(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 4 327 516 B1**

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
26.02.2025 Bulletin 2025/09

(21) Application number: **22724874.7**

(22) Date of filing: **20.04.2022**

(51) International Patent Classification (IPC):
H04L 9/40 *(2022.01)* ***H04L 9/08*** *(2006.01)*
H04L 9/32 *(2006.01)* ***G06N 20/00*** *(2019.01)*
G06N 3/02 *(2006.01)* ***G06N 3/045*** *(2023.01)*

(52) Cooperative Patent Classification (CPC):
H04L 63/0838; G06N 3/045; H04L 9/3228; H04L 63/20

(86) International application number:
PCT/IB2022/053676

(87) International publication number:
WO 2022/224153 (27.10.2022 Gazette 2022/43)

(54) METHOD OF AUTHENTICATING A CLIENT IN A CLIENT-SERVER ARCHITECTURE

VERFAHREN ZUR AUTHENTIFIZIERUNG EINES CLIENTS IN EINER CLIENT-SERVER-ARCHITEKTUR

PROCÉDÉ D'AUTHENTIFICATION D'UN CLIENT DANS UNE ARCHITECTURE CLIENT-SERVEUR

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR

(30) Priority: **21.04.2021 PT 2021117187**
11.06.2021 PT 2021117284

(43) Date of publication of application:
28.02.2024 Bulletin 2024/09

(73) Proprietor: **Universidade de Coimbra**
3004-531 Coimbra (PT)

(72) Inventors:
- **SHARMA, Rahul**
  3030-471 Coimbra (PT)
- **MARTINS RIBEIRO, Bernardete**
  3000-606 Coimbra (PT)
- **DOS SANTOS MARTINS, Alexandre Miguel**
  2625-595 Vialonga (PT)
- **BANDEIRA CARDOSO, Fernando Amílcar**
  3030-471 Coimbra (PT)

(74) Representative: **do Nascimento Gomes, Rui**
J. Pereira da Cruz, S.A.
Rua Vítor Cordon, 10-A
1249-103 Lisboa (PT)

(56) References cited:
US-A1- 2007 130 474 **US-A1- 2020 120 086**

- **SHARMA RAHUL ET AL: "Learning non-convex abstract concepts with regulated activation networks", ANNALS OF MATHEMATICS AND ARTIFICIAL INTELLIGENCE, BALTZER, BASEL, CH, vol. 88, no. 11-12, 21 March 2020 (2020-03-21), pages 1207 - 1235, XP037274759, ISSN: 1012-2443, [retrieved on 20200321], DOI: 10.1007/S10472-020-09692-5**
- **WILLIAM MELICHER ET AL: "Fast, Lean, and Accurate:Modeling Password Guessability Using Neural Networks", 6 January 2017 (2017-01-06), pages 186 - 202, XP061025084, Retrieved from the Internet <URL:https://www.usenix.org/sites/default/files/sec16_full_proceedings_interior.pdf> [retrieved on 20170106]**

EP 4 327 516 B1

Description

FIELD OF THE INVENTION

[0001] The present invention is enclosed in the field of authentication methods in a Client-Server Architecture. In particular, the present application relates to policy-based authentication methods for authenticating user(s), application(s), or device(s) and has its significance in the domain of Cybersecurity application-layer authentication.

PRIOR ART

[0002] A problem present in the state of the art is Client Identity verification. Client Identity verification can be done in the following ways: i) password-based authentication; ii) multi-factor authentication; iii) Certificate-based authentication; iv) Token-based authentication. Examples of such methodologies are disclosed in US 2020/120086 A1 and in US 2007/130474 A1, and in references [2] and [3].

[0003] However, any of these methodologies presents problems. Notably, password-based authentication can be subject to phishing attacks, password guessing, etc. The multi-factor authentication is prone to situations when the devices are lost or unavailable to generate authentication codes. In its turn, certificate-based authentication is vulnerable to the theft of private keys. Finally, token-based authentication can be expensive when generating very secure tokens. If the token is leaked due to an unsecured connection, it can create several problems.

[0004] The present solution is intended to overcome such issues innovatively.

SUMMARY OF THE INVENTION

[0005] It is, therefore, an object of the present invention, a method, as defined in the appended claims, for authenticating a Client in a Client-Server Architecture. The method developed can be used by application(s) or device(s) to authenticate user(s), application(s) or device(s). These entities are further referred to as "Client" for the present application.

[0006] This method has three main components: Data, Model, and Policies. The data is made up of synthetic and real passwords of Clients. The Model is generated using the data and a computational modeling technique named Regulated Activation Network - RAN - [1]. Policies are created using the Model.

[0007] This new method for authenticating Clients in a Client-Server architecture has its significance in the domain of cybersecurity application-layer authentication. More particularly, the Client's input passcode is encoded using the RAN Model. Depending upon the Policy shared between the Client and Server, the different combinations of encoded passcodes are selected based on the number of layers and number of nodes in each layer in the Model generated. After every successful operation, a new Policy is generated and shared between the Client and the Server.

[0008] Because of these new policies, the Client inputs the same passcode for every authentication attempt, but the passcode communicated over the network is always different. This method is different from the prior art at least because it uses a Computational Model as part of the authentication procedure and uses Policies generated from the Model and the Client's passcode to generate the Client's encoded identity verification code. The advantages in respect of the prior art include random encoded passcode generation for the client's identity verification. This randomness also ensures that the generated encoded passcodes are never repeated in two successive authentication attempts.

[0009] For that purpose, in an advantageous configuration of the present invention method, it comprises a Model creation process, a Client creation process, and a Client authentication process.

[0010] The Model creation process is responsible for generating a Hierarchical model based on feeding a RAN computational model with a randomly generated N-dimensional input Dataset.

[0011] In its turn, the Client creation process generates a Client's Encoded Passcode Hierarchy by feeding the Hierarchical model with a Client's passcode. The Client's Encoded Passcode Hierarchy comprises a set of encoded passcodes representing encoded versions of the Client's passcode, and it is saved on the Server. The Client creation process is also responsible for generating a Policy for the Client's next authentication attempt and sharing it between the Client and the Server.

[0012] On the Client-side, the Client authentication process generates a Client's Encoded Passcode Hierarchy by feeding the Hierarchical Model with the Client's passcode. It also generates a Client's Encoded passcode using the Policy shared between the Client and the Server. Additionally, it authenticates the Client at the server-side if the Client's encoded passcode matches an expected encoded passcode generated at the server, using the saved Client's Encoded Passcode Hierarchy and the shared Policy. Finally, a new Policy is generated if the Client is successfully authenticated, which is then shared between the Client and the Server for the next Client's authentication attempt.

DESCRIPTION OF FIGURES

[0013]

Figure 1 shows the Model Creation stage of the method of the present invention. As shown to create the model, an input dataset (1) is fetched from a database and fed to the Computational Model (2) named Regulated Activation Network (RAN). RAN generates a Hierarchical Model (3), as shown in Figure 2.

Figure 2 shows an example of a 4-Layered Model (3) generated using RAN's Modelling (2). Layer L.0 is the input layer where 4 nodes depict the dimension of the input data, i.e., the dataset used to build this model has 4 columns. The Layers L.1, L.2, and L.3 are the dynamically generated layers where each node is an abstract representative of the input data.

Figure 3 shows an example of the Client's input Passcode (7) transformation into an Encoded Passcode (6). The Figure shows that all the layers (L.1 to L.3), except the input Layer L.0, form the Client's Encoded Passcode Hierarchy (4), where each layer (L.1 to L.3) stores an encoded version of the Client's Passcode (7). The Policy (5) is a JSON format where the 'Key': [LIST] pair determines the Layer: [Node(s) index(s)] respectively. Based on this Policy, the encoded Passcodes are grouped to form the encoded passcode.

Figure 4 shows a Model (3) of 8 Layers (L.0 to L.7) that was generated using the RAN's modeling (2) to prove the concept of authentication using RAN's model. Layer L.0 has 5 Nodes which depicts that the dataset consists of 5 columns.

Figure 5 refers to an embodiment of the method of the present invention showing the Client registration process. In the procedure, the Client (12) sends to the server (9) the Client's ID (10), a Token (11) provided by the server (9), and the Client's input Passcode (7). The server (9) verifies the credential's provided and then registers the Client (12) and responds with the Model (3) and Policy (5) for the next authentication attempt.

Figure 6 refers to an embodiment of the method of the present invention showing the Client's authentication operations. The Figure shows all the entities required for the authentication of the Client (12), i.e., the Client's input passcode (7), Hierarchical Model (3), Shared Policy (5), Encoded Passcode Hierarchy (4), and the Encoded Passcode (6). The Figure also shows the flow of the authentication process, both from the Client (12) and Server (9).

Figure 7 refers to an embodiment showing the Client creation and Authentication procedure of the decentralized approach of the method of the present invention. For registration, the Client (12) sends (8) to the server (9) the Client's ID (10), a Token (11) provided by the server (9), the encoded passcode hierarchy (4), and the policy (5) for the next authentication after registration. The Server (9) verifies the credential's provided and then registers the client (12) and responds (13) with the registration failure or success message. For Authentication, the client (12) only sends (8) the encoded passcode (6) and the policy (5) for the next authentication, and the server responds (13) to the client with a failure or success message.

DETAILED DESCRIPTION

[0014] The more general and advantageous configurations of the present invention are described in the Summary of the invention. Such configurations are detailed below in accordance with other advantageous and/or preferred embodiments of implementation of the present invention.

[0015] The authentication method of the present invention comprises a Model creation process, a Client creation process, and a Client Authentication process. Additionally, the method can be implemented in a centralized manner such that the Model creation process, the generation of the Client's Encoded Passcode Hierarchy (4) on the Client creation process, and the generation of the Policy (5), (5.1) on both the Client creation process and Client authentication process are executed at the server's side. The Client (12) stores the Hierarchical Model (3) and the Policy (5), (5.1).

[0016] Alternatively, the method can be implemented in a decentralized manner wherein the Model creation process, the generation of the Client's Encoded Passcode Hierarchy (4), and the generation of the Policy (5), (5.1) on both the Client creation process and Client creation authentication process are executed at the Client's side. The Server (9) stores the Client's Encoded Passcode Hierarchy (4), the Policy (5), (5.1), and the respective Client's ID (10). Along with the strengths related to the randomness in encoded passcode generation, this decentralized manner empowers the Client (12) machine to have full control over the creation and store of the main authentication-related credentials (i.e., Model, Passcode, and Policies.) and limits the role of servers (9) in saving the Clients (12) related credentials. In fact, the Model (3) generation and its update are independent of the Server (9), i.e., the Client (9) only notifies the Server (9) with the passcode hierarchy (4) if a new Model (3) is generated or a new Client (12) is created. Therefore, an attack on the server (9) does not compromise the Client's and authentication model related credentials.

Model creation process

[0017] In the centralized approach, the creation of Model (3) is executed on the server-side by the server administrator, selecting the amount of data to build the model. Additionally, the server administrator also sets the hyperparameters of the RAN's model (2) that further builds the Model (3). When Model (3) is built, it is made available to all the Clients (12). Alternatively, in the case of a decentralized approach, the creation of the Model (3) is a Client (12) operation, where the client-side application randomly generates a numerical dataset considering the client's passcode minimum and maximum values. The Client's application also arbitrarily selects the hyperparameters of the RAN's model, further building the model. When the model is built, it is saved on the client's device for further usage.

[0018] When an existing Client (12) logs in to the Server (9), it is authenticated by the old model (3) first, then its stored input passcode (7) is used to generate a new encoded passcode (6) set using the newly generated model (3). Therefore, new Models (3) can be arbitrarily generated, and the Client's model (3) will be updated automatically. These intermittently changed Models (3) drastically add to the variability induced by the random policy generation during the policy generation. This has been verified by regularly changing the Model (3) while testing the authentication and Client creation operations. In Figure 1, we can see three entities and their flow, which are described as follows:

a. First, it is the dataset (1). The dataset (1) is synthetically created by the server administrator, in the case of a centralized approach, having a predefined dimension, i.e., the features (or columns, or attributes). In the case of a decentralized approach, the dataset (1) is synthetically created by the Client's application automatically. The dimension of the dataset, i.e., the features (or columns, or attributes), are decided by the Client's input passcode (7). Each data unit is an integer value between a Minimum and Maximum Integer. The number of instances, i.e., the size of the data points, is also decided by the server administrator or the Client's application, and in one embodiment of the method of the present invention, it is in the range of 1 to 2000. In the case of a centralized approach, in the process of creating a new Client (12), it's passcode (7) may be verified in light of the specifications of the dataset (1) that the server administrator defined. If there is a match, the Client (12) is registered, and its passcode (7) is appended to the dataset (1). Alternatively, in the decentralized approach, when creating a new Client (12), their passcode (7) is merged with the generated data (1) and passed for model generation (2). For this proof of concept, a dataset of 1000 instances was generated, the dataset has 5 dimensions, and the data value of each unit is an integer between 1 and 1000.

b. Second is the Computational Model (2). The computational model used is the RAN [1]. This modeling technique can generate a hierarchical model from the dataset (1), as shown in Figures 2 and 4. For the proof of concept, it is configured the RAN's modeling (2) to use K-means as concept identifier. The 'k' of the K-means was set to 15, 10, 4, 2, 7, 3, & 5 to determine the number of nodes for Layers L.1, L.2, L.3, L.4, L.5, L.6, L.7, and L.8 respectively. This configuration can be changed by the server administrator or by the Client application arbitrarily, in the case of a centralized or decentralized approach, respectively.

c. Third is the Model (3). This multi-layered model is generated using the RAN's modeling technique (2). In an embodiment, the generated RAN's model (2) has 8 Layers, wherein L.0 is the Input layer, and the other 7 Layers (L.1 to L.7) are the Abstract Concept Layers, as shown in Figure 4.

Client creation process

[0019] In a centralized approach, the Client (12) sends a Client creation request with the Client's ID (12), a passcode (7), and a token (11). Upon successful token (11) validation, the Client's passcode (7) is used as input to the RAN's model (2) to generate the encoded passcode (6). The encoded passcodes contained in the Client's encoded passcode hierarchy (4) are stored for the Client's authentication, and the Client's passcode (7) is stored to create newly encoded passcodes when the models (3) are changed. Figure 5 illustrates the Client creation process. This process may be described as follows:

a. Ensure that the RAN's model (2) is already generated and stored at the Server (9);

b. Through REST API, the Client (12) provides a passcode (7), a token (11), i.e., an arbitrarily large integer provided by the server administrator, and the client's ID (10) to the Server;

c. The Server (9) verifies the Client's passcode (7) and token (11) if it is according to the specifications of the dataset (1) created by the server administrator and the expected token at the Server (9). If any of the two fails to verify, then a FAILURE message is reverted to the Client (12);

d. If both are verified, the Client's passcode (7) is normalized between 0 and 1 and passed to the RAN's model (2) to generate an Encoded Passcode Hierarchy (4) for the Client's passcode (7) and save it for the Client's information at the Server (9);

e. The Server (9) then generates a Policy (5) of arbitrary size using the RAN's model (2) architecture specifications. The Policy (5) size varies between a minimum and maximum integer and is decided by the server administrator. In an embodiment, the range of Policy (5) was set between 5 and 60;

f. The generated Policy is saved with the Client's information as an expected Policy (5) for the next authentication attempt by the Client (12).
g. The copy of Policy (5) and the Generated Model (3) is sent back to the Client (12) through the REST interface.
h. The Model (3) and the Policy (5) are saved in the Client's system.
i. The process terminates.

[0020] Alternatively, in the case of a decentralized approach, the Client (12) sends a Client creation request with the Client's ID (10), a token (11), Encoded Passcode Hierarchy (4), and Policy (5). Upon successful token (11) validation, the Client (12) is acknowledged with a success message. Figure 7 describes the Client creation process. This process may be described as follows:

a. Ensure that the RAN's model (2) is already generated and stored at the Client's device;
b. Through REST API, the Client (12) provides a token (11), i.e., an arbitrarily large integer provided by the server administrator, Client's ID (10), and Encoded Passcode Hierarchy (4), and Policy (5) to the Server (9).
c. The Server verifies the token: If token verification is successful, then a new Client (12) is created with the unique ID (10), provided by the Client (12) along with the Client's Encoded Passcode Hierarchy (4), and Policy (5) for the next authentication. All these credentials are saved on the Server (9). A response is sent to the Client (12) stating the successful creation of the Client (12). If the token is not verified, then a 'Process Unsuccessful' response is sent to the Client (12).

Client authentication process

[0021] This process verifies the Client's identity. In this authentication process, three things are essential: first, the Client's input passcode (7); second, the Model (3) shared between Client (12) and Server (9); and third, the Policy (5) shared between Client (12) and Server (9). When a Client (12) attempts to authenticate, it's passcode (7) is passed through the Model (3) to produce the Client's Encoded Passcode Hierarchy (4). Further, an Encoded Passcode (6) is created using the Policy (5) and the Client's Encoded Passcode Hierarchy (6). This Encoded Passcode (6) is sent to the Server (9). In the case of a decentralized approach, a new Policy (5.1) for the next authentication is also sent to the Server (9).

[0022] To verify the Client (12), the Server (9) loads the Client's Encoded Passcode Hierarchy (4) saved on the Server (9) and the Policy (5) that was shared between the Client (12) and the Server (9). If both the encoded passcodes match, the Client (12) is acknowledged with an authentication success message. In the case of a centralized approach, a new Policy (5.1) is also sent to the Client (12) to be used in the next authentication approach. In the case of a decentralized approach, the current Policy (5) at the server (9) is replaced by the new Policy (5.1) sent by the Client (12). If both the encoded passcodes do not match, the Client (12) is acknowledged with an authentication failure message.

[0023] The method developed ensures that in two consecutive authentication attempts by a Client (12), the Encoded Passcode (6) is never repeated, which is confirmed by keeping a log of Policies (5) used in the Authentication operations. Table 1 lists the Iteration Log obtained that proves it. Validation was performed from 100 authentication operations until 1 million authentication attempts in different experiment sets. It can be seen that until 10,000 authentication operations, no two consecutive Policies (5) were repeated. When the authentication iteration grew to 50,000 attempts, it can be observed that Policies (5) have repeated, but two Policies (5) are never observed one after the other in two consecutive authentication attempts.

Table 1

Authentication Iteration	Index of Consecutive Repeated Policy	Count of Consecutive Repeated Policy	Unique Policy Count	Repeated Policy Count
100	[]	0	100	0
500	[]	0	500	0
1,000	[]	0	1,000	0
5,000	[]	0	5,000	0
10,000	[]	0	10,000	0
50,000	[]	0	49,995	5
3,13,353	[]	0	3,13,295	58
5,73,583	[]	0	5,73,383	200
8,33,245	[]	0	832,851	394

(continued)

Authentication Iteration	Index of Consecutive Repeated Policy	Count of Consecutive Repeated Policy	Unique Policy Count	Repeated Policy Count
10,00,000	[]	0	9,99467	533

[0024] Figure 6 shows the entire Client authentication process in case of a centralized approach. The process may be as follows:

a. Ensure that the RAN's Modelling (2) Library is available at the Client's device;
b. The new RAN's model instance is created and loaded with the RAN's model (3) provided by the Server (9);
c. The Client (12) enters its passcode (7), and it is normalized between 0 and 1 and passed to the RAN's model (3) to generate the Encoded Passcode Hierarchy (4) of the Client's passcode (7).
d. The shared Policy (5) provided by the Server (9) is applied to the Encoded Passcode Hierarchy (4) of the Client's passcode (7), and an Encoded Passcode (6) is generated for the Client's verification at the Server (9).
e. The Encoded Passcode (6) is sent to the server (9) via REST interface along with the Client's ID (10);
f. Upon receiving an authentication request, the Server (9) checks if the Client's ID (10) exists; if not, then a failure message is reverted to the Client (12);
g. If the Client's ID (10) exists, the Client's saved Encoded Passcode Hierarchy (4) is loaded along with the expected Policy (5). This Policy (5) is applied to the Client's saved Encoded Passcode Hierarchy (4) to generate an encoded passcode. This encoded passcode is compared with the Encoded Passcode (6) sent by the Client (12). If the passcodes do not match, the Client (12) is reverted with an authentication failure message;
h. If the passcodes match, the Client (12) is verified;
i. A new Policy (5.1) is generated by the Server (9), and one copy of the Policy (5.1) is saved with the Client's information at the Server (9) as an expected Policy (5) for the Client's next authentication attempt;
j. The other copy of the Policy (5.1) is sent to the Client along with a successful authentication message via REST interface;
k. At the Client (12), if the response is a successful authentication message, then save the new shared Policy (5.1) sent by the Server (9) for the next login attempt.

[0025] Figure 7 also shows the entire authentication process in the case of a decentralized approach. The Process may be as follows:

a. Ensure that the RAN's Modelling (2) Library is available at the Client's device;
b. The new RAN's model instance is created and loaded with the RAN's model (3) saved on the Client's device;
c. The Client (12) enters its passcode (7), and it is normalized between 0 and 1 and passed to the RAN's model (3) to generate the Encoded Passcode Hierarchy (4) of the Client's passcode (7);
d. The shared Policy (5) is applied to the Encoded Passcode Hierarchy (4) of the Client's passcode (7), and an Encoded Passcode (6) is generated for the Client's verification at the Server (9);
e. A new Policy (5.1) is created;
f. The Encoded Passcode (6) is sent to the Server via REST interface along with the Client's ID (10) and the new Policy (5.1);
g. Upon receiving an authentication request, the Server (9) checks if the Client's ID (10) exists; if not, then a failure message is reverted to the Client (12);
h. If the Client's ID (10) exists, the Client's saved Encoded Passcode Hierarchy (4) is loaded along with the expected Policy (5). This Policy (5) is applied to the Client's saved Encoded Passcode Hierarchy (4) to generate an encoded passcode. This encoded passcode is compared with the Encoded Passcode sent by the Client (12). If the passcodes do not match, the Client (12) is reverted with an authentication failure message;
i. If the passcodes match, the Client (12) is verified;
j. The New Policy (5.1) sent by the Client (12) is saved at the Server (9) as an expected Policy (5) for the Client's next authentication attempt.

[0026] The robustness of the developed authentication method, with respect to randomness, is achieved through the Policies (5). The Policy (5) itself is derived from the architecture of the Model (3) generated using RAN (2). A Policy (5) may be made using the following steps:

a. The size of the Policy (5) is decided by the server administrator, in the case of a centralized approach, by choosing a

number in a range of two integer numbers. Alternatively, if a decentralized approach is considered, the size of the Policy (5) is arbitrarily decided by a Client's application. The proof of concept used the range of Policy sizes between 5 and 60.

b. The randomly chosen size of the Policy (5) determines the number of elements that will be in the policy;

c. Each element of the Policy (5) is a pair of LayerID and List of Node(s) in the Layer. The Policy elements may be made as follows:

i. A Layer (L.1 to L.β-1) of the Model (3) is randomly chosen. Note, the range of layers is always between L.1 to L.β-1, where β is the number of layers in the Model (3). Layer 0 is never used in the Policy (5) because it can reveal the Client's passcode (7);

ii. Once the layer (L.1 to L.β-1) is chosen, then a list of one or more nodes is created by arbitrarily picking one or more nodes from the chosen layer (L.1 to L.β-1);

d. The generated Policy (5) looks like a JSON script, as shown in Figure 3, where the JSON Keys are the Layer-ID of the Model and JSON values are the node ID(s) of the layer.

[0027] The Client's Encoded Passcode Hierarchy (4) can be generated by propagating the Client's input passcode (7) to the generated RAN's model (3). This process encodes the Client's input passcode (7) into varied sizes based on the size of the layers of the Model (3). For example, in Figure 3 the input passcode (7) is [30, 270, 150, 210]. At the input layer, L.0, the passcode (7) is normalized between 0 and 1 (by dividing them by 300), but the size of the passcode (7) remains the same. In Layer L.1 dimension of input is reduced to 2 with encoded values [0.01, 0.68]. At Layer L.2 the dimension was expanded to 7 with encoded values [0.15, 0.23, 0.32, 0.1, 0.05, 0.08, 0.6] and the last layer L.3 the dimension is again reduced to size 3 with encoded values [0.001, 0.8, 0.17]. These encoded input passcode at different layers are seen as a hierarchy of the Client's input passcode (4). Note, the Encoded Passcode Hierarchy (4) of the Client (12) does not include the input Layer L.0 because it can reveal the Client's input passcode (7). Therefore, the Encoded Passcode Hierarchy (4) of the Client (12) consists of encoded passcodes from Layer L.1 onwards until the highest layer (L.β-1) in the Model's architecture. At the Server (9), this Encoded Passcode Hierarchy (4) exists in a persistent form, i.e., it is saved with the Client's profile in place of the Client's Passcode. This hierarchy (4) is generated on the Client's side whenever the Client (12) attempts to perform the authentication. In the case of a decentralized approach, the Encoded Passcode Hierarchy (4) is controlled by the Client (12), i.e., whenever the Client (12) wants, can replace the current Encoded Passcode Hierarchy (4) with a new one.

[0028] The Encoded Passcode (6) is the actual passcode that is sent by the Client (12) via the internet (for example) to the Server (9) for authentication. The Encoded Passcode generation is depicted in Figure 3. This Encoded passcode (6) is generated twice in one authentication attempt:

a. First, at the Client-side, where the Client (12) enters its passcode to the RAN's model (3) to generate its Encoded Passcode Hierarchy (4), then using the shared Policy (5), the Encoded Password (6) is generated. In Figure 3, the shared Policy (5) selects the encoded passcode at node 1 from Layer L.1, nodes 2, 3, & 7 at Layer L.2, and node 3 & 1 from Layer L.3. These selected encoded passcodes are grouped to form the Encoded Passcode (6), which is sent to the Server (9) for verification along with the Client's ID;

b. Second, on the Server side, upon receiving a legitimate authentication request, the server loads the Client's Encoded Passcode Hierarchy (4) saved on the Server (9). Then the Server (9) loads the expected Policy (5) and uses it to determine the expected Encoded Passcode (6) from the Client (12).

[0029] As will be clear to one skilled in the art, the present invention should not be limited to the embodiments described herein, and a number of changes are possible which remain within the scope of the present invention.

[0030] Of course, the preferred embodiments shown above are combinable, in the different possible forms, being herein avoided the repetition of all such combinations.

REFERENCES

[0031]

[1] Sharma R, Ribeiro B, Miguel Pinto A, Cardoso FA. Exploring Geometric Feature Hyper-Space in Data to Learn Representations of Abstract Concepts. Applied Sciences. 2020; 10(6):1994. https://doi.org/10.3390/app10061994.

[2] Sharma R, et al: "Learning non-convex abstract concepts with regulated activation networks", Annals of Mathematics and Artificial Intelligence, Baltzer, Basel, CH, vol.88, no. 11-12, 21 March 2020 (2020-03-21), pages 1207-1235, XP037274759, ISSN: 1012-2443, DOI:10.1007/510472-020-09692-5.

[3] William Melicher et al: "Fast, Lean, and Accurate: Modeling Password Guessability using Neural Networks",

Usenix, The Advanced Computing Systems Association, 6 January 2017, pages 186-202, XP061025084.

Claims

1. Method for authenticating a Client in a Client-Server architecture, the method comprising a Model creation process, a Client creation process, and a Client authentication process; wherein,

   the Model creation process generates:

   (i) at the Client-side or at the Server-side, a Hierarchical model (3), shared between the Client and the Server, based on feeding a Regulated Activation Network computational model (2) with a randomly generated N-dimensional input Dataset (1), said Hierarchical model (3) comprised by a plurality of layers;

   the Client creation process generates:

   (i) at the Client-side or at the Server-side, a first Client's Encoded Passcode Hierarchy (4) by feeding the Hierarchical model (3) with a Client's passcode (7); the Client's Encoded Passcode Hierarchy (4) being saved on the Server (9) and comprises a set of encoded passcodes representing encoded versions of the Client's passcode (7); the Client's Encoded Passcode Hierarchy (4) being formed by layers of the Hierarchical model (3), wherein each of said layers, except input layer, stores an encoded version from said encoded versions of the Client's passcode (7);
   (ii) at the Client-side or at the Server-side, a Policy (5) for the Client's next authentication attempt, which is shared between the Client (12) and the Server (9); and

   the Client authentication process:

   (i) generates, at the Client-side, a second Client's Encoded Passcode Hierarchy (4) by feeding the Hierarchical Model (3) with the Client's passcode (7) and a Client's Encoded passcode (6) is created using the Client's Encoded Passcode Hierarchy (4) and the Policy (5) shared between the Client (12) and the Server (9);
   (ii) authenticates the Client (12) at the server-side if the Client's encoded passcode (6), received from the Client, matches an expected encoded passcode generated at the server (9) using the saved first Client's Encoded Passcode Hierarchy (4) and the shared Policy (5);
   (iii) generates, at the Client-side or at the Server-side, a new Policy (5.1) if the Client (12) is successfully authenticated, which is shared between the Client (12) and the Server (9) for the next Client's authentication attempt.

2. Method according to claim 1 wherein the input Dataset (1) is formed by data units of an integer type; each data unit is an integer value between a minimum and maximum integer; preferably, each data unit is an integer between 1 and 2000.

3. Method according to claims 1 or 2 wherein the Regulated Activation Network computational model (2) is configured to implement a k-means algorithm as a concept identifier.

4. Method according to any of the previous claims, wherein the Hierarchical Model (3) comprises a plurality of β layers, each layer comprising at least one node, determining the size of the layer, and wherein layer L.0 is the input layer and comprises N nodes, being N the dimension of the input Dataset (1); and layers L.1 to L.β -1 are dynamically generated layers, each of said layers storing an encoded passcode.

5. Method according to claim 4, wherein in the Client creation process and the Client authentication process, the Client's passcode (7) is inputted in layer L.0 of the Hierarchical Model (3) and the layers L.1 to L.β -1 are dynamically generated layers, each of said layers storing an encoded passcode representing an encoded version of the Client's passcode (7).

6. Method according to claim 5, wherein the Client's Encoded Passcode Hierarchy (4) is formed by layers L.1 to L.β-1 of the Hierarchical Model (3).

7. Method according to any of the previous claims wherein the process of generating a Policy (5), (5.1) is based on the Hierarchical Model (3) and comprises the following steps:

- selecting a number of elements for the Policy; preferably said number being in a range of two integer numbers, even more preferably, said number being between 5 and 60;
- creating the selected number of Policy elements, each element being a pair of layer ID and a List of node(s) in said layer and being formed by:

    - randomly selecting a layer of the Hierarchical Model (3); the selection being made between layer L.1 and layer L.β-1;
    - creating a list by arbitrarily picking one or more nodes from the selecting layer.

8. Method according to claim 7, wherein the Policy (5), (5.1) is a JSON script where the JSON keys are the layer-ID of the Hierarchical Model (3) and the JSON values are the node ID(s) of the layer.

9. Method according to any of the previous claims 4 to 8, wherein in the Client creation process and the Client authentication process, the Client's Encoded Passcode Hierarchy (4) is generated by encoding the Client's passcode (7) into varied sizes based upon the sizes of each layer (L.1 to L.β-1) of the Hierarchical Model (3).

10. Method according to any of the previous claims, wherein in the Client authentication process, the generation of the Client's Encoded Passcode (6) comprises the following steps, at the Client side:

(i) entering the Client's passcode (7) to the Hierarchical Model (3) to generate the Client's Encoded Passcode Hierarchy (4);
(ii) applying the Policy (5) to the Client's Encoded Passcode Hierarchy (4) to generate the Client's Encoded Passcode (6), being formed by grouping together the respective encoded passcodes.

11. Method according to any of the previous claims, wherein in the Client authentication process, the generation of the expected encoded passcode (6) at the server-side, comprises the following steps:

(i) upon receiving an authentication request by the Client (12), the server loads the Client's Encoded Passcode Hierarchy (4), and the Policy (5) shared with the Client (12);
(ii) applying the Policy (5) to the Client's Encoded Passcode Hierarchy (4) to generate the expected encoded passcode (6), being formed by grouping together the respective encoded passcodes.

12. Method according to any of the previous claims 1 to 11, wherein the Model creation process, the generation of the Client's Encoded Passcode Hierarchy (4) on the Client creation process, and the generation of the Policy (5), (5.1) on both the Client creation process and Client authentication process are executed at the server's side; and
the Client (12) stores the Hierarchical Model (3) and the Policy (5), (5.1).

13. Method according to claim 12, wherein the Client creation process comprises the following steps:

- a Client (12) sends a Client creation request to a server administrator; the request comprising a Client's ID (10), a Client's passcode (7), and a token (11);
- upon successful token (11) validation by the server administrator, the Client's passcode (7) is used as input to the Regulated Activation Network computational model (2) to generate a Hierarchical model (3) and a respective Client's Encoded Passcode Hierarchy (4) for said Client's passcode (7);
- the Client's passcode (7) and the Client's Encoded Passcode Hierarchy (4) are saved at the server (9) as the Client's information;
- the server (9) generates a Policy (5) based on the Hierarchical model (3), and the generated policy (5) is saved at the server (9) with the Client's information as an expected Policy (5) for the next authentication attempt by the Client (12);
- a copy of the Policy (5) and the Hierarchical model (3) is sent to the Client (12) to be saved in the Client's system.

14. Method according to any of the previous claims 1 to 11, wherein the Model creation process, the generation of the Client's Encoded Passcode Hierarchy (4), and the generation of the Policy (5), (5.1) on both the Client creation process and Client creation authentication process are executed at the Client's side; and
the Server (9) stores the Client's Encoded Passcode Hierarchy (4) and the Policy (5), (5.1).

15. Method according to claim 14 wherein the Client's creation process comprises the following steps:

- the Client (12) sends a Client creation request to a server administrator;
the request comprising a Client's ID (10), a Client's Encoded Passcode Hierarchy (4), a Policy (5), and a token (11) that is provided by the server administrator;
- upon successful token validation by the server administrator, the Client (12) is created with a unique ID;
- the server (9) stores the Client's ID (10), the Client's Encoded Passcode Hierarchy (4), and the Policy (5) for the next authentication attempt by the Client (12).

Patentansprüche

1. Verfahren zur Authentifizierung eines Kündens in einer Kunden-Server-Architektur, wobei das Verfahren einen Modell-Erstellungsprozess, einen Kunden-Erstellungsprozess und einen Kunden-Authentifizierungsprozess umfasst; wobei,

   der Modell-Erstellungsprozess erzeugt:

   (i) auf der Kunden-Seite oder auf der Server-Seite ein hierarchisches Modell (3), das zwischen dem Kunden und dem Server geteilt wird, basierend auf der Speisung eines regulierten Aktivierungsnetzwerk-Rechenmodells (2) mit einem zufällig erzeugten N-dimensionalen Eingabedatensatz (1), wobei das hierarchische Modell (3) aus einer Vielzahl von Schichten besteht;

   der Kunden-Erstellungsprozess erzeugt:

   (i) auf der Kunden-Seite oder auf der Server-Seite eine erste kodierte Kunden-Passcode-Hierarchie (4), indem das hierarchische Modell (3) mit einem Kunden-Passcodes (7) gespeist wird; wobei die kodierte Kunden-Passcode-Hierarchie (4) auf dem Server (9) gespeichert wird und einen Satz von kodierten Passcodes umfasst, die kodierte Versionen des Kunden-Passcodes (7) darstellen; wobei die kodierte Kunden-Passcode-Hierarchie (4) durch Schichten des hierarchischen Modells (3) gebildet wird, wobei jede der Schichten, außer der Eingabeschicht, eine kodierte Version aus der kodierten Versionen des Kunden-Passcodes (7) speichert;
   (ii) auf der Kunden-Seite oder auf der Server-Seite eine Richtlinie (5) für den nächsten Authentifizierungsversuch des Kündens, die zwischen dem Kunden (12) und dem Server (9) geteilt wird;
   und

   den Kunden-Authentifizierungsprozess:

   (i) auf der Kunden-Seite eine zweite kodierte Kunden-Passcode-Hierarchie (4) erzeugt, indem das hierarchische Modell (3) mit dem Kunden-Passcodes (7) gespeist wird, und ein kodierter Kunden-Passcodes (6) unter Verwendung der kodierten Kunden-Passcode-Hierarchie (4) und der zwischen dem Kunden (12) und dem Server (9) geteilten Richtlinie (5) erstellt wird;
   (ii) den Kunden (12) auf der Server-Seite authentifiziert, wenn der vom Kunden empfangene kodierter Kunden-Passcodes (6) mit einem erwarteten kodierter Passcodes übereinstimmt, der auf dem Server (9) unter Verwendung der gespeicherten ersten kodierter Kunden-Passcode-Hierarchie (4) und der geteilten Richtlinie (5) erzeugt wurde;
   (iii) erzeugt auf der Kunden-Seite oder auf der Server-Seite eine neue Richtlinie (5.1), wenn der Kunden (12) erfolgreich authentifiziert wurde, die zwischen dem Kunden (12) und dem Server (9) für den nächsten Authentifizierungsversuch des Kündens geteilt wird.

2. Verfahren nach Anspruch 1, wobei der Eingabedatensatz (1) aus Dateneinheiten eines ganzzahligen Typs gebildet wird; jede Dateneinheit ist ein ganzzahliger Wert zwischen einer minimalen und maximalen ganzen Zahl; vorzugsweise ist jede Dateneinheit eine ganze Zahl zwischen 1 und 2000.

3. Verfahren nach Anspruch 1 oder 2, wobei das regulierten Aktivierungsnetzwerk-Rechenmodells (2) so konfiguriert ist, dass es einen k-means-Algorithmus als Konzeptidentifikator implementiert.

4. Verfahren nach einem der vorhergehenden Ansprüche, wobei das hierarchische Modell (3) eine Vielzahl von β Schichten umfasst, wobei jede Schicht mindestens einen Knoten umfasst, der die Größe der Schicht bestimmt, und wobei die Schicht L.0 die Eingabeschicht ist und N Knoten umfasst, wobei N die Dimension des Eingabedatensatzes

(1) ist; und die Schichten L.1 bis L.β-1 dynamisch erzeugte Schichten sind, wobei jede der Schichten einen kodierten Passcodes speichert.

5. Verfahren nach Anspruch 4, wobei im Kunden-Erstellungsprozess und im Kunden-Authentifizierungsprozess der Kunden-Passcodes (7) in die Schicht L.0 des hierarchischen Modells (3) eingegeben wird und die Schichten L.1 bis L.β-1 dynamisch erzeugte Schichten sind, wobei jede der Schichten einen codierten Passcodes speichert, der eine codierte Version des Kunden-Passcodes (7) darstellt.

6. Verfahren nach Anspruch 5, wobei die kodierte Kunden-Passcode-Hierarchie (4) durch die Schichten L.1 bis L.β-1 des hierarchischen Modells (3) gebildet wird.

7. Verfahren nach einem der vorhergehenden Ansprüche, wobei der Prozess der Erzeugung einer Richtlinie (5), (5.1) auf dem hierarchischen Modell (3) basiert und die folgenden Schritte umfasst:

- Auswahl einer Anzahl von Elementen für die Richtlinie; vorzugsweise liegt diese Anzahl in einem Bereich von zwei ganzen Zahlen, noch bevorzugter liegt diese Anzahl zwischen 5 und 60;
- Erstellen der ausgewählten Anzahl von Richtlinie-Elementen, wobei jedes Element ein Paar aus einer Schicht-ID und einer Liste von Knoten in der genannten Schicht ist und gebildet wird durch:

  - zufällige Auswahl einer Schicht des hierarchischen Modells (3); die Auswahl erfolgt zwischen Schicht L.1 und Schicht L.β-1;
  - Erstellen einer Liste durch willkürliches Auswählen eines oder mehrerer Knoten aus der Auswahlschicht.

8. Verfahren nach Anspruch 7, wobei die Richtlinie (5), (5.1) ein JSON-Skript ist, bei dem die JSON-Schlüssel die Schicht-ID des hierarchischen Modells (3) und die JSON-Werte die Knoten-ID(s) des Schicht sind.

9. Verfahren nach einem der vorhergehenden Ansprüche 4 bis 8, wobei im Kunden-Erstellungsprozess und im Kunden-Authentifizierungsprozess die kodierte Kunden-Passcode-Hierarchie (4) durch Kodierung des Kunden-Passcodes (7) in verschiedenen Größen auf der Grundlage der Größen jeder Schicht (L.1 bis L.β-1) des hierarchischen Modells (3) erzeugt wird.

10. Verfahren nach einem der vorhergehenden Ansprüche, wobei im Kunden-Authentifizierungsprozess die Erzeugung des kodierter Kunden-Passcodes (6) die folgenden Schritte auf der Kunden-Seite umfasst:

(i) Eingabe des Kunden-Passcodes (7) in das hierarchische Modell (3), um die kodierte Kunden-Passcode-Hierarchie (4) zu erzeugen;
(ii) Anwendung der Richtlinie (5) auf die kodierte Kunden-Passcode-Hierarchie (4), um den kodierter Kunden-Passcodes (6) zu erzeugen, der durch Gruppierung der jeweiligen kodierter Passcodes gebildet wird.

11. Verfahren nach einem der vorhergehenden Ansprüche, wobei im Kunden-Authentifizierungsprozess die Erzeugung des erwarteten kodierter Kunden-Passcodes (6) auf der Server-Seite die folgenden Schritte umfasst:

(i) Beim Empfang einer Authentifizierungsanforderung durch den Kunden (12) lädt der Server die kodierte Kunden-Passcode-Hierarchie (4) und die mit dem Kunden (12) geteilte Richtlinie (5);
(ii) Anwendung der Richtlinie (5) auf die kodierte Kunden-Passcode-Hierarchie (4), um den erwarteten kodierter Kunden-Passcodes (6) zu erzeugen, der durch Gruppierung der jeweiligen kodierter Passcodes gebildet wird.

12. Verfahren nach einem der vorhergehenden Ansprüche 1 bis 11, wobei der Modell-Erstellungsprozess, die Erzeugung der kodierten Kunden-Passcode-Hierarchie (4) im Kunden-Erstellungsprozess und die Erzeugung der Richtlinie (5), (5.1) sowohl im Kunden-Erstellungsprozess als auch im Kunden-Authentifizierungsprozess auf der Seite des Servers ausgeführt werden; und
der Kunde (12) speichert das hierarchische Modell (3) und die Richtlinie (5), (5.1).

13. Verfahren nach Anspruch 12, wobei der Kunden-Erstellungsprozess die folgenden Schritte umfasst:

- ein Kunden (12) sendet eine Kunden-Erstellungsanforderung an einen Server-Administrator; die Anforderung umfasst eine Kunden-ID (10), einen Kunden-Passcodes (7) und ein Token (11);
- nach erfolgreicher Validierung des Tokens (11) durch den Server-Administrator wird der Kunden-Passcodes (7)

als Eingabe für das regulierten Aktivierungsnetzwerk-Rechenmodells (2) verwendet, um ein hierarchisches Modell (3) und eine entsprechende kodierte Kunden-Passcode-Hierarchie (4) für den besagten Kunden-Passcodes (7) zu erzeugen;
- der Kunden-Passcodes (7) und die kodierte Kunden-Passcode-Hierarchie (4) werden auf dem Server (9) als Informationen des Kunden gespeichert;
- der Server (9) erzeugt eine Richtlinie (5) auf der Grundlage des hierarchischen Modells (3), und die erzeugte Richtlinie (5) wird auf dem Server (9) mit den Informationen des Kunden als erwartete Richtlinie (5) für den nächsten Authentifizierungsversuch durch den Kunden (12) gespeichert;
- eine Kopie der Richtlinie (5) und des hierarchischen Modells (3) wird an den Kunden (12) gesendet, um im System des Kunden gespeichert zu werden.

14. Verfahren nach einem der vorhergehenden Ansprüche 1 bis 11, wobei der Modell-Erstellungsprozess, die Erzeugung der kodierten Kunden-Passcode-Hierarchie (4) und die Erzeugung der Richtlinie (5), (5.1) sowohl im Kunden-Erstellungsprozess als auch im Kunden-Erstellungs-Authentifizierungsprozess auf der Seite des Kündens ausgeführt werden; und
der Server (9) speichert die kodierte Kunden-Passcode-Hierarchie (4) und die Richtlinie (5), (5.1).

15. Verfahren nach Anspruch 14, wobei der Erstellungsprozess des Kunden die folgenden Schritte umfasst:

- der Kunden (12) sendet eine Kunden-Erstellungsanforderung an einen Server-Administrator; die Anforderung umfasst eine Kunden-ID (10), eine kodierte Kunden-Passcode-Hierarchie (4), eine Richtlinie (5) und ein Token (11), das vom Server-Administrator bereitgestellt wird;
- nach erfolgreicher Validierung des Tokens durch den Server-Administrator wird der Kunden (12) mit einer eindeutigen ID erstellt;
- der Server (9) speichert die Kunden-ID (10), die kodierte Kunden-Passcode-Hierarchie (4) und die Richtlinie (5) für den nächsten Authentifizierungsversuch des Kündens (12).

Revendications

1. Méthode d'authentification d'un client dans une architecture client-serveur, la méthode comprenant un processus de création de modèle, un processus de création de client et un processus d'authentification de client; dans laquelle,

le processus de création de modèle génère:

(i) du côté du client ou du côté du serveur, un modèle hiérarchique (3), partagé entre le client et le serveur, basé sur l'alimentation d'un modèle de calcul de réseau d'activation régulé (2) avec un ensemble de données d'entrée (1) à N dimensions généré de manière aléatoire, ledit modèle hiérarchique (3) étant composé de plusieurs couches;

le processus de création de client génère:

(i) du côté du client ou du côté du serveur, une première hiérarchie de mot de passe codé (4) du client en alimentant le modèle hiérarchique (3) avec un mot de passe (7) du client; la hiérarchie de mot de passe codé (4) du client est sauvegardée sur le serveur (9) et comprend un ensemble de mots de passe codés représentant des versions codées du mot de passe (7) du client; la hiérarchie de mot de passe codé (4) du client est formée par des couches du modèle hiérarchique (3), chacune de ces couches, à l'exception de la couche d'entrée, stockant une version codée desdites versions codées du mot de passe (7) du client;
(ii) du côté du client ou du côté du serveur, une politique (5) pour la prochaine tentative d'authentification du client, qui est partagée entre le client (12) et le serveur (9);
et

le processus d'authentification de client:

(i) génère, du côté du client, une deuxième hiérarchie de mot de passe codé (4) du client en alimentant le modèle hiérarchique (3) avec le mot de passe (7) du client et un mot de passe codé (6) du client est créé à l'aide de la hiérarchie de mot de passe codé (4) du client et de la politique (5) partagée entre le client (12) et le serveur (9) ;

(ii) authentifie le client (12) du côté du serveur, si le mot de passe codé (6) du client, reçu du client, correspond à un mot de passe codé attendu, généré par le serveur (9) à l'aide de la première hiérarchie de mot de passe codé (4) du client sauvegardée et de la politique partagée (5);
(iii) génère, du côté du client ou du côté du serveur, une nouvelle politique (5.1), si le client (12) est authentifié avec succès, qui est partagée entre le client (12) et le serveur (9) pour la prochaine tentative d'authentification du client.

2. Méthode selon la revendication 1, dans laquelle l'ensemble de données d'entrée (1) est formé d'unités de données de type entier; chaque unité de données est une valeur entière comprise entre un minimum et un maximum; de préférence, chaque unité de données est un nombre entier compris entre 1 et 2000.

3. Méthode selon les revendications 1 ou 2, dans laquelle le modèle de calcul de réseau d'activation régulé (2) est configuré pour mettre en œuvre un algorithme k-means en tant qu'identificateur de concept.

4. Méthode selon l'une quelconque des revendications précédentes, dans laquelle le modèle hiérarchique (3) comprend plusieurs couches β, chaque couche comprenant au moins un nœud, déterminant la taille de la couche, et dans laquelle la couche L.0 est la couche d'entrée et comprend N nœuds, N étant la dimension de l'ensemble de données d'entrée (1); et les couches L.1 à L.β-1 sont des couches générées dynamiquement, chacune de ces couches stockant un mot de passe codé.

5. Méthode selon la revendication 4, dans laquelle, dans le processus de création de client et le processus d'authentification de client, le mot de passe (7) du client est saisi dans la couche L.0 du modèle hiérarchique (3), et les couches L.1 à L.β-1 sont des couches générées dynamiquement, chacune de ces couches stockant un mot de passe codé représentant une version codée du mot de passe (7) du client.

6. Méthode selon la revendication 5, dans laquelle la hiérarchie de mot de passe codé (4) du client est formée par des couches L.1 à L.β-1 du modèle hiérarchique (3).

7. Méthode selon l'une quelconque des revendications précédentes, dans laquelle le processus de génération d'une politique (5), (5.1) est basé sur le modèle hiérarchique (3) et comprend les étapes suivantes:

- sélection d'un nombre d'éléments pour la politique; de préférence, ce nombre est compris dans une fourchette de deux nombres entiers, de préférence encore, ce nombre est compris entre 5 et 60;
- création du nombre sélectionné d'éléments de politique, chaque élément étant une paire d'ID de couche et une liste de nœuds dans ladite couche et étant formé par:
- sélection aléatoire d'une couche du modèle hiérarchique (3); la sélection étant effectuée entre la couche L.1 et la couche L.β-1;
- création d'une liste, en choisissant arbitrairement un ou plusieurs nœuds dans la couche de sélection.

8. Méthode selon la revendication 7, dans laquelle la politique (5), (5.1) est un script JSON dont les clés JSON sont l'ID de couche du modèle hiérarchique (3) et les valeurs JSON sont les ID(s) des nœuds de la couche.

9. Méthode selon l'une des revendications précédentes 4 à 8, dans laquelle, dans le processus de création de client et le processus d'authentification de client, la hiérarchie de mot de passe codé (4) du client est générée en codant le mot de passe (7) du client dans des tailles variées, basées sur les tailles de chaque couche (L.1 à L.β-1) du modèle hiérarchique (3).

10. Méthode selon l'une quelconque des revendications précédentes, dans laquelle, dans le processus d'authentification de client, la génération du mot de code codé (6) du client comprend les étapes suivantes, du côté du client:

(i) introduction du mot de passe (7) du client dans le modèle hiérarchique (3) afin de générer la hiérarchie de mot de passe codé (4) du client;
(ii) application de la politique (5) à la hiérarchie de mot de passe codé (4) du client pour générer le mot de passe codé (6) du client, formé par le regroupement des respectifs mots de passe codés.

11. Méthode selon l'une quelconque des revendications précédentes, dans laquelle, dans le processus d'authentification de client, la génération du mot de passe codé (6) attendu du côté du serveur comprend les étapes suivantes:

(i) à la réception d'une demande d'authentification du client (12), le serveur charge la hiérarchie de mot de passe codé (4) du client et la politique (5) partagée avec le client (12);
(ii) application de la politique (5) à la hiérarchie de mot de passe codé (4) du client pour générer le mot de passe codé (6) attendu, formé par le regroupement des respectifs mots de passe codés.

12. Méthode selon l'une des revendications précédentes 1 à 11, dans laquelle le processus de création de modèle, la génération de la hiérarchie de mot de passe codé (4) du client lors du processus de création de client et la génération de la politique (5), (5.1) lors du processus de création de client et du processus d'authentification de client sont exécutés du côté du serveur; et
le client (12) stocke le modèle hiérarchique (3) et la politique (5), (5.1).

13. Méthode selon la revendication 12, dans laquelle le processus de création de client comprend les étapes suivantes:

- un client (12) envoie une demande de création de client à un administrateur de serveur; la demande comprend une ID (10) du client, un mot de passe (7) du client et un jeton (11);
- lorsque la validation du jeton (11) par l'administrateur de serveur est réussie, le mot de passe (7) du client est utilisé comme entrée dans le modèle de calcul de réseau d'activation régulé (2) pour générer un modèle hiérarchique (3) et une respective hiérarchie de mot de passe codé (4) du client pour ledit mot de passe (7) du client;
- le mot de passe (7) du client et la hiérarchie de mot de passe codé (4) du client sont sauvegardés sur le serveur (9) en tant qu'information du client;
- le serveur (9) génère une politique (5) basée sur le modèle hiérarchique (3), et la politique générée (5) est sauvegardée sur le serveur (9) avec l'information du client en tant que politique attendue (5) pour la prochaine tentative d'authentification par le client (12);
- une copie de la politique (5) et du modèle hiérarchique (3) est envoyée au client (12) pour être sauvegardée dans le système du client.

14. Méthode selon l'une des revendications précédentes 1 à 11, dans laquelle le processus de création de modèle, la génération de la hiérarchie de mot de passe codé (4) du client et la génération de la politique (5), (5.1) lors du processus de création de client et du processus d'authentification de création de client sont exécutés du côté du client; et
le serveur (9) stocke la hiérarchie de mot de passe codé (4) du client et la politique (5), (5.1).

15. Méthode selon la revendication 14, dans laquelle le processus de création du client comprend les étapes suivantes:

- le client (12) envoie une demande de création de client à un administrateur de serveur; la demande comprend une ID (10) du client, une hiérarchie de mot de passe codé (4) du client, une politique (5) et un jeton (11) fourni par l'administrateur de serveur;
- lorsque la validation du jeton par l'administrateur de serveur est réussie, le client (12) est créé avec une ID unique;
- le serveur (9) stocke l'ID (10) du client, la hiérarchie de mot de passe codé (4) du client et la politique (5) pour la prochaine tentative d'authentification du client (12).

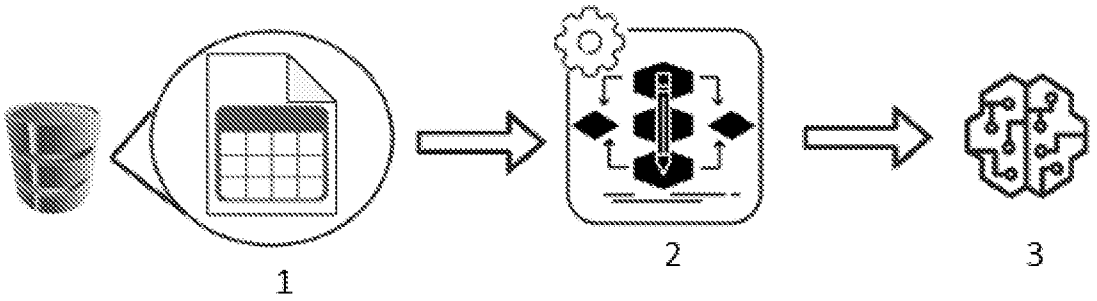

Figure 1

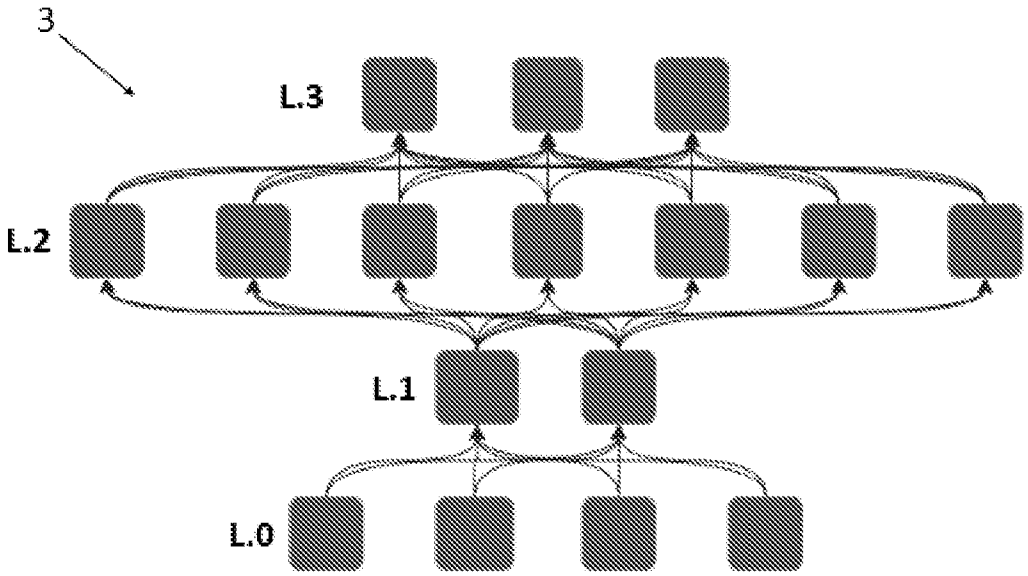

Figure 2

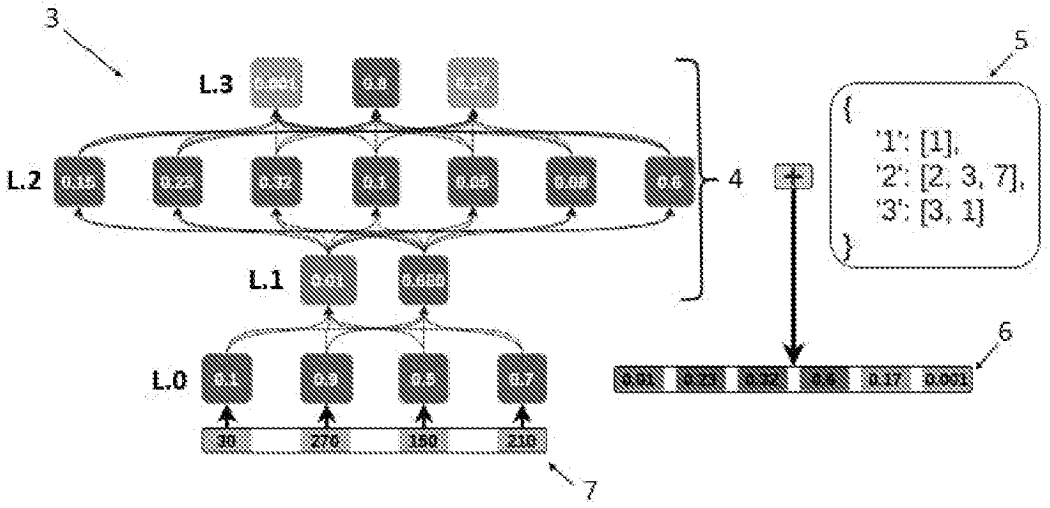

Figure 3

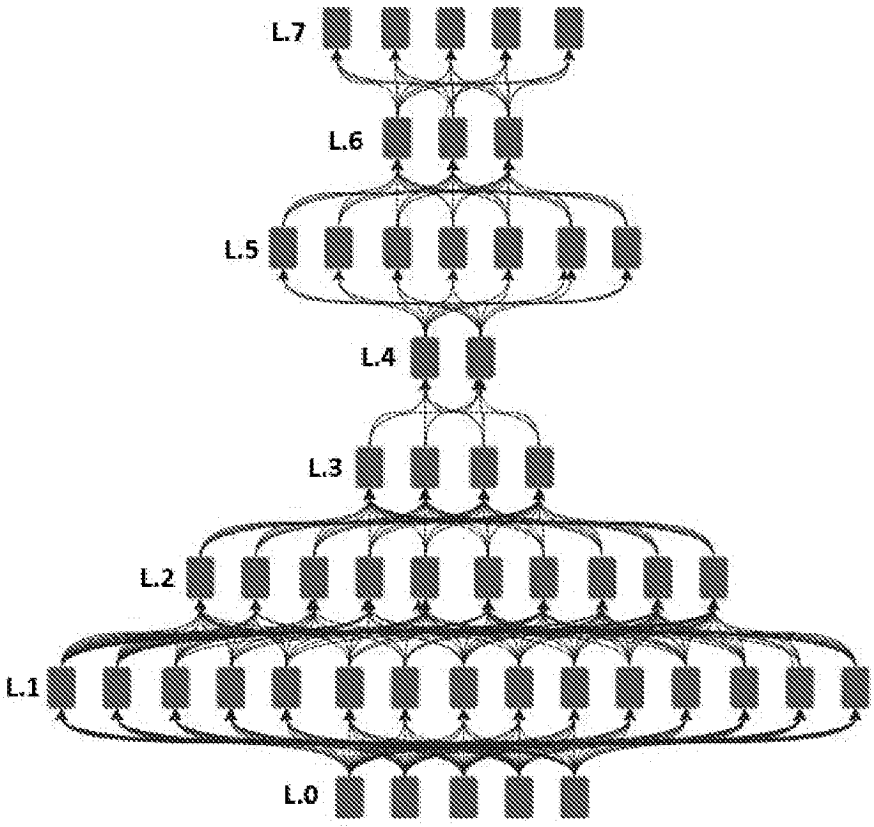

Figure 4

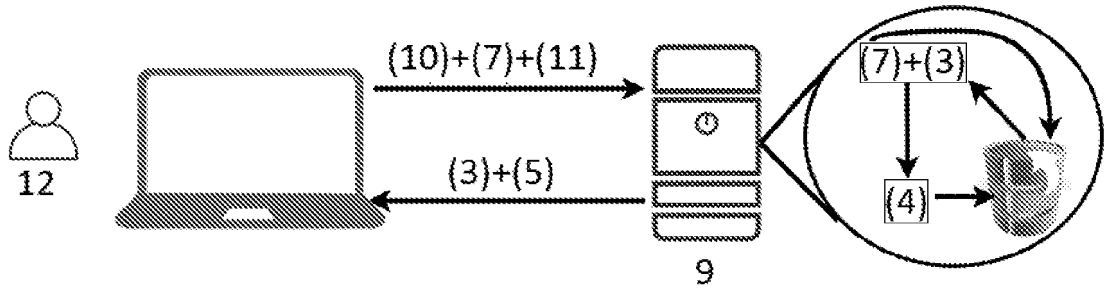

Figure 5

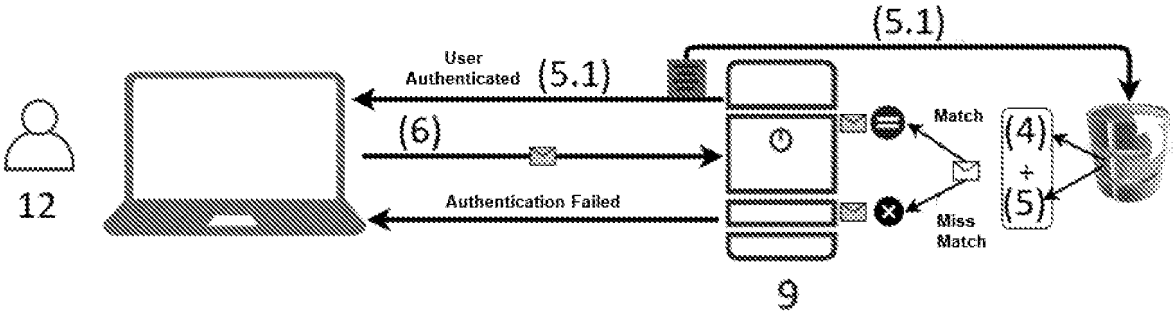

Figure 6

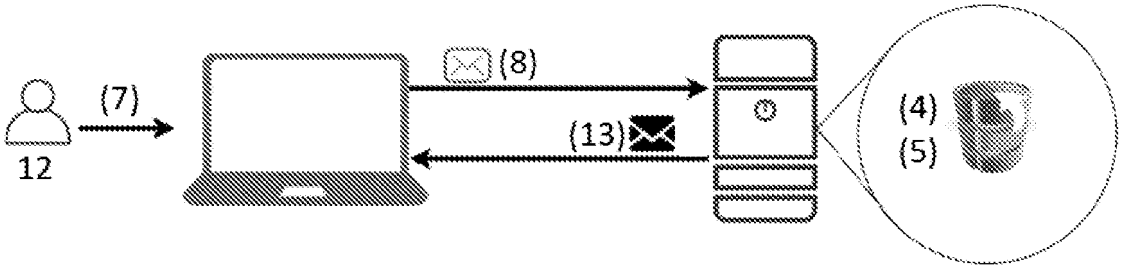

Figure 7

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 2020120086 A1 **[0002]**
- US 2007130474 A1 **[0002]**

Non-patent literature cited in the description

- **SHARMA R** ; **RIBEIRO B** ; **MIGUEL PINTO A** ; **CARDOSO FA**. Exploring Geometric Feature Hyper-Space in Data to Learn Representations of Abstract Concepts. *Applied Sciences*, 2020, vol. 10 (6), 1994, https://doi.org/10.3390/app10061994 **[0031]**
- **SHARMA R et al.** Learning non-convex abstract concepts with regulated activation networks. *Annals of Mathematics and Artificial Intelligence*, 21 March 2020, vol. 88 (11-12), 1207-1235 **[0031]**
- Fast, Lean, and Accurate: Modeling Password Guessability using Neural Networks. **WILLIAM MELICHER et al.** Usenix. The Advanced Computing Systems Association, 06 January 2017, 186-202 **[0031]**