УКРАЇНА

(19) **UA** (11) **122244** (13) **C2**

(51) МПК
H04L 9/30 (2006.01)

МІНІСТЕРСТВО РОЗВИТКУ ЕКОНОМІКИ, ТОРГІВЛІ ТА СІЛЬСЬКОГО ГОСПОДАРСТВА УКРАЇНИ

(12) ОПИС ДО ПАТЕНТУ НА ВИНАХІД

(21) Номер заявки: **a 2018 04726**

(22) Дата подання заявки: **28.09.2016**

(24) Дата, з якої є чинними права інтелектуальної власності: **13.10.2020**

(31) Номер попередньої заявки відповідно до Паризької конвенції: **14/925,769, 14/931,613**

(32) Дата подання попередньої заявки відповідно до Паризької конвенції: **28.10.2015, 03.11.2015**

(33) Код держави-учасниці Паризької конвенції, до якої подано попередню заявку: **US, US**

(41) Публікація відомостей про заявку: **27.08.2018, Бюл.№ 16**

(46) Публікація відомостей про державну реєстрацію: **12.10.2020, Бюл.№ 19**

(86) Номер та дата подання міжнародної заявки, поданої відповідно до Договору PCT: **PCT/US2016/054186, 28.09.2016**

(72) Винахідник(и):
Ні Мінь (US)

(73) Володілець (володільці):
Ні Мінь,
1050 Creekdale Dr., Clarkston, Georgia 30021, United States of America (US)

(74) Представник:
Кістерський Тимофій Арсенійович, реєстр. №457

(56) Перелік документів, взятих до уваги експертизою:
WO 2013163285 A1, 31.10.2013
US 2013021249 A1, 24.01.2013
US 2015121493 A1, 30.04.2015
US 2011167102 A1, 07.07.2011

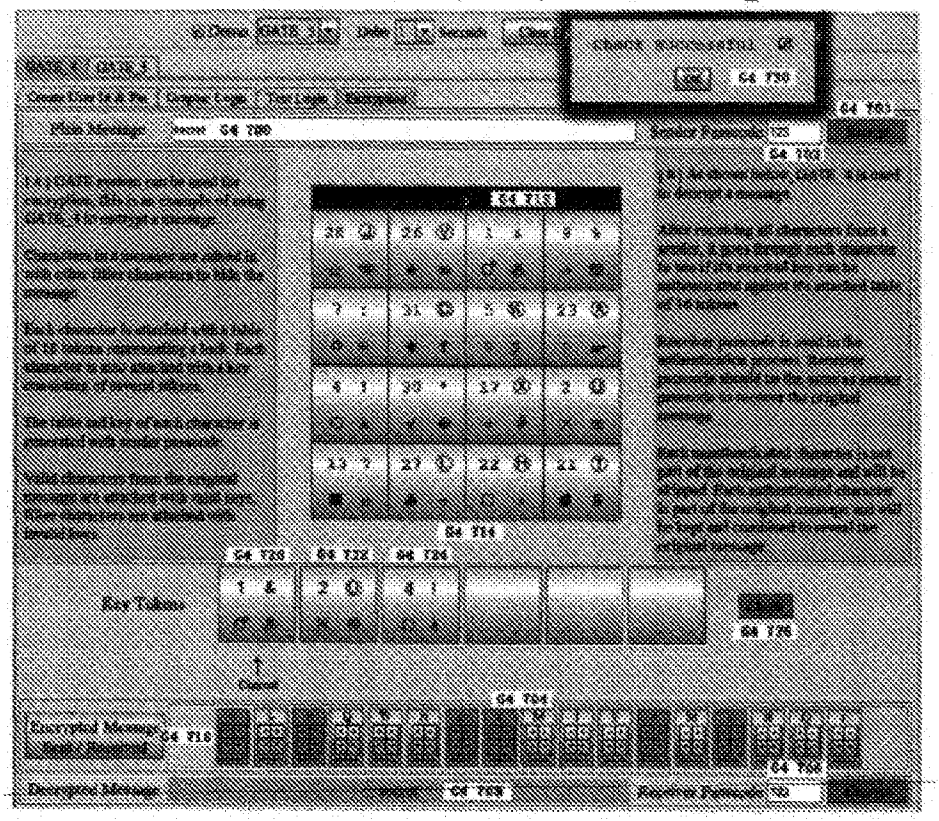
UA 122244 C2

(54) СИСТЕМА ТА СПОСІБ АУТЕНТИФІКАЦІЇ ТА ШИФРУВАННЯ ІЗ ЗАХИСТОМ ВІД ПЕРЕХОПЛЕННЯ

(57) Реферат:

Запропоновані система та спосіб для аутентифікації/шифрування із захистом від перехоплення з використанням кодів доступу з окремими пінами, що складаються з символів із сукупності символів, і токенів, що містять щонайменше два символи із зазначеної сукупності символів, використаних для коду доступу. Користувачу представляють численні токени ("сукупність токенів"), при цьому деякі або всі з попередньо вибраних пінів користувача (символів) довільно введені у деякі або всі з токенів. Користувач вибирає токен із сукупності токенів для кожного положення піна у коді доступу. Користувача аутентифікують на підставі вибраних токенів. Оскільки кожний вибраний токен може містити або може і не містити один із попередньо вибраних пінів у коді доступу користувача, а також може містити інші довільно згенеровані символи, які не є жодним із попередньо вибраних пінів у коді доступу користувача, хто-небудь, хто побачив те, які токени вибрав користувач, не зможе визначити діючий код доступу користувача.

Шифрування для варіанта реалізації «GATE_4»



Фіг. 15A

ОБЛАСТЬ ТЕХНІКИ

[1] Даний винахід відноситься до систем і способів аутентифікації та шифрування, зокрема до систем і способів аутентифікації та шифрування із захистом від перехоплення.

РІВЕНЬ ТЕХНІКИ

[2] В сучасному суспільстві у повсякденному житті необхідно використовувати, серед іншого, широку множину різних інформаційних пристроїв, таких як мобільні телефони, персональні комп'ютери, ноутбуки та банкомати. На інформаційних пристроях можуть бути збережені персональні дані користувача. Внаслідок того, що захист цих персональних даних є важливим завданням, існують способи безпечного блокування та розблокування цих пристроїв.

[3] У даний час найбільше широко використовуваний спосіб блокування та розблокування цих пристроїв являє собою процедуру аутентифікації викликів на основі паролів, при цьому для розпізнавання особистості пристрій зазвичай потребує від користувачів ввести ідентифікатор користувача та пароль ще до одержання ними доступу до своїх сервісів. Такий спосіб відомий як реєстрація входу. Процес реєстрації входу призначений для запобігання крадіжки персональних даних користувача або їх зміни шляхом обману.

[4] При швидкому щоденному зростанні мережевого покриття та доступності мережі хакерам стає все більш зручніше робити своєю мішенню паролі користувачів для одержання доступу до їхньої особистої інформації. Крім того, хакери стають все більше і більше витонченими при вгадуванні та зламуванні паролів користувача. Таким чином, прості паролі більше не забезпечують достатній захист від кібезагроз і кібершпіонажу.

[5] Через це для поліпшення захисту були розроблені різні механізми. Наприклад, користувачам необхідно створити пароль, що відповідає вимогам за довжиною пароля, складності та неможливості вгадування, так що в теорії пароль повинен мати силу, достатню для запобігання атак, заснованих на методі пошуку перебором, й атак, заснованих на методі перебору за словником. Крім того, користувачам необхідно періодично змінювати свої паролі для анулювання старих паролів, що зменшує ймовірність їх зламування. Ці механізми дійсно покращують безпеку до конкретного ступеня, що допомагає користувачам захистити свої акаунти.

[6] Однак, кожна організація може мати іншу сукупність правил створення пароля. Відповідно до деяких таких правил необхідна довжина пароля становить щонайменше 6 або 8 знаків. Деякі такі правила потребують використання комбінації великих і малих літер, а також номерів. Деякі такі правила потребують використання щонайменше одного спеціального знака, а деякі інші правила не допускають використання спеціальних знаків, так що коли ви думаєте, що ви тільки що створили дуже сильний пароль, який ви можете скрізь використовувати, з'явиться інше місце, що має іншу сукупність вимог, згідно з якими ваш чудовий пароль буде недійсним.

[7] В результаті існування різних правил створення пароля, може бути складно, якщо не неможливо, користувачам запам'ятати численні паролі, які вони встановили на різних сайтах/в різних організаціях. Таким чином, користувачі зазвичай зберігають свої паролі, наприклад, у файлі, збереженому на своєму пристрої зберігання інформації та/або додатку для зберігання паролів, що запускається на своєму пристрої зберігання інформації. Збережені паролі можуть стати метою хакерів, а якщо вони одержать доступ до пристрою, на якому збережені паролі, то вони одержують доступ до всіх паролів і всіх захищених паролем акаунтів/сайтів користувача. Таким чином, реалізація суворих правил відносно паролів для того, щоб уникнути використання занадто слабких паролів, може набути зворотній ефект щодо очікуваного ефекту (підвищений ризик розкриття додаткової інформації).

[8] У спробі вирішити вищеописані проблеми, пов'язані зі звичайними паролями, були розроблені нові способи. Ці способи можуть включати, крім іншого, використання фотографій, графічних зображень або різних форм і відтінків для ускладнення хакерам завдання із здійснення зчитування або крадіжки. Для валідації надання користувачу доступу в деяких технологіях навіть використовують жести та розміщення інформації в конкретних місцях екрана для введення даних. Однак, жоден із цих способів не може обійти сховану камеру, яка може записувати кожний крок користувачів кожний раз, коли вони реєструють вхід у пристрій. Якщо хакер може відтворити всю записану інформацію та може проаналізувати кожний крок користувача, цей хакер в остаточному підсумку одержить доступ.

[9] Основними проблемами відомих способів аутентифікації є:

(1) Звичайні паролі та контрольні питання (найбільше широко використовуваний спосіб) не захищені від зчитування;

(2) Способи на основі графічних зображень та фотографій можуть зажадати завантаження користувачем файлу зображення або фотографії, а система повинна зберігати та підтримувати

такі зображення та/або фотографії. Це збільшує навантаження на користувача та систему, а якщо хакери запишуть та відтворять процес реєстрації входу, зображення все ще можуть бути розпізнані;

(3) Нові способи аутентифікації на основі графічних зображень, жестів і/або місць розташування можуть бути використані тільки між людиною та комп'ютером, але не можуть бути використані між двома машинами.

[10] Таким чином, існує потреба в створенні систем і способів аутентифікації та шифрування, які не мають вищеописаних проблем.

РОЗКРИТТЯ СУТНОСТІ ВИНАХОДУ

[11] Завдання даного винаходу полягає у подоланні щонайменше вищеописаних проблем/недоліків і в забезпеченні щонайменше нижчеописаних переваг.

[12] Таким чином, завдання даного винаходу полягає в створенні системи та способу для аутентифікації користувача.

[13] Ще одне завдання даного винаходу полягає в створенні системи та способу для аутентифікації користувача, що намагається одержати доступ до електронного пристрою.

[14] Ще одне завдання даного винаходу полягає в створенні системи та способу для аутентифікації користувача, що запитує доступ до збереженої в електронному вигляді інформації.

[15] Ще одне завдання даного винаходу полягає в створенні системи та способу для аутентифікації користувача, що запитує доступ до пристрою за мережею.

[16] Ще одне завдання даного винаходу полягає в створенні системи та способу для аутентифікації користувача, що використовує коди доступу, які містять задану кількість символів.

[17] Ще одне завдання даного винаходу полягає в створенні системи та способу для аутентифікації користувача, що використовує численні токени, кожний з яких являє собою групу щонайменше з двох символів із сукупності символів, використаних для створення коду доступу користувача.

[18] Ще одне завдання даного винаходу полягає в створенні системи та способу для шифрування та дешифрування електронної інформації.

[19] Ще одне завдання даного винаходу полягає в створенні системи та способу для шифрування та дешифрування електронної інформації, яка використовує код доступу, що містить задану кількість символів, для шифрування та дешифрування електронної інформації.

[20] Ще одне завдання даного винаходу полягає в створенні системи та способу для шифрування та дешифрування електронної інформації, яка використовує код доступу, що містить задану кількість символів, у комбінації з численними токенами, кожний з яких являє собою групу щонайменше з двох символів із сукупності символів, використаних для створення коду доступу.

[21] Для рішення щонайменше вищеописаних завдань, повністю або частково, запропонований спосіб аутентифікації користувача з використанням заданого коду доступу, який містить задану кількість символів ("символів коду доступу"), вибраних із сукупності символів, при цьому кожний із заданих символів коду доступу характеризується заданим положенням піна, а зазначений спосіб включає: надання користувачу сукупності токенів, яка містить щонайменше два токени і в якій кожний токен містить щонайменше два символи, що належать зазначеній сукупності символів, направлення користувачу запиту на вибір токена із зазначеної сукупності токенів для кожного положення піна в заданому коді доступу та здійснення аутентифікації користувача на підставі вибраних користувачем токенів.

[22] Крім того, для рішення щонайменше вищеописаних завдань, повністю або частково, запропонована система для аутентифікації користувача з використанням заданого коду доступу, який містить задану кількість символів ("символів коду доступу"), вибраних із сукупності символів, при цьому кожний із заданих символів коду доступу характеризується заданим положенням піна, а зазначена система містить: процесор, пам'ять, до якої має доступ процесор, і модуль аутентифікації/шифрування, що містить сукупність інструкцій, що читаються комп'ютером, збережених у пам'яті та виконуваних процесором для направлення користувачу сукупності токенів, яка містить щонайменше два токени та в якій кожний токен містить щонайменше два символи, що належать зазначеній сукупності символів, направляти користувачу запит на вибір токена із зазначеної сукупності токенів для кожного положення піна в заданому коді доступу й аутентифікувати користувача на підставі вибраних користувачем токенів.

[23] Додаткові переваги, завдання й ознаки даного винаходу будуть визначені частково в наведеному далі описі та частково стануть очевидними фахівцю в даній області техніки після

ознайомлення з наведеним далі описом або можуть бути отримані при реалізації даного винаходу. Завдання та переваги даного винаходу можуть бути реалізовані й отримані таким чином, як це зазначено, зокрема, у прикладеній формулі винаходу.

КОРОТКИЙ ОПИС КРЕСЛЕНЬ

[24] Даний винахід буде докладно описаний з посиланням на нижчеперелічені креслення, на яких одні і ті самі посилальні номери привласнені однаковим елементам.

[25] На фіг. 1A показана структурна схема, яка ілюструє приклад системи аутентифікації/шифрування із захистом від перехоплення, що може бути вбудована у пристрій або сервер, до якого має доступ клієнтська система, відповідно до одного з варіантів реалізації даного винаходу.

[26] На фіг. 1B показана структурна схема, яка ілюструє систему аутентифікації/шифрування із захистом від перехоплення, що вбудована у пристрій, відповідно до одного з варіантів реалізації даного винаходу.

[27] На фіг. 1C показана структурна схема, яка ілюструє систему аутентифікації/шифрування із захистом від перехоплення, що вбудована в сервер, до якого клієнтський пристрій має доступ за допомогою мережі, відповідно до одного з варіантів реалізації даного винаходу.

[28] На фіг. 1D показана структурна схема одного з прикладів апаратної реалізації системи аутентифікації/шифрування із захистом від перехоплення відповідно до одного з варіантів реалізації даного винаходу.

[29] На фіг. 2A показані приклади символів, згрупованих у чотири вимірювання, відповідно до одного з варіантів реалізації даного винаходу.

[30] На фіг. 2B показані приклади символів, згрупованих у п'ять вимірювань, відповідно до одного з варіантів реалізації даного винаходу.

[31] На фіг. 3 показана блок-схема, яка ілюструє приклад процесу, реалізованого модулем аутентифікації/шифрування для того, щоб дозволити користувачу створити код доступу, відповідно до одного з варіантів реалізації даного винаходу.

[32] На фіг. 4 показана блок-схема, яка ілюструє приклад процесу, реалізованого модулем аутентифікації/шифрування для аутентифікації користувача, відповідно до одного з варіантів реалізації даного винаходу.

[33] На фіг. 5 показана таблиця, в якій наведені приклади правил генерації токена, що використовуються модулем аутентифікації/шифрування, відповідно до одного з варіантів реалізації даного винаходу.

[34] На фіг. 6 показана таблиця, в якій наведені правила вибору токена, що використовуються модулем аутентифікації/шифрування, відповідно до одного з варіантів реалізації даного винаходу.

[35] На фіг. 7 показана таблиця прикладів правил валідації токена, що використовуються модулем аутентифікації/шифрування, відповідно до одного з варіантів реалізації даного винаходу.

[36] На фіг. 8A і 8B показані приклади знімків екрана, які ілюструють процес створення (реєстрації) ідентифікатора користувача у варіанті реалізації "GATE_4", відповідно до одного з варіантів реалізації даного винаходу.

[37] На фіг. 9A-9D показані приклади знімків екрана, які ілюструють процес реєстрації входу користувача у варіанті реалізації "GATE_4", відповідно до одного з варіантів реалізації даного винаходу.

[38] На фіг. 10A-10D показані приклади знімків екрана, які ілюструють процес реєстрації входу користувача у варіанті реалізації "GATE_4" у текстовому форматі, відповідно до одного з варіантів реалізації даного винаходу.

[39] На фіг. 11A і 11B показані приклади знімків екрана, які ілюструють процес створення (реєстрації) ідентифікатора користувача у варіанті реалізації "GATE_5", відповідно до одного з варіантів реалізації даного винаходу.

[40] На фіг. 12A-12D показані приклади знімків екрана, які ілюструють процес реєстрації входу користувача у варіанті реалізації "GATE_5", відповідно до одного з варіантів реалізації даного винаходу.

[41] На фіг. 13A-13D показані приклади знімків екрана, які ілюструють процес реєстрації входу користувача у варіанті реалізації "GATE_5" у текстовому форматі, відповідно до одного з варіантів реалізації даного винаходу.

[42] На фіг. 14 показаний приклад знімка екрана, який ілюструє процес шифрування повідомлень з використанням варіанта реалізації "GATE_4", в якому відкрите текстове повідомлення зашифроване з використанням коду доступу відправника, відповідно до одного з варіантів реалізації даного винаходу.

[43] На фіг. 15A і 15B показані приклади знімків екрана, які ілюструють процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4", в якому відбувається успішне дешифрування, відповідно до одного з варіантів реалізації даного винаходу.

[44] На фіг. 16A і 16B показані приклади знімків екрана, які ілюструють процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4", в якому зашифроване повідомлення-заповнювач валідують на стороні одержувача, відповідно до одного з варіантів реалізації даного винаходу.

[45] На фіг. 17 показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4", в якому дешифрування виявляється неуспішним внаслідок відмінності коду доступу одержувача від коду доступу відправника, відповідно до одного з варіантів реалізації даного винаходу.

[46] На фіг. 18 показаний приклад знімка екрана, який ілюструє процес шифрування повідомлень з використанням варіанта реалізації "GATE_5", в якому відкрите текстове повідомлення зашифроване з використанням коду доступу відправника, відповідно до одного з варіантів реалізації даного винаходу.

[47] На фіг. 19A і 19B показані приклади знімків екрана, які ілюструють процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5", в якому відбувається успішне дешифрування, відповідно до одного з варіантів реалізації даного винаходу.

[48] На фіг. 20A і 20B показані приклади знімків екрана, які ілюструють процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5", в якому зашифроване повідомлення-заповнювач валідують на стороні одержувача, відповідно до одного з варіантів реалізації даного винаходу.

[49] На фіг. 21 показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5", в якому дешифрування виявляється неуспішним внаслідок відмінності коду доступу одержувача від коду доступу відправника, відповідно до одного з варіантів реалізації даного винаходу.

ЗДІЙСНЕННЯ ВИНАХОДУ

[50] У даному винаході запропонований новий механізм аутентифікації та шифрування із захистом від перехоплення, реалізований з використанням спеціально запрограмованого пристрою. Для забезпечення аутентифікації даний винахід використовує "коди доступу", сформовані з використанням символів, які є частиною операційної системи пристрою. Як приклад код доступу може виглядати наступним чином:

Ⓘ ♥ 2 ✉

[51] Для користувача вищевказаний код доступу може означати "Я люблю відправляти повідомлення електронною поштою", який просто запам'ятати, але складно довідатися будь-якій іншій людині. Кожний символ у коді доступу буде називатися "піном" (pin) з відповідним положенням піна щодо інших пінів. У вищеописаному прикладі символ «Ⓘ» знаходиться у першому положенні піна, символ «♥» знаходиться у другому положенні піна, символ "2" знаходиться у третьому положенні піна, а символ «✉» знаходиться у четвертому положенні піна.

[52] Даний винахід переважно є "внутрішньосистемним" у тому, що він використовує вибрану сукупність символів, які переважно є частиною операційної системи пристрою, та, таким чином, не потребує від користувачів завантажувати які-небудь фотографії або зображення. Символи, які використані для створення кодів доступу або зашифрованого повідомлення, мають різні типи, що переважно згруповані у дві або більше груп, які будуть називатися у даному документі "вимірювання". Це дає користувачам більше широку множину різних способів самовираження при створенні своїх кодів доступу.

[53] У даному винаході запропонований новий механізм аутентифікації та шифрування із захистом від перехоплення шляхом "приховування" символів, що використовуються для пінів, які складають код доступу користувача серед множини інших символів, що не є частиною коду доступу користувача. Таким чином, по суті ховають голку в стозі сіна. Зокрема, даний винахід використовує те, що буде називатися у даному документі "токенами". Токен являє собою групу щонайменше з двох символів. Численні токени ("сукупність токенів") представляють користувачу, при цьому в деякі з токенів або в усі токени довільно введені деякі з попередньо вибраних пінів користувача або вони всі. Зокрема, кожний пін (представлений попередньо вибраним символом) у коді доступу користувача може бути включений в один із представлених користувачу токенів. Користувач вибирає токени, які містять індивідуальні піни, що становлять код доступу, так що положення піна кожного вибраного токена відповідає положенню піна коду

доступу, який він містить. Оскільки кожний вибраний токен містить не тільки один із попередньо вибраних пінів у коді доступу користувача, але також й інші довільно згенеровані символи, що не є одним із попередньо вибраних пінів у коді доступу користувача, то хто-небудь, хто бачить, які токени вибрав користувач, не може визначити діючий код доступу користувача.

[54] Кожний раз користувача просять надати код доступу, при цьому ще одну сукупність довільно генерованих токенів генерують з використанням окремих попередньо вибраних пінів у коді доступу користувача, довільно розподілених серед численних токенів. Таким чином, користувач буде вибирати сукупність токенів кожний раз, коли його просять надати код доступу, що запобігає можливості визначення сторонньою особою коду доступу користувача на підставі результату вибору токенів.

[55] В якості ілюстративного прикладу можуть бути використані 4 групи символів (4 вимірювання), при цьому кожне вимірювання містить 36 символів. Під час процесу реєстрації входу користувачу відображають 36 токенів, кожний з яких містить символ із кожного з чотирьох вимірювань символів. Заданий символ переважно відображають тільки в одному токені (тобто, якщо символ представлений в одному токені, він вже не представлений в іншому токені). Оскільки у даному прикладі відображають 36 токенів, відображають кожний символ у кожному з чотирьох вимірювань (один символ із кожного з вимірювань у кожному з 36 токенів).

[56] Якщо кількість пінів (представлених символами) у коді доступу користувача представлена змінною "N", то у подальшому користувачу буде необхідно вибрати N токенів (токени, які містять індивідуальні піни, що складають код доступу, так що положення піна кожного вибраного токена відповідає положенню піна коду доступу, який він містить).

[57] Як описано вище, даний винахід зменшить ймовірність "зчитування та перехоплення" коду доступу користувача, оскільки користувач вводить токен, який містить 4 символи, в тому числі один із пінів у коді доступу, для кожного піна в коді доступу. Таким чином, навіть якщо запис користувача виявився замічений хакером, що дивляться на екран для реєстрації входу, або внаслідок перехоплення хакером мережевого трафіка, хакер не зміг би визначити те, який з чотирьох символів у кожному токені відповідає кожному з попередньо вибраних пінів, які складають код доступу користувача. Відповідно, якби хакер спробував увійти в акаунт користувача, йому була би представлена інша сукупність довільно згенерованих токенів, деякі з яких містять попередньо вибрані піни користувача, які складають код доступу користувача, при цьому цей хакер не зміг би довідатися, які токени слід вибрати.

[58] Однак, якщо хакер побачить процес реєстрації входу користувача достатню кількість разів, цей хакер зможе порівняти всі записані сесії реєстрації входу для з'ясування того, що кожний пін знаходиться у коді доступу, оскільки кожний пін повинен бути представлений у токені, який вводить користувач, а якщо хакер порівняє всі перші токени з різних сесій реєстрації входу, цей хакер згодом визначить перший пін, а якщо хакер порівняє всі другі токени з усіх сесій, то цей хакер з часом визначить другий пін і т.д.

[59] Таким чином, для запобігання знаходження хакером пінів у коді доступу з часом, кількість довільно згенерованих токенів, представлених користувачу, переважно менше кількості символів у кожному вимірюванні. Наприклад, якщо кожне з одного або більше вимірювань містить 36 символів, можна представити користувачу тільки 16 токенів. Результатом цього є відсутність гарантії того, що пін користувача буде представлений навіть у токені. У даному варіанті реалізації, якщо користувач робить спробу зареєструвати вхід, й один або більше пінів у коді доступу користувача не представлений у кожному з токенів, у подальшому користувач вибирає будь-який токен в якості "універсального" токена для пінів, які не представлені у кожному з токенів (так що положення піна вибраного універсального токена відповідає положенню піна відсутнього піна у коді доступу). Це значно ускладнює роботу хакерів із здійснення підбору, оскільки може матися довільно вибраний токен замість одного з пінів користувача, який у дійсності не містить пін.

[60] Ще одна перевага використання токенів у кількості, меншій кількості символів в одному або більше вимірювань (наприклад, використання 16 токенів, коли кількість символів в одному або більше вимірювань становить 36) полягає в спрощенні користувачам завдання із швидкого з'ясування, чи є попередньо вибрані піни в токенах або ж ні, при цьому для користувача екран виглядає простіше.

[61] Даний винахід переважно використовує символи, які є частиною операційної системи пристрою, що включає дану систему. Таким чином, немає необхідності в завантаженні користувачем у систему яких-небудь спеціальних графічних зображень або фотографій, що зменшує, таким чином, навантаження на користувача та систему для зберігання та підтримки цих властивостей.

[62] Система та способи згідно з даним винаходом можуть бути використані не тільки для аутентифікації користувачів, але і для аутентифікації численних фрагментів інформації, так що вони можуть бути використані для шифрування повідомлень.

[63] На фіг. 1A показана структурна схема, яка ілюструє приклад системи 100 аутентифікації/шифрування із захистом від перехоплення, що може бути вбудована у пристрій або сервер, до якого має доступ клієнтська система, відповідно до одного з переважних варіантів реалізації даного винаходу. Система 100 містить модуль 110 аутентифікації/шифрування, який забезпечує функціональні можливості з аутентифікації та/або шифрування із захистом від перехоплення. При необхідності, система 100 може бути з'єднана з мережею 130.

[64] На фіг. 1B показана структурна схема, яка ілюструє систему аутентифікації/шифрування із захистом від перехоплення, що вбудована у пристрій 150, відповідно до одного з переважних варіантів реалізації даного винаходу. Пристрій 150 переважно містить модуль 120 користувальницького інтерфейсу. Модуль 110 аутентифікації/шифрування забезпечує надійну аутентифікацію коду доступу користувача/шифрування повідомлення, а також буде більше докладно пояснений нижче. Модуль 110 аутентифікації/шифрування може бути з'єднаний з мережею 130.

[65] Модуль 120 користувальницького інтерфейсу може бути реалізований з використанням відомого користувальницького інтерфейсу будь-якого типу, такого як, наприклад, графічний користувальницький інтерфейс, веб-інтерфейс і т. п. В цілому, модуль 120 користувальницького інтерфейсу може бути реалізований з використанням інтерфейсу будь-якого типу, що дозволяє користувачу взаємодіяти з модулем 110 аутентифікації.

[66] На фіг. 1C показана структурна схема, яка ілюструє систему аутентифікації/шифрування із захистом від перехоплення, що вбудована в сервер 160, до якого клієнтський пристрій 170 має доступ за допомогою мережі 130, відповідно до одного з переважних варіантів реалізації даного винаходу. Клієнтський пристрій 170 являє собою комп'ютер будь-якого типу або пристрій, який може одержати доступ до сервера 160 за допомогою мережі 130, і переважно містить модуль 120 користувальницького інтерфейсу, який дозволяє користувачу взаємодіяти з клієнтським пристроєм 170. Модуль 110 аутентифікації/шифрування забезпечує надійну аутентифікацію коду доступу користувача та/або шифрування повідомлення, а також буде більше докладно пояснений нижче.

[67] Лінії 140 зв'язку використовують для забезпечення зв'язку між модулем 110 аутентифікації/шифрування, модулем 120 користувальницького інтерфейсу, мережею 130 та клієнтським пристроєм 170. Лінії 140 зв'язку можуть являти собою дротові лінії зв'язку, бездротові лінії зв'язку, бездротові лінії індуктивного зв'язку, лінії ємнісного зв'язку або будь-які інші механізми, що використовуються в рівні техніки для з'єднання електронних компонентів. Дротова лінія зв'язку може бути реалізована з використанням шини, такої як, наприклад, шина на основі стандартної промислової архітектури (ISA), шина на основі мікроканальної архітектури (MCA), покращена шина на основі стандартної промислової архітектури (ISA), локальна шина Асоціації із стандартів в області відеоелектроніки (VESA) або шина для підключення периферійних компонентів (PCI).

[68] Прикладами ліній бездротового зв'язку є, крім іншого, лінія зв'язку за бездротовим протоколом передачі даних (WAP), лінія зв'язку відповідно до технології пакетного радіозв'язку загального користування (GPRS), лінія зв'язку на основі стандарту глобальної системи мобільного зв'язку (GSM), лінія зв'язку на основі стандарту множинного доступу з кодовим поділом каналів (CDMA) або лінія зв'язку на основі стандарту множинного доступу з тимчасовим поділом каналів (TDMA), такий як канал стільникового телефонного зв'язку, лінія зв'язку на основі глобальної системи навігації та визначення місця розташування (GPS), лінія передачі пакетів цифрових даних за стільниковою мережею (CDPD), прийомний пристрій дуплексного пошукового зв'язку компанії "RIM", лінія радіозв'язку на основі технології "Bluetooth" або лінія радіочастотного зв'язку на основі стандарту "IEEE 802.11" (WiFi).

[69] Мережа 130 може являти собою дротову або бездротову мережу, а також може містити або мати зв'язок із будь-якою однією або більше з, наприклад, мережі Інтернет, корпоративної мережі на основі технології Інтранет, персональної мережі (PAN), локальної обчислювальної мережі (LAN), глобальної обчислювальної мережі (WAN) або міської обчислювальної мережі (MAN), мережі з виділеною зоною зберігання даних (SAN), з'єднання на основі протоколу ретрансляції кадрів, з'єднання на базі розвиненої інтелектуальної мережі (AIN), з'єднання на основі протоколу синхронної мережі передачі даних за волоконно-оптичним кабелем (SONET), цифрової лінії зв'язку типу T1, T3, E1 або E3, з'єднання на базі цифрової служби передачі даних (DDS), з'єднання на базі цифрової абонентської лінії зв'язку (DSL), з'єднання на основі

протоколу "Ethernet", лінії зв'язку на базі цифрової мережі з інтегрованими послугами (ISDN), порту з доступом за комутованою телефонною лінією, такою як V.90, з'єднання на основі аналогового модему типу "V.34bis", кабельного модему, з'єднання на базі технології асинхронного режиму передачі даних (ATM), розподіленого інтерфейсу передачі даних за волоконно-оптичними каналами (FDDI) або з'єднання за стандартом високошвидкісних локальних кабельних обчислювальних мереж із маркерним доступом (CDDI).

[70] Модуль 120 користувальницького інтерфейсу може бути реалізований з використанням відомого користувальницького інтерфейсу будь-якого типу, такого як, наприклад, графічний користувальницький інтерфейс, веб-інтерфейс і т. п. В цілому, модуль 120 користувальницького інтерфейсу може бути реалізований з використанням інтерфейсу будь-якого типу, що дозволяє користувачу взаємодіяти з модулем 110 аутентифікації/шифрування.

[71] Термін "модуль", що використовується у даному документі, означає реальний пристрій, компонент або конструкцію з компонентів, реалізованих з використанням апаратного забезпечення, який може містити, наприклад, спеціалізовану інтегральну схему (ASIC) або вентильну матрицю з програмуванням в умовах експлуатації (FPGA), або мікропроцесорну систему та сукупність інструкцій для реалізації функціональних можливостей модуля, які (при їх виконанні) перетворять мікропроцесорну систему у пристрій спеціального призначення для виконання функцій модуля.

[72] Крім того, модуль може бути реалізований у вигляді комбінації з просто апаратного забезпечення й апаратного забезпечення з програмним керуванням, при цьому конкретні функції забезпечені просто апаратним забезпеченням, а інші функції забезпечені комбінацією з апаратного забезпечення та програмного забезпечення. У конкретних реалізаціях щонайменше частина модуля, а у деяких випадках і весь модуль, може бути виконаний на процесорі або процесорах комп'ютера або пристрою (наприклад, сервер 160 і клієнтському пристрої 170), який виконує операційну систему, системні програми та прикладні програми з одночасною реалізацією модуля з використанням багатозадачної, багатопотокової, розподіленої (наприклад, хмарної) обробки або інших таких технологій. Прикладами такого комп'ютера або пристрою є, крім іншого, персональний комп'ютер (наприклад, настільний персональний комп'ютер або кишеньковий комп'ютер), сервер, автоматизований касовий апарат (ATM), касовий термінал, програмно-апаратний комплекс і мобільний обчислювальний пристрій, такий як смартфон, планшет або персональний цифровий помічник (PDA). Крім того, сервер 160 являє собою сервер будь-якого підходящого типу, такий як сервер на базі операційної системи "Windows", сервер на базі операційної системи "Linux", сервер на базі операційної системи "Unix" або т. п.

[73] На фіг. 1D показана структурна схема одного з прикладів апаратної реалізації системи 100 для аутентифікації/шифрування із захистом від перехоплення відповідно до одного з варіантів реалізації даного винаходу. У варіанті реалізації, показаному на фіг. 1D, модуль 110 аутентифікації/шифрування реалізований центральним процесором 118 і пам'яттю 112.

[74] Центральний процесор 118 одержує доступ до коду 114 операційної системи та програмного коду 116, збереженого у пам'яті 112 для його наступного виконання. Програмний код 116, який реалізує функціональні можливості модуля 110 аутентифікації/шифрування, збережений у пам'яті 112 або на зовнішньому запам'ятовуючому пристрої (не показаний) для одержання до нього доступу та його виконання центральним процесором 118.

[75] Пам'ять 112 може бути реалізована у формі, наприклад, оперативного запам'ятовуючого пристрою (RAM), кеш-пам'яті, знімного або незнімного носія даних, енергозалежного або енергонезалежного носія даних, такого як незнімний енергонезалежний накопичувач на жорстких дисках, знімного енергонезалежного накопичувача на гнучких дисках, накопичувача на оптичних дисках (такого як постійний запам'ятовуючий пристрій на компакт-дисках, постійний запам'ятовуючий пристрій на цифрових відео-дисках (DVD-ROM) або будь-який інший оптичний запам'ятовуючий пристрій засіб), флеш-накопичувача з підключенням за USB-інтерфейсом зв'язку та карти пам'яті.

[76] Нижче будуть описані функціональні можливості модуля 110 аутентифікації/шифрування. Модуль 110 аутентифікації/шифрування забезпечує аутентифікацію коду доступу/виклику у відповідь на запит реєстрації входу користувача шляхом відображення користувачу (за допомогою модуля 120 користувальницького інтерфейсу) численних токенів із попередньо вибраними пінами користувача, довільно введеними в токени. Як описано вище, кожний пін (представлений попередньо вибраним символом) у коді доступу користувача може бути включений в один із представлених користувачу токенів. В одному з токенів представлений щонайменше один пін. Користувач вибирає токени, які містять індивідуальні піни, що складають код доступу, так що положення піна кожного вибраного токена відповідає положенню піна коду доступу, який він містить. Оскільки кожний вибраний токен

містить довільно згенеровані символи, які не являють собою жоден із попередньо вибраних пінів у коді доступу користувача, а також можливо жоден із попередньо вибраних пінів у коді доступу користувача, хто-небудь, хто бачить те, які токени вибрав користувач, не може визначити фактичний код доступу користувача.

[77] Нижче будуть описані два ілюстративні варіанти реалізації функціональних можливостей модуля аутентифікації/шифрування, при цьому вони будуть називатися у даному документі як "Графічний табличний запис стосовно доступу під номером 4" ("GATE_4") і "Графічний табличний запис стосовно доступу під номером 5" ("GATE_5"). Як показано на фіг. 2A, варіант реалізації "GATE_4" використовує 4 вимірювання символів, при цьому в кожне вимірювання включені 36 символів. Як показано на фіг. 2B, варіант реалізації "GATE_5" використовує 5 вимірювань символів, при цьому в кожне вимірювання включені 26 символів. Символи, показані на фіг. 2A і 2B, є прикладами типів символів, які можуть бути використані, при цьому слід враховувати, що можуть бути використані й символи будь-яких інших типів без виходу за межі обсягу даного винаходу.

[78] Символи, показані на фіг. 2A і 2B, зазвичай є в сучасних комп'ютерних операційних системах, а також не потребують якого-небудь спеціального процесу для створення або завантаження/збереження в існуючу систему, в яку вбудований даний винахід. Всі вони є підсукупністю системи кодування символів за стандартом "Юнікод", що використовується у більшості комп'ютерних операційних систем. Кожний символ має ідентифікатор за стандартом "Юнікод", всі з добре відомих знаків: a, b, … z, 0, 1, 9, @, +, <, % є частиною системи кодування символів за стандартом "Юнікод". Наприклад:

"Юнікод" «\u0062" призначене для знака: b

"Юнікод" «\u2206" призначене для знака: ∆

"Юнікод" «\u0040" призначене для знака: @

Символи, продемонстровані у варіантах реалізації, показаних на фіг. 2A і 2B, були включені внаслідок того, що вони є різноплановими, відмінними та легко запам'ятовуються.

[79] На фіг. 3 показана блок-схема, яка ілюструє приклад процесу, реалізованого модулем 110 аутентифікації/шифрування для того, щоб дозволити користувачу створити код доступу, відповідно до одного з варіантів реалізації даного винаходу. Процес починається на етапі 310, згідно з яким користувача просять ввести необхідний ідентифікатор користувача. На етапі 320 модуль 110 аутентифікації/шифрування визначає, чи є вже ідентифікатор користувача у пам'яті 112. У випадку наявності ідентифікатора користувача процес продовжується з етапу 330. У протилежному випадку процес продовжується з етапу 340.

[80] На етапі 330 користувача запитують, чи хоче він перезаписати наявний код доступу, пов'язаний з ідентифікатором користувача. Якщо користувач указує "ні", процес повертається назад до етапу 310. Якщо користувач указує "так", процес переходить до етапу 340.

[81] На етапі 340 користувачу відображають символи, доступні користувачу для вибору для кожного піна у своєму коді доступу. Надалі на етапі 350 користувача просять вибрати один із відображених символів у якості одного з пінів для свого коду доступу. На етапі 360 процес визначає, чи запросив користувач те, щоб тільки що вибраний пін або тільки що вибрані піни були збережені у вигляді коду доступу. Це може бути реалізовано, наприклад, для відображення іконки, яку користувач може вибрати, коли користувач готовий зберегти код доступу. Якщо користувач не вказав, що код доступу повинен бути збережений, процес повертається назад до етапу 350, згідно з яким користувач вибирає інший символ для іншого піна в коді доступу. Якщо користувач указує на етапі 360, що код доступу повинен бути збережений, надалі процес переходить до етапу 370.

[82] На етапі 370 визначають, чи задовольняє вибраний код доступу заданій вимозі за довжиною (тобто, заданій мінімальній кількості пінів). Якщо вибраний код доступу задовольняє, надалі код доступу зберігають на етапі 380. Якщо код доступу не підходить, надалі процес повертається назад до етапу 350, згідно з яким користувачу нагадують вибрати символ для додаткового піна.

[83] На фіг. 4 показана блок-схема, яка ілюструє приклад процесу, реалізованого модулем 110 аутентифікації/шифрування для аутентифікації користувача, відповідно до одного з варіантів реалізації даного винаходу. Процес починається на етапі 410, згідно з яким користувачу представляють екран для реєстрації входу, в якому користувачу нагадують ввести ідентифікатор користувача. Процес надалі переходить до етапу 420, згідно з яким модуль 110 аутентифікації/шифрування визначає, чи існує ідентифікатор користувача, введений користувачем. Якщо він існує, процес переходить до етапу 430. Якщо він не існує, процес повертається назад до етапу 410, згідно з яким користувачу нагадують ввести інший ідентифікатор користувача.

[84] На етапі 430 модуль аутентифікації/шифрування генерує задану кількість токенів на підставі кількості пінів, що використовуються для коду доступу користувача. У варіанті реалізації, показаному на фіг. 4, згенеровані 16 токенів переважно відображають користувачу в таблиці токенів розміром 4 × 4, як буде описано більше докладно нижче. Токени генерують на підставі правил генерації токена, що показані на фіг. 5. У прикладах у таблиці є 16 токенів, однак у дійсності може бути будь-яка кількість токенів, яка більше кількості пінів користувача, і представлених у вигляді 3 × 4, 2 × 5 або будь-якої іншої комбінації, при цьому варіант 4 × 4 є просто переважним способом їх відображення.

[85] Процес надалі переходить до етапу 440, згідно з яким користувач вибирає токени, які містять піни в його коді доступу, у порядку, в якому піни представлені у коді доступу, відповідно до правил вибору токена, що показані на фіг. 6. На етапі 450 модуль 110 аутентифікації/шифрування визначає, чи вибрав користувач токени відповідно до правил вибору токена, що показані на фіг. 6. Якщо додержуватися правил вибору токена, процес переходить до етапу 460, згідно з яким користувача аутентифікують та дозволяють доступ. Якщо не додержуватися правил вибору токена, процес переходить до етапу 470, згідно з яким користувача не аутентифікують, і йому забороняють доступ.

[86] Як описано вище, на фіг. 5 показана таблиця, в якій наведені приклади правил генерації токена, що використовуються модулем 110 аутентифікації/шифрування, відповідно до одного з варіантів реалізації даного винаходу. Щонайменше один із пінів у коді доступу користувача буде перебувати щонайменше в одному з 16 токенів. Іноді в токенах будуть виявлятися всі піни користувача, у більшості випадків від 1 до N (N є довжиною коду доступу користувача з N пінами) попередньо вибраних користувачем пінів будуть перебувати в 16 токенах.

[87] У варіанті реалізації, показаному на фіг. 5, генерують 16 токенів, а не 36 токенів (у випадку варіанта реалізації "GATE_4") і не 26 токенів (у варіанті реалізації "GATE_5"). Таким чином, у випадку варіанта реалізації "GATE_4" тільки 16/36=44 % періоду часу один із пінів у коді доступу користувача може бути представлений в одному з 16 токенів. У випадку варіанта реалізації "GATE_5" тільки 16/26=62 % періоду часу один із пінів у коді доступу користувача буде представлений в одному з 16 токенів. Є ймовірність, що всі піни у коді доступу користувача можуть бути представлені у таблиці токенів, при цьому гарантується, що у таблиці токенів буде представлений щонайменше один пін користувача. Більшу частину часу деякі з пінів у коді доступу користувача будуть відсутні, а деякі будуть представлені в токенах. В альтернативних варіантах реалізації правила можуть бути змінені таким чином, що в таблиці токенів будуть представлені щонайменше 2 або 3 із пінів у коді доступу користувача.

[88] Це і є тією невизначеністю, яка робить даний винахід ефективним. У випадку зчитування або перехоплення процесу реєстрації входу хакеру стає відомою тільки довжина коду доступу, оскільки якщо користувач вводить токени в кількості, більшій або меншій довжини коду доступу, реєстрація входу буде невдалою. Єдиною дією, яка призведе, але без яких-небудь гарантій, до успішної реєстрації входу є введення токенів у кількості, ідентичній кількості пінів у коді доступу користувача.

[89] Однак, навіть якщо хакер довідається довжину коду доступу, цей хакер не зможе визначити тотожність окремих пінів. Причина полягає в тому, що навіть якщо пін і не представлений у силу необхідності в одному з 16 токенів, користувач усе ще може успішно зареєструвати вхід. Це обумовлено тим, що, як зазначено у правилах вибору токена, показаних на фіг. 6, користувач може вибрати довільний токен для піна, який не є присутнім у жодному з представлених токенів.

[90] Крім того, навіть якщо всі піни у коді доступу користувача представлені в 16 токенах, хакер усе ще не зможе сказати те, який символ у кожному токені являє собою попередньо вибраний пін, оскільки є 4 символи у варіанті реалізації "GATE_4" і 5 символів у варіанті реалізації "GATE_5". Ця невизначеність забезпечує захист системи та способу згідно з даним винаходом від зчитування та перехоплення.

[91] Як зазначено у правилах вибору токена, показаних на фіг. 6, правила вибору дійсного токена можуть бути зведені воєдино наступним чином:

- Користувач повинен вибрати N токенів із таблиці токенів, що відповідають N пінам у коді доступу користувача. Таким чином, якщо код доступу користувача має 4 піна, то користувач вибирає 4 токени. Аналогічним чином, якщо код доступу користувача має 6 пінів, то користувач повинен вибрати 6 токенів.

- Якщо пін-код користувача представлений в одному з 16 токенів, то користувач повинен вибрати цей токен для введення піна.

- Якщо один із пінів у коді доступу користувача не представлений в жодному з 16 токенів, користувач повинен вибрати будь-який з 16 токенів для цього піна (що називається далі "універсальним токеном").

[92] Як описано вище, на фіг. 7 показана таблиця прикладів правил валідації токена, що використовуються модулем 110 аутентифікації/шифрування відповідно до одного з варіантів реалізації даного винаходу. Ці правила призначені для валідації токенів, вибраних користувачем у процесі реєстрації входу. Наприклад, правила визначають, чи рівна кількість токенів, ввведених користувачем, довжині коду доступу користувача. Якщо не рівна, то реєстрація входу користувача буде невдалою. Якщо рівна, то надалі правила зажадають перевірки кожного піна в коді доступу користувача для того, щоб довідатися, чи дійсно зазначений символ є в одному з 16 токенів. Якщо один із пінів у коді доступу користувача не представлений в одному з токенів, користувач повинен вибрати довільний токен. Якщо пін у коді доступу користувача не представлений в одному з токенів, користувач повинен вибрати цей токен.

[93] На фіг. 8A показаний приклад знімка екрана, який ілюструє порожній екран реєстрації для введення ідентифікатора користувача, для етапу 310, показаного на фіг. 3, відповідно до одного з варіантів реалізації даного винаходу. Креслення ілюструють те, як екран реєстрації варіанта реалізації "GATE_4" може бути показаний до введення користувачем ідентифікатора користувача.

[94] На фіг. 8B показаний приклад знімка екрана, який ілюструє кадр виконання процесу реєстрації для створення ідентифікатора користувача, представленого системою 100 відповідно до одного з варіантів реалізації даного винаходу. Фіг. 8B ілюструє те, як може виглядати екран реєстрації варіанта реалізації "GATE_4" при проходженні користувачем процесу реєстрації (створення ідентифікатора користувача), показаного на фіг. 3 у такий спосіб:

- Користувач вводить новий ідентифікатор користувача: "адміністратор" (G4 206), надалі натискає на кнопку "Перевірити наявність" (G4 208). Система здійснює перевірку для того, щоб довідатися, чи знаходиться вже ідентифікатор "адміністратор" в її пам'яті (етап 320, показаний на фіг. 3). Якщо знаходиться, то вона відобразить діалогове вікно (не показане), яке запитує: "Ідентифікатор користувача вже існує, ти прагнеш перезаписати наявний код доступу?» Якщо користувач не прагне перезаписувати старий код доступу, процес закриє діалогове вікно та буде очікувати введення користувачем іншого ідентифікатора користувача. Якщо ідентифікатор користувача "адміністратор" не існує, або якщо він існує, але користувач прагне перезаписати наявний код доступу, система активує кнопки у G4 212, G4 222, G4 232 і G4 242, кожна з яких містить 36 символів із заданого вимірювання, як показано на фіг. 2A. Наприклад, G4 212 містить всі 36 чисел із першого вимірювання, і ті символи будуть показані в токені у положенні «[1]» (верхньому лівому), як показано у G4 210 (такими числами є числа від 1 до 36). G4 222 містить 36 символів від «Ⓐ» до «?», і вони будуть показані у будь-якому токені у положенні «[2]» (G4 220: верхнє праве). G4 232 містить 36 символів від «○» до «₸», які будуть показані у положенні «[3]» (G4 230: нижній лівий) у токені, а G4 242 ілюструє 36 символів від «+» до «₪», при цьому вони представлені у будь-якому токені у положенні «[4]» (G4 240: нижній правий). На цьому етапі процесу система активує вищеописані кнопки у G4 212, G4 222, G4 232 і G4 242, так що користувач може натиснути на кожну з них. В якості порівняння, на фіг. 8A ці кнопки не активовані та виглядають тьмяними, оскільки користувач ще не ввів ідентифікатор користувача. При відсутності ідентифікатора користувача цьому користувачу не дозволено створювати код доступу.

- Користувач натискає на «①» серед символів у G4 222 для вибору першого піна, і він буде показаний у G4 250 (етап 350, показаний на фіг. 3).

- Користувач натискає на «♥» серед символів у G4 232 для вибору другого піна, і він буде показаний у G4 252 (етап 350, показаний на фіг. 3).

- Користувач натискає на "2" серед символів у G4 212 для вибору третього піна, і він буде показаний у G4 254 (етап 350, показаний на фіг. 3).

- Користувач натискає на «✉» серед символів у G4 242 для вибору четвертого піна, і він буде показаний у G4 256 (етап 350, показаний на фіг. 3).

- У цьому прикладі користувач вибирає наявність 4 пінів у коді доступу, так що довжина коду доступу становить 4.

- У цьому прикладі положення G4 258 і G4 260 залишені безбарвними.

- Користувач надалі завершує процес створення (реєстрації) ідентифікатора користувача шляхом натискання на кнопку "Зберегти" - G4 270 (етап 370, показаний на фіг. 3). Система збереже код доступу «① ♥ 2 ✉» з ідентифікатором користувача "адміністратор" у пам'ять (етап 380, показаний на фіг. 3).

[95] Як описано вище, розмірні варіанти не обмежені символами, продемонстрованими у прикладі, показаному на фіг. 8B. Розмірні варіанти можуть містити будь-які інші символи. У вищеописаному ілюстративному варіанті реалізації є чотири піна користувача, однак без виходу за межі обсягу даного винаходу може бути використана будь-яка кількість пінів. Якщо кількість пінів є занадто невеликою, то код доступу буде занадто вразливим. Якщо кількість пінів є занадто великою, то користувач може і не запам'ятати код доступу. Відповідно, переважне значення довжини знаходиться між 4 і 6 пінами.

[96] На фіг. 9A показаний приклад знімка екрана з кадром, який ілюструє виконання екрана для реєстрації входу, представленого системою 100 відповідно до одного з варіантів реалізації даного винаходу, до введення користувачем будь-якої інформації. Фіг. 9A використана для порівняння з фіг. 9B.

[97] На фіг. 9B показаний приклад знімка екрана з кадром, який ілюструє виконання процесу реєстрації входу, що виконується системою 100 відповідно до одного з варіантів реалізації даного винаходу. Фіг. 9B ілюструє те, як екран реєстрації варіанта реалізації "GATE_4" може виглядати в міру проходження користувачем процесу реєстрації входу за фіг. 4, що полягає в наступному:

- Користувач уводить ідентифікатор користувача: "адміністратор" (G4 306) (етап 410, показаний на фіг. 4).

- Користувач натискає на кнопку "Введення" (G4 309). Система здійснює перевірку для того, щоб довідатися, чи перебуває вже ідентифікатор "адміністратор" в її пам'яті (етап 420, показаний на фіг. 4), Якщо він не існує, система відобразить повідомлення (не показане), що ілюструє "Ідентифікатор користувача не існує, введіть, будь ласка, дійсний ідентифікатор користувача". Якщо він існує, система відобразить таблицю розміром 4 × 4 (G4 320). Для кращого опису таблиці рядки переважно промарковані зверху вниз у такий спосіб: A, B, C, D. Крім того, стовпці переважно промарковані зліва направо у такий спосіб: 1, 2, 3, 4. Токени у цій таблиці згенеровані відповідно до правил, показаних на фіг. 5.

- Оскільки ми знаємо з фіг. 8B, що кодом доступу, пов'язаним з ідентифікатором користувача "адміністратор", є: «Ⓘ ♥ 2 ✉», так що користувачу необхідно почати з перебору всіх 16 токенів у таблиці для знаходження першого піна: «Ⓘ». Оскільки символ Ⓘ належить до другого вимірювання, то у цьому прикладі він представлений у верхньому правому положенні будь-якого токена, так що користувачу необхідно відсканувати верхню праву частину кожного токена для того, щоб довідатися, чи є символ Ⓘ. У даному прикладі він знаходиться в токені D2. Цей знімок екрана був отриманий у демонстраційному режимі, а у демонстраційному режимі програма виділяє кольором співпадаючий символ користувачу для поліпшення розуміння процесу. У дійсності немає необхідності в його виділенні кольором. На даному кресленні, символ Ⓘ у D2 виділений кольором. Оскільки цей символ виявлений у токені D2, то користувач повинен натиснути на цей токен відповідно до правил, показаних на фіг. 6.

- Після того, як користувач натискає на D2, токен копіюють у перше положення коду доступу (G4 350) (етап 440, показаний на фіг. 4).

- Другим піном у коді доступу користувача є: «♥», при цьому цей символ належить до третього вимірювання. У даному прикладі символи з третього вимірювання представлені в нижній лівій стороні будь-якого токена. Таким чином, користувачу необхідно тільки глянути на нижню ліву сторону кожного з 16 токенів. У даному прикладі він знаходиться в токені D3. Таким чином, користувач повинен натиснути на D3 (етап 440, показаний на фіг. 4). На даному кресленні він виділений кольором.

- Після того, як користувач натискає на D3 (етап 440, показаний на фіг. 4), токен копіюють у друге положення коду доступу (G4 353).

- Третім піном у коді доступу є "2", при цьому він належить до першого вимірювання. Таким чином, у даному прикладі користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У даному прикладі він не існує. Згідно з правилами вибору токена (див. фіг. 6) користувач може і повинен вибрати універсальний токен (будь-який токен) для цього положення піна. У даному прикладі користувач довільно натискає на токен C3. Цей токен копіюють у третє положення піна - G4 356.

- Четвертим й останнім піном є «✉», при цьому цей символ належить до четвертого вимірювання. Символи четвертого вимірювання у цьому прикладі представлені у нижній правій стороні будь-якого токена. Таким чином, користувачу необхідно перевірити тільки нижню праву сторону кожного з 16 токенів. У даному прикладі він знаходиться в токені B3, так що користувачу необхідно натиснути на B3 (етап 440, показаний на фіг. 4). На даному кресленні він виділений кольором. Після того, як користувач натискає на B3, токен копіюють у четверте положення коду доступу (G4 359).

- Оскільки є тільки 4 піна у коді доступу, положення G4 362 і G4 365 залишені безбарвними, і вони повинні залишатися безбарвними. Якщо користувач вводить токен в одному місці або в обох місцях, система відхилить надання користувачу доступу, оскільки цей користувач ввів токени в кількості, більшій первинних чотирьох пінів (етап 450, показаний на фіг. 4).

- Після того, як користувач вводить всі чотири токени, користувач натисне на кнопку "Зареєструвати вхід" (G4 370) для того, щоб повідомити систему про те, що користувач завершив процес вибору токена, а система буде здійснювати перевірку для того, щоб довідатися, чи є діючими введені токени відповідно до правил, показаних на фіг. 7 (етап 450, показаний на фіг. 4).

- У даному прикладі введені користувачем токени є діючими, а система відображає повідомлення "Успішна реєстрація входу" (G4 380) та надає користувачу доступ (етап 460, показаний на фіг. 4).

[98] На фіг. 9C показаний приклад знімка екрана з кадром, який ілюструє виконання процесу реєстрації входу, що виконується системою 100, для невдалого процесу реєстрації входу відповідно до одного з варіантів реалізації даного винаходу. Фіг. 9C ілюструє те, як екран реєстрації варіанта реалізації "GATE_4" може виглядати, якщо користувач проходить через невдалий процес реєстрації входу (див. фіг. 4), який полягає у наступному:

- Користувач вводить ідентифікатор користувача: "адміністратор" (G4 307) (етап 410, показаний на фіг. 4).

- Користувач натискає на кнопку "Введення" (G4 310). Система здійснює перевірку для того, щоб довідатися, чи перебуває вже ідентифікатор "адміністратор" в її пам'яті (етап 420, показаний на фіг. 4), Якщо він не існує, система відобразить повідомлення (не показане), що ілюструє "Ідентифікатор користувача не існує, введіть, будь ласка, дійсний ідентифікатор користувача". Якщо він існує, система відобразить таблицю розміром 4 × 4 (G4 321). Для кращого опису таблиці рядки переважно промарковані зверху вниз у такий спосіб: A, B, C, D. Крім того, стовпці переважно промарковані зліва направо у такий спосіб: 1, 2, 3, 4. Токени у цій таблиці згенеровані відповідно до правил, показаних на фіг. 5.

- Оскільки ми знаємо з фіг. 8B, що кодом доступу, пов'язаним з ідентифікатором користувача "адміністратор", є: «ⓘ ♥ 2 ✉», так що користувачу необхідно почати з перебору всіх 16 токенів у таблиці для знаходження першого піна: «ⓘ». Оскільки символ ⓘ належить до другого вимірювання, у цьому прикладі він представлений у верхньому правому положенні будь-якого токена, так що користувачу необхідно відсканувати верхню праву частину кожного токена для того, щоб побачити, чи є ⓘ. У даному прикладі він знаходиться в токені A1. Користувач повинен натиснути на цей токен відповідно до правил, показаних на фіг. 6.

- Після того, як користувач натискає на A1, токен копіюють у перше положення коду доступу (G4 351) (етап 440, показаний на фіг. 4).

- Другим піном у коді доступу користувача є: «♥», при цьому цей символ належить до третього вимірювання. У даному прикладі символи третього вимірювання представлені в нижній лівій стороні будь-якого токена. Таким чином, користувачу необхідно тільки глянути на нижню ліву сторону кожного з 16 токенів. У даному прикладі він знаходиться в токені D1. Таким чином, користувач повинен натиснути на D1 (етап 440, показаний на фіг. 4). Однак, у цьому прикладі користувач не натиснув на цей токен, а замість цього натиснув на B4. Цей токен є невірним, а система запише це як помилку та надасть користувачу доступ.

- Після того, як користувач натискає на B4 (етап 440, показаний на фіг. 4), токен копіюють у друге положення коду доступу (G4 354).

- Третім піном у коді доступу є "2", при цьому він належить до першого вимірювання. Таким чином, у даному прикладі користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У даному прикладі він не існує. Згідно з правилами вибору токена (див. фіг. 6) користувач може та повинен вибрати універсальний токен (будь-який токен) для цього положення піна. У даному прикладі користувач довільно натискає на токен C4. Цей токен копіюють у третє положення піна - G4 357.

• Четвертим й останнім піном є «✉», при цьому цей символ належить до четвертого вимірювання. У цьому прикладі символи четвертого вимірювання представлені у нижній правій стороні будь-якого токена. Таким чином, користувачу необхідно перевірити тільки нижню праву сторону кожного з 16 токенів. У даному прикладі він знаходиться в токені A2, так що користувачу необхідно натиснути на A2 (етап 440, показаний на фіг. 4). Після того, як користувач натискає на A2, токен копіюють у четверте положення коду доступу (G4 360).

- Оскільки є тільки 4 піна у коді доступу, положення G4 363 і G4 366 залишені безбарвними, і вони повинні залишатися безбарвними. Якщо користувач вводить токен в одному місці або в

обох місцях, система відхилить надання користувачу доступу, оскільки цей користувач ввів токени в кількості, більшій первинних 4 пінів (етап 450, показаний на фіг. 4).

- Після того, як користувач вводить всі чотири токени, користувач натисне на кнопку "Зареєструвати вхід" (G4 371) для того, щоб повідомити систему про те, що він завершив процес вибору токена, а система здійснить перевірку для того, щоб довідатися, чи є діючими введені токени відповідно до правил, показаних на фіг. 7 (етап 450, показаний на фіг. 4).

- У даному прикладі введені користувачем токени є недіючими, а система відображає повідомлення "Неуспішна реєстрація входу" (G4 381) та відхиляє надання користувачу доступу (етап 470, показаний на фіг. 4).

[99] На фіг. 9D показаний приклад знімка екрана з кадром, який ілюструє виконання процесу реєстрації входу, що виконується системою 100 для іншого невдалого процесу реєстрації входу відповідно до одного з варіантів реалізації даного винаходу. Фіг. 9D ілюструє те, як екран реєстрації варіанта реалізації "GATE_4" може виглядати у міру проходження користувачем через невдалий процес реєстрації входу (див. фіг. 4), що полягає у наступному:

- Користувач вводить ідентифікатор користувача: "адміністратор" (G4 308) (етап 410, показаний на фіг. 4).

- Користувач натискає на кнопку "Введення" (G4 311). Система здійснює перевірку для того, щоб довідатися, чи перебуває вже ідентифікатор "адміністратор" в її пам'яті (етап 420, показаний на фіг. 4), Якщо він не існує, вона відобразить повідомлення (не показане), що ілюструє "Ідентифікатор користувача не існує, введіть, будь ласка, дійсний ідентифікатор користувача". Якщо він існує, система відобразить таблицю розміром 4 × 4 (G4 322). Для кращого опису таблиці рядки переважно промарковані зверху вниз у такий спосіб: A, B, C, D. Крім того, стовпці переважно промарковані зліва направо у такий спосіб: 1, 2, 3, 4. Токени у цій таблиці генерують відповідно до правил, показаних на фіг. 5.

- Оскільки ми знаємо з фіг. 8B, що кодом доступу, пов'язаним з ідентифікатором користувача "адміністратор", є: «Ⓘ ♥ 2 ✉», так що користувачу необхідно почати з перебору всіх 16 токенів у таблиці для знаходження першого піна: «Ⓘ». Оскільки символ Ⓘ належить до другого вимірювання, у цьому прикладі він представлений у верхньому правому положенні будь-якого токена, так що користувачу необхідно відсканувати верхню праву частину кожного токена для того, щоб побачити, чи є Ⓘ. У даному прикладі він знаходиться в токені B2. Користувач повинен натиснути на цей токен відповідно до правил, показаних на фіг. 6.

- Після того, як користувач натискає на B2, токен копіюють у перше положення коду доступу (G4 352) (етап 440, показаний на фіг. 4).

- Другим піном у коді доступу користувача є: «♥», при цьому цей символ належить до третього вимірювання. У даному прикладі символи третього вимірювання представлені в нижній лівій стороні будь-якого токена. Таким чином, користувачу необхідно тільки глянути на нижню ліву сторону кожного з 16 токенів. У даному прикладі він не існує. Згідно з правилами вибору токена (див. фіг. 6) користувач може та повинен вибрати універсальний токен (будь-який токен) для цього положення піна. У даному прикладі користувач довільно натискає на токен A4. Цей токен копіюють у друге положення піна - G4 355.

- Третім піном у коді доступу є "2", при цьому він належить до першого вимірювання. Таким чином, у даному прикладі користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У даному прикладі він знаходиться в токені B3. У даному прикладі користувач натискає на B3, а цей токен копіюють у третє положення піна - G4 358.

- Четвертим й останнім піном є «✉», при цьому цей символ належить до четвертого вимірювання. Символи четвертого вимірювання у цьому прикладі представлені у нижній правій стороні будь-якого токена. Таким чином, користувачу необхідно перевірити тільки нижню праву сторону кожного з 16 токенів. У даному прикладі він знаходиться в токені D1, так що користувачу необхідно натиснути на D1 (етап 440, показаний на фіг. 4). Після того, як користувач натискає на D1, токен копіюють у четверте положення коду доступу (G4 361).

- Оскільки є тільки 4 піна у коді доступу, положення G4 364 і G4 367 повинні бути залишені безбарвними. У даному прикладі користувач ввів додатковий токен D2, у результаті чого система відхилить надання користувачу доступу, оскільки користувач ввів піни у кількості, що перевищує первинні 4 піна.

- Після того, як користувач вводить всі п'ять токенів, користувач натисне на кнопку "Зареєструвати вхід" (G4 372) для того, щоб повідомити систему про те, що він завершив процес вибору токена, а система здійснить перевірку для того, щоб довідатися, чи є дійсними введені токени відповідно до правил, показаних на фіг. 7 (етап 450, показаний на фіг. 4).

- У даному прикладі користувач ввів занадто багато токенів, а система відображає повідомлення "Неуспішна реєстрація входу" (G4 382) та відхиляє надання користувачу доступу (етап 470, показаний на фіг. 4).

[100] На фіг. 10A показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_4" у текстовому форматі. Він відображає порожній екран при запуску програми. Це зображення використовують як основу порівняння для наведених нижче фіг. 10B, 10C і 10D. Цей екран показаний винятково в якості пояснення того, що відбувається поза полем зору. Його не показують під час реєстрації входу користувача в реальному часі.

[101] На фіг. 10B показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_4" у текстовому форматі. Цей знімок екрана ілюструє те, що відбувається поза полем зору, коли відбувається процес реєстрації входу користувача, показаний на фіг. 9B, й ілюструє те, як виглядає послідовність операцій процесу, показаного на фіг. 4, у типовому варіанті реалізації. Цей знімок екрана є винятково демонстраційною версією, при цьому його не показують під час реєстрації входу користувача в реальному часі. Цей знімок екрана використовують для візуалізації правил валідації токена, показаних на фіг. 7.

[102] Є 3 стовпці: "Клієнтська сторона" (ліва сторона), "Мережеве з'єднання" (середина) та "Серверна сторона" (права сторона). Процес починається з клієнтської сторони, коли користувач вводить ідентифікатор користувача, надалі інформацію передають через мережеве з'єднання на серверну сторону. Сервер генерує 16 токенів і передає їх у мережу, надалі токени передають на клієнтську сторону.

[103] Користувач вибирає токени відповідно до правил вибору токена, показаних на фіг. 6. Вибрані токени передають у мережу, надалі їх передають на серверну сторону для їх валідації, а результат у вигляді надання або відхилення доступу передають через мережу на серверну сторону. Послідовність операцій процесу зазначена стрілками на фіг. 10B. Більше докладне пояснення наведене нижче.

- Користувач уводить ідентифікатор користувача: "адміністратор" (G4 406).
- Користувач натискає на кнопку "Введення" (G4 409).
- Ідентифікатор користувача "адміністратор" (G4 406) показують на серверній стороні (G4 411) та передають (G4 421) у мережеве з'єднання (G4 413), а надалі його знову передають (G4 422) на серверну сторону (G4 415).
- На серверній стороні система здійснює перевірку своєї пам'яті для того, щоб довідатися, чи є ідентифікатор "адміністратор". Якщо він не існує, то система покаже повідомлення: "Ідентифікатор користувача не існує, введіть, будь ласка, дійсний ідентифікатор користувача" (не показано). Використовують той же самий приклад, що і приклад, показаний на фіг. 8B, при цьому кодом доступу у пам'яті є: «① ♥ 2 ✉», а система знаходить його у пам'яті (G4 417).
- Система генерує 16 токенів (G4 423) відповідно до правил генерації токена, показаних на фіг. 5.
- 16 токенів передають (G4 424) у мережу.
- Мережа передає (G4 425) токени на серверну сторону.
-16 токенів відображають на екрані для реєстрації входу користувача, як показано у фіг. 9B, у таблиці розміром 4 × 4 (G4 320 на фіг. 9B).
- Користувач вибирає (G4 426) чотири токени: G4 350, G4 353, G4 356 і G4 359.
- Чотири вибрані користувачем токени передають (G4 427) у мережу після того, як користувач натискає "Зареєструвати вхід" (G4 370 на фіг. 9B).
- Чотири вибрані користувачем токени надалі передають (G4 428) на серверну сторону.
- На серверній стороні система перевіряє всі чотири токени один за одним: C11, C12, C13 і C14, при цьому у цьому прикладі всі вони є правильними.
- Вищеописаний результат успішної реєстрації входу (G4 429) передають (G4 430) у мережу.
- Мережа передає (G4 431) результат на серверну сторону та відображає (G4 432) повідомлення, показане у G4 380, що показаний на фіг. 9B.

[104] На фіг. 10C показаний приклад знімка екрана, що ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_4" у текстовому форматі. Цей знімок екрана ілюструє те, що відбувається поза полем зору, коли відбувається процес реєстрації входу користувача, показаний на фіг. 9C, й ілюструє те, як виглядає послідовність операцій процесу, показана на фіг. 4, у типовому варіанті реалізації. Цей знімок екрана є винятково демонстраційною версією, при цьому його не показують під час реєстрації входу користувача в реальному часі. Цей знімок екрана використовують для візуалізації правил валідації токена, показаних на фіг. 7.

[105] Є 3 стовпця: "Клієнтська сторона" (ліва сторона), "Мережеве з'єднання" (середина) та "Серверна сторона" (права сторона). Процес починається з клієнтської сторони, коли

користувач вводить ідентифікатор користувача, надалі інформацію передають через мережеве з'єднання на серверну сторону. Сервер генерує 16 токенів і передає їх у мережу, а надалі ці токени передають на серверну сторону.

[106] Користувач вибирає токени відповідно до правил вибору токена, показаних на фіг. 6. Вибрані токени передають у мережу, а надалі передають на серверну сторону для їх валідації, при цьому результат надання або відхилення доступу передають через мережу на серверну сторону. Послідовність операцій процесу зазначена стрілками на фіг. 10C. Більше докладне пояснення наведене нижче.

- Користувач вводить ідентифікатор користувача: "адміністратор" (G4 506).
- Користувач натискає на кнопку "Введення" (G4 509).
- Ідентифікатор користувача "адміністратор" (G4 506) показують на серверній стороні (G4 511) та передають (G4 521) у мережеве з'єднання (G4 513), а надалі його знову передають (G4 522) на серверну сторону (G4 515).
- На серверній стороні система здійснює перевірку своєї пам'яті для того, щоб довідатися, чи є ідентифікатор "адміністратор". Якщо він не існує, то система покаже повідомлення: "Ідентифікатор користувача не існує, введіть, будь ласка, дійсний ідентифікатор користувача" (не показано). Використовують той же самий приклад, що і приклад, показаний на фіг. 8B, при цьому кодом доступу у пам'яті є: «🄸 ♥ 2 ✉», а система знаходить його у пам'яті (G4 517).
- Система генерує 16 токенів (G4 523) відповідно до правил генерації токена, показаних на фіг. 5.
- 16 токенів передають (G4 524) у мережу.
- Мережа передає (G4 525) ці токени на серверну сторону.
- 16 токенів відображають на екрані для реєстрації входу користувача, як показано у фіг. 9C, у таблиці розміром 4 × 4 (G4 321 на фіг. 9C).
- Користувач вибирає (G4 526) чотири токени: G4 351, G4 354, G4 357 і G4 360.
- Чотири вибрані користувачем токени передають (G4 527) у мережу після того, як користувач натискає "Зареєструвати вхід" (G4 371 на фіг. 9C).
- Чотири вибрані користувачем токени надалі передають (G4 528) на серверну сторону.
- На серверній стороні система перевіряє всі чотири токени один за одним: C21, C22, C23 і C24, при цьому у цьому прикладі другий токен є неправильним (оскільки другий пін «♥» є в токені D1, однак користувач вибрав B4 токен, який був невірним. Таким чином результат полягає у невдалій реєстрації входу).
- Вищеописаний результат невдалої реєстрації входу (G4 529) передають (G4 530) у мережу.
- Мережа передає (G4 531) результат на серверну сторону та відображає (G4 532) повідомлення, показане у G4 381, що показаний на фіг. 9C.

[107] На фіг. 10D показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_4" у текстовому форматі. Цей знімок екрана ілюструє те, що відбувається поза полем зору, коли відбувається процес реєстрації входу користувача, показаний на фіг. 9D, й ілюструє те, як виглядає послідовність операцій процесу, показана на фіг. 4, у типовому варіанті реалізації. Цей знімок екрана є тільки демонстраційною версією, при цьому його не показують під час реєстрації входу користувача в реальному часі. Цей знімок екрана використовують для візуалізації правил валідації токена, показаних на фіг. 7.

[108] Є 3 стовпця: "Клієнтська сторона" (ліва сторона), "Мережеве з'єднання" (середина) та "Серверна сторона" (права сторона). Процес починається з клієнтської сторони, коли користувач вводить ідентифікатор користувача, надалі інформацію передають через мережеве з'єднання на серверну сторону. Сервер генерує 16 токенів і передає їх у мережу, а надалі ці токени передають на серверну сторону.

[109] Користувач вибирає токени відповідно до правил вибору токена, показаних на фіг. 6. Вибрані токени передають у мережу, а надалі передають на серверну сторону для їх валідації, при цьому результат надання або відхилення доступу передають через мережу на серверну сторону. Послідовність операцій процесу зазначена стрілками на фіг. 10D. Більше докладне пояснення наведене нижче.

- Користувач уводить ідентифікатор користувача: "адміністратор" (G4 606).
- Користувач натискає на кнопку "Введення" (G4 609).
- Ідентифікатор користувача "адміністратор" (G4 606) показують на серверній стороні (G4 611) та передають (G4 621) у мережеве з'єднання (G4 613), а надалі його знову передають (G4 622) на серверну сторону (G4 615).
- На серверній стороні система здійснює перевірку своєї пам'яті для того, щоб довідатися, чи є ідентифікатор "адміністратор". Якщо він не існує, то система покаже повідомлення:

"Ідентифікатор користувача не існує, введіть, будь ласка, дійсний ідентифікатор користувача" (не показано). Використовують той же самий приклад, що і приклад, показаний на фіг. 8B, при цьому кодом доступу у пам'яті є: «① ♥ 2 ✉», а система знаходить його у пам'яті (G4 617).

- Система генерує 16 токенів (G4 623) відповідно до правил генерації токена, показаних на фіг. 5.
- 16 токенів передають (G4 624) у мережу.
- Мережа передає (G4 625) ці токени на серверну сторону.
- 16 токенів відображають на екрані для реєстрації входу користувача, як показано у фіг. 9D, у таблиці розміром 4 × 4 (G4 322 на фіг. 9D).
- Користувач вибирає (G4 626) п'ять токенів: G4 352, G4 355, G4 358, G4 361 і G4 364.
- П'ять вибраних користувачем токенів передають (G4 627) у мережу після того, як користувач натискає "Зареєструвати вхід" (G4 372 на фіг. 9D).
- П'ять вибраних користувачем токенів надалі передають (G4 628) на серверну сторону.
- На серверній стороні система перевіряє всі п'ять токенів один за одним: C31, C32, C33, C34 і C35, при цьому у цьому прикладі п'ятий токен є неправильним (оскільки код доступу має тільки 4 піна, однак користувач ввів п'ятий токен, який був невірним. Таким чином результат полягає у невдалій реєстрації входу).
- Вищеописаний результат невдалої реєстрації входу (G4 629) передають (G4 630) у мережу.
- Мережа передає (G4 631) результат на серверну сторону та відображає (G4 632) повідомлення, показане у G4 382, що показаний на фіг. 9D.

[110] На фіг. 11A показаний приклад знімка екрана, який ілюструє процес створення (реєстрації) ідентифікатора користувача у варіанті реалізації "GATE_5". Фіг. 11A відображає порожній екран, у якому запускаються програми. Це зображення використовують як основу для порівняння з наведеною нижче фіг. 11B.

[111] На фіг. 11B показаний приклад знімка екрана, який ілюструє процес створення (реєстрації) ідентифікатора користувача у варіанті реалізації "GATE_5". Фіг. 11B ілюструє те, де в токені розташоване кожне вимірювання символів. Крім того, фіг. 11B відображає те, як користувач створює новий ідентифікатор користувача й як користувач вибирає та зберігає піни для формування коду доступу, пов'язаного з ідентифікатором користувача. Процес переходить до наступного:

- Користувач вводить новий ідентифікатор користувача: "адміністратор" (G5 206), надалі натискає на кнопку "Перевірити наявність" (G5 208). Система здійснює перевірку для того, щоб довідатися, чи знаходиться вже ідентифікатор "адміністратор" у своїй пам'яті, а якщо це так, то ця система відображає діалогове вікно (не показано), яке запитує: "Ідентифікатор користувача вже існує, ти прагнеш перезаписати наявний код доступу?» Якщо користувач не прагне перезаписати старий код доступу, процес закриє діалогове вікно та буде очікувати введення користувачем іншого ідентифікатора користувача. Якщо ідентифікатор користувача "адміністратор" не існує або якщо він існує, але користувач прагне перезаписати наявний код доступу, система активує кнопки у G5 212, G5 222, G5 232, G5 242 і G5 248, кожна з яких містить 26 символів із заданого вимірювання, як описано на фіг. 2B.

Наприклад, G5 212 містить всі 26 символів із першого вимірювання, ці символи опиняться в токені у положенні «[1]» (верхнє ліве), як показано у G5 210. Символами в кількості 26 штук є символи від «Ⓐ» до «Ⓩ». G5 222 ілюструє 26 символів від «α» до «ϟ», вони з другого вимірювання, і вони опиняться у будь-якому токені у положенні «[2]» (G5 220: верхнє праве). G5 232 ілюструє 26 чисел від 1 до 26, вони з третього вимірювання, який показаний у положенні «[3]» (G5 230: посередині) в токені, а G5 242 ілюструє 26 символів від «○» до «≠», вони з четвертого вимірювання, при цьому вони виявляться у будь-якому токені у положенні «[4]» (G5 240: нижнє ліве). G5 248 ілюструє 26 символів від «+» до «₪», вони з п'ятого вимірювання, при цьому вони виявляться у будь-якому токені у положенні «[5]» (G5 246: нижнє праве).

У цей момент часу у процесі система активує вищеописані кнопки у G5 212, G5 222, G5 232, G5 242 і G5 248, так що користувач може натиснути на кожну з них. В якості порівняння, на фіг. 11A ці кнопки є неактивними та виглядають тьмяними, оскільки користувач ще не ввів який-небудь ідентифікатор користувача. При відсутності ідентифікатора користувача, це не дозволить користувачу вибрати будь-який пін.

- Користувач натискає на «$» серед символів у G5 248 для вибору першого піна, при цьому це показано у G5 250.
- Користувач натискає на «=» серед символів у G5 242 для вибору другого піна, при цьому це показано у G5 252.
- Користувач натискає на «Ⓜ» серед символів у G5 212 для вибору третього піна, при цьому

це показано у G5 254.

- Користувач натискає на «©» серед символів у G5 212 для вибору четвертого піна, при цьому це показано у G5 256.

- Користувач натискає на "2" серед символів у G5 232 для вибору п'ятого піна, при цьому це показано у G5 258.

- Користувач натискає на «☺» серед символів у G5 242 для вибору шостого піна, при цьому це показано у G5 260.

- У цьому прикладі користувач вибирає наявність 6 пінів в його коді доступу, так що довжина його коду доступу становить 6.

- Користувач надалі завершує процес створення (реєстрації) ідентифікатора користувача шляхом натискання на кнопку "Зберегти" (G5 270). Система збереже коди доступу «$ = Ⓜ © 2 ☺» з ідентифікатором користувача "адміністратор" у пам'ять.

[112] На фіг. 12A показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5". Це відображає порожній екран при запуску програми. Це зображення використовують як основу порівняння з наведеними нижче фіг. 12B, 12C і 12D.

[113] На фіг. 12B показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5". На фіг. 12B показана таблиця розміром 4 × 4 із 16 токенами, з яких користувач вибирає коди доступу. Фіг. 12B відображає те, як працює процес вибору токена, а також на ній виділені кольором символи в токенах, які вибирає користувач й які мають у них піни користувача як частини коду доступу користувача. Крім того, на фіг. 12B показано як може виглядати успішна реєстрація входу. Цей ілюстративний процес слідує прикладу, показаному на фіг. 11B, тому що використовують той же самий код доступу. Процес переходить до наступного:

- Користувач уводить новий ідентифікатор користувача: "адміністратор" (G5 306).

- Користувач натискає на кнопку "Введення" (G5 309). Система здійснює перевірку для того, щоб довідатися, чи знаходиться вже ідентифікатор "адміністратор" у своїй пам'яті. Якщо він не існує, то система відобразить повідомлення (не показано), яке ілюструє "Ідентифікатор користувача не існує, введіть, будь- ласка, дійсний ідентифікатор користувача". Якщо він існує, то система відобразить таблицю розміром 4 × 4 (G5 320), при цьому для кращого опису таблиці рядки промарковані зверху вниз: A, B, C, D. Крім того, стовпці промарковані зліва направо: 1, 2, 3, 4. Токени у цій таблиці генерують відповідно до правил, показаних на фіг. 5.

- Оскільки ми знаємо з фіг. 11B, що кодом доступу, пов'язаним з ідентифікатором користувача "адміністратор", є: «$ = Ⓜ © 2 ☺», так що користувачу необхідно почати з перебору всіх 16 токенів у таблиці для знаходження першого піна: «$».

Оскільки символ $ належить до п'ятого вимірювання, він буде представлений тільки у нижньому правому положенні будь-якого токена, так що користувачу необхідно відсканувати нижню праву частину кожного токена для того, щоб побачити, чи є символ $. У нашому прикладі він знаходиться в токені B4, при цьому екран використаний у демонстраційному режимі, а у демонстраційному режимі програма виділяє кольором співпадаючий символ для користувача для поліпшення розуміння процесу. В дійсності, він не буде виділений кольором. У нашому випадку символ $ у B4 виділений кольором, оскільки він виявлений в токені B4, при цьому користувач повинен натиснути на цей токен відповідно до правил, показаних на фіг. 6.

- Після того, як користувач натискає на B4, токен копіюють у перше положення коду доступу (G5 350).

- Другим піном у коді доступу користувача є: «=», при цьому цей символ належить до четвертого вимірювання, при цьому символи четвертого вимірювання представлені винятково в нижній лівій стороні будь-якого токена, так що користувачу необхідно тільки глянути на нижню ліву сторону кожного з 16 токенів. У нашому випадку він знаходиться в токені D2, так що користувач повинен натиснути на D2. Це виділено кольором на знімку екрана.

- Після того, як користувач натискає на D2, токен копіюють у друге положення коду доступу (G5 353).

- Третім піном у коді доступу є: «Ⓜ», при цьому він належить до першого вимірювання, так що користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У нашому випадку він знаходиться в токені D4, так що користувач повинен натиснути на D4. Це виділено кольором на знімку екрана.

- Після того, як користувач натискає на D4, токен копіюють у третє положення коду доступу (G5 356).

- Четвертим піном є: «©», при цьому цей символ належить до першого вимірювання, а символи першого вимірювання представлені тільки у верхній лівій стороні будь-якого токена,

так що користувачу необхідно тільки перевірити у верхній лівій стороні кожного з 16 токенів. У нашому випадку він не існує, при цьому згідно з правилами вибору токена користувач може та повинен вибрати універсальний токен (будь-який токен) у положенні піна, так що цей користувач натискає на довільний токен A4, при цьому цей токен копіюють у четверте положення піна G5 359.

- П'ятим піном у коді доступу є: "2", при цьому він належить до третього вимірювання, так що користувачу необхідно тільки глянути в центр кожного з 16 токенів. У нашому випадку він знаходиться в токені D2, так що користувач повинен натиснути на D2. Це виділено кольором на знімку екрана. Після того, як користувач натискає, токен копіюють у п'яте положення коду доступу (G5 362).

- Шостим й останнім піном є: «☺», при цьому цей символ належить до четвертого вимірювання, а символи четвертого вимірювання представлені винятково в нижній лівій стороні будь-якого токена, так що користувачу необхідно тільки перевірити нижню ліву сторону кожного з 16 токенів. У нашому випадку він знаходиться в токені A4, так що користувачу необхідно натиснути на A4. Крім того, це зазначено у демонстраційній версії програми. Після того, як користувач здійснить натискання, токен копіюють у шосте й останнє положення коду доступу (G5 365).

- Після того, як користувач введе всі шість токенів, він натисне на кнопку "Зареєструвати вхід" (G5 370) для того, щоб повідомити систему, що він завершив процес вибору токена, а система здійснить перевірку для того, щоб довідатися, чи є дійсними введені токени відповідно до правил, показаних на фіг. 7.

- У даному прикладі введені користувачем токени є дійсними, а система відобразила повідомлення "Успішний вхід у систему" та надала доступ користувачу (G5 380).

- У даному прикладі токени у G5 353 і G5 362 є одними і тими самими, що і токени у G5 359 і G5 365, при цьому даний випадок є всього лише збігом, такі ситуації, подібні до цієї, можуть відбуватися досить часто. Це може сильно спантеличити того, хто намагається вгадати код доступу.

[114] На фіг. 12C показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5". На фіг. 12C показана таблиця розміром 4 × 4 із 16 токенами, з яких користувач вибирає код доступу. Фіг. 12C відображає те, як працює процес вибору токена. Фіг. 12C ілюструє те, як може виглядати невдала реєстрація входу з використанням 3 невірних пінів. Цей ілюстративний процес слідує прикладу, показаному на фіг. 11B, так що буде використаний той же самий код доступу. Процес продовжується наступним чином:

- Користувач уводить ідентифікатор користувача: "адміністратор" (G5 307).

- Користувач натискає на кнопку "Введення" (G5 310). Система здійснює перевірку для того, щоб довідатися, чи знаходиться вже ідентифікатор "адміністратор" в її пам'яті. Якщо він не існує, то система відобразить повідомлення (не показано), що ілюструє "Ідентифікатор користувача не існує, введіть, будь-ласка, дійсний ідентифікатор користувача". Якщо він існує, то система відобразить таблицю розміром 4 × 4 (G5 321), при цьому для кращого опису таблиці рядки промарковані зверху вниз: A, B, C, D. Крім того, стовпці промарковані зліва направо: 1, 2, 3, 4. Токени у цій таблиці згенеровані відповідно до правил, показаних на фіг. 5.

- Оскільки ми знаємо з фіг. 11B, що кодом доступу, пов'язаним з ідентифікатором користувача "адміністратор", є: «$ = Ⓜ Ⓒ 2 ☺», так що користувачу необхідно почати з перебору всіх 16 токенів у таблиці для знаходження першого піна: «$»

Оскільки символ $ належить до п'ятого вимірювання, він буде представлений тільки у нижньому правому положенні будь-якого токена, так що користувачу необхідно відсканувати нижню праву частину кожного токена для того, щоб побачити, чи є символ $. У нашому прикладі він знаходиться в токені A2, користувач повинен натиснути на цей токен відповідно до правил, показаних на фіг. 6.

- Після того, як користувач натискає на A2, токен копіюють у перше положення коду доступу (G5 351).

- Другим піном у коді доступу користувача є: «=», цей символ належить до четвертого вимірювання, а символи четвертого вимірювання представлені винятково в нижній лівій стороні будь-якого токена, так що користувачу необхідно тільки глянути на нижню ліву сторону кожного з 16 токенів. У нашому випадку він не існує, при цьому згідно з правилами вибору токена користувач може та повинен вибрати універсальний токен (будь-який токен) у положенні піна, так що цей користувач натискає на довільний токен A3.

- Після того, як користувач натискає на A3, токен копіюють у друге положення коду доступу (G5 354).

- Третім піном у коді доступу є: «Ⓜ», при цьому він належить до першого вимірювання, так що користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У нашому випадку він знаходиться в токені C1, так що користувачу необхідно натиснути на C1. У нашому випадку користувач дійсно натиснув на C1.

- Після того, як користувач натискає на C1, токен копіюють у третє положення коду доступу (G5 357).

- Четвертим піном є: «Ⓒ», при цьому цей символ належить до першого вимірювання, так що користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У нашому випадку він знаходиться в токені D2, так що користувачу необхідно натиснути на D2. У нашому випадку користувач не натиснув на D2, а замість цього натиснув на C2. Це є невірним, а система відхилить надання користувачу доступу.

- Після того, як користувач натискає на C2, токен копіюють у четверте положення коду доступу (G5 360).

- П'ятим піном у коді доступу є: "2", при цьому він належить до третього вимірювання, так що користувачу необхідно тільки глянути в центр кожного з 16 токенів. У нашому випадку він знаходиться в токені C3, при цьому відповідно до правил вибору токена, показаних на фіг. 6, користувач повинен вибрати цей токен, однак у прикладі користувач натомість вибрав токен B2, який є невірним, а система здійснить перевірку та повідомить.

- Після того, як користувач натискає на невірний токен B2, його копіюють у п'яте положення коду доступу (G5 363).

- Шостим й останнім піном є: «☺», при цьому цей символ належить до четвертого вимірювання, а символи четвертого вимірювання тільки представлені в нижній лівій стороні будь-якого токена, так що користувачу необхідно тільки перевірити нижню ліву сторону кожного з 16 токенів. У нашому випадку він знаходиться в токені D1, так що користувачу необхідно натиснути на D1. У нашому випадку користувач не натиснув на D1, а замість цього користувач натиснув на токен C4, що є невірним, при цьому система помітить це.

- Після того, як користувач натискає на невірний токен C4, його копіюють у шосте положення коду доступу (G5 366).

- Після того, як користувач ввів всі шість токенів, він натискає на кнопку "Зареєструвати вхід" (G5 371) для того, щоб повідомити систему, що він завершив процес вибору токена, при цьому система здійснить перевірку для того, щоб довідатися, чи є діючими введені токени, відповідно до правил, показаних на фіг. 7.

- У даному прикладі введені користувачем токени є недійсними, а система відобразила повідомлення "Неуспішна реєстрація входу" та відхилила надання користувачу доступу (G5 381).

[115] На фіг. 12D показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5". На фіг. 12D показана таблиця розміром 4 × 4 із 16 токенами, з яких користувач вибирає код доступу. Фіг. 12D відображає те, як працює процес вибору токена. Крім того, на фіг. 12D показане те, як може виглядати невдала реєстрація входу при пропущеному піні. Цей ілюстративний процес слідує прикладу, показаному на фіг. 11B, так що буде використаний той же самий код доступу. Процес продовжується наступним чином:

- Користувач вводить ідентифікатор користувача: "адміністратор" (G5 308).

- Користувач натискає на кнопку "Введення" (G5 311). Система здійснює перевірку для того, щоб довідатися, чи знаходиться вже ідентифікатор "адміністратор" в її пам'яті. Якщо він не існує, то система відобразить наступне повідомлення (не показано) "Ідентифікатор користувача не існує, введіть, будь-ласка, дійсний ідентифікатор користувача". Якщо він існує, то система відобразить таблицю розміром 4 × 4 (G5 322), при цьому для переважного опису таблиці рядки промарковані зверху вниз: A, B, C, D. Крім того, стовпці промарковані зліва направо: 1, 2, 3, 4. Токени у цій таблиці згенеровані відповідно до правил, показаних на фіг. 5.

- Оскільки ми знаємо з фіг. 11B, що кодом доступу, пов'язаним з ідентифікатором користувача "адміністратор", є: «$ = Ⓜ Ⓒ 2 ☺», користувачу необхідно почати з перебору всіх 16 токенів у таблиці для знаходження першого піна: «$».

Оскільки символ $ належить до п'ятого вимірювання, він буде представлений тільки у нижньому правому положенні будь-якого токена, так що користувачу необхідно тільки відсканувати нижню праву частину кожного токена для того, щоб побачити, чи є символ $. У нашому прикладі він не існує, при цьому користувач може та повинен вибрати універсальний токен (будь-який токен) у положенні піна, так що цей користувач натискає на довільний токен D3.

- Після того, як користувач натискає на D3, токен копіюють у перше положення коду доступу (G5 352).

- Другим піном у коді доступу користувача є: «=», при цьому цей символ належить до четвертого вимірювання, а символи з четвертого вимірювання представлені винятково в нижній лівій стороні будь-якого токена, так що користувачу необхідно тільки глянути на нижню ліву сторону кожного з 16 токенів. У нашому випадку він знаходиться в токені C1, так що користувач повинен натиснути на цей токен.

- Після того, як користувач натискає на C1, токен копіюють у друге положення коду доступу (G5 355).

- Третім піном у коді доступу є: «Ⓜ», при цьому він належить до першого вимірювання, так що користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У нашому випадку він знаходиться в токені A1, так що користувачу необхідно натиснути на A1. У нашому випадку користувач дійсно натиснув на A1.

- Після того, як користувач натискає на A1, токен копіюють у третє положення коду доступу (G5 358).

- Четвертим піном є: «Ⓒ», при цьому цей символ належить до першого вимірювання, так що користувачу необхідно тільки глянути на верхнє ліве положення кожного з 16 токенів. У нашому випадку він знаходиться в токені D3, так що користувачу необхідно натиснути на D3. У нашому випадку користувач не натиснув на D3.

- Після того, як користувач натискає на D3, токен копіюють у четверте положення коду доступу (G5 361).

- П'ятим піном у коді доступу є: "2", при цьому він належить до третього вимірювання, так що користувачу необхідно тільки глянути в центр кожного з 16 токенів. У нашому випадку він знаходиться в токені C4, так що відповідно до правил вибору токена, показаних на фіг. 6, користувач повинен вибрати цей токен, однак у прикладі користувач не вибрав токен C4.

- Після того, як користувач натискає на токен C4, його копіюють у п'яте положення коду доступу (G5 364).

- Шостим й останнім піном є: «☺», при цьому цей символ належить до четвертого вимірювання, а символи з четвертого вимірювання представлені винятково в нижній лівій стороні будь-якого токена, так що користувачу необхідно тільки перевірити нижню ліву сторону кожного з 16 токенів. У нашому випадку він знаходиться в токені C2, так що користувачу необхідно натиснути на C2. Однак у нашому випадку користувач не натиснув на C2, а замість цього цей користувач залишив останнє положення безбарвним і ввів тільки п'ять пінів. Це є невірним.

- Після того, як користувач ввів понад п'ять токенів, він натиснув на кнопку "Реєстрація входу" (G5 372) для того, щоб повідомити систему, що він завершив процес вибору токена, при цьому система здійснить перевірку для того, щоб довідатися, чи є діючими введені токени, відповідно до правил, показаних на фіг. 7.

- У даному прикладі введені користувачем токени є недійсними, оскільки первинний код доступу мав 6 пінів, а у нашому прикладі користувач ввів тільки п'ять токенів, так що запит користувача на одержання доступу був відхилений, а система відобразила повідомлення "Неуспішна реєстрація входу" (G5 382).

[116] На фіг. 13A показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5" у текстовому форматі. Фіг. 12A відображає порожній екран при запуску програми. Це зображення використовують як основу для порівняння з наведеними нижче фіг. 13B, 13C і 13D. Цей екран показаний винятково в якості пояснення того, що відбувається поза полем зору. Цей екран не показують під час реєстрації входу користувача в реальному часі.

[117] На фіг. 13B показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5" у текстовому форматі. Цей знімок екрана ілюструє те, що відбувається поза полем зору, коли відбувається процес реєстрації входу користувача, показаний на фіг. 12B, й ілюструє як виглядає послідовність операцій процесу, показана на фіг. 4, у типовому варіанті реалізації. Цей знімок екрана є тільки демонстраційною версією, при цьому його не показують під час реєстрації входу користувача в реальному часі. Цей знімок екрана використовують для візуалізації правил валідації токена, показаних на фіг. 7.

[118] Є 3 стовпця: "Клієнтська сторона" (ліва сторона), "Мережеве з'єднання" (середина) та "Серверна сторона" (права сторона). Процес починається з клієнтської сторони, коли користувач вводить ідентифікатор користувача, при цьому надалі інформацію передають через мережеве з'єднання на серверну сторону. Сервер генерує 16 токенів і передає їх у мережу, при цьому надалі ці токени передають на серверну сторону.

[119] Користувач вибирає токени відповідно до правил вибору токена, показаних на фіг. 6, при цьому надалі вибрані токени передають у мережу, а потім передають на серверну сторону

для їх валідації. Результат надання або відхилення доступу передають через мережу на серверну сторону. Послідовність операцій процесу зазначена стрілками на фіг. 13B. Процес переходить до наступного:

- Користувач уводить ідентифікатор користувача "адміністратор" (G5 406).
- Користувач натискає на "Введення" (G5 409).
- Ідентифікатор користувача "адміністратор" (G5 406) показують на серверній стороні (G5 411) та передають (G5 421) у мережеве з'єднання (G5 413), при цьому надалі його знову передають (G5 422) на серверну сторону (G5 415).
- На серверній стороні система здійснює перевірку своєї пам'яті для того, щоб довідатися, чи є ідентифікатор "адміністратор". Якщо він не існує, то система покаже повідомлення: "Ідентифікатор користувача не існує, введіть, будь-ласка, дійсний ідентифікатор користувача" (не показано). Використовують той же самий приклад, що і приклад, показаний на фіг. 11B, так що кодом доступу у пам'яті є: «$ = Ⓜ Ⓒ 2 ☺», при цьому система знаходить його у пам'яті (G5 417).
- Система генерує 16 токенів (G5 423) згідно з фіг. 5.
- 16 токенів передають (G5 424) у мережу.
- Мережа передала (G5 425) токени на серверну сторону.
- 16 токенів відображають на екрані для реєстрації входу користувача, як показано на фіг. 12B, у таблиці розміром 4 × 4 (G5 320).
- Користувач вибирає (G5 426) шість токенів: G5 350, G5 353, G5 356, G5 359, G5 362 і G5 365.
- Шість вибраних користувачем токенів передають (G5 427) у мережу після того, як користувач натискає "Зареєструвати вхід" (G5 370) на фіг. 12B.
- Шість вибраних користувачем токенів надалі передають (G5 428) на серверну сторону.
- На серверній стороні система здійснює перевірку всіх шести токенів один за одним: K11, K12, K13, K14, K15 і K16, при цьому у нашому прикладі всі вони є правильними.
- Вищеописаний результат успішної реєстрації входу (G5 429) передають (G5 430) у мережу.
- Мережа передає (G5 431) результат на серверну сторону та відображає (G5 432) повідомлення, продемонстроване у G5 380, що показаний на фіг. 12B.

[120] На фіг. 13C показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5" у текстовому форматі. Цей знімок екрана ілюструє те, що відбувається поза полем зору, коли відбувається процес реєстрації входу користувача, показаний на фіг. 12C, й ілюструє те, як виглядає послідовність операцій процесу, показана на фіг. 4, у типовому варіанті реалізації. Цей знімок екрана є тільки демонстраційною версією, при цьому його не показують під час реєстрації входу користувача в реальному часі. Цей знімок екрана використовують для візуалізації правил валідації токена, показаних на фіг. 7.

[121] Є 3 стовпця: "Клієнтська сторона" (ліва сторона), "Мережеве з'єднання" (середина) та "Серверна сторона" (права сторона). Процес починається з клієнтської сторони, коли користувач уводить ідентифікатор користувача, при цьому надалі інформацію передають через мережеве з'єднання на серверну сторону. Сервер генерує 16 токенів і передає їх у мережу, при цьому надалі ці токени передають на серверну сторону.

[122] Користувач вибирає токени відповідно до правил вибору токена, показаних на фіг. 6. Вибрані токени передають у мережу, при цьому надалі їх передають на серверну сторону для їх валідації, а результат надання або відхилення доступу передають через мережу на серверну сторону. Послідовність операцій процесу зазначена стрілками на фіг. 13C. Більше докладне пояснення наведене нижче.

- Користувач уводить ідентифікатор користувача: "адміністратор" (G5 506).
- Користувач натискає на кнопку "Введення" (G5 509).
- Ідентифікатор користувача "адміністратор" (G5 506) показують на серверній стороні (G5 511) та передають (G5 521) у мережеве з'єднання (G5 513), при цьому надалі його знову передають (G5 522) на серверну сторону (G5 515).
- На серверній стороні система здійснює перевірку своєї пам'яті для того, щоб довідатися, чи є ідентифікатор "адміністратор". Якщо він не існує, то система покаже повідомлення: "Ідентифікатор користувача не існує, введіть, будь-ласка, дійсний ідентифікатор користувача" (не показано). Використовують той же самий приклад, що і приклад, показаний на фіг. 11B, так що кодом доступу у пам'яті є: «$ = Ⓜ Ⓒ 2 ☺», при цьому система знаходить його у пам'яті (G5 517).
- Система генерує 16 токенів (G5 523) відповідно до правил генерації токена, показаних на фіг. 5.
- 16 токенів передають (G5 524) у мережу.

- Мережа передає (G5 525) токени на серверну сторону.

- 16 токенів відображають на екрані для реєстрації входу користувача, як показано на фіг. 12C, у таблиці розміром 4 × 4 (G5 321 на фіг. 12C).

- Користувач вибирає (G5 526) шість токенів: G5 351, G5 354, G5 357, G5 360, G5 363 і G5 366.

- Шість вибраних користувачем токенів передають (G5 527) у мережу після того, як користувач натискає "Зареєструвати вхід" (G5 371 на фіг. 12C).

- Шість вибраних користувачем токенів надалі передають (G5 528) на серверну сторону.

- На серверній стороні система здійснює перевірку всіх шести токенів один за одним: K21, K22, K23, K24, K25 і K26. У цьому прикладі щонайменше три вибрані токени є неправильними (користувачу було необхідно вибрати D2 токен, C3 токен і D1 токен відповідно для четвертого, п'ятого та шостого токенів, однак замість цього користувач вибрав C2 токен, B2 токен і токен C4, які є невірними. Таким чином, результат полягає у невдалій реєстрації входу).

- Вищеописаний результат невдалої реєстрації входу (G5 529) передають (G5 530) у мережу.

- Мережа передає (G5 531) результат на серверну сторону та відображає (G5 532) повідомлення, продемонстроване у G5 381, що показаний на фіг. 12C.

[123] На фіг. 13D показаний приклад знімка екрана, який ілюструє процес реєстрації входу користувача у варіанті реалізації "GATE_5" у текстовому форматі. Цей знімок екрана ілюструє те, що відбувається поза полем зору, коли відбувається процес реєстрації входу користувача, показаний на фіг. 12D, й ілюструє те, як виглядає послідовність операцій процесу, показана на фіг. 4, у типовому варіанті реалізації. Цей знімок екрана є тільки демонстраційною версією, при цьому його не показують під час реєстрації входу користувача в реальному часі. Цей знімок екрана використовують для візуалізації правил валідації токена, показаних на фіг. 7.

[124] Є 3 стовпця: "Клієнтська сторона" (ліва сторона), "Мережеве з'єднання" (середина) та "Серверна сторона" (права сторона). Процес починається з клієнтської сторони, коли користувач уводить ідентифікатор користувача, при цьому надалі інформацію передають через мережеве з'єднання на серверну сторону. Сервер генерує 16 токенів і передає їх у мережу, при цьому надалі ці токени передають на серверну сторону.

[125] Користувач вибирає токени відповідно до правил вибору токена, показаних на фіг. 6, при цьому надалі вибрані токени передають у мережу, а потім передають на серверну сторону для їх валідації. Результат надання або відхилення доступу передають через мережу на серверну сторону. Послідовність операцій процесу зазначена стрілками на фіг. 13D. Процес переходить до наступного:

- Користувач уводить ідентифікатор користувача "адміністратор" (G5 606).

- Користувач натискає на "Введення" (G5 609).

- Ідентифікатор користувача "адміністратор" (G5 606) показують на серверній стороні (G5 611) та передають (G5 621) у мережеве з'єднання (G5 613), при цьому надалі його знову передають (G5 622) на серверну сторону (G5 615).

- На серверній стороні система здійснює перевірку своєї пам'яті для того, щоб довідатися, чи є ідентифікатор "адміністратор". Якщо він не існує, то система покаже повідомлення: "Ідентифікатор користувача не існує, введіть, будь-ласка, дійсний ідентифікатор користувача" (не показано). Використовують той же самий приклад, що і приклад, показаний на фіг. 11B, так що кодом доступу у пам'яті є: «$ = Ⓜ Ⓒ 2 ☺», при цьому система знаходить його у пам'яті (G5 617).

- Система генерує 16 токенів (G5 623) відповідно до правил генерації токена, показаних на фіг. 5.

- 16 токенів передають (G5 624) у мережу.

- Мережа передала (G5 625) токени на серверну сторону.

- 16 токенів відображають на екрані для реєстрації входу користувача, як показано на фіг. 12D, у таблиці розміром 4 × 4 (G5 322).

- Користувач вибирає (G5 626) 5 токенів: G5 352, G5 355, G5 358, G5 361 і G5 364. Слід звернути увагу, що у G5 367 говориться "відсутні вхідні дані у формі символу «[☺]», що означає, що система очікує символ «☺» в якості останнього піна, так що повинен бути шостий токен, а вхідні дані з шостого токена все ще відсутні, що є помилкою, при цьому система відхилить надання користувачу доступу.

- П'ять вибраних користувачем токенів передають (G5 627) у мережу після того, як користувач натискає "Зареєструвати вхід" (G5 372) на фіг. 12D.

- П'ять вибраних користувачем токенів надалі передають (G5 628) на серверну сторону.

- На серверній стороні система перевіряє всі п'ять токенів один за одним: K31, K32, K33, K34

і K35. У нашому прикладі всі п'ять токенів є правильними, однак все-таки пропущений шостий токен (користувач не натиснув на C2, а замість цього користувач залишив останнє положення безбарвним і ввів тільки п'ять пінів), так що K36 отримав мітку [x], яка означає помилку. Це є невірним. Таким чином, результат полягає у невдалій реєстрації входу.

- Вищеописаний результат невдалої реєстрації входу (G5 629) передають (G5 630) у мережу.

- Мережа передає (G5 631) результат на серверну сторону та відображає (G5 632) повідомлення, продемонстроване у G5 382, що показаний на фіг. 12D.

[126] Цей спосіб може бути додатково розповсюджений і на наступну ознаку для того, щоб зробити завдання вгадування коду доступу ще більше складним: призначають конкретну кількість "схованих" пінів, так що коли ці сховані піни представлені у таблиці вибору, їх не слід вибирати. Замість цього користувач повинен вибрати будь-який інший токен, який не містить ці піни, а також повинен уникати токени, які містять ці піни.

[127] Таким чином, наприклад, якщо користувач має код доступу "123($#)456", довжина коду доступу становить 8, 2 піна в середині є схованими пінами, показаними між «(і)». Для пінів 1, 2, 3, 4, 5, 6 додержуються вищеописаних правил. Для «$» і «#»додержуються "правил схованого піна", які полягають у наступному: якщо жодні з них не представлені в жодному з 16 токенів, користувач може та повинен вибрати універсальний токен в їх місці, однак якщо жоден із них не представлений в одному з 16 токенів у таблиці вибору, користувач повинен уникати цього піна та вибрати тільки один із будь-яких інших 15 токенів.

[128] Для того, щоб сховати пін, може бути відображений нижчеописаний блок перевірки, розташований під кожним піном на екрані створення (реєстрації) коду доступу, так що коли користувач здійснює перевірку блока під піном, такий пін стає схованим, при цьому під час процесу валідації токена використовують перераховані вище правила для схованого піна для валідації реєстрації входу користувача.

[129] Крім того, даний винахід може бути використаний у будь-якому процесі зв'язку для прив'язки таблиці вибору з 16 токенами [створеними з використанням коду доступу відправника відповідно до правил генерації токена, показаних на фіг. 5] і ключа з деякими токенами до повідомлення. Якщо цей ключ є дійсним щодо цієї таблиці, то надалі це повідомлення є дійсним повідомленням. Якщо ключ є недійсним щодо таблиці вибору, то надалі прив'язане повідомлення є неправильним повідомленням. Шляхом збереження дійсного повідомлення та відкидання неправильного повідомлення, хто-небудь роздобуде остаточне [первісне] правильне повідомлення. Одержувач використовує той же самий код доступу для дешифрування повідомлення, так що процес може продовжитися аналогічно наступним прикладам:

- Повідомлення_1: "I will go home at 7 pm" (Я піду додому в 19:00) [+ таблиця токенів із 16 токенами + недійсний ключ]

→ Відхилити повідомлення

Повідомлення_2: "I will go home at 3 pm" (Я піду додому о 15:00) [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "I will go home at 3 pm" (Я піду додому в 15:00)

Повідомлення_3: "We will abandon attack on the 3rd" (Ми відмовимося від атаки на третій суб'єкт) [+ таблиця токенів із 16 токенами + недійсний ключ] → Відхилити повідомлення

Повідомлення_4: "We will attack at noon on the 3rd" (Ми зробимо напад опівночі на третій суб'єкт) [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "We will attack at noon on the 3rd" (Ми зробимо напад опівночі на третій суб'єкт

Таким чином, правильним остаточним повідомленням є: "I will go home at 3 pm" (Я піду додому о 15:00). "We will attack at noon on the 3rd" (Ми зробимо напад опівночі на третій суб'єкт).

- Повідомлення_1: "I" [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "I»

Повідомлення_2: "will" [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "will»

Повідомлення_3: "not" [+ таблиця токенів із 16 токенами + недійсний ключ] → Відхилити повідомлення

Повідомлення_4: "go" [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "go»

Таким чином, правильним остаточним повідомленням є: "I will go" (Я піду).

- Повідомлення_1: u [+ таблиця токенів із 16 токенами + недійсний ключ] → Відхилити повідомлення

Повідомлення_2: n [+ таблиця токенів із 16 токенами + недійсний ключ] → Відхилити повідомлення

Повідомлення_3: t [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "t»

Повідомлення_4: r [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "r»

Повідомлення_5: u [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "u»

Повідомлення_6: e [+ таблиця токенів із 16 токенами + дійсний ключ] ==> "e»

Таким чином, правильним остаточним повідомленням є: "true" (істина).

[130] Крім того, даний винахід може захистити від фішингу, який являє собою спробу одержати конфіденційну інформацію, таку як імена користувачів, паролі та дані кредитних карт, часто зі злим наміром, шляхом проникнення під виглядом благонадійного суб'єкта при встановленні зв'язку з використанням електронних засобів. Оскільки з використанням даного винаходу вся інформація, передана між користувачем і сервером, має форму токенів, а кожний токен містить множину символів, не відбувається розголошення самого піна користувача.

[131] На фіг. 14 показаний приклад знімка екрана, який ілюструє процес шифрування повідомлень з використанням варіанта реалізації "GATE_4". На фіг. 14 показаний приклад того, як відкрите текстове повідомлення зашифроване з використанням коду доступу відправника, і того, як може виглядати зашифроване повідомлення. Процес переходить до наступного:

- Відправник повідомлення вводить відкрите текстове повідомлення "секрет" у G4 700.
- Відправник вводить код доступу "123" у G4 702 і натискає на кнопку "Зашифрувати» [G4 703].
- Первинне повідомлення "секрет" змішують з деякими довільними знаками, що заповнюють, і перетворять у наведене далі результуюче повідомлення "sLeQWNcrMfYeMtHQr", показане у G4 704.
- Для дешифрування повідомлення одержувач буде використовувати той же самий код доступу "123" [G4 706] на стороні одержувача.

[132] На фіг. 15A показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4". На фіг. 15A показаний приклад того, як зашифроване первинне повідомлення може бути успішно дешифроване з використанням коду доступу одержувача. Процес переходить до наступного:

- Кожний знак у результуючому повідомленні [G4 704] прив'язаний до таблиці токенів розміром 4 × 4, при цьому кожний токен у таблиці має 4 символи, кожний знак у повідомленні також прив'язаний до "ключа", у ключ входять деякі токени, а кількість токенів у ключі може знаходитися в діапазоні від 2 до 6.
- Для дешифрування повідомлення одержувач використовує той же самий код доступу "123" [G4 706], при цьому дешифроване повідомлення показане у G4 704 у вигляді виділених кольором знаків. Результуючим дешифрованим повідомленням є "секрет" [G4 708].
- На фіг. 15A показаний приклад того, як це може виглядати для кожного знака. На знімку екрана перший знак у повідомленні "s" [G4 710] відображений як приклад після того, як користувач натискає на це [G4 710], а G4 712 ілюструє, що поточним знаком, що відображається є знак "s". Таблиця токенів розміром 4 × 4, прив'язана до цього символу, показана у таблиці G4 714.
- Ключі-токени, прив'язані до знака "s", також показані як G4 720, G4 722 і G4 724.
- Знаки, що заповнюють, у повідомленні: L, Q, W, N, M, f, Y, M, H, Q і r спеціально прив'язані до таблиць токенів і ключів, що є недійсними, так що вони будуть недійсними на стороні одержувача.
- У показаному прикладі користувач може натиснути на кожний знак у G4 704 для того, щоб показати його вміст і ключі-токени, при цьому надалі він натискає кнопку "Зберегти" [G4 726] для того, щоб довідатися, чи є дійсним зазначений знак. На знімку екрана продемонстровано, що знак "s" є дійсним, а перевірка була успішною [G4 730].

[133] На фіг. 15B показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4". Фіг. 15B ілюструє те, що відбувається поза полем зору, коли здійснюють процес, показаний на фіг. 15A, і те, як повідомлення валідують на стороні одержувача. Процес переходить до наступного:

- Знак повідомлення "s" [G4 750] зашифрований з використанням коду доступу відправника "123" [G4 752] і прив'язаний до 16 токенів [G4 754] і ключа з деякими токенами [G4 720, G4 722 і G4 724]. Цю інформацію відправляють у мережу [G4 756], а надалі одержувачу [G4 758].
- На стороні одержувача для декодування повідомлення використовують той же самий код доступу "123" [G4 760]. Ключі-токени проходять через [G4 762] процес валідації G4 764, G4 766 для перевірки кожного ключа-токена. З C51, C52 і C53 хто-небудь може побачити, що вони все є дійсними, так що приймають остаточне рішення про те [G4 768], що повідомлення є дійсним [G4 770], як показано на фіг. 15A [G4 730].

[134] На фіг. 16A показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4". На фіг. 16A показаний приклад того, як зашифроване повідомлення-заповнювач може бути дешифроване та розпізнане в якості недійсної інформації за допомогою коду доступу одержувача. Процес переходить до наступного:

- На фіг. 16A показаний приклад того, як це може виглядати для кожного заповнюючого знака, що не є частиною первинного повідомлення. На фіг. 16A показаний вміст другого знака "L" [G4 711] у повідомленні [G4 704] після того, як користувач натискає на нього. G4 713 демонструє, що знаком є "L". Таблиця розміром 4 × 4 із 16 токенами, прив'язаними до цього символу, показана у G4 715. Чотири ключі-токени, прив'язаних до знака "L", показані як G4 721, G4 723, G4 725 і G4 727.

- Оскільки знак "L" є заповнюючим знаком і не є частиною первинного повідомлення, відправник спеціально прив'язав його до таблиці токенів розміром 4 × 4 і ключа, які би не забезпечили валідацію. Хто-небудь може чітко розглянути, що код доступу відправника [G4 702] збігається з кодом доступу одержувача [G4 706] та містить 3 піна "1", "2" і "3", так що дійсний ключ повинен містити не більше і не менше трьох токенів. У даному прикладі він має чотири ключі-токени, так що він є недійсним, а знак "L" повинен бути зігнорований та не міг би являти собою частину результуючого дешифрованого повідомлення.

- На знімку екрана, коли користувач натискає кнопку "Перевірити" G4 726, продемонстровано, що процес валідації був невдалим [G4 731].

[135] На фіг. 16B показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4". Фіг. 16B ілюструє те, що відбувається поза полем зору, коли здійснюється процес, показаний на фіг. 16A, і те, як заповнююче повідомлення валідують на стороні одержувача. Процес переходить до наступного:

- Знак "L" у повідомленні [G4 751] шифрують з використанням коду доступу відправника "123" [G4 752] та прив'язують до таблиці розміром 4 × 4 із 16 токенами [G4 755] і ключа з деякими токенами [G4 721, G4 723, G4 725 і G4 727]. Наведену вище інформацію відправляють у мережу [G4 757], а надалі й одержувачу [G4 759].

- На стороні одержувача для декодування повідомлення використовують той же самий код доступу "123" [G4 760]. Ключі-токени проходять через [G4 763] процес валідації G4 765, G4 767 для перевірки кожного ключа-токена з C61, C62, C63 і C64. Хто-небудь може довідатися, що щонайменше два ключі-токени є недійсними.

- У третьому токені [G4 790] є третій код доступу "3", який і повинен бути вибраний. Однак, був вибраний восьмий токен [G4 792], показаний у третьому ключі-токені G4 725. Це є неправильним.

- Код доступу відправника збігається з кодом доступу одержувача та містить 3 піна, однак прив'язаний ключ містить чотири токени. Останній токен [G4 727] також є недійсним.

- Приймають остаточне рішення [G4 769], що повідомлення є недійсним [G4 771], як показано на фіг. 16A [G4 731].

[136] На фіг. 17 показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_4". На фіг. 17 показаний приклад того, як зашифроване первинне повідомлення може і не бути успішно дешифроване за допомогою коду доступу одержувача, що відрізняється від коду доступу відправника. Процес переходить до наступного:

- Користувач, що надрукував відкрите текстове повідомлення "секрет" у G4 700, надалі ввів код доступу "123" [G4 702] і натиснув на кнопку "Зашифрувати" [G4 703].

- Повідомлення шифрують, відправляють, приймають та демонструють у G4 705 як: "sLeQWNcrMfYeMtHQr" [G4 705].

- Одержувач використовує код доступу "567" [G4 707] для дешифрування прийнятого від відправника повідомлення, яке було зашифроване з використанням коду доступу "123" [G4 702].

- Дешифроване повідомлення виділене кольором у G4 705: "ecrQ".

- Результуюче повідомлення показане як "ecrQ" [G4 709].

- Результуюче повідомлення відрізняється від первісного повідомлення від відправника: "секрет" [G4 700], оскільки одержувач використовував інший код доступу для дешифрування повідомлення.

[137] На фіг. 18 показаний приклад знімка екрана, який ілюструє процес шифрування повідомлень з використанням варіанта реалізації "GATE_5". На фіг. 18 показаний приклад того, як відкрите текстове повідомлення зашифроване з використанням коду доступу відправника, і те, як може виглядати зашифроване повідомлення. Процес переходить до наступного:

- Відправник повідомлення вводить відкрите текстове повідомлення "FYEO" у G5 700.

- Відправник вводить код доступу "123" у G5 702 і натискає кнопку "Зашифрувати" [G5 703].

- Первинне повідомлення "FYEO" змішують з деякими довільними заповнюючими знаками та перетворюють у наведене далі результуюче повідомлення "FlPRojcYnEbAO" [G5 704].

- Одержувач буде використовувати той же самий код доступу "123" [G5 706] на стороні одержувача для дешифрування повідомлення.

[138] На фіг. 19A показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5". На фіг. 19A показаний приклад того, як зашифроване первинне повідомлення може бути успішно дешифроване з використанням коду доступу одержувача. Процес переходить до наступного:

- Кожний знак у прийнятому повідомленні [G5 704] прив'язаний до таблиці токенів розміром 4 × 4, при цьому кожний токен у таблиці містить 5 символів, кожний знак у повідомленні також прив'язаний до "ключа", у ключ входять деякі токени, а кількість токенів у ключі може знаходитися в діапазоні від 2 до 6.

- Для декодування повідомлення одержувач використовує той же самий код доступу "123" [G5 706], при цьому дешифроване повідомлення показане у G5 704 у вигляді виділених кольором знаків. Результуючим дешифрованим повідомленням є "FYEO" [G5 708].

- На фіг. 19A показаний приклад того, як це може виглядати для кожного знака. На знімку екрана перший знак "F" у повідомленні [G5 710] відображений як приклад. G5 712 ілюструє те, що поточним знаком, що відображається є "F", а таблиця токенів розміром 4 × 4, прив'язана до цього символу, показана у таблиці G5 714.

- Ключі-токени, прив'язані до знака "F", також показані як G5 720, G5 722 і G5 724.

- Знаки, що заповнюють, у повідомленні: l, P, R, o, j, c, n, b і A спеціально прив'язані до таблиць токенів і ключів, що є недійсними, так що вони будуть недійсними на стороні одержувача.

- У даному прикладі користувач може натиснути на кожний знак у G5 704 для того, щоб показати його вміст і ключі-токени, при цьому надалі він натискає кнопку "Перевірити" [G5 726] для того, щоб довідатися, чи є дійсним зазначений знак. На знімку екрана продемонстровано, що знак "F" є дійсним, а перевірка була успішною [G5 730].

[139] На фіг. 19B показаний приклад знімка екрана, що ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5". Це ілюструє те, що відбувається поза полем зору, коли відбувається процес, показаний на фіг. 19A, і те, як повідомлення валідують на стороні одержувача. Процес переходить до наступного:

- Знак "F" у повідомленні [G5 750] шифрують з використанням коду доступу відправника "123" [G5 752] та прив'язують до таблиці розміром 4 × 4 із 16 токенами [G5 754] і ключем із деякими токенами [G5 720, G5 722 і G5 724]. Цю інформацію відправляють у мережу [G5 756], а надалі й одержувачу [G5 758].

- На стороні одержувача для дешифрування повідомлення він використовує той же самий код доступу "123" [G5 760]. Ключі-токени проходять через [G5 762] процес валідації G5 764, G5 766 для перевірки кожного ключа-токена. З K51, K52 і K53 хто-небудь може побачити, що всі вони є дійсними, так що приймають остаточне рішення про те [G5 768], що повідомлення є дійсним [G5 770], як показано на фіг. 19A [G5 730].

[140] На фіг. 20A показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5". На фіг. 20A показаний приклад того, як зашифроване повідомлення-заповнювач може бути дешифроване та розпізнане в якості недійсної інформації за допомогою коду доступу одержувача. Процес переходить до наступного:

- На фіг. 20A показаний приклад того, як це може виглядати для кожного заповнюючого знака, що не є частиною первісного повідомлення. Фіг. 20A ілюструє вміст третього знака "P" [G5 711] у повідомленні [G5 704]. G5 713 демонструє, що знаком є "P", а таблиця розміром 4 × 4 із 16 токенами, прив'язаними до цього символу, показана у G5 715. Три ключі-токени, прив'язаних до знака "P", показані як G5 721, G5 723 і G5 725.

- Оскільки знак "P" є заповнюючим знаком і не є частиною первинного повідомлення, відправник спеціально прив'язав його до таблиці токенів розміром 4 × 4 і ключа, який не забезпечив би валідацію, оскільки хто-небудь може чітко розгледіти, що код доступу відправника [G5 702] збігається з кодом доступу одержувача [G5 706] та містить 3 піна "1", "2" і "3". Перший пін "1" у коді доступу представлений в останньому токені у таблиці розміром 4 × 4 [G5 716], так що токен повинен бути вибраний в якості першого ключа-токена. Однак, другий токен [G5 718] у таблиці був вибраний та показаний у положенні G5 721 першого ключа-токена. Це є неправильним, що не дозволило би визнати недійсним це повідомлення.

- На знімку екрана, коли користувач натискає кнопку "Перевірити" G5 726, продемонстровано, що процес валідації був невдалим [G5 731] для цього знака "P".

[141] На фіг. 20B показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5". Фіг. 20B ілюструє те, що

відбувається поза полем зору, коли відбувається процес, показаний на фіг. 20A, і те, як заповнююче повідомлення валідують на стороні одержувача. Процес переходить до наступного:

- Знак повідомлення "P" [G5 751] шифрують з використанням коду доступу відправника "123" [G5 752] та прив'язують до таблиці розміром 4 × 4 із 16 токенами [G5 755] і ключа з деякими токенами [G5 721, G5 723 і G5 725]. Цю інформацію відправляють у мережу [G5 757], а надалі й одержувачу [G5 759].

- На стороні одержувача для дешифрування повідомлення він використовує той же самий код доступу "123" [G5 760]. Ключі-токени проходять через [G5 763] процес валідації G5 765, G5 767 для перевірки кожного ключа-токена. З K61, K62 і K63 хто-небудь може довідатися, що перший токен є недійсним.

- В останньому токені [G5 790] представлений перший код доступу "1", який повинен бути вибраний. Однак був вибраний другий токен [G5 792], показаний у положенні [G5 721] першого ключа-токена. Таким чином, він є недійсним.

- Приймають остаточне рішення [G5 769], що повідомлення є недійсним [G5 771], як показано на фіг. 20A [G5 731].

[142] На фіг. 21 показаний приклад знімка екрана, який ілюструє процес дешифрування повідомлень з використанням варіанта реалізації "GATE_5". На фіг. 21 показаний приклад того, як зашифроване первинне повідомлення може і не бути успішно дешифроване за допомогою коду доступу одержувача, що відрізняється від коду доступу відправника. Процес переходить до наступного:

- Користувач, що надрукував відкрите текстове повідомлення "FYEO" у G5 700, надалі ввів код доступу "123" [G5 702] та натиснув на кнопку "Зашифрувати" [G5 703].

- Повідомлення шифрують, відправляють, приймають та надалі показують у G5 705 у наступному вигляді: "FlPRojcYnEbAO" [G5 705].

- Одержувач використовує код доступу "680" [G5 707] для дешифрування повідомлення, прийнятого від відправника та зашифрованого з використанням коду доступу "123" [G5 702].

- Дешифроване повідомлення виділяють кольором у G5 705: "nE".

- Результуюче повідомлення показують як "nE" [G5 709].

- Результуюче повідомлення відрізняється від первинного повідомлення від відправника: "FYEO" [G5 700], оскільки одержувач використовував інший код доступу для дешифрування повідомлення.

[143] Для кожного піна у коді доступу для генерування дійсних ключів-токенів щодо таблиці розміром 4 × 4 із 16 токенами для кожного дійсного повідомлення переважно використовують наступні етапи, згідно з якими:

- Здійснюють перебір всіх 16 токенів: (a) якщо пін виявлений в токені, вибирають цей токен; й (b) якщо пін не представлений у будь-якому токені, вибирають довільний токен із 16 токенів у таблиці.

[144] Для генерування недійсних ключів-токенів щодо таблиці розміром 4 × 4 із 16 токенами для кожного недійсного повідомлення переважно використовують наступні етапи, згідно з якими:

<1> Задають булеву змінну "Done_Fixing" у значення "брехня".

<2> Перетворюють всі 16 токенів, при цьому виконують нижче описані етапи <3> і <4> для кожного піна у коді доступу.

<3> <A> Якщо пін виявлений у токені:

(1) Якщо булева змінна "Done_Fixing" рівна "брехня", вибирають будь-який інший токен за винятком цього токена для того, щоб навмисне вибрати неправильний токен, а також задають булеву змінну "Done_Fixing" у значенні "істина".

(2) Якщо булева змінна "Done_Fixing" рівна "істина", вибирають цей токен.

<B> Якщо пін не представлений у будь-якому токені, вибирають довільний токен із 16 токенів.

<4> Зберігають ключ-токен, згенерований вище у вектор.

<5> Генерують довільне число N у наступному діапазоні: -1 до 1.

<A> Якщо N = -1, видаляють останній ключ-токен із вектора.

<B> Якщо N = 0, нічого не роблять.

<C> Якщо N = 1, а довжина піна користувача < 6, додають довільний токен із 16 токенів у таблиці до вектора.

<6> Токени у векторі будуть являти собою остаточні ключи-токени.

[145] Вищеописані варіанти реалізації та переваги є по суті ілюстративними, при цьому не слід вважати, що вони обмежують даний винахід. Опис даного винаходу слід вважати

ілюстративним, при цьому не слід вважати, що він обмежує обсяг формули винаходу. Фахівцям в даній області техніки будуть очевидні численні альтернативи, модифікації та зміни. Різні зміни можуть бути виконані без виходу за межі сутності й обсягу даного винаходу, заданого у прикладеній формулі винаходу.

[146] Наприклад, незважаючи на те, що даний винахід був описаний стосовно до варіантів реалізації "GATE_4" і "GATE_5", в яких використовують відповідно чотири вимірювання та п'ять вимірювань символів, може бути використана будь-яка кількість вимірювань (у тому числі тільки одне вимірювання) без виходу за межі обсягу даного винаходу. В цілому, оскільки кожний токен має більше одного символу, то може бути використана будь-яка кількість символів, розподілених у будь-якій кількості вимірювань. Крім того, слід вважати, що вищеописані варіанти реалізації "GATE_4" і "GATE_5" та відповідні знімки екрана є ілюстративними і не обмежують обсяг даного винаходу.

ФОРМУЛА ВИНАХОДУ

1. Спосіб надання користувачу доступу до збереженої в електронному вигляді інформації ("аутентифікації") з використанням збереженого в електронному вигляді заданого коду доступу ("коду доступу"), що містить задану кількість символів ("символів коду доступу"), вибраних із попередньо вибраної сукупності символів, при цьому кожний з символів коду доступу характеризується заданим положенням піна, зазначений спосіб включає:
надання, за допомогою користувацького інтерфейсу (120) електронного пристрою (150), користувачу сукупності токенів, яка містить щонайменше два токени і в якій кожний токен із сукупності токенів містить щонайменше два символи, що належать до зазначеної сукупності символів;
направлення, за допомогою користувацького інтерфейсу, користувачу запиту на вибір токена із зазначеної сукупності токенів для кожного положення піна в коді доступу; і
здійснення аутентифікації користувача на підставі вибраних користувачем токенів, при цьому користувач є аутентифікованим, якщо:
кількість токенів, вибраних користувачем, дорівнює кількості символів у коді доступу,
щонайменше один із вибраних токенів містить відповідний один із символів коду доступу, і
для кожного положення піна символ коду доступу в зазначеному положенні піна або міститься в токені, вибраному для цього положения піна, або не міститься ні в якому з токенів сукупності токенів, наданих користувачу.

2. Спосіб за п. 1, згідно з яким кількість символів у зазначеній сукупності символів дорівнює кількості токенів, наданих користувачу.

3. Спосіб за п. 1, згідно з яким кількість символів у зазначеній сукупності символів більше кількості токенів, наданих користувачу.

4. Спосіб за п. 1, згідно з яким зазначена сукупність символів розділена щонайменше на дві підсукупності ("вимірювання"), і кожний токен містить символ із кожного із зазначених щонайменше двох вимірювань.

5. Спосіб за п. 1, згідно з яким кожний токен містить чотири символи, які належать зазначеній сукупності символів, і додатково кожна сукупність символів розділена на чотири підсукупності ("вимірювання"), при цьому кожний токен містить символ із кожного вимірювання символів.

6. Спосіб за п. 1, згідно з яким кожний токен містить п'ять символів, які належать зазначеній сукупності символів, і додатково кожна сукупність символів розділена на п'ять підсукупностей ("вимірювань"), при цьому кожний токен містить символ із кожного вимірювання символів.

7. Спосіб за п. 1, згідно з яким зазначена сукупність символів заснована на системі кодування символів за стандартом "Юнікод".

8. Система для надання користувачу доступу до збереженої в електронному вигляді інформації ("аутентифікації") з використанням збереженого в електронному вигляді заданого коду доступу ("коду доступу"), що містить задану кількість символів ("символів коду доступу"), вибраних із попередньо вибраної сукупності символів, при цьому кожний з символів коду доступу характеризується заданим положенням піна, зазначена система містить:
процесор;
пам'ять, до якої має доступ процесор; і
модуль (110) аутентифікації шифрування, що містить сукупність інструкцій, що читаються комп'ютером, які збережені у пам'яті та виконуються процесором для того, щоб:
надати користувачу сукупність токенів, яка містить щонайменше два токени і в якій кожний токен містить щонайменше два символи, що належать зазначеній сукупності символів;

направляти користувачу запит на вибір, за допомогою користувацького інтерфейсу (120), токена із зазначеної сукупності токенів для кожного положення піна в коді доступу; й
аутентифікувати користувача на підставі токенів, які вибрав користувач, при цьому процесор визначає, що користувач є аутентифікованим, якщо:
кількість токенів, вибраних користувачем, дорівнює кількості символів у коді доступу,
щонайменше один із вибраних токенів містить відповідний один із символів коду доступу, і
для кожного положення піна символ коду доступу в зазначеному положенні піна або міститься в токені, вибраному для цього положення піна, або не міститься ні в якому з токенів сукупності токенів, наданих користувачу.

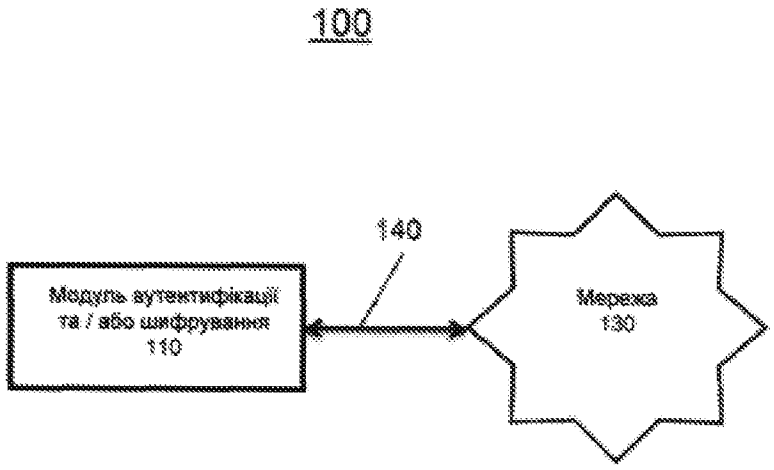
9. Система за п. 8, в якій кількість символів у зазначеній сукупності символів дорівнює кількості токенів, наданих користувачу, або в якій кількість символів у зазначеній сукупності символів більше кількості токенів, наданих користувачу.

10. Система за п. 8, в якій зазначена сукупність символів розділена щонайменше на дві підсукупності ("вимірювання"), і кожний токен містить символ із кожного із зазначених щонайменше двох вимірювань.

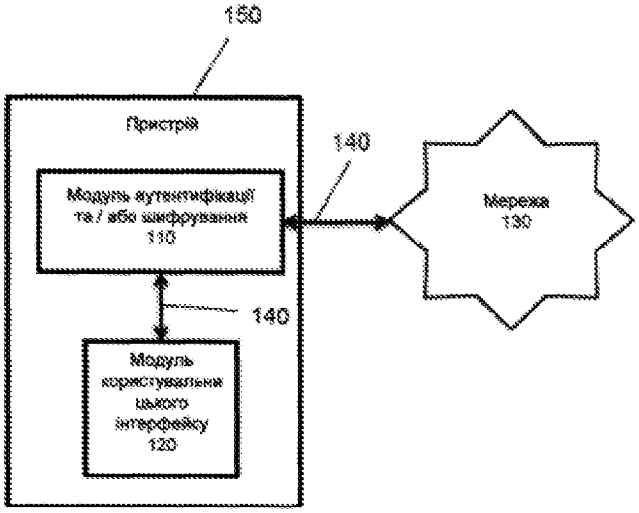
11. Система за п. 8, в якій кожний токен містить чотири символи, що належать зазначеній сукупності символів, причому кожна сукупність символів розділена на чотири підсукупності ("вимірювання"), а кожний токен містить символ із кожного вимірювання символів.

12. Система за п. 8, в якій кожний токен містить п'ять символів, що належать зазначеній сукупності символів, і в якій додатково кожна сукупність символів розділена на п'ять підсукупностей ("вимірювання"), а кожний токен містить символ із кожного вимірювання символів.

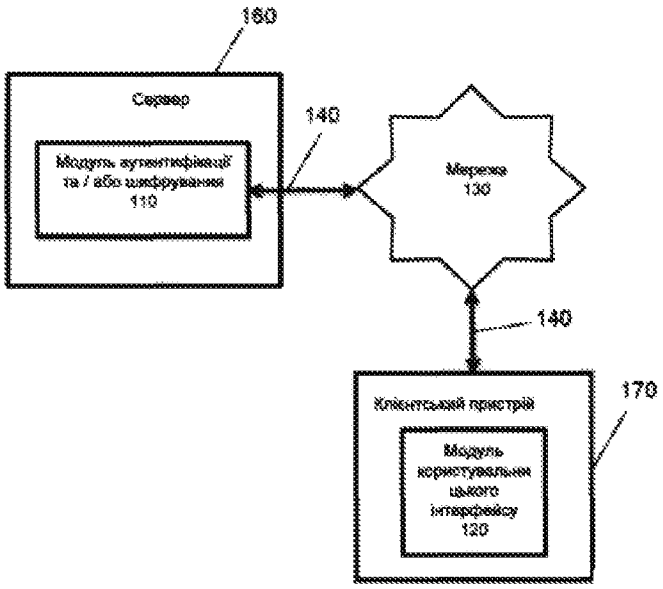
13. Система за п. 8, в якій зазначена сукупність символів заснована на системі кодування символів за стандартом "Юнікод".



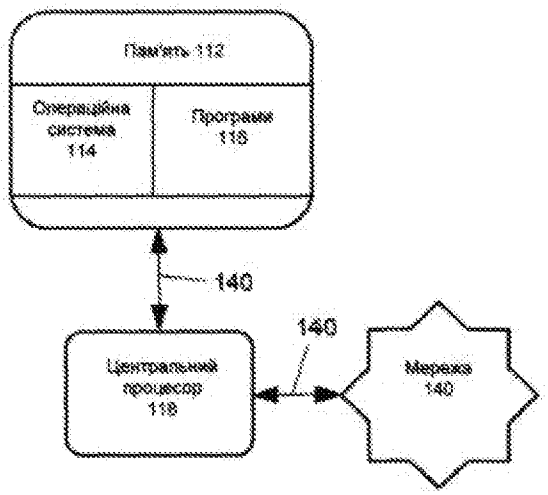
Фіг. 1А


150
Пристрій
Модуль аутентифікації та / або шифрування
110
140
Мережа
130
140
Модуль користувальницького інтерфейсу
120

Фіг. 1B


160
Сервер
Модуль аутентифікації та / або шифрування
110
140
Мережа
130
140
170
Клієнтський пристрій
Модуль користувальницького інтерфейсу
120

Фіг. 1C



Фіг. 1D

(GATE_4)

Перше кодування

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36

Друге кодування

Третє кодування

Четверте кодування

Фіг. 2A

(GATE_5)

Перше кодування

Друге кодування

α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ ς σ τ υ φ χ ψ ω ϟ

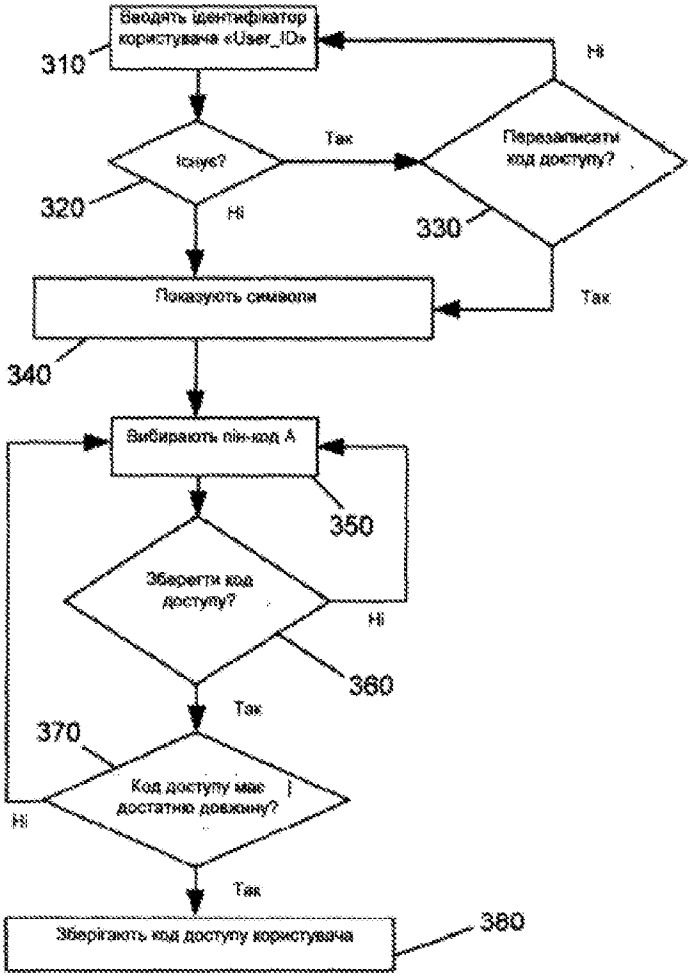
Третє кодування

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

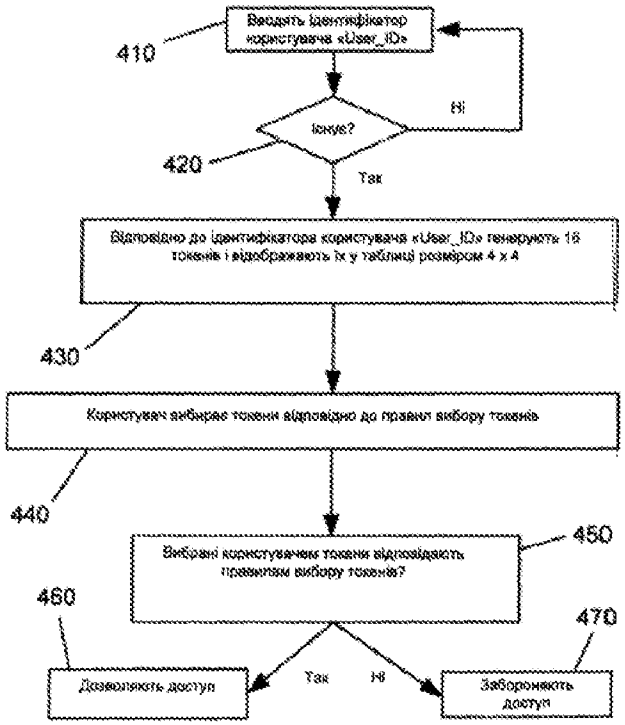
Четверте кодування

П'яте кодування

Фіг. 2B


Вводять ідентифікатор користувача «User_ID»
310
Ні
Існує?
Так
320
Ні
Перезаписати код доступу?
330
Так
Показують символи
340
Вибирають пін-код А
350
Зберегти код доступу?
Ні
360
Так
370
Код доступу має достатню довжину?
Ні
Так
Зберігають код доступу користувача
380

Фіг. 3


Вводять ідентифікатор користувача «User_ID»
410
Ні
Існує?
420
Так
Відповідно до ідентифікатора користувача «User_ID» генерують 16 токенів і відображають їх у таблиці розміром 4 x 4
430
Користувач вибирає токени відповідно до правил вибору токенів
440
Вибрані користувачем токени відповідають правилам вибору токенів?
450
460
470
Так
Ні
Дозволяють доступ
Забороняють доступ

Фіг. 4

(Правила генерування токенів)

[В якості ілюстрації використаний варіант реалізації GATE_4 (див. Фіг. 2А)]
Завдання полягає в генеруванні 16 токенів без наявності в них зайвих символів, при цьому кожний токен матиме 4 символи, по одному з кожного вимірювання, наприклад: Token_1, Token_2, ... Token_16. В одному з 16 токенів повинен бути присутнім ідентифікатор одно пін-код користувача, а можливо і більша кількість пін-кодів, а можливо і всі пін-коди.
Слід звернути увагу на порядок наведених вище символів: перший символ - завжди з першого вимірювання, а другий символ - з другого вимірювання ...

Правила й етапи

[1] Створюють 4 вектори, кожний з яких містить 36 символів з одного з чотирьох вимірювань, описаних на фіг. 2. Доступні символи для варіанта реалізації GATE_4. Кожний раз, коли з вектора будуть видалятися символи, його розмір зменшується на одиницю. Причина видалення символів із векторів полягає в запобіганні дублювання. Будемо називати вищеописані вектори як розширені вектори: V_1, V_2, V_3, V_4. Кожний етап генерування токена буде видаляти один запланованій символ з кожного з 4 вищеописаних векторів, так що ми не отримаємо і двох токенів, які будуть мати одні і ті самі символи.
[2] Отримують ідентифікатор користувача «User_ID», введений користувачем під час процесу реєстрації акаунт.
[3] Отримують код доступу ** з ідентифікатора користувача «User_ID», збереженого у пам'яті під час реєстрації.
[4] Зберігають довільний пін-код з коду доступу користувача у вектор пін-кодів користувача «User_Pin_Vector» ***
[5] Зберігають довільний номер (від 1 до 16) у положення (частоти) пін-коду користувача «User_Pin_Show_Up_Location».
[6] Для (i = 1; i <= 16; i ++) виконують етапи [7] і [8] для генерування 16 токенів.
[7] Створюють порожній токен: вектор для зберігання 4 символів.
[8] Для (j = 1; j <= 4; j ++) виконують етап [8.1] або етап [8.2] для додавання в токен одного символу у кожній розмірності. Етап [8.1] забезпечує використання щойновибраної першого пін-коду користувача.
[8.1] Якщо і дорівнює «User_Pin_Show_Up_Location», а «i»-е з «User_Pin_Vector» все ще знаходиться в «V_j», його видаляють з «V_j» і додають у поточний токен.
[8.2] В іншому випадку видаляють один довільний символ із «V_j» для його додавання у поточний токен. Розмір «V_j» зменшується на одиницю слідом за видаленням цього символу.
[9] Після виконання вищеописаних етапів ми будемо мати у своєму розпорядженні 16 токенів щонайменше з одним пін-кодом користувача.

Приклади результатів

[1] V_1 (1, 2, ... 36, 36)
V_2(①, ②, ..., ?)
V_3(♤, ♠, ... ♡, ♥)
V_4(+, -, ... $, ∞)

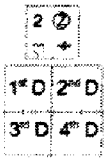
[2] admin
[3] ① ♥ 2 ✉
[4] ♥
[5] 7
[6]
[7] Token_i ()
[8] Додати в масив 4 символи.
[8.1] Token_7
[8.2] Token_j
[9] Token_1, ..., Token_16

Примітки:
* Пін-код - кожний символ у коді доступу ** є пін-кодом. Наприклад символ ① є першим пін-кодом, а символ ✉ є четвертим пін-кодом у наступному коді доступу: ①♥2✉
** код доступу - складний зі звичайними паролем, однак містить символи з результатів вибору в кожному вимірюванні (наприклад, ① ♥ 2✉), при цьому вимірювання описані на фіг. 2А і 2В.
*** «User_Pin_Vector» - вектор комп'ютерних мовою «Java», який містить будь-яку кількість елементів будь-якого типу, у даному випадку - символи.

Фіг. 5

(Правила вибору токенів)

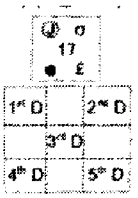
[1] Користувачу в таблиці розміром 4 x 4 буде показано 16 токенів, кожний з яких буде містити 4 символи, наприклад токен буде зображений таким чином:
Кожне вимірювання має фіксоване положення в токені, при цьому верхнє ліве положення призначене для символів із першого вимірювання, верхнє праве положення призначене для символів із другого вимірювання, нижнє ліве положення призначене для символів із третього вимірювання, а нижнє праве положення призначене для символів із четвертого вимірювання. Фіксоване положення буде допомагати користувачу швидко визначити місцезнаходження символів у токені.



A	13 &	34 Ⓜ	29 Ⓟ	28 !
B	3 Ⓠ	36 Ⓥ	6 :	19 Ⓦ
C	20 Ⓩ	26 Ⓙ	17 Ⓢ	12 Ⓒ
D	32 Ⓚ	35 Ⓕ	16 Ⓑ	10 Ⓤ

4

Третій пін-код користувача також пропущений, так що можна вибрати B1: (). Однак четвертий пін-код користувача ✉ знаходиться в токені A1, при цьому користувач повинен вибрати його з тим, щоб він був дійсним, так що можна вибрати A1: (). Таким чином, в кінці користувач вибрав наступні чотири токена: (), (), () і (), при цьому ці 4 токени будуть відправлені на сервер для їх валідації.



Фіг. 6

(Правила валідації токенів)

Для варіанта реалізації GATE_4: продовжимо з прикладу, показаного на фіг. 6.

[1] Користувач повинен слідувати порядку коду доступу користувача ⓘ ♥ 2 ✉ для вибору токенів, які містять пін-коди користувача.

[2] У розташованій праворуч таблиці показані початкові 16 токенів.

[3] Користувачем були вибрані наступні 4 токена: (34 Ⓜ ◊ ♫), (16 Ⓑ ● ☜), (3 Ⓠ ← Ⓡ) і (13 & ☺ ✉).

[4] Процес валідації перевірить їх по черзі, при цьому якщо один із них не пройде перевірку, то запит користувача на реєстрацію входу буде відхилений.

[5] Якщо користувач вибрав більшу або меншу кількість токенів у порівнянні з кількістю пін-кодів у своєму коді доступу, то запит на реєстрацію входу також буде відхилений.

[6] У даному прикладі реалізації довжина кодів доступу користувача може досягати до 6 пін-кодів.

[7] Оскільки першим пін-кодом користувача є ⓘ, необхідно буде перебрати всі символи у всіх 16 токенах і дізнатися, чи існує символ ⓘ.

	1	2	3	4
A	13 & ☺ ✉	34 Ⓜ ◊ ♫	29 Ⓟ → ☢	28 ! ✓ ÷
B	3 Ⓠ ← Ⓡ	36 Ⓥ = ◒	6 : ↓ π	19 Ⓦ X +
C	20 Ⓩ ☪ ☚	26 Ⓙ ₣ €	17 Ⓢ ♤ ×	12 Ⓒ ♡ ▣
D	32 Ⓚ ♠ $	35 Ⓕ ☆ ✹	16 Ⓑ ● ☜	10 Ⓤ ☪ ☎

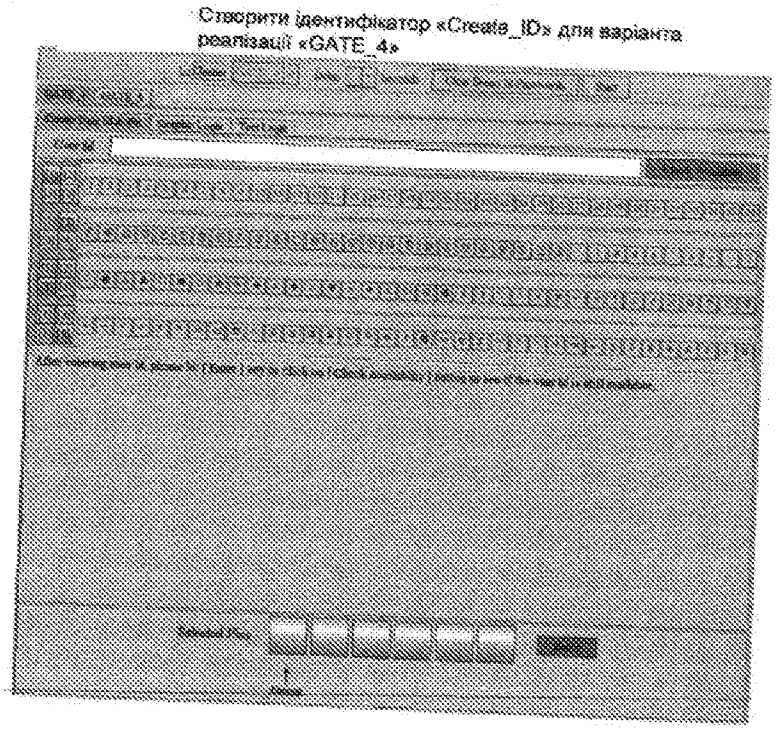
Оскільки цей символ не існував, користувач може та повинен вибрати універсальний токен на його місці, при цьому у нашому прикладі вибраний користувачем перший токен (34 Ⓜ ◊ ♫) є дійсним.

[8] У всіх 16 токенах також відсутній другий пін-код, так що другим вибраним користувачем токеном є: токен (16 Ⓑ ● ☜) є дійсним, те саме справедливо і для третього вибраного користувачем токена (3 Ⓠ ← Ⓡ).

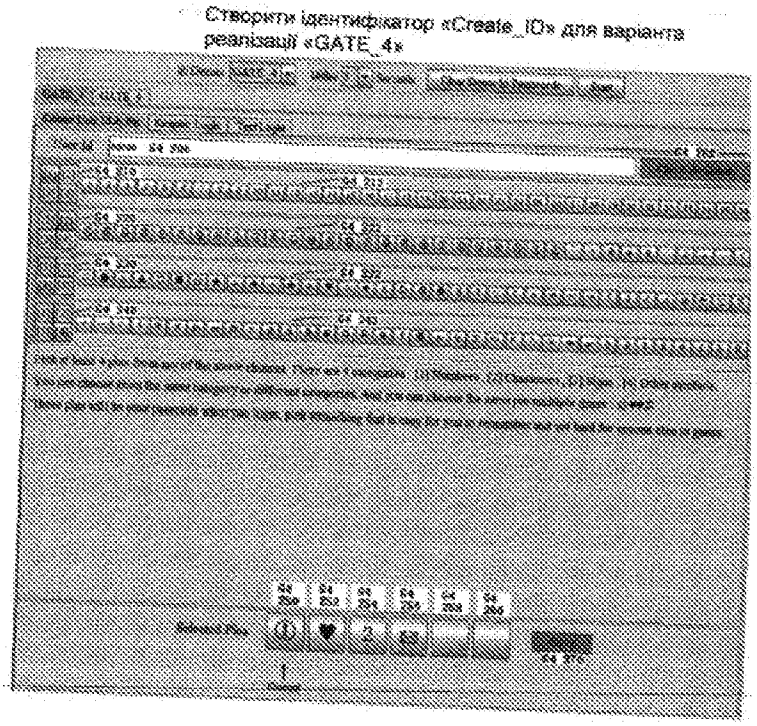
[9] Четвертим пін-кодом був символ ✉, при цьому при переборі у перерахованих вище 16 токенах всіх символів, ми можемо дізнатися, що цей символ знаходиться в токені А1, так що для того, щоб він був дійсним, користувач повинен і може вибрати тільки токен А 1, при цьому у нашому прикладі він вибрав токен (13 & ☺ ✉) в якості свого четвертого й останнього токена, а в його первинному коді доступу є не більше та не менше чотирьох пін-кодів, в результаті чого запит користувача на реєстрацію доступу був задоволений.

Для варіанта реалізації «GATE_5» використовують той же самий процес, при цьому просто додають ще одне вимірювання.

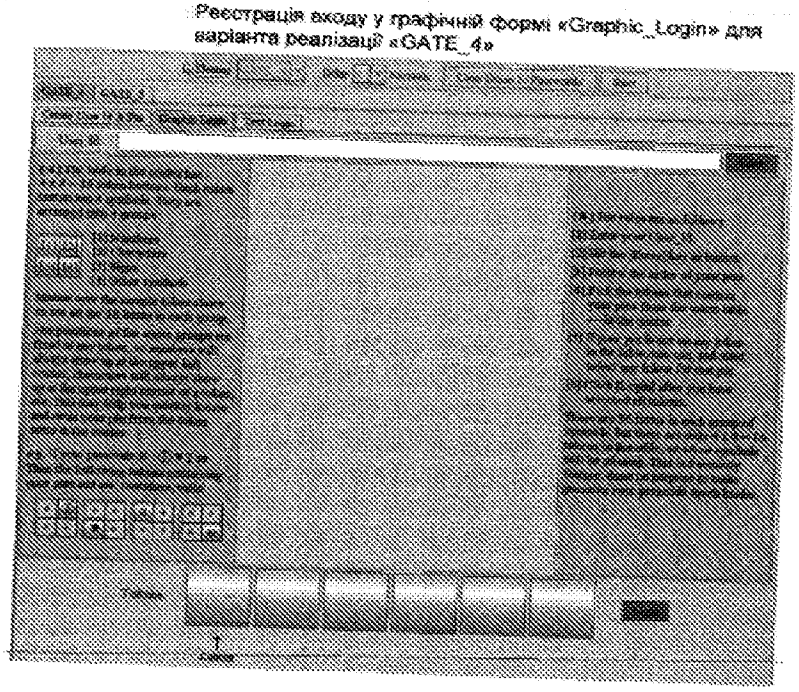
Фіг. 7



Фіг. 8А

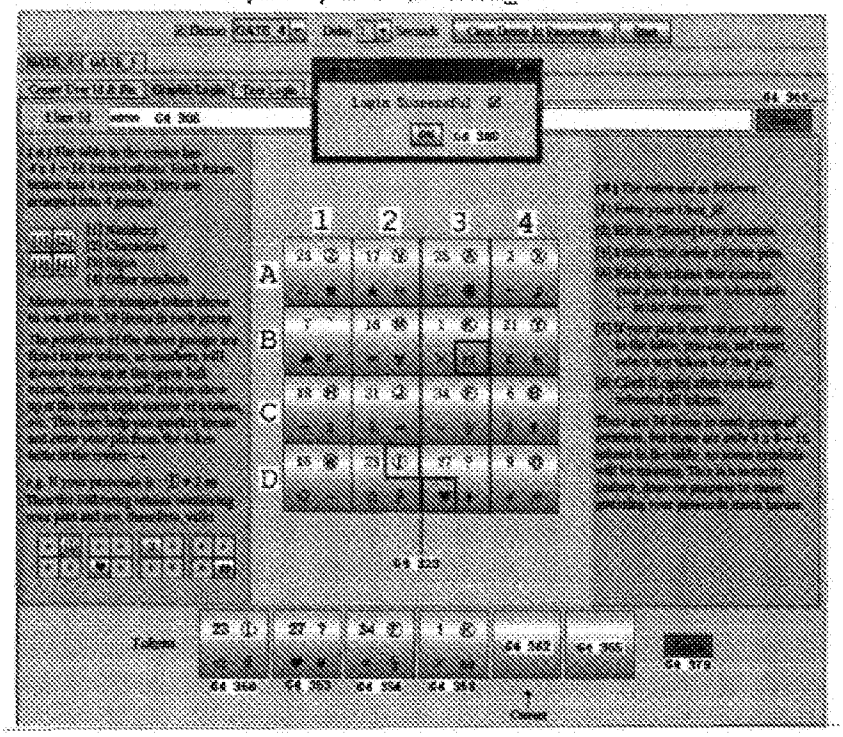

Створити ідентифікатор «Create_ID» для варіанта реалізації «GATE_4»

Фіг. 8B


Реєстрація входу у графічній формі «Graphic_Login» для варіанта реалізації «GATE_4»

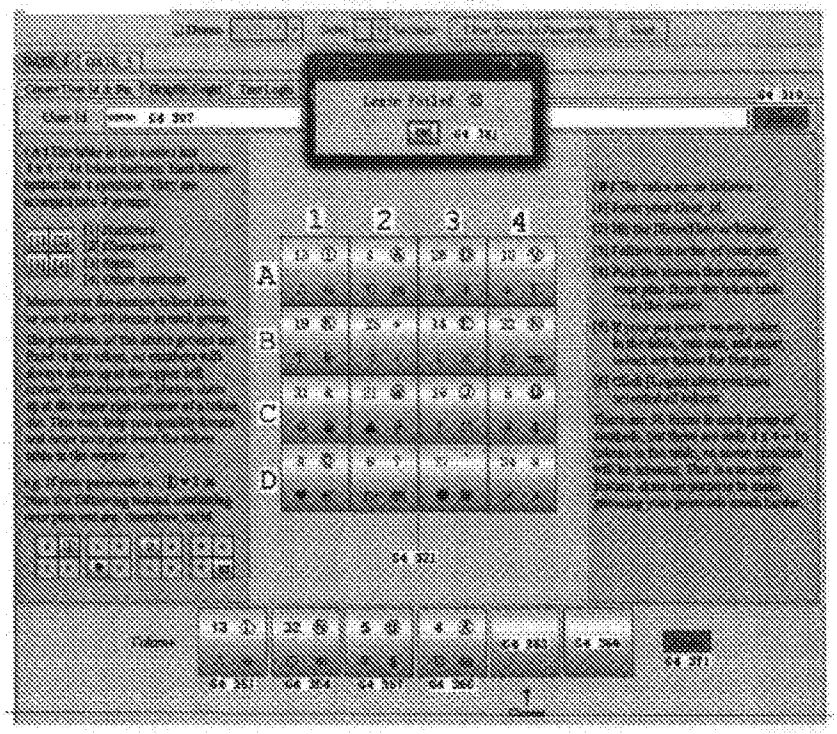
Фіг. 9A

Реєстрація входу у графічній формі «Graphic_Login» для варіанта реалізації «GATE_4»



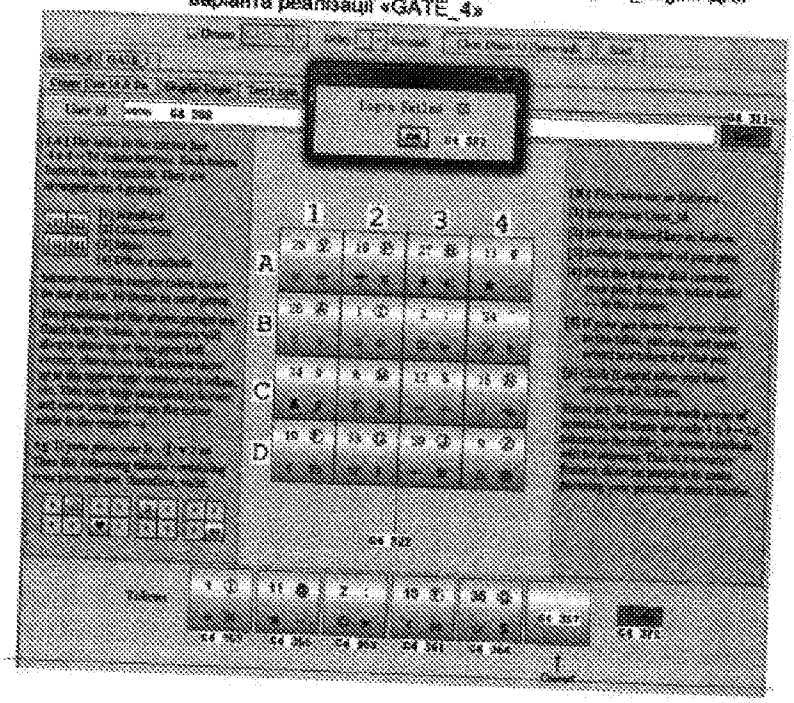
Фіг. 9B

Реєстрація входу у графічній формі «Graphic_Login» для варіанта реалізації «GATE_4»



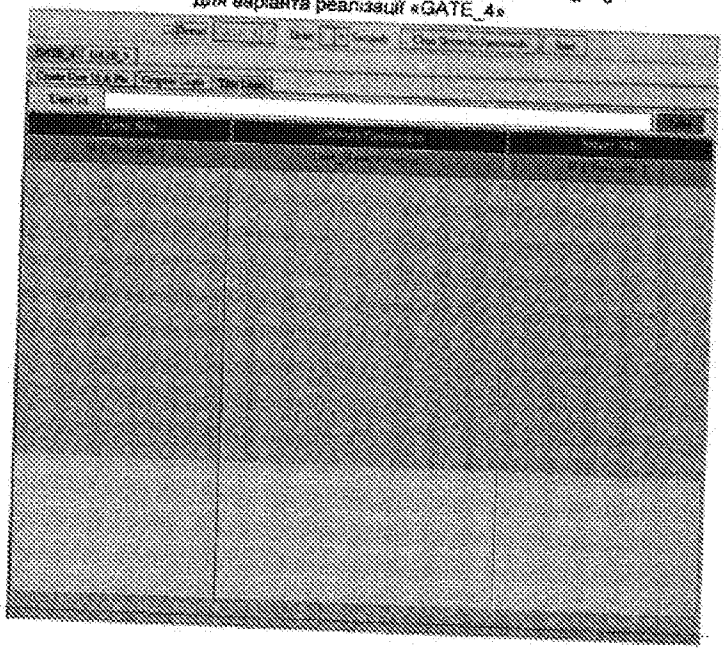
Фіг. 9C

Реєстрація входу у графічній формі «Graphic_Login» для варіанта реалізації «GATE_4»



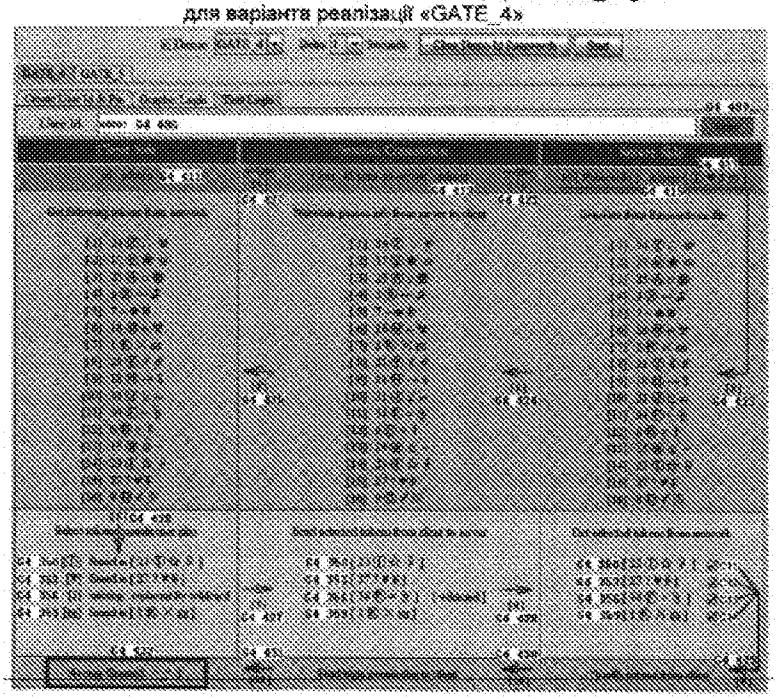
Фіг. 9D

Реєстрація входу у текстовій формі «Text_Login» для варіанта реалізації «GATE_4»



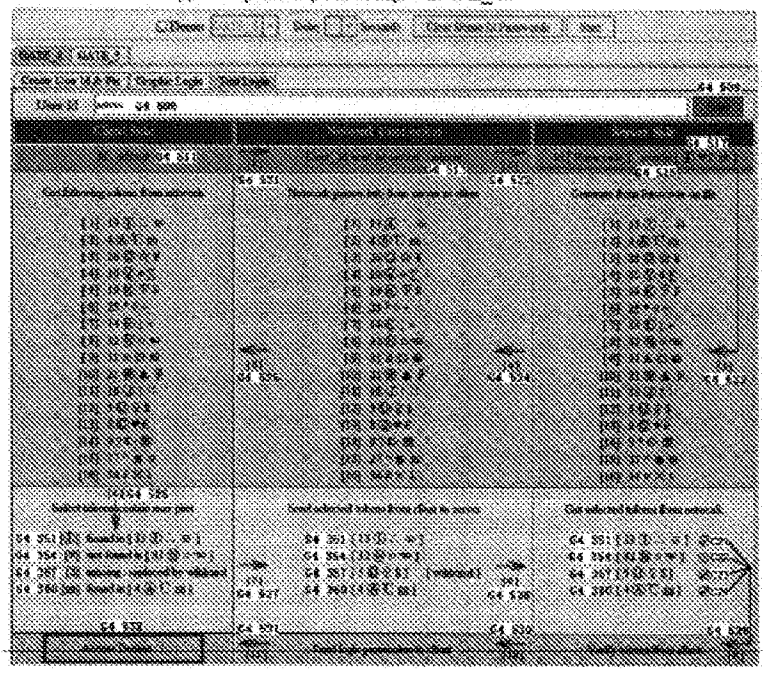
Фіг. 10A

Реєстрація входу у текстовій формі «Text_Login»
для варіанта реалізації «GATE_4»



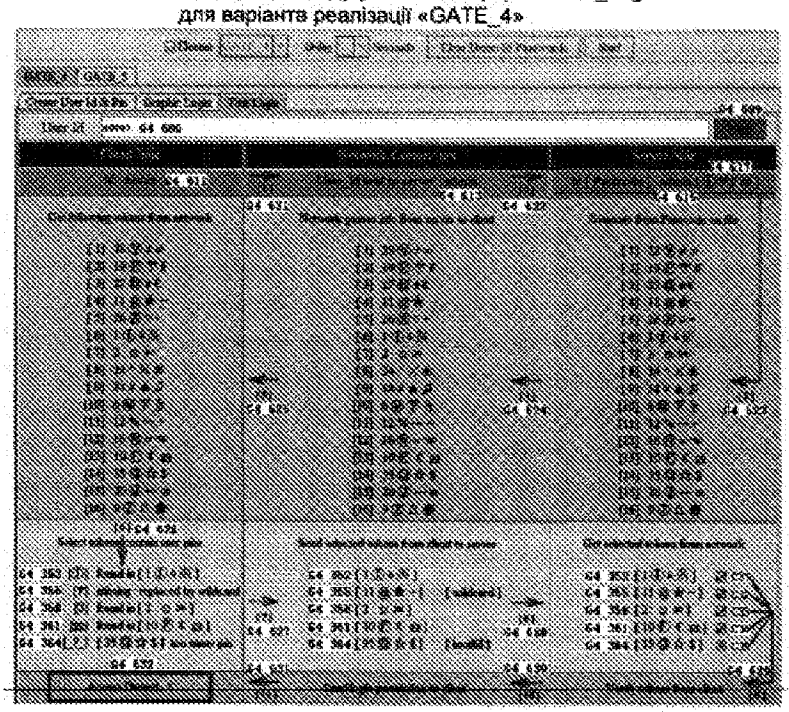
Фіг. 10B

Реєстрація входу у текстовій формі «Text_Login»
для варіанта реалізації «GATE_4»



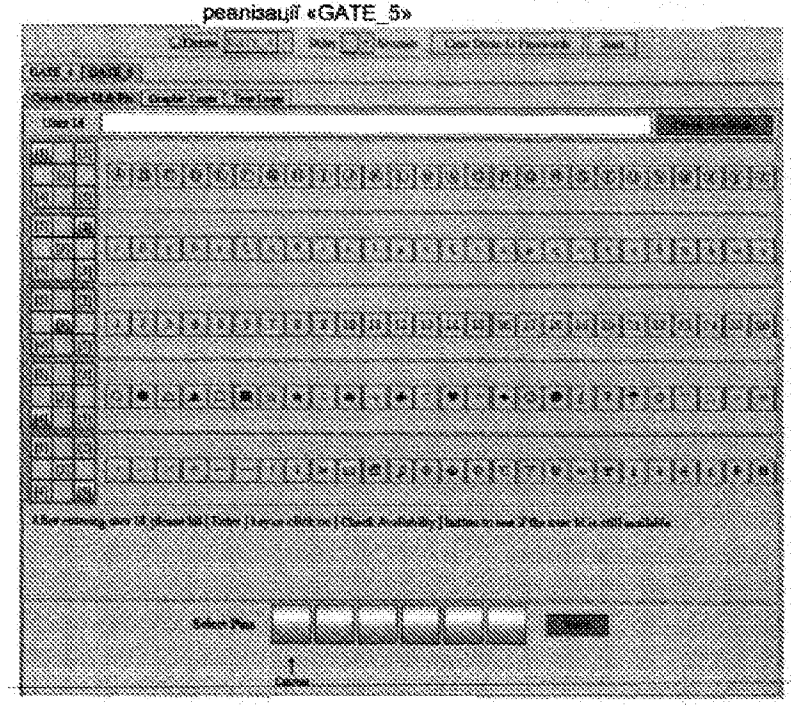
Фіг. 10C

Реєстрація входу у текстовій формі «Text_Login» для варіанта реалізації «GATE_4»



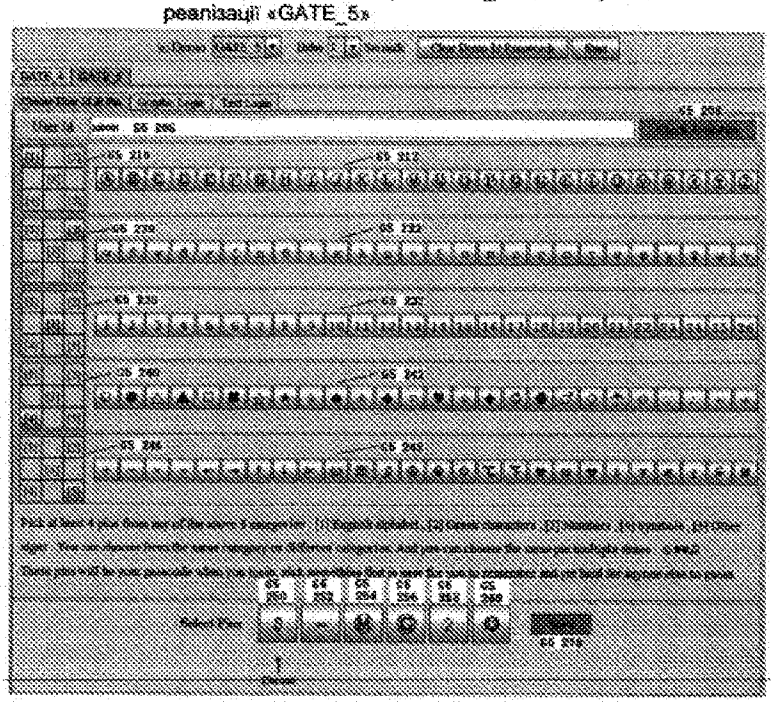
Фіг. 10D

Створити ідентифікатор «Create_ID» для варіанта реалізації «GATE_5»



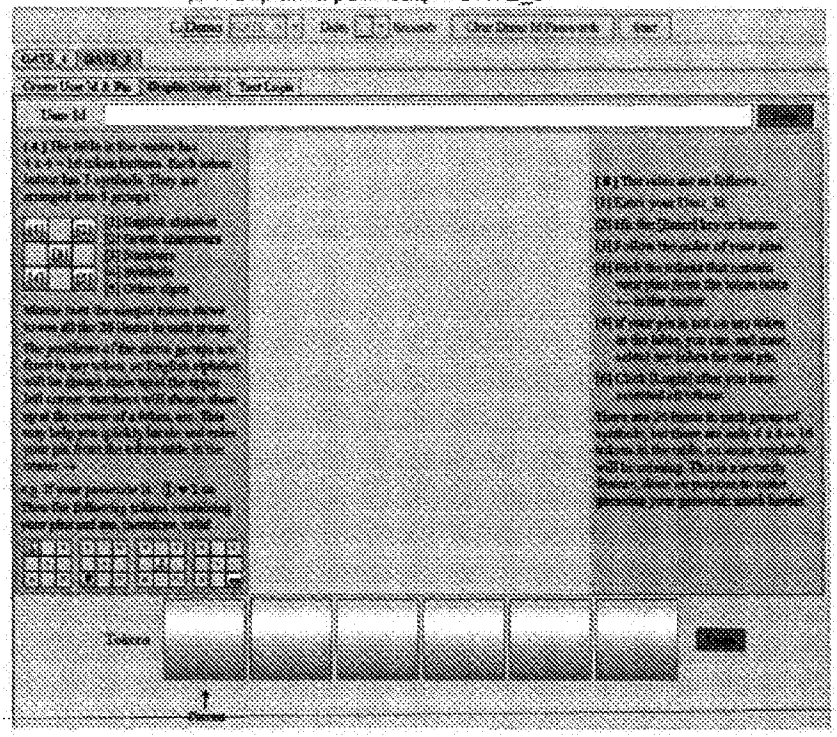
Фіг. 11A

Створити ідентифікатор «Create_ID» для варіанта реалізації «GATE_5»



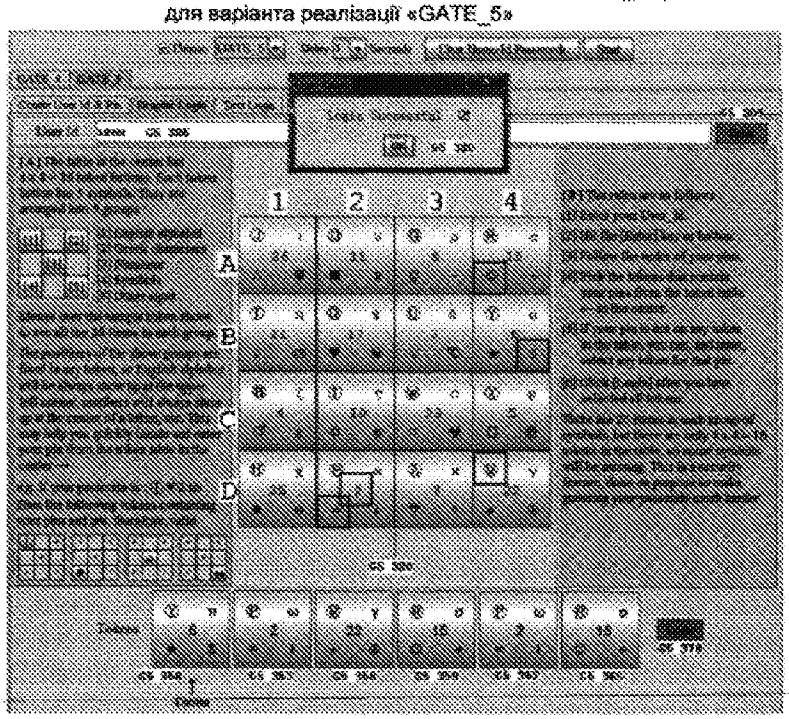
Фіг. 11B

Реєстрація входу у графічній формі «Graphic_Login» для варіанта реалізації «GATE_5»



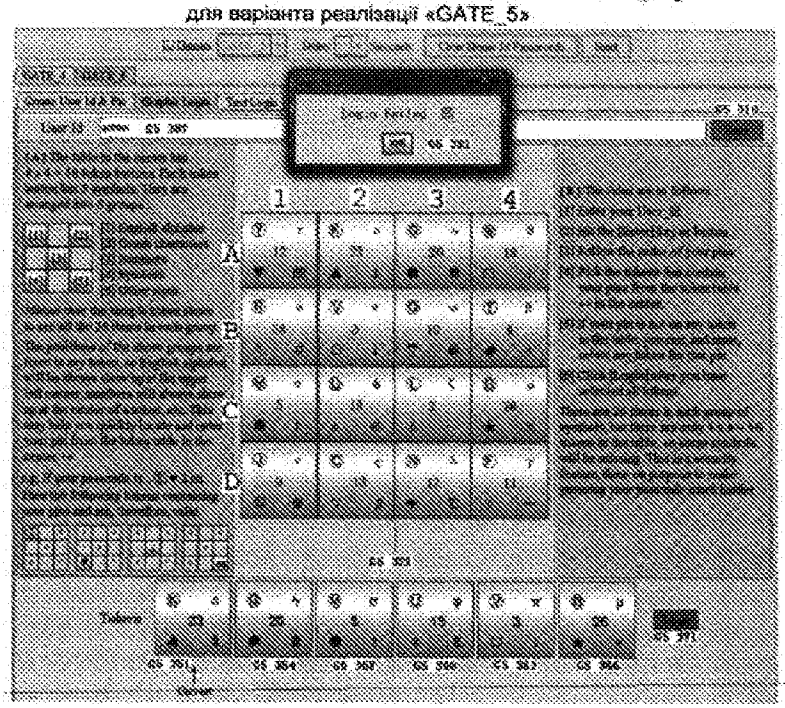
Фіг. 12A

Реєстрація входу у графічній формі «Graphic_Login»
для варіанта реалізації «GATE_5»



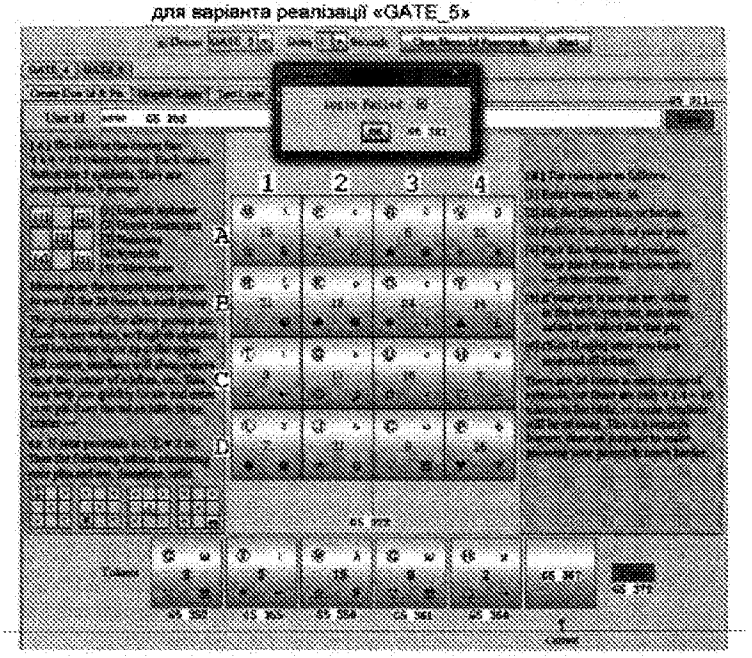
Фіг. 12B

Реєстрація входу у графічній формі «Graphic_Login»
для варіанта реалізації «GATE_5»



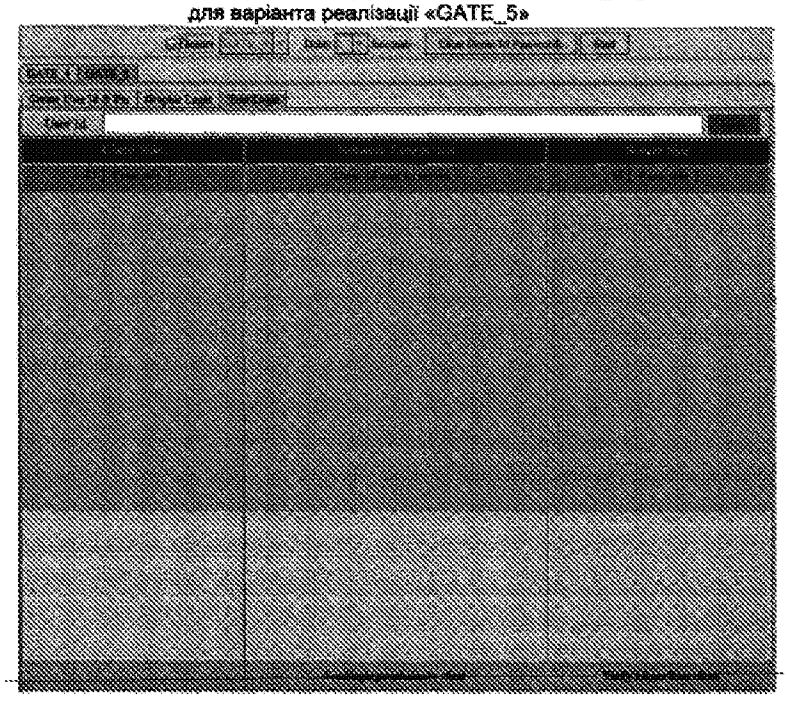
Фіг. 12C

Реєстрація входу у графічній формі «Graphic_Login»
для варіанта реалізації «GATE_5»



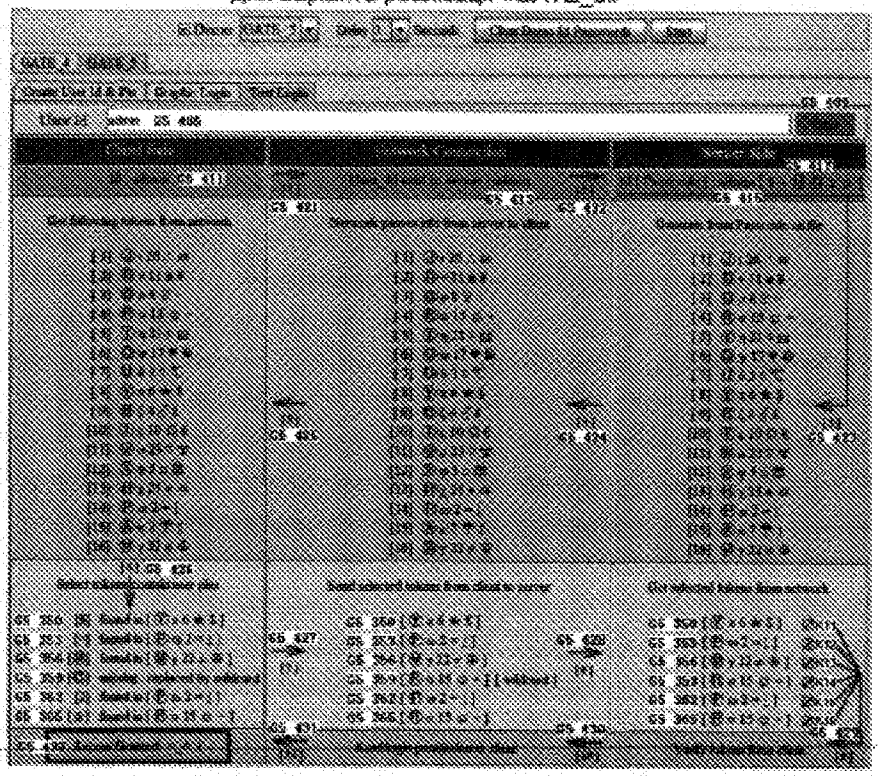
Фіг. 12D

Реєстрація входу у текстовій формі «Text_Login»
для варіанта реалізації «GATE_5»



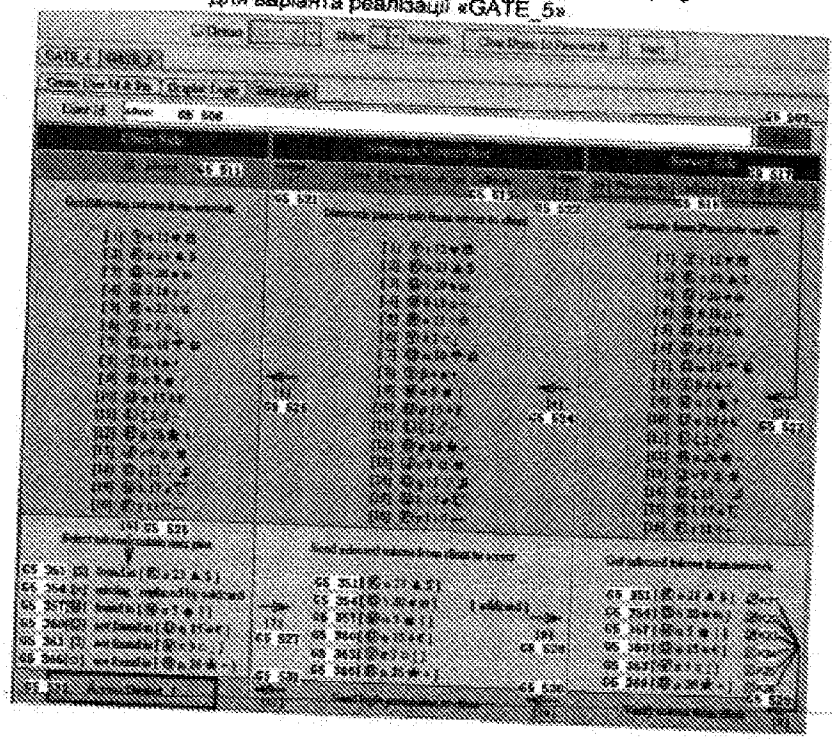
Фіг. 13A

Реєстрація входу у текстовій формі «Text_Login»
для варіанта реалізації «GATE_5»

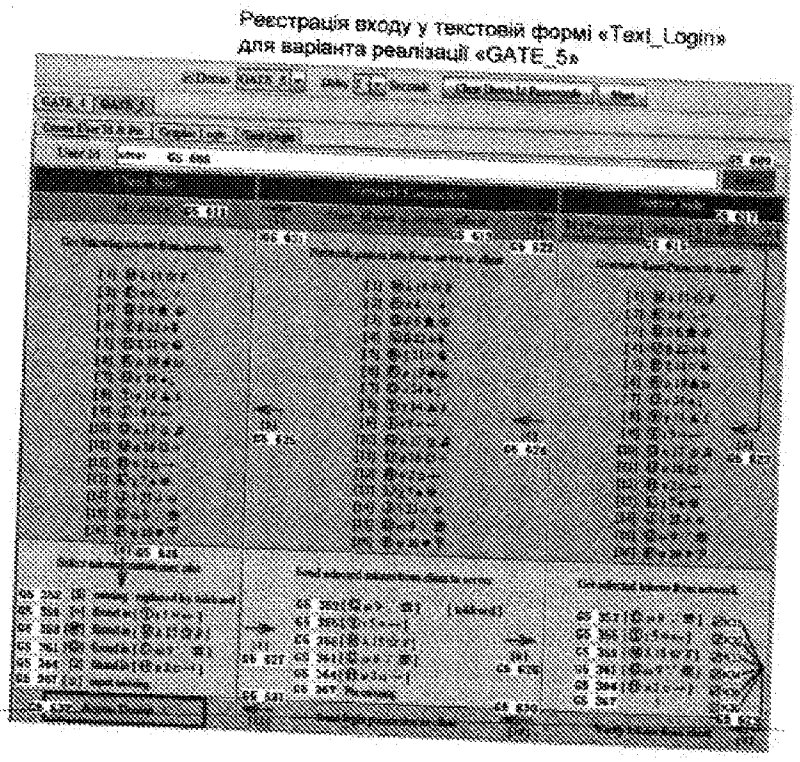


Фіг. 13B

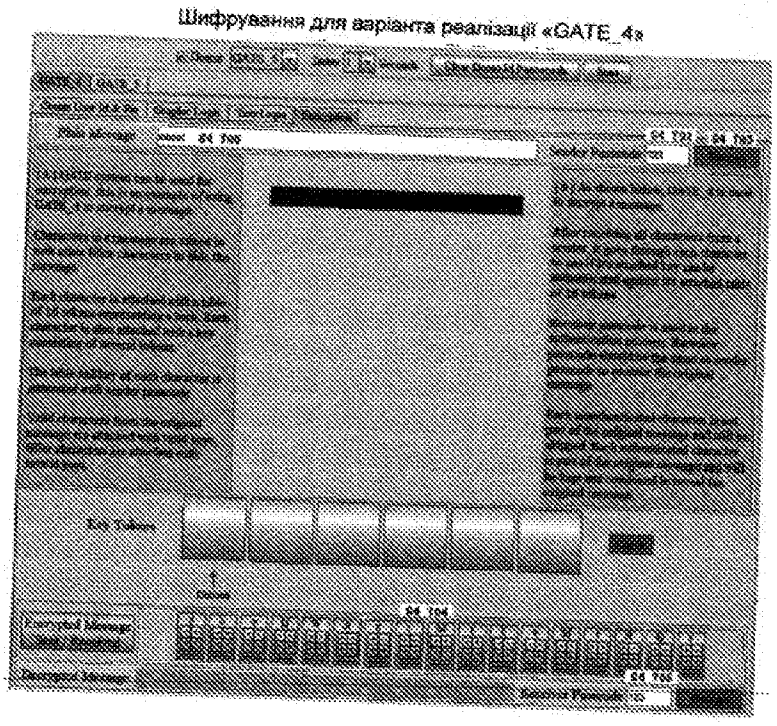
Реєстрація входу у текстовій формі «Text_Login»
для варіанта реалізації «GATE_5»



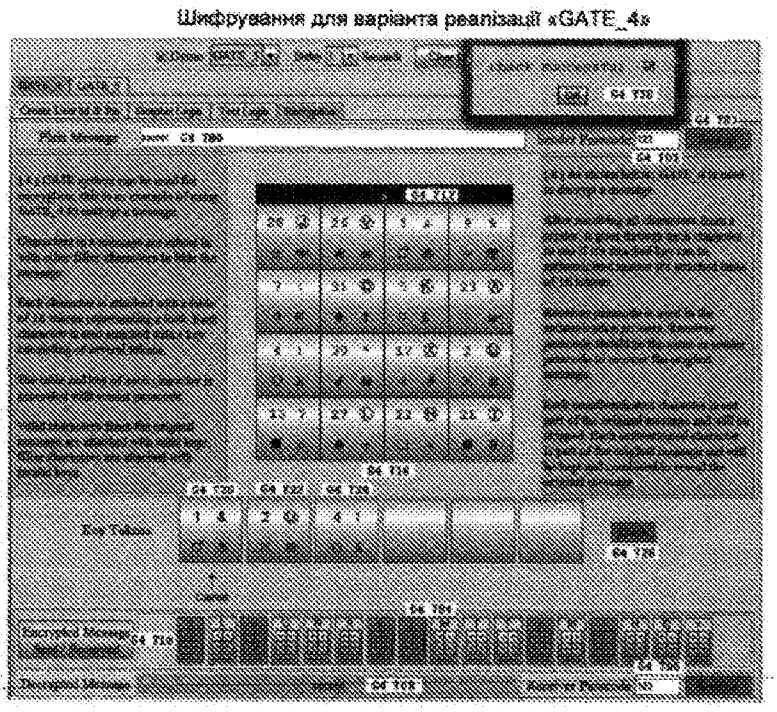
Фіг. 13C



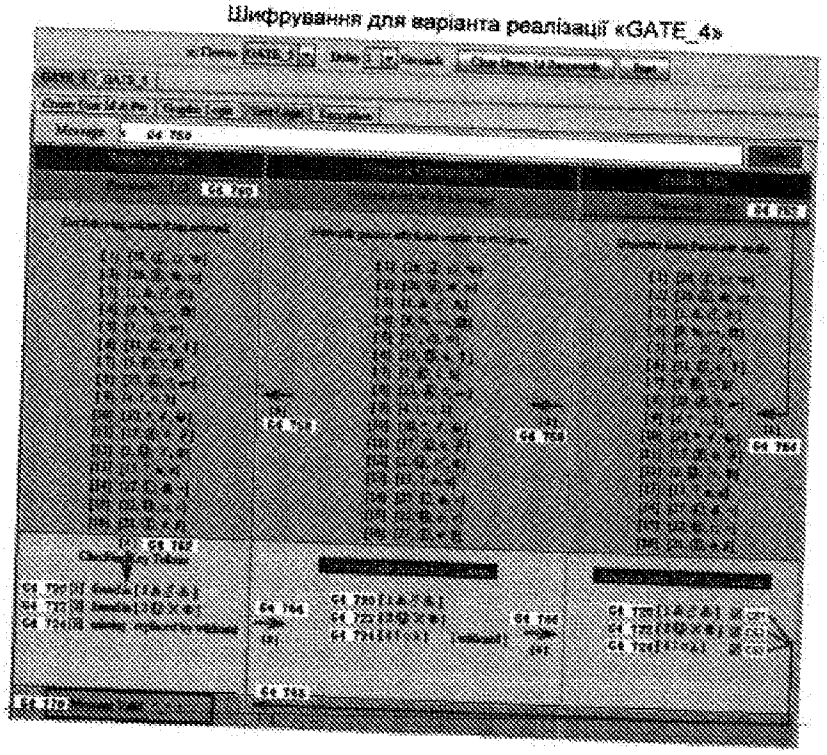
Фіг. 13D



Фіг. 14

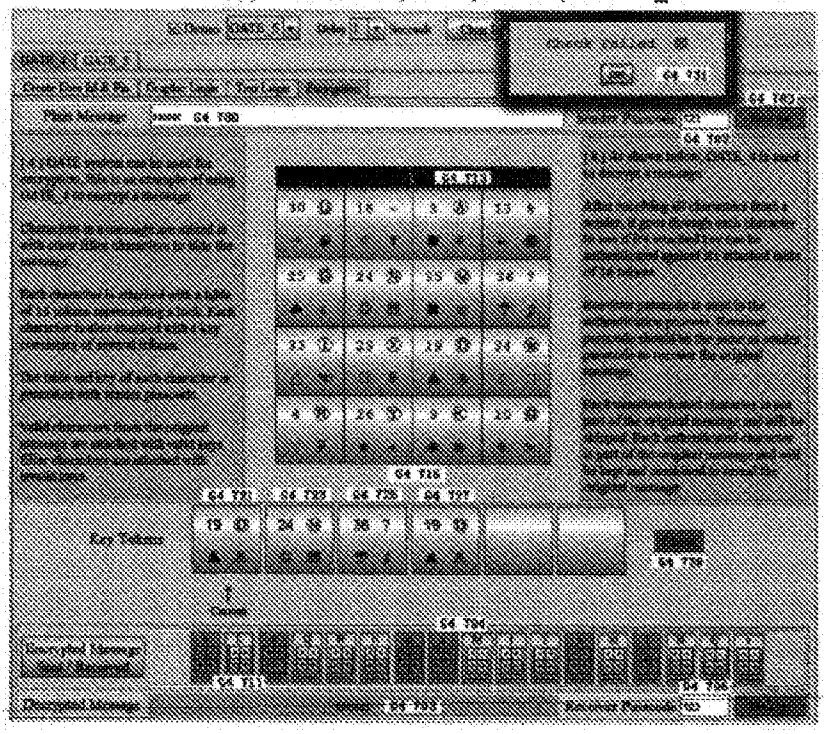


Фіг. 15A



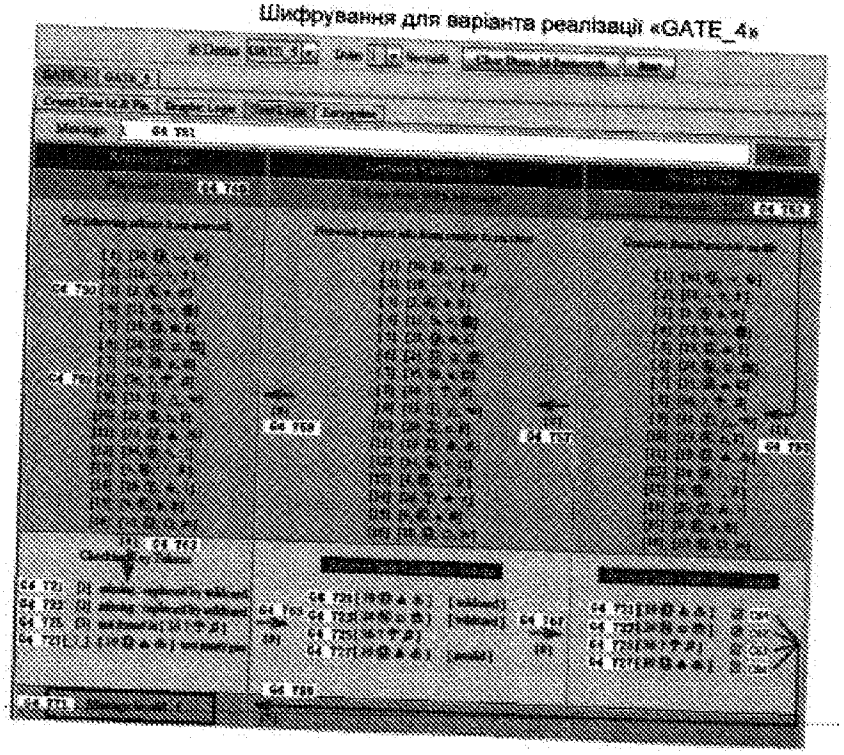
Фіг. 15B

Шифрування для варіанта реалізації «GATE_4»



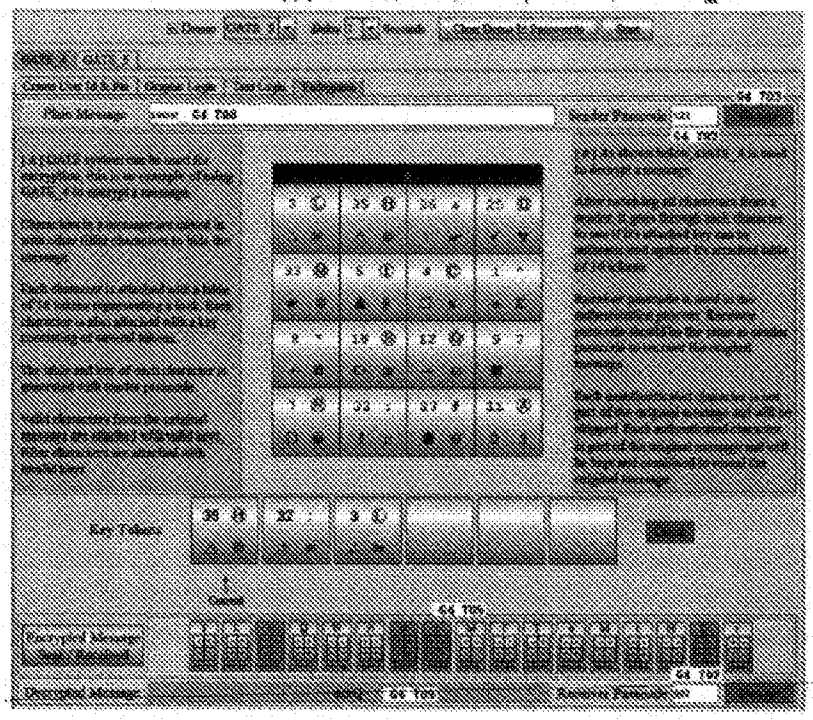
Фіг. 16A

Шифрування для варіанта реалізації «GATE_4»



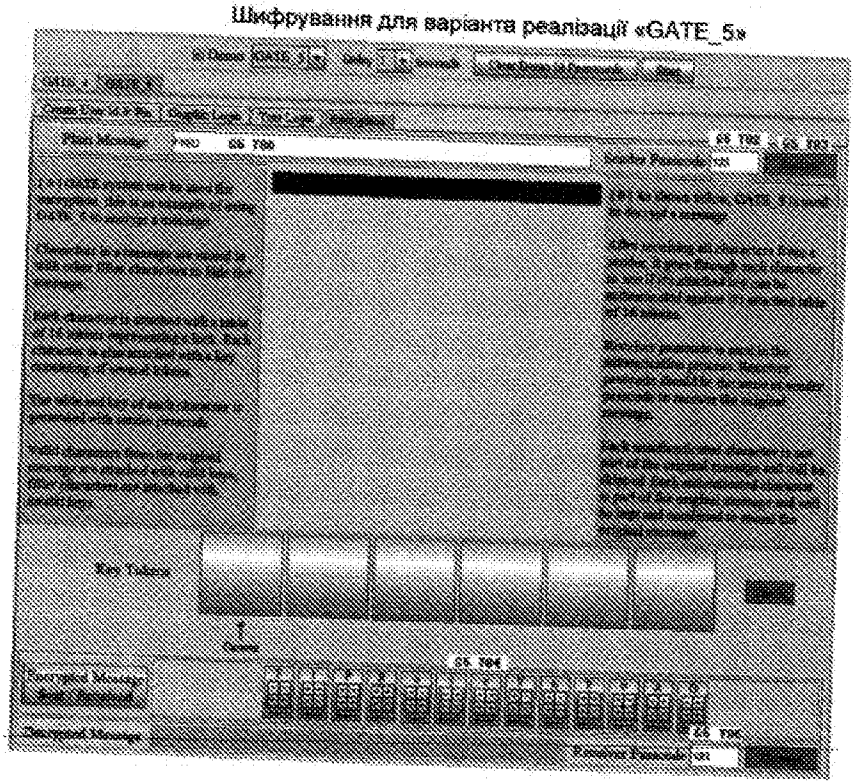
Фіг. 16B

Шифрування для варіанта реалізації «GATE_4»



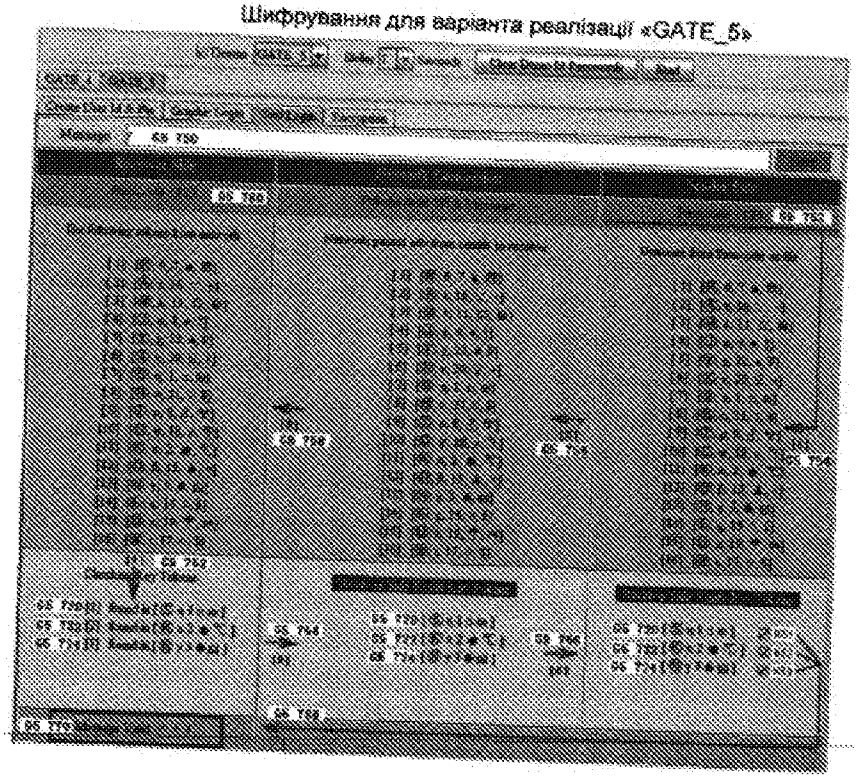
Фіг. 17

Шифрування для варіанта реалізації «GATE_5»

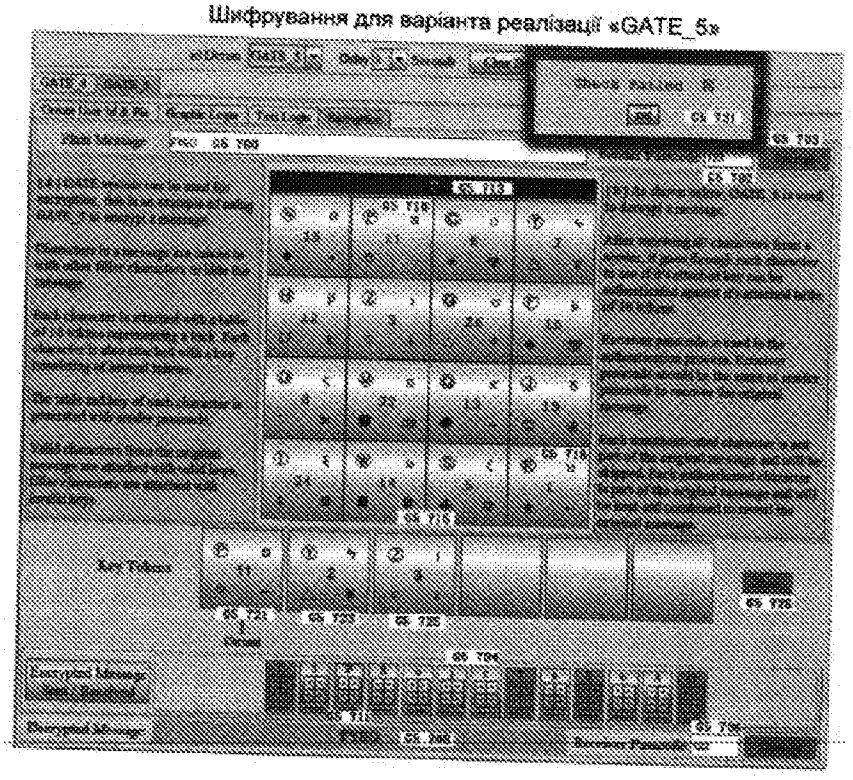


Фіг. 18

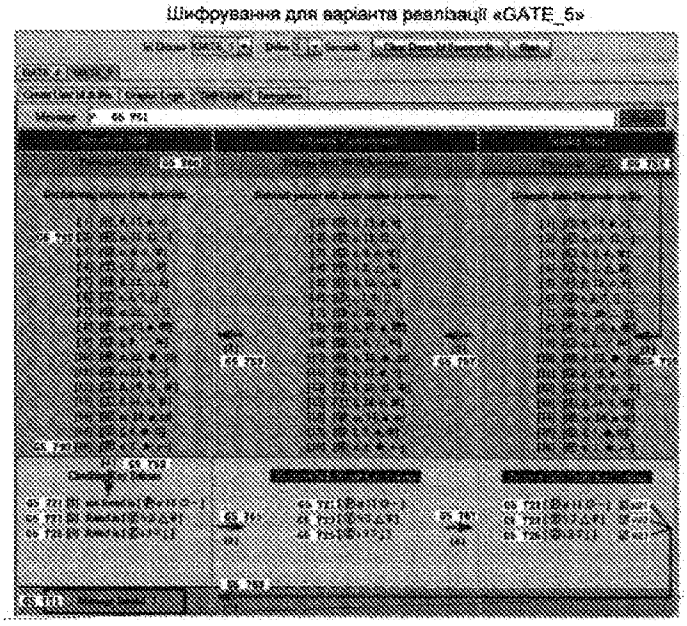
Фіг. 19A



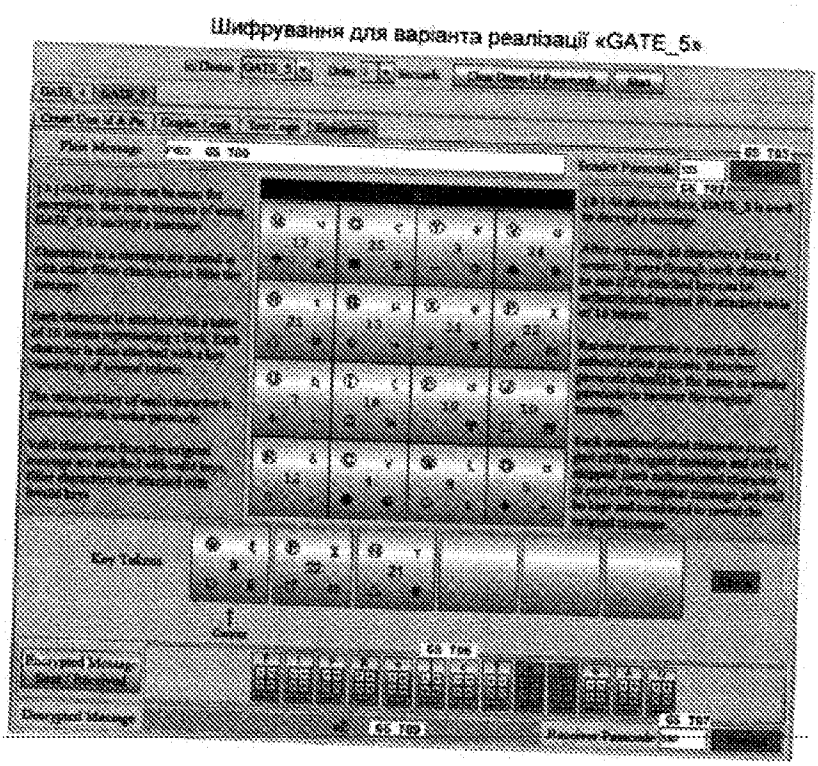
Фіг. 19B



Фіг. 20A



Фіг. 20B



Фіг. 21

Комп'ютерна верстка С. Чулій

Міністерство розвитку економіки, торгівлі та сільського господарства України,
вул. М. Грушевського, 12/2, м. Київ, 01008, Україна

ДП "Український інститут інтелектуальної власності", вул. Глазунова, 1, м. Київ – 42, 01601