system, in which it is used both for logging in and for authorizing transactions. The screenshot in Fig. 14, shows a form for entering the personal data and personalizing the virtual token, visualized as an iPad tablet. The form in the background shows a special wizard for designing of a graphic identifier and secret key, i.e. its image on the right side and the associated code table on the left side. This token implements the iChip protocol and the Turbo-Chip overlays. The token window opens on the form to be authorized. The response for the Click-Chip challenge is shown on the token screen.

The e-banking system is available online [25].



Phone

☐ User ID - QR

☐ Graphic Identifier b5b7b9

☑ Secret Key

&	Inputs	Outputs	Off
0		28	
1	556575	44	
2	95a		
3	bat		
4	7b6		
5	d5c		
6			
7			
8			
9			

7	0	3	2	7	2	8	7	1	0	7	8	9	3	7	2	6
7	6	6	1	1	1	8	0	1	9	6	2	6	9	7	4	2
7	8	4	8	2	8	8	1	0	3	4	8	0	1	1	6	8
0	5	4	2	7	4	9	0	9	5	8	9	4	7	9	7	0
6	9	1	4	9	6	9	3	5	8	8	6	1	1	5	9	5
3	9	0	3	2	6	7	4	6	5	6	4	1	2	7	3	5
5	5	6	7	3	9	1	2	1	6	6	0	8	3	9	7	1
8	7	0	7	9	7	8	5	3	7	8	2	8	5	7	5	1
7	2	1	9	2	2	9	0	7	4	5	2	9	5	7	2	6
3	4	7	5	6	1	6	5	0	7	5	2	3	9	2	3	3
5	4	8	7	2	0	2	4	7	6	3	9	9	0	0	1	5
0	9	8	1	3	7	6	7	6	4	4	8	8	5	9	3	5
8	7	0	5	2	8	5	5	3	8	6	4	9	5	1	8	7
6	7	5	5	8	1	3	1	0	2	9	1	9	7	0	4	2
3	4	7	7	7	9	7	2	7	2	2	3	2	1	0	8	7
2	8	5	1	4	3	2	1	4	2	5	5	0	5	2	0	9
7	2	4	8	7	0	4	6	1	6	8	8	8	6	3	4	5

a2bde7

☐ Pho

Secure

First Na

Last Na

Informa

Address

Fig. 14. Screenshot of authorisation by iChip in the e-banking system