

(12) DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITÉ DE COOPÉRATION EN MATIÈRE DE BREVETS (PCT)

(19) Organisation Mondiale de la
Propriété Intellectuelle
Bureau international

(43) Date de la publication internationale
23 juin 2022 (23.06.2022)



(10) Numéro de publication internationale
WO 2022/129772 A1

(51) Classification internationale des brevets :

G06F 21/36 (2013.01) H04W 12/06 (2021.01)
G06F 21/31 (2013.01)

(21) Numéro de la demande internationale :

PCT/FR2021/052316

(22) Date de dépôt international :

14 décembre 2021 (14.12.2021)

(25) Langue de dépôt :

français

(26) Langue de publication :

français

(30) Données relatives à la priorité :

FR2013557 17 décembre 2020 (17.12.2020) FR

(71) Déposant : **ORANGE** [FR/FR] ; 111, quai du Président
Roosevelt, 92130 Issy-les-Moulineaux (FR).

(72) Inventeurs : **MITTIG, Karel** ; Orange - Intellectual Property and Licensing, Orange Gardens - 44 avenue de la République -, CS 50010, 92326 Châtillon Cedex (FR). **BIGNON, Fabien** ; Orange - Intellectual Property and Licensing, Orange Gardens - 44 avenue de la République -, CS 50010, 92326 Châtillon Cedex (FR).

(81) États désignés (sauf indication contraire, pour tout titre de protection nationale disponible) : AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA,

(54) Title: METHOD FOR GENERATING A FORM BASED ON A REFERENCE FORM AND FOR TRANSMISSION OF SAID FORM TO A TERMINAL

(54) Titre : PROCÉDÉ DE GÉNÉRATION D'UN FORMULAIRE À PARTIR D'UN FORMULAIRE DE RÉFÉRENCE ET D'ENVOI À UN TERMINAL

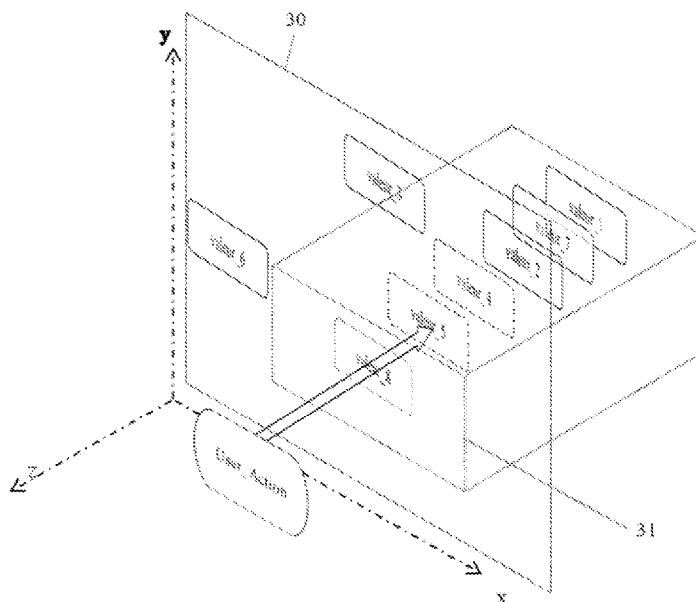


FIG. 3b

(57) Abstract: The invention relates to a method for generating a form based on a reference form and for transmission of said form to a terminal (T), the reference form comprising at least one validation zone to be protected, the generation of the form comprising the following steps: - replacing (E3) the validation zone to be protected with at least one new validation zone, the new validation zone comprising at least two validation components, - generating (E4) a unique value which belongs to the at least two validation components, - selecting (E6) one of the validation components as the correct component, the other components of the at least one component becoming decoys, - generating (E7) a positioning code in the new validation zone for the correct component and for the others of the at least one decoy, the positioning code allowing the correct component to be activated by the user, - transmitting (E8)

[Suite sur la page suivante]



WO 2022/129772 A1

SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR,
TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) États désignés (*sauf indication contraire, pour tout titre de protection régionale disponible*) : ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), eurasién (AM, AZ, BY, KG, KZ, RU, TJ, TM), européen (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Publiée:

— avec rapport de recherche internationale (Art. 21(3))

to the terminal (T) the form comprising the new validation zone.

(57) Abrégé : L'invention porte sur un procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal (T), ledit formulaire de référence comprenant au moins une zone de validation à protéger, la génération du formulaire comprenant les étapes suivantes : - substitution (E3) de ladite zone de validation à protéger par au moins une nouvelle zone de validation, ladite nouvelle zone de validation comprenant au moins deux composants de validation, - génération (E4) d'une valeur unique propre auxdits au moins deux composants de validation, - sélection (E6) d'un desdits composants de validation comme composant correct, les autres desdits au moins un composant devenant des leurres, - génération (E7) d'un code de positionnement dans la nouvelle zone de validation pour le composant correct et pour les autres desdits au moins un leurre, le code de positionnement permettant au composant correct d'être activé par l'utilisateur, - envoi (E8) au terminal (T) du formulaire comprenant la nouvelle zone de validation.

Procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal

La présente invention se rapporte au domaine général de l'informatique et concerne plus précisément un procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal. Plus précisément, le procédé est destiné à filtrer des automates malveillants sur des contenus web destinés à un être humain, de manière transparente pour l'utilisateur.

L'invention s'applique de façon privilégiée à la sécurisation de transactions de sites web et a pour but de bloquer des automates malveillants, plus précisément d'empêcher de tels automates d'effectuer des actions malveillantes telles que :

- exploiter des formulaires d'authentification pour effectuer des attaques par dictionnaire,
- exploiter des formulaires de contact pour envoyer du spam,
- exploiter des formulaires de création de comptes pour créer de faux profils utilisateurs,
- exploiter des formulaires de publication de contenus sur des blogs et forums pour du spam et des codes malveillants,
- aspirer des sites web et contenus à accès limité,
- etc.

Distinguer un automate, on parle habituellement de « bot », d'un être humain est une problématique majeure pour les sites web sur Internet. En effet, une portion importante du trafic web mondial est liée à des automates, parmi lesquels plus de la moitié correspondent à des codes malveillants.

Depuis les années 2000, la solution pour bloquer ces automates malveillants repose sur la mise en place de tests de Turing permettant de différencier de manière automatisée un utilisateur humain d'un automate. Ces tests, plus connus sous le rétro-acronyme de CAPTCHATM (pour « Completely Automated Public Turing test to tell Computers and Humans Apart »), consistent en une boîte de dialogue demandant la résolution d'un exercice simple pour un humain mais complexe pour une machine. Par exemple, ces tests peuvent consister en la reconnaissance de texte à partir d'une image du texte déformée, la reconnaissance d'éléments au sein d'une image, ou encore l'ordonnancement d'images.

Les mécanismes mis en place par les bots malveillants pour contourner les CAPTCHA reposent tous sur une analyse du contenu et de la structure des pages HTML (pour « HyperText Markup Language ») afin d'interagir avec le code ciblé. Les CAPTCHA ont donc évolué de manière à complexifier l'analyse de contenu.

Cependant, la disponibilité en accès libre de services de reconnaissance d'images de plus en plus efficaces a permis aux bots malveillants de résoudre la majorité des CAPTCHA, entraînant une

escalade avec des exercices de plus en plus complexes à résoudre. Certains de ces tests sont devenus maintenant tellement complexes que des automates obtiennent parfois de meilleurs résultats que les humains sur des figures trop complexes. L'impact ergonomique est important pour des utilisateurs légitimes qui se voient obligés de répondre à des problèmes absurdes et répétitifs
5 afin d'accéder à leurs services, alors même que l'efficacité de ces méthodes est de plus en plus discutable.

Un des buts de l'invention est de remédier à des insuffisances/inconvénients de l'état de la technique, et/ou d'y apporter des améliorations.

A cette fin, l'invention propose un procédé de génération d'un formulaire à partir d'un formulaire
10 de référence et d'envoi à un terminal, ledit formulaire de référence comprenant au moins une zone de validation à protéger, la génération du formulaire comprenant les étapes suivantes :

- substitution de ladite zone de validation à protéger par au moins une nouvelle zone de validation, ladite nouvelle zone de validation comprenant au moins deux composants de validation,
- génération d'une valeur unique propre auxdits au moins deux composants de validation,
- 15 - sélection d'un desdits composants de validation comme composant correct, les autres desdits au moins un composant devenant des leurres,
- génération d'un code de positionnement dans la nouvelle zone de validation pour le composant correct et pour les autres desdits au moins un leurre, ledit code de positionnement permettant au composant correct d'être activé par l'utilisateur,
- 20 - envoi au terminal du formulaire comprenant la nouvelle zone de validation.

Le procédé propose une solution permettant de lutter efficacement contre des bots malveillants dits « basiques » qui effectuent une analyse textuelle de formulaires de référence afin d'identifier une valeur à envoyer au serveur correspondant à une action de validation de ce formulaire. De tels automates sont alors capables de remplir automatiquement de tels formulaires et de perpétrer
25 des actions malveillantes telles que des attaques par dictionnaires, l'envoi de spam, etc. En substituant dans le formulaire de référence la zone de validation par une nouvelle zone de validation qui comprend au moins deux composants de validation, le procédé complique, voire empêche, l'analyse textuelle réalisée par de tels bots qui se trouvent alors bloqués. L'action qui permet à l'utilisateur de soumettre le formulaire est remplacée par un ensemble de possibilités, variables
30 dans le temps, c'est-à-dire qui change à chaque chargement du formulaire. Cela devient donc très difficilement analysable par le programme malveillant qui repose sur une analyse textuelle du contenu de la page correspondant au formulaire.

De façon avantageuse, le procédé de génération d'un formulaire et d'envoi à un terminal comprend :

- 35 - réception en provenance du terminal d'une valeur de sélection, ladite valeur de sélection étant la valeur unique propre au composant de validation sélectionné par l'utilisateur,

- lorsque la valeur de sélection est égale à la valeur propre au composant de validation correct, traitement des données du formulaire.

Le procédé est transparent pour un utilisateur légitime : le rendu d'une page d'un formulaire proposée par le serveur en réponse à une requête est inchangé par rapport au rendu du formulaire de référence qui serait proposé si le procédé n'était pas mis en œuvre.

Avec le procédé, il n'est pas nécessaire d'ajouter au traitement du formulaire un test de type CAPTCHA afin de distinguer un utilisateur d'un bot. L'impact ergonomique du procédé est donc nul pour un utilisateur.

Par ailleurs, les éléments à remplir par l'utilisateur, par exemple, son adresse e-mail, son mot de passe, restent identifiables et accessibles dans le formulaire. Ainsi, si l'utilisateur utilise un gestionnaire de mot de passe, le procédé de l'invention est sans impact pour ce gestionnaire dont le fonctionnement repose sur une analyse du formulaire.

Avantageusement, la sélection du composant de validation correct et de sa valeur unique propre parmi les au moins deux composants de validation est pseudo-aléatoire.

Le choix du composant de validation correct et de sa valeur unique propre associée étant pseudo-aléatoire, il est donc impossible pour un bot qui effectue une analyse purement textuelle du formulaire d'anticiper le choix d'un composant de validation correct, le choix de celui-ci variant d'un chargement du formulaire à un autre.

Dans un exemple de réalisation, un desdits au moins deux composants de validation peut être dans un état compris dans un ensemble d'états comprenant : superposé à un autre desdits au moins un composant, distribués dans l'espace, visible, transparent, masqué, apte à se déplacer dynamiquement lors du chargement du formulaire, apte à se déplacer dynamiquement sur une action de l'utilisateur

Ainsi, les possibilités de positionnement et de représentation du composant correct et des leurres sont nombreuses, ce qui complexifie, voire rend impossible une analyse textuelle du formulaire. En effet, il est ainsi possible d'obtenir de nombreux formulaires de code différents avec un rendu parfaitement identique pour un utilisateur.

Dans un premier exemple de réalisation, le formulaire de référence étant constitué d'une page HTML, la nouvelle zone de validation est constituée d'une sous-page HTML imbriquée dans la page HTML associée au formulaire de référence sous forme d'une balise iFrame.

Dans un deuxième exemple de réalisation, le formulaire de référence étant constitué d'une page HTML, ledit formulaire de référence comprend une balise HTML spécifique, associée à une librairie javascript, ladite balise spécifique étant remplacée par un code substituant la zone de validation par la nouvelle zone de validation.

Da façon avantageuse, le formulaire comprend au moins deux nouvelles zones de validation, lesdites nouvelles zones de validation étant cloisonnées les unes par rapport aux autres.

Cet exemple de réalisation permet de lutter contre des bots qualifiés de « standards », qui disposent d'un moteur de rendu HTML et ont le plus souvent la capacité d'exécuter du code javascript.

5 De tels bots sont capables de récupérer dans une page HTML l'élément visible, tel qu'un composant de validation, situé à des coordonnées données. On comprend qu'une fois cet élément récupéré il est facile pour le bot d'extraire la valeur associée et de la soumettre au serveur ou encore de simuler de façon programmatique une interaction avec un formulaire de validation, tel qu'un clic souris. Avec l'utilisation de plusieurs zones de validation, chacune étant considérée comme
10 un espace cloisonné, il est impossible pour un bot standard de travailler sur un rendu global d'une combinaison ces nouvelles zones de validation. Ainsi, un bot standard, et a fortiori un bot basique, est dans l'incapacité d'identifier le composant de validation correct et activable par l'utilisateur parmi les composants de validation des combinaisons de nouvelles zones de validation.

Dans un exemple de réalisation, lesdites au moins deux nouvelles zones de validation étant sous
15 forme d'au moins deux sous-pages, le procédé comprend la génération d'un décalage pseudo-aléatoire des abscisses et des ordonnées entre les au moins deux sous-pages, la génération du décalage garantissant une intersection entre une zone visible de l'utilisateur et les au moins deux sous-pages supérieure à la taille d'un composant de validation.

Cela garantit de toujours pouvoir afficher un composant de validation, tout en introduisant une
20 certaine flexibilité dans le positionnement des différentes zones de validation les unes par rapport aux autres.

Dans un exemple de réalisation, la nouvelle zone de validation comprend une zone visible par l'utilisateur, ladite zone visible étant de taille égale à un multiple de la taille d'un composant de validation visible, ledit composant de validation visible étant positionné de manière pseudo-aléa-
25 toire dans la nouvelle zone de validation.

Cet exemple de réalisation permet de lutter contre des bots qualifiés de « évolués » qui sont capables de simuler des actions de l'utilisateur, comme des frappes au clavier, des déplacements, ou des clics souris. Ces actions sont strictement équivalentes, d'un point de vue logiciel, à celles générées par de vrais périphériques d'entrée, et peuvent en absolu être distinguées par des solu-
30 tions basées sur l'analyse comportementale et l'Intelligence Artificielle. Ce type d'approche se base sur une modélisation du comportement de l'utilisateur avant que l'action, tel un clic souris, n'intervienne. Cependant l'analyse comportementale pose de réels problèmes de préservation de la vie privée. En effet, elle consiste à enregistrer des données propres à l'utilisateur afin de modéliser son comportement. La solution proposée ici offre une solution qui permet de lutter à elle
35 seule contre les bots évolués et rend donc optionnelle une analyse comportementale. Le caractère

aléatoire introduit dans le positionnement du composant de validation visible de l'utilisateur ne permet pas de simuler le comportement de l'utilisateur. On peut qualifier cette approche de respectueuse de la vie privée car indépendante de toute analyse comportementale.

L'invention concerne également un serveur apte à générer un formulaire de validation à partir
5 d'un formulaire de référence et à l'envoyer à un terminal, ledit formulaire de référence comprenant au moins une zone de validation à protéger, le serveur comprenant :

- des moyens de substitution, agencés pour substituer ladite zone de validation à protéger par au moins une nouvelle zone de validation, ladite nouvelle zone de validation comprenant au moins deux composants de validation,
- 10 - des premiers moyens de génération, agencés pour générer une valeur unique propre auxdits au moins deux composants de validation,
- des moyens de sélection, agencés pour sélectionner un desdits composants de validation comme composant correct, les autres desdits au moins un composant devenant des leurres,
- des deuxièmes moyens de génération, agencés pour générer un code de positionnement dans la
15 nouvelle zone de validation pour le composant correct et pour les autres desdits au moins leurre, le code de positionnement permettant au composant correct d'être activé par l'utilisateur,
- des moyens d'envoi, agencés pour envoyer au terminal le formulaire comprenant la nouvelle zone de validation.

L'invention porte aussi sur un programme d'ordinateur sur un support de données et chargeable
20 dans la mémoire d'un ordinateur, le programme comprenant des portions de code pour exécuter les étapes du procédé de génération d'un formulaire de validation et d'envoi à un terminal tel que décrit précédemment, lorsque le programme est exécuté sur ledit ordinateur.

L'invention concerne enfin également un support de données dans lequel est enregistré le programme tel que présenté ci-dessus.

25 D'autres caractéristiques et avantages de la présente invention seront mieux compris de la description détaillée, des annexes et des figures annexées parmi lesquels :

- la figure 1 présente les étapes d'un procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal, selon un premier exemple de réalisation ;
- la figure 2 présente un exemple de code de formulaire de référence et de code modifié au moyen
30 d'une balise iFrame ;
- la figure 3a présente un exemple de code HTML généré conformément aux étapes du procédé de génération et d'envoi d'un formulaire, selon un premier exemple de réalisation ;
- la figure 3b représente une modélisation du rendu de la sous-page présentée à la figure 3a ;

- la figure 4a présente un exemple de code HTML généré conformément aux étapes du procédé de génération et d'envoi d'un formulaire, selon un deuxième exemple de réalisation ;

- la figure 4b représente une modélisation du rendu des sous-pages présentées à la figure 4a ;

5 - la figure 5 représente une illustration d'un formulaire généré à partir des étapes du procédé, selon un troisième exemple de réalisation ;

- la figure 6 est une représentation schématique d'un serveur agencé pour mettre en œuvre les étapes du procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal, selon un premier exemple de réalisation.

10 Les étapes d'un procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal, selon un premier exemple de réalisation, vont maintenant être décrites en relation avec la figure 1.

15 Le procédé décrit ici est destiné à distinguer un automate (ou « bot » en anglais), c'est-à-dire un logiciel informatique effectuant des actions automatiques, d'un être humain lors du traitement d'un formulaire de validation. L'exemple est décrit ici dans le cas de formulaire de type HTML (pour « HyperText Markup Language »), embarquant ou non du code javascript. L'invention peut s'appliquer à d'autres types de formulaires. A noter cependant que les langages actuels connus adaptés pour créer de tels formulaires sont en général des métalangages agencés pour générer des pages HTML. Lorsqu'il est malveillant, un tel automate peut être agencé pour par exemple, remplir et envoyer des formulaires d'authentification de manière à perpétrer des attaques par dictionnaires contre des sites sensibles. Par la suite le mot « bot » désigne un bot malveillant.

20 Le procédé découle du constat que les mécanismes mis en place par les bots existants pour traiter automatiquement des formulaires de validation reposent tous sur une analyse du contenu de la structure de pages HTML qui constituent ces formulaires afin d'interagir directement avec le code ciblé. Ces bots renseignent automatiquement le formulaire le cas échéant, et identifient une portion de code associée à un composant de validation tel qu'un bouton à cliquer, puis génèrent
25 programmatiquement un événement associé à ce composant, tel qu'un clic souris par exemple, afin que le formulaire soit traité comme si c'était un être humain qui l'avait renseigné. Au contraire, un humain s'appuie sur le rendu visuel de la page pour naviguer : il visualise un composant de validation et opère une action sur ce composant pour accéder à l'étape suivante dans un processus donné. Le procédé décrit ici exploite cette différence et « noie » en quelque sorte l'information perceptible par l'utilisateur et qui permet à celui-ci de valider les données d'une page d'un formulaire, dans un ensemble de composants de validation dont un seul est activable par l'utilisateur, les autres servant alors de leurres pour un programme automatisé analysant la page correspondant au formulaire. On entend par « activable » un composant qui, sélectionné par l'utilisateur
30 par un moyen de pointage ou une touche de validation, provoque la génération d'un événement
35

associé qui déclenche l'envoi des données de la page et leur traitement par le serveur. Il peut s'agir par exemple de valider l'envoi de données d'identification et d'authentification saisies dans un formulaire.

Un terminal informatique T est agencé pour émettre des requêtes afin d'accéder à des ressources d'un réseau, par exemple le réseau Internet. Les ressources, telles que des services, sont accédées au moyen d'un protocole de communication client-serveur, par exemple « HTTP » (« HyperText Transfer Protocol »). Les clients HTTP les plus connus sont des navigateurs web qui permettent à des utilisateurs d'accéder au moyen de leur terminal à des données stockées sur un serveur S. Dans l'exemple décrit ici, le terminal informatique T est un ordinateur personnel, un terminal mobile, une tablette, un « PDA » (« Personal Digital Assistant »), etc. Le serveur S est un ordinateur, hébergeur de services, apte à traiter des requêtes de services émanant du terminal T. Par la suite on appelle ressource, ou « URL » (pour « Uniform Resource Locator »), une donnée proposée par un serveur ou le serveur lui-même qui sont concernés par la requête. Le procédé s'intéresse plus particulièrement aux pages ou formulaires de validation de pages web. De tels formulaires sont destinés à valider l'envoi de données au serveur, ou valider le passage à une page suivante dans un enchaînement de pages web qui constitue par exemple un parcours client. Un tel formulaire peut comprendre par exemple une zone dans laquelle l'utilisateur saisie des données telles qu'un identifiant, et un bouton, ou composant de validation, permettant de valider l'envoi au serveur des données saisies. Dans un autre exemple de formulaire de validation associé à un parcours client, la zone de saisie est absente et l'utilisateur doit appuyer sur le bouton pour accéder à une page suivante.

Ainsi, dans une étape initiale E0 d'envoi d'une requête, le terminal utilisateur T envoie une requête d'accès REQ à une ressource du serveur S. Cette requête REQ correspond à une demande d'accès au serveur S faite par l'utilisateur depuis un navigateur du terminal T en sélectionnant une URL associée au serveur.

La requête est reçue par le serveur S dans une étape E1 de réception. On suppose que l'URL sélectionnée par l'utilisateur est associée sur le serveur à une page HTML correspondant à un formulaire de validation dit « de référence ». Ce formulaire comprend le cas échéant des éléments pour lesquels l'utilisateur est censé saisir des informations, soit sous forme de texte, soit sous forme d'éléments à sélectionner tels que des cases à cocher, des listes à sélectionner, etc., et une zone de validation à protéger, par exemple un bouton « envoyer » ou un bouton « suivant » d'un parcours client. C'est cette zone de validation qui est protégée par les étapes du procédé. Le formulaire est dit « de référence » car les étapes du procédé vont en quelque sorte transformer ce formulaire de référence sur réception de la requête en un formulaire FORM, et substituer à la zone de validation à protéger au moins une nouvelle zone de validation, difficilement analysable par un bot dit « basique ». Un bot basique correspond à un simple script automatisé capable de ne

percevoir une page HTML que comme un contenu textuel et de ne faire qu'une analyse textuelle d'une telle page. En effet, l'action qui permet de soumettre le formulaire de référence est remplacée dans le formulaire envoyé au terminal T par un ensemble de possibilités, variables dans le temps.

- 5 Dans une étape suivante E2 de génération d'un identifiant, le serveur S génère un identifiant unique IDU associé à la requête REQ.

Dans une étape suivante E3 de substitution, le serveur S génère au moins une nouvelle zone de validation et substitue la zone de validation à protéger par la au moins une nouvelle zone de validation. Une nouvelle zone de validation comprend au moins deux composants de validation.

- 10 Dans un premier exemple de réalisation le serveur S génère une seule nouvelle zone de validation, sous forme d'une sous-fenêtre. Dans le cas de pages HTML décrit ici, le serveur S génère la nouvelle zone de validation sous forme d'une sous-page HTML destinée à se substituer à la zone de validation à protéger du formulaire de référence. Cette sous-page est imbriquée dans la page HTML correspondant au formulaire de référence ; elle est destinée à constituer une nouvelle zone
15 de validation dans le formulaire de validation à envoyer à l'utilisateur. Ce formulaire correspond ainsi au formulaire de référence dans lequel la zone de validation à protéger a été substituée par la nouvelle zone de validation contenue dans la sous-page HTML.

- Dans un premier exemple de réalisation, la sous-page HTML générée est imbriquée dans le formulaire de référence, sous forme d'une balise « iFrame ». La balise iFrame permet d'intégrer dans
20 une page HTML le contenu d'une autre page HTML. La figure 2 présent un exemple de code de formulaire de référence comprenant un bouton de validation initial « Envoyer ! » et de code d'un formulaire modifié. La sous-page « invention.html » ainsi créée en remplacement du bouton de validation initial comprend des instructions HTML décrivant une nouvelle zone de validation qui comprend un ensemble d'au moins deux composants de validation. Cette sous-page est de taille
25 variable mais généralement de la taille de l'écran du terminal T de l'utilisateur. Cependant, seule une portion de cette sous-fenêtre est rendue visible à l'utilisateur et constitue une zone visible. Cette zone visible correspond en quelque sorte à la zone dans laquelle un nouveau composant de validation visible est susceptible d'apparaître aux yeux de l'utilisateur. La taille de la zone visible est adaptée de manière à ne pas recouvrir du texte qui serait affiché à l'attention de l'utilisateur,
30 ou des zones de saisies comprises dans la page initiale du formulaire de référence. Les paramètres de la taille de la zone visible sont définis dans le code comme paramètre d'inclusion de la sous-page iFrame.

Dans un deuxième exemple de réalisation, le formulaire de référence comprend un champ HTML spécifique sous forme de balise, associé à une librairie javascript et identifiant la zone de valida-

tion à protéger. Sur réception de la requête de l'utilisateur correspondant à l'obtention du formulaire de référence, la balise est remplacée par un code javascript qui substitue la zone de validation à protéger par la nouvelle zone de validation, conformément aux étapes du procédé.

Dans une étape E4 de génération de composants de validation, le serveur S génère au moins deux composants de validation destinés à la nouvelle zone de validation. Les composants de validation générés, notés $C_1, \dots, C_n$, avec $n \geq 2$, possèdent une information générale identique : même classe (« button »), même nom (« login »), même contenu (« submit »). Ils n'ont aucun élément distinctif en dehors de leur style qui correspond à la façon dont ils apparaissent dans la sous-page. En revanche, chacun des composants de validation possède une valeur unique. C'est à cette valeur qu'est associé un événement programmatique destiné à effectuer un traitement associé.

Ainsi, dans une étape suivante E5 de génération d'une valeur, le serveur S génère pour chacun des composants C_j de validation de la sous-page une valeur pseudo-aléatoire unique V_j . Ainsi, la valeur unique propre à chacun des composants est susceptible de changer à chaque fois qu'un formulaire de référence est traité conformément aux étapes du procédé.

Dans une étape suivante E6 de sélection d'un composant correct, le serveur S désigne de manière pseudo-aléatoire un des composants de validation C_j de la sous-page comme composant correct. Ainsi, un seul composant et sa valeur associée sont considérés comme valides à un instant t . Ce composant de validation C_j est destiné à être celui que l'utilisateur pourra activer avec un moyen de pointage, tel qu'une souris ou un écran tactile. La valeur V_j associée au composant correct C_j devient de ce fait une valeur correcte. Il est important de noter qu'avec ce tirage pseudo-aléatoire du composant correct et la génération pseudo-aléatoire de la valeur propre unique associée au cours de l'étape précédente à ce composant, seul le serveur S connaît pour le formulaire destiné à être transmis à l'utilisateur en réponse à la requête reçue au cours de l'étape E1, la valeur associée au bouton de validation activable par l'utilisateur et qui est celle attendue par le serveur S pour traiter les données du formulaire. Le composant correct C_j , choisi de manière pseudo-aléatoire, varie donc à chaque chargement de la page.

Les autres composants de validation $C_1, \dots, C_{j-1}, C_{j+1}, \dots, C_n$ deviennent alors des leurres destinés à tromper des bots. A noter que le composant de validation correct C_j n'est pas forcément le composant de validation visible par l'utilisateur du fait des différentes caractéristiques attribuables aux composants de validation de la page.

A noter que le caractère pseudo-aléatoire dans le choix du composant correct et dans le choix de la valeur associée aux composants de validation ne permet pas à un bot de s'appuyer sur l'une ou l'autre de ces informations lors d'une analyse à un instant t d'un formulaire afin d'utiliser ces informations lors du chargement ultérieur à un instant $t + 1$ de la page HTML associée au formulaire.

Dans une étape E7 de génération d'un code de positionnement, le serveur S génère pour le composant de validation correct C_j , et pour chacun des leurres, un code de positionnement HTML au moyen de feuilles de style css et de code javascript. Les codes des leurres et du composant de validation correct C_j sont générés de manière à positionner les leurres de façon à ce qu'une interaction entre l'utilisateur et le composant de validation correct C_j puisse avoir lieu. En particulier, le code des leurres et du composant correct C_j utilisent astucieusement l'ordre d'ajout de ces codes dans la sous-page. A noter que le code associé au composant de validation correct C_j peut être contraint à un certain emplacement. En effet, il peut être positionné systématiquement à un endroit correspondant à l'emplacement d'un composant de validation de la zone de validation à protéger du formulaire de référence. Pour autant, l'analyse de la sous-page ainsi générée n'en est pas plus facile. En effet, des leurres peuvent être positionnés au-dessus et/ou en dessous du composant de validation correct, complexifiant ainsi une analyse textuelle de la sous-pageHTML par un bot.

Dans une étape suivante E8 d'envoi du formulaire, le formulaire FORM d'identifiant IDU ainsi généré est envoyé au terminal T en réponse à la requête.

Dans une étape E9 de réception et d'affichage, le formulaire FORM est affiché sur l'écran du terminal T de l'utilisateur.

Dans une étape E10 d'action utilisateur, l'utilisateur remplit le cas échéant le formulaire de validation et sélectionne le composant de validation activable, noté C_k . Dans cet exemple de réalisation le composant de validation activable C_k correspond au composant de validation C_l visible par l'utilisateur. Dans un autre exemple de réalisation le bouton de validation activable C_k est différent du composant de validation visible C_l . Par exemple le composant activable C_k est transparent et positionné au-dessus du composant visible C_l .

Cela provoque dans une étape E11 d'envoi, l'envoi au serveur S de données du formulaire DATAFORM(data, IDU', V_k) comprenant des données data saisies par l'utilisateur le cas échéant, un identifiant IDU' du formulaire et la valeur V_k associée au composant de validation C_k activé par l'utilisateur.

Les données du formulaire sont reçues par le serveur S dans une étape E12 de réception.

Dans une étape E13 de comparaison, il est vérifié par le serveur S que le couple formé par l'identifiant IDU' du formulaire et la valeur V_k et reçu du terminal T est égal au couple formé par l'identifiant IDU et la valeur correcte V_j générée pour ce formulaire.

Dans un premier cas où la comparaison est positive (cas « ok » sur la figure 1), alors l'action est considérée comme valide et les données du formulaire sont traitées dans une étape suivante E14 de traitement.

Dans le cas contraire (cas « nok » sur la figure 1) alors l'action est considérée comme invalide et dans une étape suivante E15 de traitement d'une erreur, les données du formulaire sont refusées

et une mesure est mise en œuvre. Par exemple, il y a génération et envoi d'un nouveau formulaire, conformément aux étapes E2 de génération d'un identifiant à E9 d'envoi du formulaire telles que décrites précédemment. Dans un deuxième exemple de réalisation, le terminal T est suspecté d'attaque à l'encontre du serveur S et est mis en quarantaine. Dans un autre exemple de réalisation,

5 le terminal T est redirigé vers une page de validation au moyen d'un CAPTCHA classique.

Un exemple de code de la sous-page HTML « invention.html » tel que généré à partir des étapes décrites précédemment est fourni à la figure 3a, et une modélisation du rendu de cette sous-page est fourni à la figure associée 3b. Dans la figure 3b, la zone de validation à protéger du formulaire de référence est substituée par une sous-page HTML 30.

10 La sous-page 30 comprend une zone visible 11 et huit composants de validation de valeur notées valeur_1, ... , valeur_8. Chaque composant de validation comporte une information générale identique. Ainsi, aucun bouton ne comporte d'élément distinctif en dehors de son style css. Une seule de ces valeurs est considérée comme valide à un instant *t* et correspond au bouton activable par l'utilisateur avec un moyen de pointage, tel qu'une souris.

15 Sur la figure 3a, qui correspond au code HTML de la sous-page dont la modélisation du rendu est présentée à la figure 3b, le composant valide avec lequel l'utilisateur peut interagir correspond au composant de validation ayant pour valeur valeur_5 ; c'est un élément transparent actif situé en avant des autres composants. Ainsi, dans l'exemple des figure 3a et 3b, $V_{correcte} = \text{valeur_5}$. Tous les autres composants de validation, de par leur présence dans le code de la sous-page HTML

20 servent de leurres pour un programme automatisé destiné à analyser la page.

Les composants situés programmatiquement au-dessus sont en effet respectivement :

- désactivé et non visible pour le composant de valeur valeur_8 ;
- déplacé dynamiquement en dehors du champ visible pour le composant de valeur valeur_6, dès que l'utilisateur le sélectionne avec son moyen de pointage.

25 Les autres composants sont positionnés :

- programmatiquement en-dessous du composant de valeur correcte en utilisant la propriété css « z-index » ; c'est le cas du composant de valeur valeur_2. Bien que ce composant soit celui qui soit visible pour l'utilisateur, les composants positionnés devant lui étant transparents et actifs, ce n'est pas celui qui recevra les interactions ;

30 - implicitement en-dessous du composant de valeur valide ; c'est le cas des composants de valeur valeur_1 et valeur_7, visibles et actifs à l'instar du composant de valeur valeur_2. La propriété « z-index » est par défaut considérée comme étant égale à 1 par les moteurs de rendu HTML ;

- hors champ de manière statique grâce aux propriétés css : c'est le cas du composant de valeur valeur_3 ;

35 - en-dessous du composant de valeur valide du fait de leur ordre d'ajout dans la page HTML ;

c'est le cas du composant de valeur valeur_4.

Les techniques de positionnement utilisées, telles que le style css associé à un composant de validation, un code javascript ou l'ordre d'ajout des composants peuvent varier et être présentes alternativement ou simultanément dans une même sous-page HTML selon les itérations.

- 5 Ainsi, rien ne distingue un composant de validation d'un autre dans la sous-page hormis son style css. Un bot basique sera donc impuissant pour identifier le composant de validation actif lors d'une analyse textuelle d'une sous-page HTML. Cela est d'autant plus vrai que le contenu de la sous-page est généré pseudo-aléatoirement de telle façon que la distribution des composants de validation dans l'espace varie à chaque itération. Une analyse du code de la sous-page HTML du
- 10 formulaire pour un service donné ne peut donc permettre un remplissage automatique du formulaire à des fins frauduleuses. Par ailleurs, la perception qu'a l'utilisateur du formulaire est inchangée, le même formulaire chargé à différent instants ayant toujours le même aspect.

- Avec les étapes du procédé décrites précédemment, le processus de blocage d'un bot est totalement transparent pour l'utilisateur. Il n'est donc pas nécessaire d'ajouter un test supplémentaire
- 15 sous forme de CAPTCHA lors de la validation d'un formulaire afin de différencier un utilisateur d'un bot. La validation d'un formulaire est donc plus ergonomique du point de vue de l'utilisateur et simplifiée du point de vue traitements des formulaires.

- Dans un exemple de réalisation, le nombre de composants de validation compris dans le formulaire de validation ainsi généré est compris entre 4 et 8. Cet exemple n'est pas limitatif. Cependant,
- 20 un nombre trop important de composants de validation peut entraîner un mauvais rendu du formulaire au niveau du terminal utilisateur T. Un nombre trop faible peut par contre augmenter les chances d'un bot d'exploiter le formulaire. Des tests ont montré que l'utilisation de six composants de validation donnait de très bons résultats.

- Dans l'exemple décrit précédemment, la zone de validation à protéger dans le formulaire de référence est substituée par une nouvelle zone de validation représentée par une unique sous-page
- 25 HTML. Cet exemple est adapté pour pallier des attaques de bots basiques. Cependant, certains types de bots, qualifiés de « standards » utilisent un moteur de rendu HTML et ont le plus souvent la capacité d'exécuter un code javascript. Pour ces bots, le langage javascript offre une fonctionnalité permettant de récupérer dans une page HTML l'élément visible situé à des coordonnées données. En effet, le langage javascript dispose notamment d'une méthode « document.element-
- 30 FromPoint(x, y) » qui permet pour une page HTML donnée, de récupérer l'élément visible situé aux coordonnées (x, y) fournies. Une fois cet élément récupéré, il devient possible pour le bot d'extraire la valeur associée pour la soumettre au serveur S, ou encore de simuler de façon programmatique une interaction avec un formulaire de validation, tel qu'un clic souris.

- 35 Afin de pallier des attaques de bots standards, dans un deuxième exemple de réalisation, illustré

par les figures 4a et 4b, la zone de validation à protéger du formulaire de référence est substituée par au moins deux nouvelles zones de validation, correspondant chacune à une sous-fenêtre. La figure 4b est une modélisation du rendu du code HTML présenté à la figure 4a. Dans l'exemple de code HTML présenté ici, le code de la zone de validation à protéger du formulaire de référence est substitué par trois sous-pages HTML imbriquées. Dans cet exemple, chacune des sous-pages est une iFrame HTML qui comprend, en plus de la partie du formulaire HTML qui permet sa validation et des composants de validation générés selon les principes des étapes E4 de génération de composants de validation à E7 de génération d'un code de positionnement décrites précédemment, et de manière récursive, une autre sous-page générée selon les mêmes principes.

Cet ensemble de sous-pages imbriquées, chacune étant associée à une nouvelle zone de validation, dispose des propriétés suivantes :

- chaque sous-page possède un code permettant de soumettre le formulaire d'origine ;
 - les valeurs associées aux composants de validation, tels que les boutons HTML, des sous-pages sont uniques, une même valeur ne pouvant se retrouver dans différentes sous-pages ;
 - une seule de ces valeurs est considérée comme valide à un instant t ;
 - la seule valeur considérée comme valide à l'instant t est celle qui est accessible au moyen d'un périphérique de pointage, par exemple, une souris, un doigt sur un écran tactile, etc.
- A l'instar des composants de validation qui peuvent être représentés au sein d'une sous-page, chaque sous-page peut :
- être distribuée dans l'espace, en fonction de ses coordonnées x et y ;
 - être visible, transparente et/ou masquée ;
 - se déplacer dynamiquement, au chargement de la sous-page ou sur action de l'utilisateur, par exemple lors du déplacement de la souris.

Dans un exemple de réalisation, dans une étape de génération d'un décalage, non représentée sur la figure 1 et préalable à l'étape E7 de génération de code de positionnement, un décalage pseudo-aléatoire des coordonnées x et y des différentes sous-pages est généré tout en garantissant une intersection desdites sous-pages sur le plan (x, y) supérieure à la taille d'un composant de validation. Cela garantit de toujours pouvoir afficher un composant de validation tout en introduisant un décalage entre les différentes sous-pages.

Les sous-pages étant positionnables dynamiquement les unes par rapport aux autres, les coordonnées d'affichage réelles d'un composant sont donc dépendantes :

- de la position de ce composant dans sa sous-fenêtre ;
- de la position de cette sous-page par rapport aux sous-pages parentes, c'est-à-dire celles dans lesquelles elle est imbriquée.

A noter que la profondeur d'une sous-page est prédéfinie et correspond à l'ordre d'inclusion des

sous-pages : la première sous-page est positionnée au premier plan, la deuxième vient se positionner au-dessous, la troisième au-dessous de la deuxième, etc.

Ainsi, dans l'exemple de la figure 4b, correspondant à une modélisation du rendu du code HTML présenté à la figure 4a, trois sous-fenêtres 40, 41 et 42 correspondant aux trois sous-pages HTML de la figure 4a sont visibles. La première sous-fenêtre 40 apparaît derrière les deux autres sous-fenêtres, conformément à l'ordre d'inclusion des sous-pages HTML. La première sous-fenêtre 40 comprend les composants de validation associés aux valeurs valeur__1 et valeur__2. La deuxième sous-fenêtre 41 comprend les composants de validation associés aux valeurs valeur__3 et valeur__4 et enfin, la troisième sous-fenêtre 42, qui apparaît au-dessus des deux autres car imbriquée en dernier, comprend les composants de validation associés aux valeurs valeur__5 et valeur__6. A noter que certains composants de validation ne sont pas visibles, car situés en dehors de la zone visible de l'utilisateur. Dans cet exemple de réalisation, les coordonnées (x, y) des deuxième et troisième sous-page 41, 42 ont fait l'objet d'un léger décalage généré dans les sous-pages HTML conformément aux instructions « translate(-150%, 50%) ». Dans l'exemple de la figure 4a et de la figure 4b associée, le composant de valeur valeur__4 est celui qui est activable par l'utilisateur.

La profondeur de récursivité peut être variable selon la complexité souhaitée. Cependant une valeur trop faible, c'est-à-dire inférieure à trois, peut rendre le problème d'identification du composant de validation correct trop simple pour un bot standard, alors qu'une valeur trop élevée peut induire une consommation de ressources importante au niveau du terminal T et des problèmes d'affichage. A titre d'exemple, et suite à différents tests, une valeur égale à six est adaptée.

L'avantage procuré par ce deuxième mode de réalisation est que chaque sous-page étant considérée par le langage javascript comme un espace cloisonné, il n'est pas possible de travailler sur un rendu global d'une combinaison de ces sous-pages. Le langage javascript nécessite à l'heure actuelle de parcourir chacune des sous-pages pour accéder à ses éléments. La conséquence est qu'une méthode du type « document.elementFromPoint(x, y) », exploitable dans le premier mode de réalisation pour déterminer le composant visible aux coordonnées fournies, devient inopérante dans ce deuxième mode de réalisation. En effet, une telle méthode ne retourne que le composant positionné dans la sous-page considérée, et non le composant résultant de la combinaison des sous-pages. La position (x, y) fournie sera celle de la sous-page considérée et non celle de l'espace visible de l'utilisateur.

Ainsi, ce deuxième mode de réalisation est efficace pour éviter à des bots standards, et a fortiori à des bots basiques, de déterminer le composant accessible par l'utilisateur et la valeur valide associée.

Cependant, certains bots, dits « avancés », sont capables de simuler des actions utilisateurs,

comme des frappes de clavier, des déplacements ou des clics de souris. Ces actions sont strictement équivalentes d'un point de vue système ou logiciel à celles générées par de véritables périphériques d'entrée et ne peuvent être éventuellement distinguées que par une analyse comportementale, connue pour soulever des problèmes de préservation de la vie privée. De tels bots peuvent ainsi simuler une action de la souris en se basant uniquement sur la position du curseur, sans interagir ni analyser le contenu HTML généré. Ainsi, si un composant de validation est systématiquement positionné à une position (x, y) de l'écran qui constitue la zone visible de l'utilisateur, alors un bot avancé n'a qu'à simuler une action de la souris sur cette position pour reproduire le comportement humain. Il peut ainsi exploiter des formulaires d'authentification et perpétrer une attaque par dictionnaire, exploiter des formulaires de contact et envoyer du spam, etc.

Afin de contrer ces bots avancés sans avoir à mettre en œuvre une analyse comportementale pour les distinguer d'un utilisateur réel, il est introduit dans un troisième mode de réalisation un caractère aléatoire dans le positionnement du composant de validation visible de l'utilisateur. Ainsi, dans ce troisième mode de réalisation, la taille de la zone visible d'une nouvelle zone de validation possède une surface supérieure à celle du composant de validation visible et ce composant possède une position qui varie à l'intérieur de cette zone de manière aléatoire, d'une génération de formulaire à une autre.

La nouvelle zone de validation est générée conformément aux étapes E3 de génération d'au moins une nouvelle zone de validation à E6 de sélection d'un composant correct. Dans l'étape E7 de génération d'un code, la génération du code associé au composant visible comprend en outre une sélection pseudo-aléatoire de la position du composant de validation parmi les positions possibles dans la zone visible. Le formulaire envoyé au terminal T comprend ainsi ce composant de validation qui peut, d'un chargement de formulaire à un autre, être positionné à un endroit différent dans la zone visible. La zone visible restant circonscrite, cela ne perturbe pas l'utilisateur outre mesure. Dans un premier exemple de réalisation, ce composant visible est aussi celui qui est activable par l'utilisateur. Dans un deuxième exemple de réalisation, le composant visible est associé à un composant activable, par exemple un composant transparent, positionné systématiquement au même endroit que le composant visible, en l'occurrence au-dessus du composant visible.

Dans un exemple de réalisation de ce troisième mode de réalisation, la taille de la zone visible est un multiple de la taille du composant de validation tel que visible par l'utilisateur. Ce multiple peut aller de un jusqu'à un nombre qui peut être important. Une valeur trop importante peut cependant avoir un impact négatif sur la visibilité et l'ergonomie pour l'utilisateur. En pratique, une surface visible égale à neuf fois la taille du composant de validation à afficher paraît raisonnable. Elle peut être réduite afin d'améliorer l'ergonomie, ou augmenter afin d'améliorer la sécurité. Ce mode de réalisation peut paraître moins agréable et/ou esthétique pour l'utilisateur qui peut constater, lors de différents accès au serveur, des modifications dans l'affichage d'un formulaire du

fait de variations dans la position d'un composant de validation. Cependant, il est très efficace pour lutter contre des bots.

Ce troisième mode de réalisation est illustré par la figure 5 qui représente un formulaire 50 généré conformément aux étapes du procédé selon le troisième mode de réalisation. Dans cet exemple, la taille de la zone visible 501 des nouvelles zones de validation 502 et 503 est égale à neuf fois la taille du composant de validation 504 visible affiché. Le formulaire comprend par ailleurs des éléments du formulaire de référence non modifiés par les étapes du procédé, en l'espèce les zones de saisies 505 et 506.

Aux nouvelles zones de validation 502 et 503 de la figure 5 sont associées deux sous-pages HTML imbriquées de manière récursives et associées au formulaire. Dans un autre exemple de réalisation, et à l'instar du premier mode de réalisation décrit précédemment, une seule sous-page HTML est imbriquée dans le formulaire.

Dans un exemple de réalisation, dans une sous-étape de mémorisation (non représentée sur la figure 1) de l'étape E11 de réception et d'affichage, des actions éventuelles à côté du composant de validation, par exemple des clics souris sur une zone vide de la zone visible, sont mémorisées pour être envoyées avec les données du formulaire au cours de l'étape E11 d'envoi et analysées par le serveur S. Ainsi, dans une sous-étape d'analyse (non représentée sur la figure 1) de l'étape E14 de traitement, le serveur S analyse les actions mémorisées au niveau du terminal T et associées au formulaire. Une telle analyse permet de distinguer un automate, qui effectue des tests multiples dans la zone visible jusqu'à cliquer statistiquement sur le composant de validation, d'un être humain accédant immédiatement ou avec un nombre d'erreurs très restreint au composant de validation.

Un serveur S, apte à mettre en œuvre les étapes du procédé de génération d'un formulaire de validation à partir d'un formulaire de référence et d'envoi à un terminal va maintenant être décrit en relation avec la figure 6.

Le serveur S est un équipement informatique, accessible depuis un réseau de données tel qu'Internet. Il comprend :

- une unité de traitement 61, ou CPU (de l'anglais « Central Processing Unit »), destinée à charger des instructions en mémoire, à les exécuter, à effectuer des opérations ;
- un ensemble de mémoires, dont une mémoire volatile 62, ou RAM (pour « Random Access Memory ») utilisée pour exécuter des instructions de code, stocker des variables, etc., et une mémoire de stockage 63 de type EEPROM (de l'anglais « Electronically-Erasable Programmable Read-Only Memory »). En particulier, la mémoire de stockage 63 est agencée pour mémoriser un module logiciel de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal, tel que décrit précédemment.

Le serveur S comprend également :

- un module de substitution 64, agencé pour générer au moins une nouvelle zone de validation et pour substituer une zone de validation à protéger du formulaire de référence par la au moins une nouvelle zone de validation, ladite nouvelle zone de validation comprenant au moins deux composants de validation. Le module de substitution 64 est agencé pour mettre en œuvre l'étape E3 du procédé de génération d'un formulaire et d'envoi à un terminal tel que décrit précédemment ;
 - un premier module de génération 65, agencé pour générer une valeur unique propre auxdits au moins deux composants de validation. Le premier module de génération 65 est agencé pour mettre en œuvre l'étape E4 de génération de composants de validation du procédé de génération d'un formulaire et d'envoi tel que décrit précédemment ;
 - un module de sélection 66, agencé pour sélectionner l'un desdits composants de validation comme composant correct, les autres desdits au moins un composant devenant des leurres. Le module de sélection 66 est agencé pour mettre en œuvre l'étape E6 de sélection d'un composant correct du procédé de génération d'un formulaire et d'envoi tel que décrit précédemment ;
 - un deuxième module de génération 67, agencé pour générer un code de positionnement dans la nouvelle zone de validation pour le composant correct et pour les autres desdits au moins un leurre. Le code de positionnement d'un leurre permet au composant correct d'être activé par l'utilisateur. Le deuxième module de génération 67 est agencé pour mettre en œuvre l'étape E7 de génération d'un code, du procédé de génération d'un formulaire et d'envoi tel que décrit précédemment ;
 - un module d'envoi 68, agencé pour envoyer au terminal T le formulaire comprenant la nouvelle zone de validation. Le module d'envoi 68 est agencé pour mettre en œuvre l'étape E8 d'envoi du procédé de génération d'un formulaire et d'envoi tel que décrit précédemment.
- Le module de substitution 64, le premier module de génération 65, le module de sélection 66, le deuxième module de génération 67 et le module d'envoi 68 sont de préférence des modules logiciels comprenant des instructions logicielles pour mettre en œuvre les étapes du procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal qui sont mises en œuvre par le serveur S.

L'invention concerne donc également :

- un programme d'ordinateur comportant des instructions de code pour la mise en œuvre du procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal tel que décrit précédemment et mises en œuvre par le serveur S lorsque ce programme est exécuté par un processeur du serveur S, et
- un support d'enregistrement lisible sur lequel est enregistré le programme d'ordinateur ci-dessus.

Revendications

[Revendication 1] Procédé de génération d'un formulaire à partir d'un formulaire de référence et d'envoi à un terminal (T), ledit formulaire de référence comprenant au moins une zone de validation à protéger, la génération du formulaire comprenant les étapes suivantes :

- substitution (E3) de ladite zone de validation à protéger par au moins une nouvelle zone de validation, ladite nouvelle zone de validation comprenant au moins deux composants de validation,
- génération (E4) d'une valeur unique propre auxdits au moins deux composants de validation,
- sélection (E6) d'un desdits composants de validation comme composant correct, les autres desdits au moins un composant devenant des leurres,
- génération (E7) d'un code de positionnement dans la nouvelle zone de validation pour le composant correct et pour les autres desdits au moins un leurre, ledit code de positionnement permettant au composant correct d'être activé par l'utilisateur,
- envoi (E8) au terminal (T) du formulaire comprenant la nouvelle zone de validation.

[Revendication 2] Procédé de génération d'un formulaire et d'envoi à un terminal selon la revendication 1, comprenant :

- réception (E12) en provenance du terminal (T) d'une valeur de sélection, ladite valeur de sélection étant la valeur unique propre au composant de validation sélectionné par l'utilisateur,
- lorsque la valeur de sélection est égale à la valeur propre au composant de validation correct, traitement (E14) des données du formulaire.

[Revendication 3] Procédé de génération d'un formulaire et d'envoi à un terminal selon la revendication 1 ou la revendication 2, dans lequel la sélection du composant de validation correct et de sa valeur unique propre parmi les au moins deux composants de validation est pseudo-aléatoire.

[Revendication 4] Procédé de génération d'un formulaire de validation et d'envoi à un terminal selon l'une des revendications précédentes dans lequel un desdits au moins deux composants de validation peut être dans un état compris dans un ensemble d'états comprenant : superposé à un autre desdits au moins un composant, distribués dans l'espace, visible, transparent, masqué, apte à se déplacer dynamiquement lors du chargement du formulaire, apte à se déplacer dynamiquement sur une action de l'utilisateur.

[Revendication 5] Procédé de génération d'un formulaire et d'envoi à un terminal selon l'une des revendications précédentes, dans lequel le formulaire de référence étant constitué d'une

page HTML, la nouvelle zone de validation est constituée d'une sous-page HTML imbriquée dans la page HTML associée au formulaire de référence sous forme d'une balise iFrame.

[Revendication 6] Procédé de génération d'un formulaire et d'envoi à un terminal selon l'une des revendications 1 à 4, dans lequel le formulaire de référence étant constitué d'une page HTML, ledit formulaire de référence comprend une balise HTML spécifique, associée à une librairie javascript, ladite balise spécifique étant remplacée par un code substituant la zone de validation par la nouvelle zone de validation.

[Revendication 7] Procédé de génération d'un formulaire et d'envoi à un terminal selon l'une des revendications précédentes, dans lequel le formulaire comprend au moins deux nouvelles zones de validation, lesdites nouvelles zones de validation étant cloisonnées les unes par rapport aux autres.

[Revendication 8] Procédé de génération d'un formulaire et d'envoi à un terminal selon la revendication précédente, dans lequel lesdites au moins deux nouvelles zones de validation étant sous forme d'au moins deux sous-pages, le procédé comprend la génération d'un décalage pseudo-aléatoire des abscisses et des ordonnées entre les au moins deux sous-pages, la génération du décalage garantissant une intersection entre une zone visible de l'utilisateur et les au moins deux sous-pages supérieure à la taille d'un composant de validation.

[Revendication 9] Procédé de génération d'un formulaire et d'envoi à un terminal selon l'une des revendications précédentes dans lequel la nouvelle zone de validation comprend une zone visible par l'utilisateur, ladite zone visible étant de taille égale à un multiple de la taille d'un composant de validation visible, ledit composant de validation visible étant positionné de manière pseudo-aléatoire dans la nouvelle zone de validation.

[Revendication 10] Serveur (S) apte à générer un formulaire de validation à partir d'un formulaire de référence et à l'envoyer à un terminal (T), ledit formulaire de référence comprenant au moins une zone de validation à protéger, le serveur comprenant :

- des moyens de substitution (64), agencés pour substituer ladite zone de validation à protéger par au moins une nouvelle zone de validation, ladite nouvelle zone de validation comprenant au moins deux composants de validation,
- des premiers moyens de génération (65), agencés pour générer une valeur unique propre auxdits au moins deux composants de validation,
- des moyens de sélection (66), agencés pour sélectionner un desdits composants de validation comme composant correct, les autres desdits au moins un composant devenant des leurres,

- des deuxièmes moyens de génération (67), agencés pour générer un code de positionnement dans la nouvelle zone de validation pour le composant correct et pour les autres desdits au moins leurre, le code de positionnement permettant au composant correct d'être activé par l'utilisateur,
- des moyens d'envoi (68), agencés pour envoyer au terminal (T) le formulaire comprenant la nouvelle zone de validation.

[Revendication 11] Programme d'ordinateur sur un support de données et chargeable dans la mémoire d'un ordinateur, le programme comprenant des portions de code pour exécuter les étapes du procédé de génération d'un formulaire de validation et d'envoi à un terminal selon l'une des revendications 1 à 9, lorsque le programme est exécuté sur ledit ordinateur.

[Revendication 12] Support de données dans lequel est enregistré le programme selon la revendication 11.

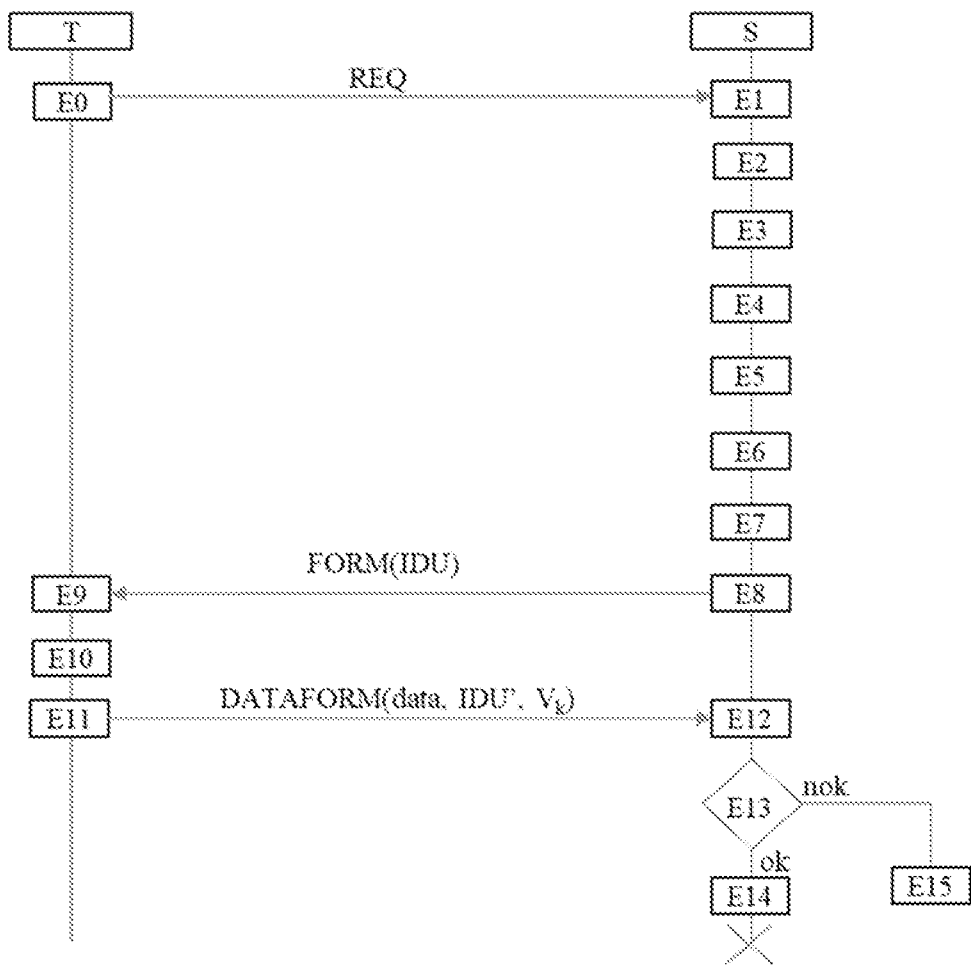


FIG. 1

Code_ init.	<pre><form id="myForm"> <input id="mydata" name="data" value="please enter data"> <input type="submit" value="Envoyer !"> </form></pre>
Code_ modif.	<pre><form id="myForm"> <input id="mydata" name="data" value="please enter data"> <iframe scr="/invention.html"></iframe> </form></pre>

FIG. 2

```

<html>
<head>
</head>
<body>
  <form action="post" URL="//serveur_de_validation/page_de_validation">
    <button type="submit" name="login" value="valeur_1">login</button>
    <button type="submit" name="login" value="valeur_2" style="z-index:10;">login</button>
    <button type="submit" name="login" value="valeur_3" style="z-index:20; transform: translate(-150%, -50%);">login</button>
    <button type="submit" name="login" value="valeur_4" style="z-index:21;filter:alpha(opacity=0); opacity:0.0;">login</button>
    <button type="submit" name="login" value="valeur_5" style="z-index:21;filter:alpha(opacity=0); opacity:0.0;">login</button>
    <button type="submit" name="login" value="valeur_6" style="z-index:25;filter:alpha(opacity=0); opacity:0.0;"
      onmouseover="this.style.transform='translate(-150%, 250%)'">login</button>
    <button type="submit" name="login" value="valeur_7">login</button>
    <button type="submit" name="login" value="valeur_8" style="z-index:30;display:none;">login</button>
  </form>
</body>
</html>

```

FIG. 3a

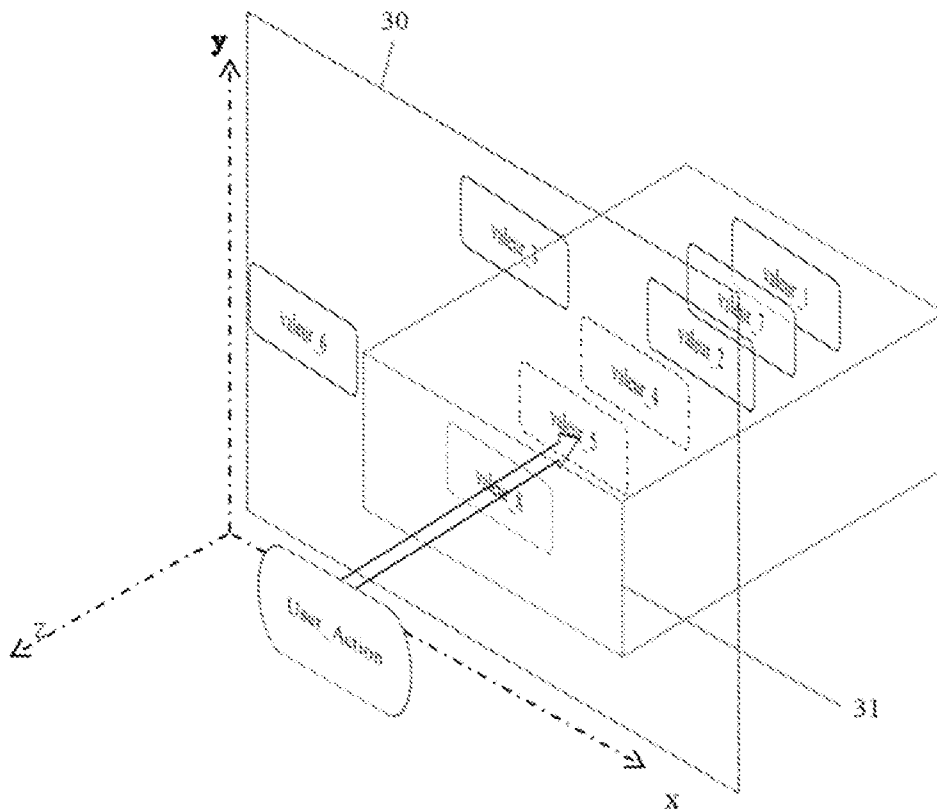


FIG. 3b

```

<html>
<head></head>
<body>
<form action="post" URL="//serveur_de_validation/page_de_validation">
  <button type="submit" name="login" value="valeur_1" style="z-index:10;">login</button>
  <button type="submit" name="login" value="valeur_2" style="z-index:20; transform: translate(-150%, -50%);">login</button>
</form>
<iframe>
  <html>
  <head></head>
  <body style="transform: translate(-150%, -50%);">
    <form action="post" URL="//serveur_de_validation/page_de_validation">
      <button type="submit" name="login" value="valeur_5" style="z-index:20; transform: translate(150%, 50%);">login</button>
      <button type="submit" name="login" value="valeur_6">login</button>
    </form>
    <iframe></iframe>
  </body>
  </html>
</iframe>
</body>
</html>
</iframe>
</body>
</html>

```

FIG. 4a

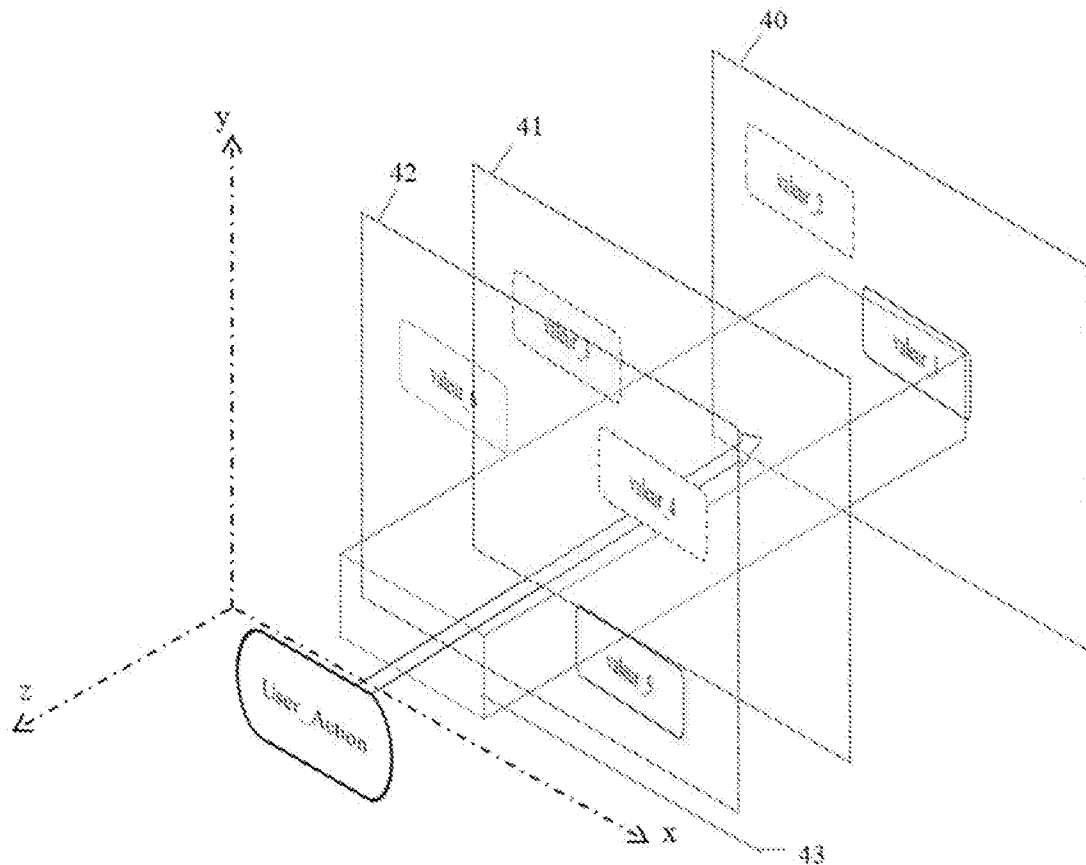


FIG. 4b

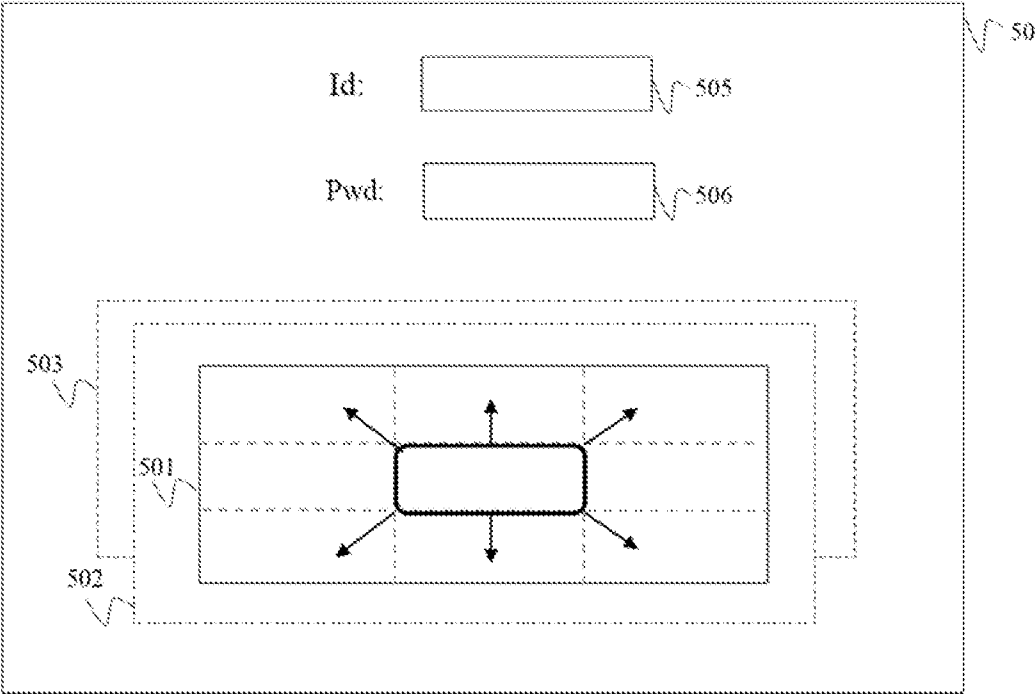


FIG. 5

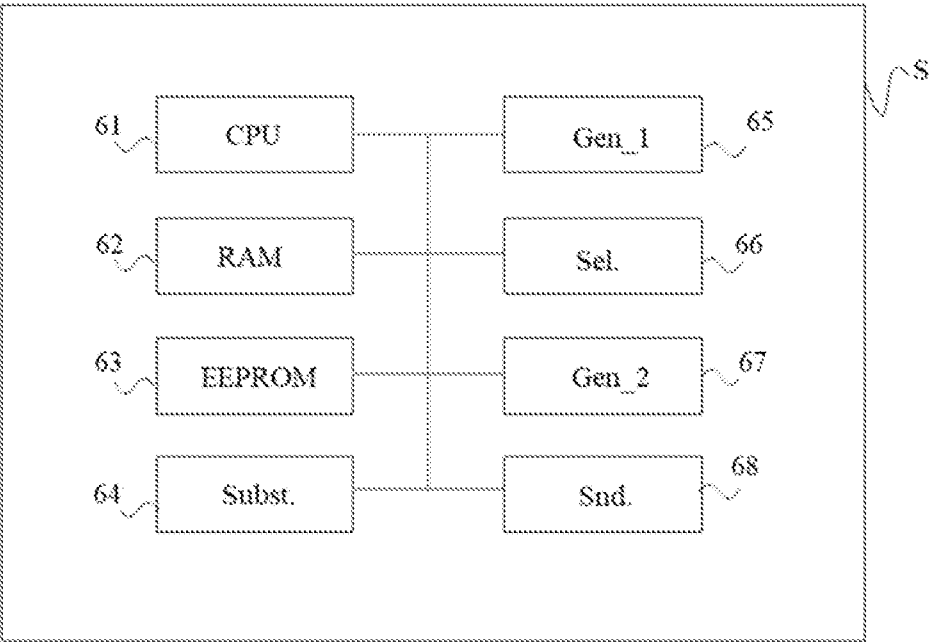


FIG. 6

INTERNATIONAL SEARCH REPORT

International application No.

PCT/FR2021/052316

A. CLASSIFICATION OF SUBJECT MATTER**G06F 21/36**(2013.01)i; **G06F 21/31**(2013.01)i; **H04W 12/06**(2021.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F; H04L; H04W

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	VIKRAM SHARDUL ET AL. "NOMAD: Towards non-intrusive moving-target defense against web bots" <i>2013 IEEE CONFERENCE ON COMMUNICATIONS AND NETWORK SECURITY (CNS)</i> , <i>IEEE</i> , 14 October 2013 (2013-10-14), pages 55-63 DOI: 10.1109/CNS.2013.6682692 XP032529046	1-6,9-12
Y	abstract sections I, III-V	7,8
Y A	US 2019373012 A1 (MATTSON ELI [US] ET AL) 05 December 2019 (2019-12-05) abstract paragraph [0002] - paragraph [0007] paragraph [0134] - paragraph [0145] paragraph [0064] - paragraph [0075]	7,8 1-6,9-12
A	US 2017034148 A1 (RAPAPORT SHAY [IL] ET AL) 02 February 2017 (2017-02-02) abstract paragraph [0001] - paragraph [0010] paragraph [0019] - paragraph [0045]	1-12

☐ Further documents are listed in the continuation of Box C.☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

28 March 2022

Date of mailing of the international search report

07 April 2022

Name and mailing address of the ISA/EP

European Patent Office
p.b. 5818, Patentlaan 2, 2280 HV Rijswijk
Netherlands

Telephone No. (+31-70)340-2040

Facsimile No. (+31-70)340-3016

Authorized officer

Jakob, Gregor

Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/FR2021/052316

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
US	2019373012	A1	05 December 2019	EP	3414695	A1	19 December 2018
				US	2017237766	A1	17 August 2017
				US	2019373012	A1	05 December 2019
				WO	2017139709	A1	17 August 2017
US	2017034148	A1	02 February 2017	EP	2673708	A2	18 December 2013
				ES	2679286	T3	23 August 2018
				US	2014250538	A1	04 September 2014
				US	2017034148	A1	02 February 2017
				WO	2012107879	A2	16 August 2012

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°

PCT/FR2021/052316

A. CLASSEMENT DE L'OBJET DE LA DEMANDE INV. G06F21/36 G06F21/31 H04W12/06 ADD.		
Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB		
B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTE Documentation minimale consultée (système de classification suivi des symboles de classement) G06F H04L H04W		
Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche		
Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si cela est réalisable, termes de recherche utilisés) EPO-Internal		
C. DOCUMENTS CONSIDERES COMME PERTINENTS		
Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	VIKRAM SHARDUL ET AL: "NOMAD: Towards non-intrusive moving-target defense against web bots", 2013 IEEE CONFERENCE ON COMMUNICATIONS AND NETWORK SECURITY (CNS), IEEE, 14 octobre 2013 (2013-10-14), pages 55-63, XP032529046, DOI: 10.1109/CNS.2013.6682692	1-6, 9-12
Y	abrégé Sections I, III-V	7, 8
Y	US 2019/373012 A1 (MATTSON ELI [US] ET AL) 5 décembre 2019 (2019-12-05)	7, 8
A	abrégé alinéa [0002] - alinéa [0007] alinéa [0134] - alinéa [0145] alinéa [0064] - alinéa [0075]	1-6, 9-12
	----- -/-	
☒ Voir la suite du cadre C pour la fin de la liste des documents ☒ Les documents de familles de brevets sont indiqués en annexe		
* Catégories spéciales de documents cités: "A" document définissant l'état général de la technique, non considéré comme particulièrement pertinent "E" document antérieur, mais publié à la date de dépôt international ou après cette date "L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée) "O" document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens "P" document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée "T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention "X" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément "Y" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier "&" document qui fait partie de la même famille de brevets		
Date à laquelle la recherche internationale a été effectivement achevée 28 mars 2022		Date d'expédition du présent rapport de recherche internationale 07/04/2022
Nom et adresse postale de l'administration chargée de la recherche internationale Office Européen des Brevets, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016		Fonctionnaire autorisé Jakob, Gregor

C(suite). DOCUMENTS CONSIDERES COMME PERTINENTS		
Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
A	US 2017/034148 A1 (RAPAPORT SHAY [IL] ET AL) 2 février 2017 (2017-02-02) abrégé alinéa [0001] - alinéa [0010] alinéa [0019] - alinéa [0045] -----	1-12

RAPPORT DE RECHERCHE INTERNATIONALE

Renseignements relatifs aux membres de familles de brevets

Demande internationale n°

PCT/FR2021/052316

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
US 2019373012 A1	05-12-2019	EP 3414695 A1	19-12-2018
		US 2017237766 A1	17-08-2017
		US 2019373012 A1	05-12-2019
		WO 2017139709 A1	17-08-2017

US 2017034148 A1	02-02-2017	EP 2673708 A2	18-12-2013
		ES 2679286 T3	23-08-2018
		US 2014250538 A1	04-09-2014
		US 2017034148 A1	02-02-2017
		WO 2012107879 A2	16-08-2012
