(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2023년09월13일
(11) 등록번호 10-2577973
(24) 등록일자 2023년09월08일

(51) 국제특허분류(Int. Cl.)
H04L 9/32 (2006.01) *G06F 21/36* (2013.01)
H04L 9/40 (2022.01)
(52) CPC특허분류
H04L 9/3226 (2013.01)
G06F 21/36 (2013.01)
(21) 출원번호 10-2018-7012319
(22) 출원일자(국제) 2016년09월28일
심사청구일자 2021년07월06일
(85) 번역문제출일자 2018년04월30일
(65) 공개번호 10-2018-0088377
(43) 공개일자 2018년08월03일
(86) 국제출원번호 PCT/US2016/054186
(87) 국제공개번호 WO 2017/105579
국제공개일자 2017년06월22일
(30) 우선권주장
14/925,769 2015년10월28일 미국(US)
14/931,613 2015년11월03일 미국(US)
(56) 선행기술조사문헌
JP2007086873 A*
JP2009169857 A*
US20130291096 A1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
니, 민
미국, 30021 조지아, 클라크스턴, 크리크데일 드라이브 1050
(72) 발명자
니, 민
미국, 30021 조지아, 클라크스턴, 크리크데일 드라이브 1050
(74) 대리인
특허법인에이아이피

전체 청구항 수 : 총 42 항 심사관 : 나용수

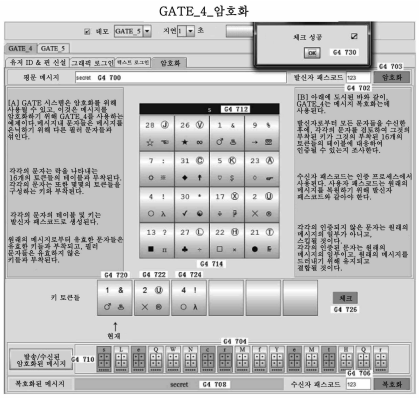
(54) 발명의 명칭 **인터셉션 방지 인증 및 암호화 시스템 및 방법**

(57) 요 약

심벌들의 세트로부터의 심벌들로 이루어진 개별 핀들을 갖는 패스코드들, 및 패스코드를 위해 사용되는 심벌들의 세트로부터 적어도 두개의 심벌들을 함유하는 토큰들을 이용하는 인터셉션 방지 인증 및 암호화 시스템 및 방법이 제공된다. 다수의 토큰들 (토큰 세트)이 유저에게 제시되고, 유저의 사전 선택된 핀들(심벌들)의 일부 또는

(뒷면에 계속)

대 표 도 - 도15a



전부가 토큰들의 일부 또는 전부에 무작위적으로 삽입된다. 유저는 패스코드내 각각의 핀 위치에 대하여 토큰 세트로부터 토큰을 선택한다. 유저는 선택된 토큰들에 기초하여 인증된다. 각각의 선택된 토큰은 유저의 패스코드에 사전-선택된 핀들 중 하나를 함유할 수 있거나 또는 함유하지 않을 수 있고, 또한 유저의 패스 코드에서 사전 선택된 핀들 중 하나가 아닌 다른 무작위적으로 생성된 심벌들을 함유하기 때문에, 어느 토큰들을 유저가 선택하였는지 관찰하는 사람은 유저의 실제 패스코드가 무엇인지 결정할 수 없다.

(52) CPC특허분류
H04L 63/0435 (2013.01)
H04L 63/083 (2013.01)

명 세 서

청구범위

청구항 1

심벌들의 세트에서 선택된 미리 결정된 수의 심벌들 (“패스코드 심벌들(passcode symbols)”)을 포함하는 미리 결정된 전자적으로 저장된 패스코드 (“패스코드(passcode)”)를 이용하여 전자적으로 저장된 정보에 대한 유저 액세스를 허용하는 (“인증하는(authenticating)”) 방법에 있어서, 상기 패스코드 심벌들의 각각은 미리 결정된 핀 위치에 의해 특징되고, 메모리에 저장된 컴퓨터 판독가능한 지시들의 세트를 프로세서에 의해 실행함으로써 수행되는,

전자 디바이스의 유저 인터페이스를 통하여 상기 유저에게 토큰 세트를 제공하는 단계로서, 상기 토큰 세트는 적어도 두개의 토큰들을 포함하고, 상기 토큰 세트내 각각의 토큰은 상기 심벌들의 세트에 속하는 적어도 두개의 심벌들을 포함하는, 상기 제공하는 단계;

상기 유저가 상기 유저 인터페이스를 통하여 상기 패스코드내 각각의 핀 위치에 대하여 상기 토큰 세트로부터 토큰을 선택할 것을 요구하는 단계; 및

상기 유저가 선택한 상기 토큰들에 기초하여 상기 유저를 인증하는 단계로서, 상기 유저는 만약:

상기 유저에 의해 선택된 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고,

선택된 상기 토큰들 중 적어도 하나가 상기 패스코드 심벌들의 개별 심벌을 함유하고, 및

상기 패스코드 심벌들의 개별 심벌을 함유하는 상기 선택된 토큰들의 각각의 핀 위치가 상기 패스코드내 개별 패스코드 심벌의 핀 위치에 대응하면 인증되는, 상기 인증하는 단계를 포함하는, 방법.

청구항 2

청구항 1에 있어서, 상기 유저는 만약 :

상기 유저에 의해 선택된 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고,

선택된 각각의 토큰이 상기 패스코드 심벌들의 개별 심벌을 함유하고; 및

상기 선택된 토큰들의 각각의 핀 위치가 각각의 상기 패스코드내 상기 심벌들 중 어느 심벌이 상기 선택된 토큰들의 각각에 포함되는지에 기초하여 상기 패스코드 심벌들의 핀 위치에 대응하면 인증되는, 방법.

청구항 3

청구항 1에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 유저에게 제공된 토큰들의 수와 같은, 방법.

청구항 4

청구항 1에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 유저에게 제공된 토큰들의 수보다 더 큰, 방법.

청구항 5

청구항 1에 있어서, 상기 심벌들의 세트는 적어도 두개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰은 상기 적어도 두개의 차원의 각각으로부터의 심벌을 포함하는, 방법.

청구항 6

청구항 1에 있어서, 각각의 토큰은 상기 심벌들의 세트에 속하는 네개의 심벌들을 포함하는, 방법.

청구항 7

청구항 6에 있어서, 심벌들의 각각의 세트는 네개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 방법.

청구항 8

청구항 1에 있어서, 각각의 토큰은 상기 심벌들의 세트에 속하는 다섯개의 심벌들을 포함하는, 방법.

청구항 9

청구항 8에 있어서, 심벌들의 각각의 세트는 다섯개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 방법.

청구항 10

청구항 1에 있어서, 상기 심벌들의 세트는 유니코드 체계(Unicode system)에 기초하는, 방법.

청구항 11

심벌들의 세트에서 선택된 미리 결정된 수의 심벌들 (“패스코드 심벌들(passcode symbols)”)을 포함하는 미리 결정된 전자적으로 저장된 패스코드 (“패스코드(passcode)”)를 이용하여 전자적으로 저장된 정보에 대한 유저 액세스를 허용하기 위한 (“인증하는(authenticating)”) 시스템에 있어서, 상기 패스코드 심벌들의 각각은 미리 결정된 핀 위치에 의해 특징되고 :

프로세서;

상기 프로세서에 의해 액세스 가능한 메모리; 및

상기 프로세서에 의해 실행가능한 메모리에 저장된 컴퓨터 판독가능한 지시들의 세트를 포함하는 인증/암호화 모듈로서,

상기 유저에게 토큰 세트를 제공하고, 상기 토큰 세트는 적어도 두개의 토큰들을 포함하고, 상기 토큰 세트내 각각의 토큰은 상기 심벌들의 세트에 속하는 적어도 두개의 심벌들을 포함하고;

상기 유저가 유저 인터페이스를 통하여 상기 패스코드내 각각의 핀 위치에 대하여 상기 토큰 세트로부터 토큰을 선택할 것을 요구하고; 및

상기 유저가 선택한 상기 토큰들에 기초하여 상기 유저를 인증하는, 상기 인증/암호화 모듈을 포함하되, 상기 프로세서는 만약:

상기 유저에 의해 선택된 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고,

선택된 상기 토큰들 중 적어도 하나가 상기 패스코드 심벌들의 개별 심벌을 함유하고, 및

상기 패스코드 심벌들의 개별 심벌을 함유하는 상기 선택된 토큰들의 각각의 핀 위치가 상기 패스코드내 개별 패스코드 심벌의 핀 위치에 대응하면 상기 유저가 인증되었다고 결정하는, 시스템.

청구항 12

청구항 11에 있어서, 상기 프로세서는 만약 :

상기 유저에 의해 선택된 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고,

선택된 각각의 토큰이 상기 패스코드 심벌들의 개별 심벌을 함유하고; 및

상기 선택된 토큰들의 각각의 핀 위치가 각각의 상기 패스코드내 상기 심벌들 중 어느 심벌이 상기 선택된 토큰들의 각각에 포함되는지에 기초하여 상기 패스코드 심벌들의 핀 위치에 대응하면 상기 유저가 인증되었다고 결정하는, 시스템.

청구항 13

청구항 11에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 유저에게 제공된 토큰들의 수와 같은, 시스템.

청구항 14

청구항 11에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 유저에게 제공된 토큰들의 수보다 더 큰, 시스템.

청구항 15

청구항 11에 있어서, 상기 심벌들의 세트는 적어도 두개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰은 상기 적어도 두개의 차원의 각각으로부터의 심벌을 포함하는, 시스템.

청구항 16

청구항 11에 있어서, 각각의 토큰은 상기 심벌들의 세트에 속하는 네개의 심벌들을 포함하는, 시스템.

청구항 17

청구항 16에 있어서, 심벌들의 각각의 세트는 네개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 시스템.

청구항 18

청구항 11에 있어서, 각각의 토큰은 상기 심벌들의 세트에 속하는 다섯개의 심벌들을 포함하는, 시스템.

청구항 19

청구항 18에 있어서, 심벌들의 각각의 세트는 다섯개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 시스템.

청구항 20

청구항 11에 있어서, 상기 심벌들의 세트는 유니코드 체계(Unicode system)에 기초하는, 시스템.

청구항 21

심벌들의 세트에서 선택된 미리 결정된 수의 심벌들 (“패스코드 심벌들(passcode symbols)”)을 포함하는 미리 결정된 전자적으로 저장된 패스코드 (“패스코드(passcode)”)를 이용하는 정보에 대한 암호화/복호화 방법에 있어서, 상기 패스코드 심벌들의 각각은 미리 결정된 핀 위치에 의해 특징되고, 메모리에 저장된 컴퓨터 판독가능한 지시들의 세트를 프로세서에 의해 실행함으로써 수행되는,

암호화될 원래의 정보 (“원래의 정보(original information)”)를 수신하는 단계로서, 상기 원래의 정보는 개별 정보 엘리먼트 위치들에 미리 결정된 수의 원래의 정보 엘리먼트들을 포함하고, 상기 원래의 정보 엘리먼트들은 정보 엘리먼트들의 세트에서 선택되는, 상기 수신하는 단계;

상기 원래의 암호화된 정보를 생성하기 위해 상기 정보 엘리먼트들의 세트로부터 무작위적으로 선택된 정보 엘리먼트들 (“랜덤 정보 엘리먼트(random information element)들”)을 랜덤 정보 엘리먼트 위치들에서 상기 원래의 정보에 삽입하는 단계 ;

각각의 원래 및 랜덤 정보 엘리먼트에 대한 토큰 세트를 생성하는 단계로서, 각각의 토큰 세트는 적어도 두개의 토큰들을 포함하고, 상기 토큰 세트내 각각의 토큰은 상기 심벌들의 세트에 속하는 적어도 두개의 심벌들을 포함하는, 상기 토큰 세트를 생성하는 단계;

상기 암호화된 정보에 각각의 원래의 및 랜덤 정보 엘리먼트에 대한 개별 키를 생성하는 단계로서, 각각의 키는 적어도 두개의 키 토큰들을 포함하고, 각각의 키 토큰은 상기 심벌들의 세트로부터의 적어도 두개의 심벌들을 포함하고, 각각의 키 토큰내 심벌들의 수는 각각의 토큰 세트의 각각의 토큰내 심벌들의 수와 같은, 상기 개별 키를 생성하는 단계;를 포함하되,

상기 키 토큰들 및 토큰 세트들이 생성되어 상기 키 토큰들 및 토큰 세트들은 총괄하여 상기 암호화된 정보를 복호화하기 위해 상기 패스코드와 함께 사용되는, 방법.

청구항 22

청구항 21에 있어서, 복호화 프로세스 동안, 상기 암호화된 정보내 정보 엘리먼트는 만약 :

상기 정보 엘리먼트의 대응하는 키내 키 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고;

상기 정보 엘리먼트의 대응하는 키내 상기 키 토큰들 중 적어도 하나가 상기 패스코드 심벌들 중 개별 심벌을

함유하고; 및

상기 패스코드 심벌들의 개별 심벌을 함유하는 상기 정보 엘리먼트의 대응하는 키내 상기 키 토큰들의 각각의 핀 위치가 상기 패스코드내 개별 패스코드 심벌의 핀 위치에 대응하면 유효한 정보 엘리먼트인 것으로 결정되는, 방법.

청구항 23

청구항 21에 있어서, 복호화 프로세스 동안, 상기 암호화된 정보내 정보 엘리먼트는 만약 :

상기 정보 엘리먼트의 대응하는 키내 키 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고;

상기 정보 엘리먼트의 대응하는 키내 각각의 키 토큰은 상기 패스코드 심벌들 중 개별 심벌을 함유하고; 및

상기 패스코드 심벌들의 개별 심벌을 함유하는 상기 정보 엘리먼트의 대응하는 키내 상기 키 토큰들의 각각의 핀 위치가 상기 패스코드내 개별 패스코드 심벌의 핀 위치에 대응하면 유효한 정보 엘리먼트인 것으로 결정되는, 방법.

청구항 24

청구항 21에 있어서, 상기 심벌들의 세트는 적어도 두개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰 및 각각의 키 토큰은 상기 적어도 두개의 차원의 각각으로부터의 심벌을 포함하는, 방법.

청구항 25

청구항 21에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 토큰 세트내 토큰들의 수와 같은, 방법.

청구항 26

청구항 21에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 토큰 세트내 토큰들의 수보다 더 큰, 방법.

청구항 27

청구항 21에 있어서, 각각의 키 토큰은 상기 심벌들의 세트에 속하는 네개의 심벌들을 포함하는, 방법.

청구항 28

청구항 27에 있어서, 심벌들의 각각의 세트는 네개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰 및 각각의 키 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 방법.

청구항 29

청구항 21에 있어서, 각각의 키 토큰은 상기 심벌들의 세트에 속하는 다섯개의 심벌들을 포함하는, 방법.

청구항 30

청구항 29에 있어서, 심벌들의 각각의 세트는 다섯개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰 및 각각의 키 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 방법.

청구항 31

청구항 21에 있어서, 상기 정보 엘리먼트들의 세트는 유니코드 체계(Unicode system)에 기초하는, 방법.

청구항 32

심벌들의 세트에서 선택된 미리 결정된 수의 심벌들 (“패스코드 심벌들(passcode symbols)”)을 포함하는 미리 결정된 전자적으로 저장된 패스코드 (“패스코드(passcode)”)를 이용하는 정보의 암호화/복호화를 위한 시스템에 있어서, 상기 패스코드 심벌들의 각각은 미리 결정된 핀 위치에 의해 특징되고 :

프로세서;

상기 프로세서에 액세스가능한 메모리; 및

상기 프로세서에 의해 실행가능한 메모리에 저장된 컴퓨터 판독가능한 지시들의 세트를 포함하는 인증/암호화

모듈로서,

암호화될 원래의 정보 (“원래의 정보(original information)”)를 수신하고, 상기 원래의 정보는 개별 정보 엘리먼트 위치들에 미리 결정된 수의 원래의 정보 엘리먼트들을 포함하고, 상기 원래의 정보 엘리먼트들은 정보 엘리먼트들의 세트에서 선택되고;

상기 원래의 암호화된 정보를 생성하기 위해 상기 정보 엘리먼트들의 세트로부터 무작위적으로 선택된 정보 엘리먼트들 (“랜덤 정보 엘리먼트(random information element)들”)을 랜덤 정보 엘리먼트 위치들에서 상기 원래의 정보에 삽입하고;

각각의 원래 및 랜덤 정보 엘리먼트에 대한 토큰 세트를 생성하고, 각각의 토큰 세트는 적어도 두개의 토큰들을 포함하고, 상기 토큰 세트내 각각의 토큰은 상기 심벌들의 세트에 속하는 적어도 두개의 심벌들을 포함하고;

상기 암호화된 정보에 각각의 원래의 및 랜덤 정보 엘리먼트에 대한 개별 키를 생성하고, 각각의 키는 적어도 두개의 키 토큰들을 포함하고, 각각의 키 토큰은 상기 심벌들의 세트로부터의 적어도 두개의 심벌들을 포함하고, 각각의 키 토큰내 심벌들의 수는 각각의 토큰 세트의 각각의 토큰내 심벌들의 수와 같은, 상기 인증/암호화 모듈;을 포함하되,

상기 키 토큰들 및 토큰 세트들이 생성되어 상기 키 토큰들 및 토큰 세트들은 총괄하여 상기 암호화된 정보를 복호화하기 위해 상기 패스코드와 함께 사용되는, 시스템.

청구항 33

청구항 32에 있어서, 복호화 프로세스 동안, 상기 프로세서는 만약 :

상기 정보 엘리먼트의 대응하는 키내 키 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고;

상기 정보 엘리먼트의 대응하는 키내 상기 키 토큰들 증 적어도 하나가 상기 패스코드 심벌들 중 개별 심벌을 함유하고; 및

상기 패스코드 심벌들의 개별 심벌을 함유하는 상기 정보 엘리먼트의 대응하는 키내 상기 키 토큰들의 각각의 핀 위치가 상기 패스코드내 개별 패스코드 심벌의 핀 위치에 대응하면 상기 암호화된 정보내 정보 엘리먼트가 유효한 정보 엘리먼트인 것으로 결정하는, 시스템.

청구항 34

청구항 32에 있어서, 복호화 프로세스 동안, 상기 프로세서는 만약 :

상기 정보 엘리먼트의 대응하는 키내 키 토큰들의 수가 상기 패스코드내 심벌들의 수와 같고;

상기 정보 엘리먼트의 대응하는 키내 각각의 키 토큰은 상기 패스코드 심벌들 중 개별 심벌을 함유하고; 및

상기 패스코드 심벌들의 개별 심벌을 함유하는 상기 정보 엘리먼트의 대응하는 키내 상기 키 토큰들의 각각의 핀 위치가 상기 패스코드내 개별 패스코드 심벌의 핀 위치에 대응하면 상기 암호화된 정보내 정보 엘리먼트가 유효한 정보 엘리먼트인 것으로 결정하는, 시스템.

청구항 35

청구항 32에 있어서, 상기 심벌들의 세트는 적어도 두개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰 및 각각의 키 토큰은 상기 적어도 두개의 차원의 각각으로부터의 심벌을 포함하는, 시스템.

청구항 36

청구항 32에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 토큰 세트내 토큰들의 수와 같은, 시스템.

청구항 37

청구항 32에 있어서, 상기 심벌들의 세트내 심벌들의 수는 상기 토큰 세트내 토큰들의 수보다 더 큰, 시스템.

청구항 38

청구항 32에 있어서, 각각의 키 토큰은 상기 심벌들의 세트에 속하는 네개의 심벌들을 포함하는, 시스템.

청구항 39

청구항 38에 있어서, 심벌들의 각각의 세트는 네개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰 및 각각의 키 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 시스템.

청구항 40

청구항 32에 있어서, 각각의 키 토큰은 상기 심벌들의 세트에 속하는 다섯개의 심벌들을 포함하는, 시스템.

청구항 41

청구항 40에 있어서, 심벌들의 각각의 세트는 다섯개의 서브세트들 (“차원(dimension)”)으로 분할되고, 각각의 토큰 및 각각의 키 토큰은 심벌들의 각각의 차원으로부터의 심벌을 포함하는, 시스템.

청구항 42

청구항 32에 있어서, 상기 정보 엘리먼트들의 세트는 유니코드 체계(Unicode system)에 기초하는, 시스템.

발명의 설명

기 술 분 야

[0001] 본 발명은 인증 및 암호화 시스템들 및 방법들에 관한 것으로, 보다 상세하게는, 인터셉션 방지 인증 및 암호화 시스템들 및 방법들에 관한 것이다.

배 경 기 술

[0002] 현대 사회에서, 일상생활은 예를 들자면, 이동 전화기들, PC들, 노트북들, 및 ATM들과 같은 매우 다양한 정보 디바이스들을 요구한다. 정보 디바이스들은 유저들의 퍼스널 데이터를 유지할 수 있다. 이 퍼스널 데이터 보호의 중요성 때문에, 이들 디바이스들을 안전하게 락(lock)하고 언락(unlock)하는 방법들이 있다.

[0003] 현재, 이들 디바이스들을 락하고 언락하는 가장 흔히 사용하는 방법은 패스워드기반 챌린지 인증 절차이며, 이에 의해 디바이스는 전형적으로 그것의 서비스들에 액세스하기 전에 유저들은 아이덴티티 인식을 위해 유저 ID와 패스워드를 입력하는 것을 요구한다. 이는 로그인(login)으로 알려져 있다. 이 로그인 프로세스는 유저들' 퍼스널 데이터가 도난되거나 또는 부정하게 바뀌는 것을 방지하도록 디자인된다.

[0004] 네트워크 커버리지 및 접근성의 빠른 날마다의 증가로, 해커(hacker)들은 유저들의 개인 정보에 대한 액세스를 획득하기 위해 유저들' 패스워드들을 보다 쉽게 타겟팅할 수 있다. 추가하여, 해커들은 유저들' 패스워드들의 추측하고 크랙(crack) 하는데 점점 더 정교해지고 있다. 따라서, 간단한 패스워드들은 더 이상 사이버 위협 및 스파이 행위(espionage)로부터 적절한 보호를 제공하지 않는다.

[0005] 이런면에서, 더 나은 보호를 제공하기 위해 다양한 메커니즘들이 구현되었다. 예를 들어, 유저들은 패스워드의 길이, 복잡도 및 예측 불가능성의 요건들을 충족시키는 패스워드를 신설하는 것이 요구되어, 패스워드의 강도는 이론적으로는 무차별 대입 검색 공격(brute-force search attack) 및 사전 공격(dictionary attack)을 막을 수 있기에 충분하다. 더욱이, 유저들은 이전 패스워드를 무효화하기 위해 정기적으로 패스워드들을 변경함으로써 그들의 패스워드들이 크랙될 가능성을 줄인다. 이들 메커니즘들은 어느 정도까지 보안을 강화하여서 유저들이 그들의 계정들을 보호하는데 도움을 준다.

[0006] 그러나, 각각의 조직은 상이한 세트의 패스워드 규칙들을 가질 수 있다. 일부는 패스워드 길이가 적어도 6 또는 8 문자들일 것을 요구한다. 일부는 대문자와 소문자가 혼합된 문자와 숫자를 사용해야 한다. 일부는 적어도 하나의 특수 문자가 필요하지만 일부는 특수 문자를 허용하지 않으므로 모든 경우에 사용할 수 있는 매우 강력한 황금 패스워드를 신설한때 황금 패스워드가 유효하지 않게되는 다른 요구 사항이 있는 다음 경우가 있다.

[0007] 이러한 서로 다른 패스워드 규칙의 결과로 유저들이 여러 사이트 / 조직에서 설정한 많은 패스워드를 기억하는 것이 불가능하지는 않더라도 어려울 수 있다. 따라서, 유저들은 전형적으로 자신의 정보 디바이스에 저장된 파일 및 / 또는 자신의 정보 디바이스에서 실행되는 패스워드 스토리지 애플리케이션과 같은 그들의 패스워드를 저장할 것이다. 저장된 패스워드는 해커의 표적이 될 수 있으며 패스워드가 저장된 디바이스에 액세스하면 모든

해커들은 모든 패스워드들을 액세스할 것이고 모든 유저의 패스워드로 보호 된 계정 / 사이트를 액세스 할 수 있다. 따라서, 너무 약한 패스워드를 피하기 위해 패스워드에 엄격한 규칙을 구현하는 것은 의도한 효과와 반대가 될 수 있다 (더 많은 정보를 노출하는 증가된 위험).

[0008] 전통적인 패스워드들의 이들 문제들의 면에서, 새로운 방법들이 이들 문제들을 해결하기 위해 개발되어 왔다. 이러한 방법들은 해커가 훔쳐 보거나 훔치기가 더 어렵게 하기 위해 사진, 그래픽 이미지 또는 다양한 모양과 음영을 사용하는 것을 포함할 수 있지만, 이것에 한정되지는 않는다. 일부 기술은 심지어 입력 스크린의 특정 위치에 제스처와 정보 측위를 사용하여 유저 액세스를 확인한다. 그러나, 이 방법들 중 어떤 것도 디바이스에 로그인 할 때마다 유저의 모든 움직임을 기록 할 수 있는 몰래 카메라를 무위로 만들 수는 없다. 해커가 모든 녹음을 재생하고 유저의 모든 움직임을 분석 할 수 있다면 해커는 결국 액세스를 획득할 수 있다.

[0009] 현존하는 인증 방법들의 일차 문제들은:

[0010] (1) 전통적인 패스워드 및 보안 질문 (가장 일반적으로 사용되는 방법)은 훔쳐보기방지(peek-proof)가 없다;

[0011] (2) 그래픽 이미지 및 사진 기반 방법들은 유저들이 이미지 또는 사진 파일을 업로드해야 할 수 있고, 시스템은 이미지 및 / 또는 사진을 저장하고 유지해야한다. 이것은 유저 및 시스템 부담을 증가시키고 해커가 로그인 프로세스를 기록하고 재생할 때, 이미지들도 또한 인식될 수 있다;

[0012] (3) 새로운 그래픽 및 제스처 및 / 또는 위치 기반 인증 방법은 인간과 컴퓨터 사이에서만 사용할 수 있으므로 기계 대 기계에는 사용할 수 없다.

[0013] 전술한 문제점을 보이지 않는 인증 및 암호화 시스템 및 방법에 대한 요구가 있다.

발명의 내용

과제의 해결 수단

[0014] 본 발명의 목적은 적어도 상기의 문제들 및/또는 단점들의 해결하고 적어도 이하에서 설명되는 장점들을 제공하는 것이다.

[0015] 따라서, 본 발명의 목적은 유저의 인증을 위한 시스템 및 방법을 제공하는 것이다.

[0016] 본 발명의 다른 목적은 전자 디바이스 액세스를 시도하는 유저의 인증을 위한 시스템 및 방법을 제공하는 것이다.

[0017] 본 발명의 다른 목적은 전자적으로 저장된 정보에 대한 액세스를 요청하는 유저의 인증을 위한 시스템 및 방법을 제공하는 것이다.

[0018] 본 발명의 다른 목적은 네트워크상의 디바이스에 대한 액세스를 요청하는 유저의 인증을 위한 시스템 및 방법을 제공하는 것이다.

[0019] 본 발명의 다른 목적은 미리 결정된 수의 심벌들을 함유하는 패스코드(passcode)들을 이용하는 유저의 인증을 위한 시스템 및 방법을 제공하는 것이다.

[0020] 본 발명의 다른 목적은 다수의 토큰(token)들을 이용하는 유저의 인증을 위한 시스템 및 방법을 제공하는 것이며, 여기서 각각의 토큰은 유저 패스코드를 신설하기 위해 사용되는 심벌들의 세트에서 적어도 두개의 심벌들의 그룹이다.

[0021] 본 발명의 다른 목적은 전자 정보의 암호화 및 복호화를 위한 시스템 및 방법을 제공하는 것이다.

[0022] 본 발명의 다른 목적은 전자 정보를 암호화 및 복호화하기 위한 미리 결정된 수의 심벌들을 함유하는 패스 코드를 이용하는 전자 정보의 암호화 및 복호화를 위한 시스템 및 방법을 제공하는 것이다.

[0023] 본 발명의 다른 목적은 다수의 토큰들과 조합하여 미리 결정된 수의 심벌들을 함유하는 패스코드를 이용하는 전자 정보의 암호화 및 복호화를 위한 시스템 및 방법을 제공하고, 여기서 각각의 토큰은 상기 패스코드를 신설하는데 사용되는 심벌들의 세트에서 적어도 두개의 심벌들의 그룹이다.

[0024] 전체로 또는 부분으로 적어도 상기의 목적들을 달성하기 위해서, 심벌들의 세트에서 선택된 미리 결정된 수의 심벌들 (“패스코드 심벌들(passcode symbols)”)을 포함하는 미리 결정된 패스코드를 이용하여 유저를 인증하는 방법이 제공되고, 상기 미리 결정된 패스코드 심벌들의 각각은 미리 결정된 핀 위치에 의해 특징되고 : 상기

유저에게 토큰 세트를 제공하는 단계로서, 상기 토큰 세트는 적어도 두개의 토큰들을 포함하고, 상기 토큰 세트 내 각각의 토큰은 상기 심벌들의 세트에 속하는 적어도 두개의 심벌들을 포함하는, 상기 제공하는 단계, 상기 유저가 상기 미리 결정된 패스코드내 각각의 핀 위치에 대하여 상기 토큰 세트로부터 토큰을 선택할 것을 요구하는 단계; 및 상기 유저가 선택한 상기 토큰들에 기초하여 상기 유저를 인증하는 단계를 포함한다.

[0025] 전체로 또는 부분으로 적어도 상기의 목적들을 달성하기 위해서, 심벌들의 세트에서 선택된 미리 결정된 수의 심벌들 (“패스코드 심벌들(passcode symbols)”)을 포함하는 미리 결정된 패스코드를 이용하여 유저를 인증하기 위한 시스템이 또한 제공되고, 상기 미리 결정된 패스코드 심벌들의 각각은 미리 결정된 핀 위치에 의해 특징되고 : 프로세서, 상기 프로세서에 액세스 가능한 메모리; 및 상기 프로세서에 의해 실행가능한 메모리에 저장된 컴퓨터 판독가능한 지시들의 세트를 포함하는 인증/암호화 모듈로서,상기 유저에게 토큰 세트를 제공하고, 상기 토큰 세트는 적어도 두개의 토큰들을 포함하고, 상기 토큰 세트내 각각의 토큰은 상기 심벌들의 세트에 속하는 적어도 두개의 심벌들을 포함하고, 상기 유저가 상기 미리 결정된 패스코드내 각각의 핀 위치에 대하여 상기 토큰 세트로부터 토큰을 선택할 것을 요구하고; 및 상기 유저가 선택한 상기 토큰들에 기초하여 상기 유저를 인증하는, 상기 인증/암호화 모듈을 포함한다.

[0026] 본 발명의 추가적인 장점들, 목적들 및 특징들은 이하의 설명에서 어느 정도는 개시 될 것이고, 어느 정도는 이하의 검토를 통해 당업자에게 명백해질 것이며, 또는 본 발명의 실시로부터 학습 할 수 있을 것이다. 상기 본 발명의 목적들 및 장점들은 상기 첨부된 청구항들에 언급된 것으로 실현될 수 있고 이루어질 수 있다.

도면의 간단한 설명

[0027] 본 발명은 같은 도면 번호들이 같은 엘리먼트들을 나타내는 이하의 도면들을 참고로하여 상세하게 설명될 것이다:

도 1a은 본 발명의 일 실시예에 따른 클라이언트 시스템에 의해 액세스되는 디바이스 또는 서버에 통합 될 수 있는 예시적인 인터셉션 방지 인증 / 암호화 시스템을 예시하는 블럭 다이어그램이다;

도 1b는 본 발명의 일 실시예에 따른 디바이스에 통합되는 인터셉션 방지 인증 / 암호화 시스템을 예시하는 블럭 다이어그램이다;

도 1c는 본 발명의 일 실시예에 따른 네트워크를 통하여 클라이언트 디바이스에 의해 액세스되는 서버에 통합된 인터셉션 방지 인증 / 암호화 시스템을 예시하는 블럭 다이어그램이다;

도 1d는 본 발명의 일 실시예에 따른 인터셉션 방지 인증/암호화 시스템의 일 예시적인 하드웨어 구현예의 개략도이다;

도 2a는 본 발명의 일 실시예에 따른 네개의 차원(dimension)들로 그룹화된 심벌들의 예들을 도시한다;

도 2b는 본 발명의 일 실시예에 따른 다섯개의 차원들로 그룹화된 심벌들의 예들을 도시한다;

도 3은 본 발명의 일 실시예에 따른 유저가 패스코드를 신설하는 것을 가능하게 하기 위한 인증/암호화 모듈에 의해 구현된 예시적인 프로세스를 예시하는 플로우 차트이다;

도 4는 본 발명의 일 실시예에 따른 유저를 인증하기 위한 인증/암호화 모듈에 의해 구현된 예시적인 프로세스를 예시하는 플로우 차트이다;

도 5 는 본 발명의 일 실시예에 따른 인증/암호화 모듈에 의해 사용되는 예시적인 토큰 생성 규칙을 리스트한 테이블이다;

도 6은 본 발명의 일 실시예에 따른 인증/암호화 모듈에 의해 사용되는 예시적인 토큰 선택 규칙을 리스트한 테이블이다;

도 7은 본 발명의 일 실시예에 따른 인증/암호화 모듈에 의해 사용되는 예시적인 토큰 확인 규칙을 리스트한 테이블이다;

도면들 8a 및 8b은 본 발명의 일 실시예에 따른 GATE_4 실시예에 유저 ID 신설 (등록) 프로세스의 샘플 스크린샷들이다;

도면들 9a-9d은 본 발명의 일 실시예에 따른 GATE_4 실시예에 유저 로그인 프로세스의 샘플 스크린샷들이다;

도면들 10a-10d은 본 발명의 일 실시예에 따른 텍스트 포맷에 GATE_4 실시예에 유저 로그인 프로세스의 샘플 스

크린샷들이다;

도면들 11a 및 11b은 본 발명의 일 실시예에 따른 GATE_5 실시예에 유저 ID 신설 (등록) 프로세스의 샘플 스크린샷들이다;

도면들 12a-12d은 본 발명의 일 실시예에 따른 GATE_5실시예에 유저 로그인 프로세스의 샘플 스크린샷들이다;

도면들 13a-13d은 본 발명의 일 실시예에 따른 텍스트 포맷에 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷들이다;

도 14는 본 발명의 일 실시예에 따른 평문 문자 메시지(plain text message)가 발신자 패스 코드로 암호화되는 GATE_4 실시예를 사용하는 메시지 암호화 프로세스의 샘플 스크린샷이다;

도면들 15a 및15b는 본 발명의 일 실시예에 따른 성공적인 복호화가 이루어진 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷들이다;

도면들 16a 및16b는 본 발명의 일 실시예에 따른 암호화된 필러 메시지(filler message)가 수신자 측에서 무효화되는 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷들이다;

도 17은 본 발명의 일 실시예에 따른 수신자 패스코드가 발신자 패스코드와 다르기 때문에 복호화가 실패한 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다;

도 18은 본 발명의 일 실시예에 따른 평문 문자 메시지가 발신자 패스코드로 암호화되는 GATE_5 실시예를 사용하는 메시지 암호화 프로세스의 샘플 스크린샷이다;

도면들 19a 및19b는 본 발명의 일 실시예에 따른 성공적인 복호화가 이루어진 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷들이다;

도면들 20a 및20b는 본 발명의 일 실시예에 따른 암호화된 필러 메시지가 수신자 측에서 무효화되는 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷들이다; 및

도 21은 본 발명의 일 실시예에 따른 수신자 패스코드가 발신자 패스코드와 다르기 때문에 복호화가 실패한 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다.

발명을 실시하기 위한 구체적인 내용

[0028] 본 발명은 특별히 프로그래밍된 장치로 구현된 새로운 인터셉션 방지 인증 및 암호화 메커니즘을 제공한다. 인증을 위해, 본 발명은 디바이스의 운영 체제(operating system)의 일부인 심벌들로 형성된 “패스코드(passcode)들” 을 사용한다. 일 예로서, 패스코드는 다음과 같을 수 있다:

[0029] ⓘ ♥ 2 ✉

[0030] 유저에게, 상기의 패스코드는 "나는 이메일 보내기를 좋아한다"를 의미 할 수 있고, 이는 기억하기 쉽지만 다른 사람이 알기가 어렵다. 패스코드내 각각의 심벌은 다른 핀들과 관련한 대응하는 핀 위치와 함께 "핀(pin)"으로 지칭될 것이다. 상기의 예제에서, 심벌 “ⓘ” 은 제 1 핀 위치에 있고, 심벌 “♥” 는 제 2 핀 위치에 있고, 심벌 “2” 는 제 3 핀 위치에 있고 심벌 “ ✉ ” 는 제 4 핀 위치에 있다.

[0031] 본 발명은 바람직하게는 디바이스의 운영 체제의 일부인 심벌들의 선택 세트를 사용한다는 점에서 "인-시스템(in-system)"이고, 따라서 유저들이 임의의 사진들 또는 이미지들을 업로드하는 것을 필요로 하지 않는다. 패스코드들 또는 암호화된 메시지를 신설하기 위해 사용되는 심벌들은 바람직하게는 본 출원에서 "차원(dimension)"으로 지칭될 두개 이상의 그룹들로 그룹화되는 상이한 유형들을 가진다. 이것은 패스코드들을 신설할 때 유저들에게 그 자신들을 표현하기 위한 더 많은 다양한 방법들을 제공한다.

[0032] 본 발명은 유저의 패스코드의 일부가 아닌 다수의 다른 심벌들 사이에 유저의 패스코드를 구성하는 핀들에 사용된 심벌들을 "은닉(hide)"함으로써 새로운 인터셉션 방지 인증 및 암호화 메커니즘을 제공한다. 따라서, 본질적으로, 건초 더미(haystack)에 바늘을 은닉하는 것. 구체적으로, 본 발명은 “토큰(token)들” 로 본 출원에 언급되는 것을 사용한다. 토큰은 적어도 두개의 심벌들의 그룹이다. 다수의 토큰들 ("토큰 세트")이 유저에게 제시되고, 유저의 사전 선택된 핀의 일부 또는 전부가 토큰들의 일부 또는 전부에 무작위적으로 삽입된다. 구체적으

로, 유저의 패스코드내 각각의 핀 (사전 선택된 심벌에 의해 표현된)은 유저에게 제시되는 토큰 중 하나에 포함될 수 있다. 유저는 패스코드를 구성하는 개별 핀들을 함유하는 토큰들을 선택하여 각각의 선택된 토큰의 핀 위치가 그것이 함유된 패스코드 핀의 핀 위치에 대응한다. 각각의 선택된 토큰은 유저의 패스코드에 사전-선택된 핀들 중 하나 뿐 아니라 유저의 패스 코드에서 사전 선택된 핀들 중 하나가 아닌 다른 무작위적으로 생성된 심벌들을 함유하기 때문에 어느 토큰들을 유저가 선택하였는지 관찰하는 사람은 유저의 실제 패스코드가 무엇인지 결정할 수 없다.

[0033] 유저가 패스코드를 제공 할 것을 요청받을 때마다, 다수의 토큰 사이에 무작위적으로 분포된 유저의 패스코드에 사전-선택된 개별 핀들로 무작위적으로 생성된 토큰들의 다른 세트가 생성된다. 따라서, 유저는 패스코드를 제공하라는 요청이 있을 때마다 다른 토큰들의 세트를 선택할 것이므로, 관측자가 토큰들의 선택에 기초하여 유저의 패스코드를 결정하는 것을 방지한다.

[0034] 예시적인 예제로서, 심벌들의 4 그룹들 (4 차원)이 사용될 수 있고, 각각의 차원은 36 심벌들을 함유한다. 로그인 프로세스 동안에, 유저에게 36 개의 토큰들이 디스플레이되며, 각각의 토큰은 심벌들의 네개의 차원의 각각으로부터 심벌을 함유한다. 주어진 심벌은 바람직하게는 하나의 토큰에만 디스플레이된다 (즉, 하나의 토큰에 심벌이 나타나면 다른 토큰에는 다시 나타나지 않는다). 이 예제에서, 36 개의 토큰들이 디스플레이되기 때문에, 네개의 차원의 각각에 모든 심벌이 디스플레이된다 (36 개의 각각의 토큰에 각각의 차원으로부터 하나의 심벌).

[0035] 만약 유저의 패스 코드에 핀들의(심벌들로 표시된) 수가 변수 "N"으로 표시되면, 그러면 유저는 N 토큰들을 선택해야 할 필요가 있다 (패스코드를 구성하는 개별 핀들을 함유하는 토큰들, 각각의 선택된 토큰의 핀 위치는 그것이 함유된 패스코드 핀의 핀 위치에 대응한다).

[0036] 상기에서 논의된 바와 같이, 본 발명은 유저가 패스코드 내의 각각의 핀에 대해 패스코드의 핀들 중 하나를 포함하는 4 개의 심벌을 함유하는 토큰을 입력하기 때문에 유저의 패스코드의 "훔쳐보기 및 인터셉션(peek and interception)"가능성을 감소시킬 것이다. 따라서, 설사 유저 엔트리(entry)가 로그인 스크린을 지켜보는 해커에 의해 또는 네트워크 트래픽을 인터셉트하는 해커에 의해 관측된다 하더라도 해커는 각각의 토큰에 4 개의 심벌들 중 어느 것이 유저의 패스코드를 구성하는 사전-선택된 핀에 해당하는지를 결정할 수 없다. 따라서, 만약 해커가 유저의 계정에 로그인하려고 시도하면 해커는 무작위적으로 생성된 토큰들의 다른 세트를 제공 받게되며 그 중 일부는 유저의 패스코드를 구성하는 사전-선택된 핀들을 함유하고 해커는 어느 토큰을 선택할지를 알지 못할 것이다.

[0037] 그러나, 만약 해커가 유저 로그인 프로세스를 충분히 관찰하면 해커는 모든 기록된 로그인 세션들을 비교하여 각각의 핀이 패스코드에 있는 것을 찾을 수 있는데 이는 각각의 핀은 유저가 입력한 토큰에 나타나기 때문이고 만약 해커가 다른 로그인 세션의 첫 번째 토큰들 전부를 비교하면, 해커는 결국에는 첫 번째 핀을 결정하고 만약 해커가 모든 세션의 모든 두 번째 토큰들을 비교하면 해커는 결국 두 번째 핀을 등등 결정할 것이다.

[0038] 따라서, 시간이 흘러 해커가 패스코드에서 핀을 찾는 것을 방지하기 위해, 유저에게 제시된 무작위적으로 생성된 토큰들의 수는 바람직하게는 각 각의 차원의 심벌들의 수보다 적다. 예를 들어, 만약 하나 이상의 차원의 각각이 36 개의 심벌들을 함유하는 경우, 유저에게 16 개의 토큰들만 제시하도록 선택될 수 있다. 이 결과는 유저 핀이 토큰에 나타날 수도 있다는 보장이 없다는 것이다. 이 실시예에서, 만약 유저가 로그인을 시도하고 유저의 패스코드에 하나 이상의 핀들이 임의의 토큰들에 존재하지 않으면, 그러면 유저는 임의의 토큰을 임의의 토큰에 존재하지 않는 핀의 "와일드 카드(wildcard)" 토큰으로 선택한다 (선택된 와일드카드 토큰의 핀 위치는 누락된 패스 코드 핀의 핀 위치에 대응한다). 이것은 실제로 핀을 함유하지 않는 유저의 핀들 중 하나에 무작위적으로 선택된 토큰이 있을 수 있기 때문에 해커의 추측 작업이 훨씬 어려워진다.

[0039] 하나 이상의 차원에서 심벌의 수보다 적은 토큰을 사용하는 또 다른 장점은 (예를 들어, 하나 이상의 차원에 심벌들의 수가 36 일 때 16 개의 토큰 사용하는 것)은 유저가 신속하게 사전-선택된 핀이 토큰들에 있는지 없는지를 찾는 것을 더 쉽게 하고 유저에게 스크린이 더 단순해 보인다는 것이다.

[0040] 본 발명은 바람직하게는 본 시스템을 통합한 디바이스의 운영 체제의 일부인 심벌들을 사용한다. 따라서, 유저가 시스템에 특별한 그래픽스나 사진들을 로드할 필요가 없고 따라서 유저와 시스템이 이러한 특성을 저장하고 유지 관리하는 로드가 줄어든다.

[0041] 본 발명의 시스템 및 방법은 유저를 인증하는 것뿐만 아니라 다수의 정보를 인증하는 데에도 사용될 수 있으며, 따라서 메시지들을 암호화하는데 사용될 수 있다.

[0042] 도 1a은 본 발명의 일 선호되는 실시예에 따른 클라이언트 시스템에 의해 액세스되는 디바이스 또는 서버에 통합 될 수 있는 예시적인 인터셉션 방지 인증 / 암호화 시스템(100)을 예시하는 블럭 다이어그램이다. 시스템(100)은 인터셉션 방지 인증 및/또는 암호화 기능을 제공하는 인증/암호화 모듈 (110)을 포함한다. 시스템(100)은 옵션으로 네트워크 (130)에 연결될 수 있다.

[0043] 도 1b는 본 발명의 일 선호되는 실시예에 따른 디바이스(150)에 통합되는 인터셉션 방지 인증 / 암호화 시스템을 예시하는 블럭 다이어그램이다. 디바이스 (150)는 바람직하게는 유저 인터페이스 모듈 (120)을 포함한다. 인증/암호화 모듈 (110)은 이하에서 더 상세하게 설명될 메시지의 암호화 및/또는 유저의 패스코드의 안전한 인증을 제공한다. 인증/암호화 모듈 (110)은 네트워크 (130)에 연결될 수 있다.

[0044] 유저 인터페이스 모듈 (120)은 예를 들어, 그래픽 유저 인터페이스, 웹 기반 인터페이스 및 유사한 것과 같은 관련 기술 분야에서 알려진 임의의 유형의 유저 인터페이스로 구현될 수 있다. 일반적으로, 유저 인터페이스 모듈 (120)은 유저가 인증 모듈 (110)과 상호작용하는 것을 가능하게 하는 임의 유형의 인터페이스로 구현될 수 있다.

[0045] 도 1c는 본 발명의 일 선호되는 실시예에 따른 네트워크를 통하여 클라이언트 디바이스(170)에 의해 액세스되는 서버(160)에 통합된 인터셉션 방지 인증 / 암호화 시스템을 예시하는 블럭 다이어그램이다. 클라이언트 디바이스 (170)는 네트워크 (130)를 통하여 서버 (160)를 액세스할 수 있는 임의의 유형의 컴퓨터 또는 디바이스이고, 바람직하게는 유저가 클라이언트 디바이스 (170)와 상호작용하는 것을 가능하게 하는 유저 인터페이스 모듈 (120)을 포함한다. 인증/암호화 모듈 (110)은 이하에서 더 상세하게 설명될 메시지의 암호화 및/또는 유저의 패스코드의 안전한 인증을 제공한다.

[0046] 통신 링크들 (140)은 인증/암호화 모듈 (110), 유저 인터페이스 모듈 (120), 네트워크 (130) 및 클라이언트 디바이스 (170) 간의 통신을 위해 사용된다. 통신 링크들 (140)은 유선 링크들, 무선 링크들, 무선 유도성(inductive) 링크들, 정전용량성(capacitive) 링크들 또는 전자 컴포넌트들을 연결하기 위해 관련 기술 분야에서 알려진 임의의 다른 메커니즘들일 수 있다. 하드와이어(hardwire) 링크는 예를 들어, ISA (Industry Standard Architecture) 버스, MCA(Micro Channel Architecture) 버스, 강화된 ISA 버스, VESA(Video Electronics Standards Association) 로컬 버스 또는 PCI(Peripheral Component Interconnect) 버스와 같은 버스로 적절하게 구현될 수 있다.

[0047] 무선 링크들의 예들은, WAP (Wireless Application Protocol) 링크, GPRS (General Packet Radio Service) 링크, GSM (Global System for Mobile Communication) 링크, CDMA (Code Division Multiple Access) 또는 TDMA (Time Division Multiple Access) 링크, 예컨대 셀룰러 폰 채널, GPS (Global Positioning System) 링크, CDPD (Cellular Digital Packet Data), RIM (Research in Motion, Limited) 듀플렉스 페이징 (duplex paging) 유형 디바이스, 블루투스 라디오 링크, 또는 IEEE 802.11기반 라디오 주파수 링크 (와이파이)를 포함하지만, 이것에 한정되지는 않는다.

[0048] 네트워크 (130)는 유선 또는 무선 네트워크일 수 있고, 예를 들어, 인터넷, 인트라넷, PAN (Personal Area Network), LAN (Local Area Network), WAN (Wide Area Network) 또는 MAN (Metropolitan Area Network), SAN(storage area network), 프레임 릴레이 연결, AIN(Advanced Intelligent Network) 연결, SONET(synchronous optical network) 연결, 디지털 T1, T3, E1 또는 E3 라인, DDS(Digital Data Service) 연결, DSL (Digital Subscriber Line) 연결, 이더넷 연결, ISDN (Integrated Services Digital Network) 라인, 다이얼-업 포트 예컨대 V.90, V.34bis 아날로그 모뎀 연결, 케이블 모뎀, ATM (Asynchronous Transfer Mode) 연결, FDDI (Fiber Distributed Data Interface) 또는 CDDI (Copper Distributed Data Interface) 연결 중 임의의 하나 이상을 포함할 수 있거나 또는 임의의 하나 이상에 인터페이스 할 수 있다.

[0049] 유저 인터페이스 모듈 (120)은 예를 들어, 그래픽 유저 인터페이스, 웹 기반 인터페이스 및 유사한 것과 같은 관련 기술 분야에서 알려진 임의의 유형의 유저 인터페이스로 구현될 수 있다. 일반적으로, 유저 인터페이스 모듈 (120)은 유저가 인증/암호화 모듈 (110)과 상호작용하는 것을 가능하게 하는 임의 유형의 인터페이스로 구현될 수 있다.

[0050] 본 출원에서 사용되는 용어 “모듈(module)”은 실제 디바이스, 컴포넌트, 또는 예를 들어, ASIC(application specific integrated circuit) 또는 FPGA(field-programmable gate array)를 포함할 수 있는 하드웨어를 이용하여 구현된 컴포넌트들의 배열,또는 모듈의 기능들을 수행하기 위해 마이크로프로세서 시스템을 특정 목적 디바이스로 변환하는(실행되는 동안) 모듈의 기능을 구현하기 위한 지시들의 세트 및 마이크로프로세서 시스템을

의미한다.

[0051] 모듈은 또한 하드웨어 단독 및 소프트웨어로 제어 되는 하드웨어의 조합으로서 구현 될 수 있고, 특정 기능들은 하드웨어만으로 가능하게되고 다른 기능들은 하드웨어와 소프트웨어의 조합에 의해 가능하게 된다. 어떤 구현예들에서, 모듈의 적어도 일부, 및 일부 경우들에서, 모듈의 전부는 운영 체제, 시스템 프로그램 및 애플리케이션 프로그램을 실행하는 컴퓨터 또는 디바이스 (예를 들어, 서버 (160) 및 클라이언트 디바이스 (170)와 같은)의 프로세서 (들)상에서 실행될 수 있고, 한편 또한 멀티 태스킹, 멀티 스레딩, 분산 (예를 들어, 클라우드) 프로세싱 또는 다른 이런 기술들을 이용하여 모듈을 구현한다. 이런 컴퓨터 또는 디바이스의 예들은 퍼스널 컴퓨터 (예를 들어, 데스크탑 컴퓨터 또는 노트북 컴퓨터), 서버, 현금 자동 입출금기 (ATM), POS(point-of-sale) 단말, 기기, 및 모바일 컴퓨팅 디바이스, 예컨대 스마트폰, 태블릿, 또는 개인 디지털 보조장치 (PDA)를 포함하지만, 이것에 한정되지는 않는다. 더구나, 서버 (160)는 적절하게 윈도우즈 서버, 리눅스 서버, 유닉스 서버 또는 유사한 것과 같은 임의의 유형의 서버이다.

[0052] 도 1d는 본 발명의 일 실시예에 따른 인터셉션 방지 인증/암호화 시스템(100)의 일 예시적인 하드웨어 구현예의 개략도이다. 도 1d의 실시예에서, 인증/암호화 모듈 (110)은 CPU (118) 및 메모리 (112)에 의해 구현된다.

[0053] CPU (118)는 실행을 위해 메모리 (112)에 저장된 운영 체제 코드(114) 및 다른 프로그램 코드 (116)를 액세스한다. 인증/암호화 모듈 (110)의 기능을 구현하는 프로그램 코드 (116)는 CPU (118)에 의한 액세스 및 실행을 위해 메모리 (112)에, 또는 외부 스토리지 디바이스 (미도시)상에 저장된다.

[0054] 메모리 (112)는 예를 들어, RAM(random-access memory), 캐시 메모리, 착탈 가능한/비-착탈 가능한 스토리지 매체, 휘발성/비-휘발성 스토리지 매체, 예컨대 비-착탈 가능한 비-휘발성 하드 디스크 드라이브, 착탈 가능한 비-휘발성 플로피 디스크 드라이브, 광 디스크 드라이브 (예컨대 CD-ROM, DVD-ROM, 또는 임의의 다른 광 스토리지 매체), USB 플래시 드라이브, 및 메모리 카드에 의해 구현될 수 있다.

[0055] 인증/암호화 모듈 (110)의 기능이 이제 설명될 것이다. 인증 / 암호화 모듈 (110)은 토큰에 무작위로 삽입 된 유저의 사전-선택된 핀들을 갖는 다수의 토큰들을 (유저 인터페이스 모듈 (120)을 통해) 유저에게 디스플레이함으로써 유저 로그인 요청에 응답하여 패스코드 /챌린지(challenge) 인증을 제공한다. 상기에서 논의된 바와 같이, 유저의 패스코드내 각각의 핀 (사전-선택된 심벌에 의해 표현된)은 유저에게 제시되는 토큰 중 하나에 포함될 수 있다. 적어도 하나의 핀은 토큰들 중 하나에 제시된다. 유저는 패스코드를 구성하는 개별 핀들을 함유하는 토큰들을 선택하여 각각의 선택된 토큰의 핀 위치가 그것이 함유된 패스코드 핀의 핀 위치에 대응한다. 각각의 선택된 토큰은 유저의 패스코드에 사전-선택된 핀들 중 하나 뿐만 아니라 어쩌면 유저의 패스 코드에서 사전-선택된 핀들 중 하나인 무작위적으로 생성된 심벌들을 함유하기 때문에 어느 토큰들을 유저가 선택하였는지 관찰하는 사람은 유저의 실제 패스코드가 무엇인지 결정할 수 없다.

[0056] 인증 / 암호화 모듈 기능의 2 가지 예시적인 실시 예가 이하에서 설명 될 것이고, 본 출원에서는 "GATE_4"(Graphical Access Tabular Entry_4) 및 "GATE_5"(Graphical Access Tabular Entry_5)로 언급 될 것이다. 도 2a에 도시된 바와 같이, GATE_4 실시예는 4 차원의 심벌들을 사용하고, 36 심벌들이 각각의 차원에 포함된다. 도 2b에 도시된 바와 같이, GATE_5 실시예는 5 차원의 심벌들을 사용하고, 26 심벌들이 각각의 차원에 포함된다. 도면들 2a 및 2b에 도시된 심벌들은 사용될 수 있는 심벌들의 유형들의 예들이고, 임의의 다른 유형들의 심벌들이 사용될 수 있고 또한 본 발명의 범위내에 있다는 것이 이해되어야 한다.

[0057] 도면들 2a 및 2b에 도시된 심벌들은 현대의 컴퓨터 운영 체제들에 이용가능하고, 본 발명을 통합하는 현존하는 시스템으로 신설 또는 업로드/저장하기 위해 임의의 특별한 프로세스를 요구하지 않는다. 그것들은 대부분의 컴퓨터 운영 체제들에서 발견되는 표준 유니코드(Unicode) 체계의 서브세트이다. 각각의 심벌은 유니코드 ID를 가지며, 익숙한 문자들 : a, b, … z, 0, 1, 9, @, +, <, %은 전부 유니코드 체계의 일부이다. 예를 들어:

[0058] 유니코드 \u0062는 문자: b를 위한 것이다

[0059] 유니코드 \u2206은 문자: Δ를 위한 것이다

[0060] 유니코드 \u0040은 문자: @를 위한 것이다

[0061] 도면들 2a 및도 2b의 실시예들에 도시된 심벌들은 다양하고 특유하며 기억하기 쉽기 때문에 포함되었다.

[0062] 도 3은 본 발명의 일 실시예에 따른 유저가 패스코드를 신설하는 것을 가능하게 하기 위한 인증/암호화 모듈 (110)에 의해 구현된 예시적인 프로세스를 예시하는 플로우 차트이다. 프로세스는 단계 (310)에서 시작하고, 여기서 유저는 희망하는 유저 ID를 입력하도록 요구된다. 단계 (320)에서, 인증/암호화 모듈 (110)은 만약 유저

ID가 메모리 (112)에 이미 존재하는지 여부를 결정한다. 만약 유저 ID가 존재하면, 프로세스는 단계 (330)로 이어진다. 그렇지 않으면, 프로세스는 단계 (340)으로 이어진다.

[0063] 단계 (330)에서, 유저는 그들이 유저 ID와 관련된 현존하는 패스코드에 덮어 쓰기(overwrite)를 원하는지가 질문된다. 만약 유저가 “아니오(no)”를 표시하면, 프로세스는 단계 (310)로 다시 점프한다. 만약 유저가 “예(yes)”를 표시하면, 프로세스는 단계 (340)로 진행한다.

[0064] 단계 (340)에서, 유저가 패스 코드의 각 핀에 대해 선택할 수 있는 심벌이 유저에게 디스플레이된다. 그런다음, 단계 (350)에서, 유저는 패스코드에 대한 핀들 중 하나로서 디스플레이된 심벌들 중 하나를 선택할 것이 요청된다. 단계 (360)에서, 프로세스는 유저가 현재 선택된 핀 (들)이 패스 코드로서 저장되어야 하는지를 요구했는지 여부를 결정한다. 이것은, 예를 들어 유저가 패스코드를 저장할 준비가 되었을 때 유저가 선택할 수 있는 아이콘을 디스플레이함으로써 구현 될 수 있다. 만약 유저가 패스코드가 저장되어야 한다고 표시하지 않으면, 프로세스는 단계 (350)으로 되돌아 가며, 여기서 유저는 패스코드 내의 다른 핀에 대한 다른 심벌을 선택한다. 만약 유저가 단계 (360)에서, 해당 패스코드는 저장되어야 한다고 표시하면, 그러면 프로세스는 단계 (370)로 진행한다.

[0065] 단계 (370)에서, 선택된 패스코드가 미리 결정된 길이 요건 (즉, 미리 결정된 최소 수의 핀들)을 준수하는 여부가 결정된다. 만약 선택된 패스코드가 준수하면, 단계 (380)에서 패스코드가 저장된다. 만약 패스코드가 준수하지 않으면, 프로세스는 단계 (350)으로 되돌아 가며, 여기서 유저는 추가 핀을 위한 심벌을 선택하도록 프롬프트(prompt)된다.

[0066] 도 4는 본 발명의 일 실시예에 따른 유저를 인증하기 위한 인증/암호화 모듈(110)에 의해 구현된 예시적인 프로세스를 예시하는 플로우 차트이다. 프로세스는 단계 (410)에서 시작하고, 여기서 유저가 유저 ID를 입력하도록 프롬프트되는 로그인 스크린이 유저에게 제공된다. 그 다음, 프로세스는 단계 (420)으로 진행하며, 여기서 인증 / 암호화 모듈 (110)은 유저에 의해 입력된 유저 ID가 존재 하는지를 결정한다. 만일 그렇다면, 프로세스는 단계 (430)으로 진행한다. 만약 그렇지 않으면, 프로세스는 단계 (410)으로 되돌아 가서, 여기서 유저는 다른 유저 ID를 입력하도록 프롬프트된다.

[0067] 단계 (430)에서, 인증 / 암호화 모듈은 유저의 패스 코드를 위해 사용된 핀들의 수에 기초하여 미리 결정된 수의 토큰들을 생성한다. 도 4에 설명된 실시예에서, 16 개의 토큰들이 생성되고, 바람직하게는 아래에서 보다 상세하게 설명되는 바와 같이 4 x 4 토큰 테이블로 유저에게 디스플레이된다. 토큰은 도 5에 도시된 토큰 생성 규칙에 기초하여 생성된다. 예들에서, 테이블에 16 개의 토큰들이 있지만, 실제로는 유저 핀 수보다 큰 임의의 개수의 토큰이 될 수 있고, 3 x 4, 2 x 5 또는 임의의 다른 조합을 가정하면, 4 x 4가 그것들을 디스플레이하기에 가장 선호되는 방법이다.

[0068] 그런 다음, 프로세스는 단계 (440)로 진행하여, 여기서 유저는 도 6에 도시된 토큰 선택 규칙들에 따라, 그의 패스코드에서 핀들이 나타나는 순서로 자신의 패스코드내 핀들을 함유하는 토큰들을 선택한다. 단계 (450)에서, 인증 / 암호화 모듈 (110)은 유저 선택된 토큰들이 도 6에 도시된 토큰 선택 규칙을 따르는지를 결정한다. 토큰 선택 규칙이 따르면, 프로세스는 단계 (460)으로 진행하고, 여기서 유저는 인증되고 액세스가 허용된다. 만약 토큰 선택 규칙들을 따르지 않으면, 프로세스는 단계 (470)로 진행하고, 여기서 유저는 인증되지 않고 액세스가 거부된다.

[0069] 상기에서 논의된 바와 같이, 도 5 는 본 발명의 일 실시예에 따른 인증/암호화 모듈(110)에 의해 사용되는 예시적인 토큰 생성 규칙을 리스트한 테이블이다. 유저의 패스코드에 핀들 중 적어도 하나는 16개의 토큰들 중 적어도 하나에 있을 것이다. 때때로 모든 유저 핀들이 토큰들에서 발견 될 수 있으며 대부분 1 내지 N (N은 N 개의 핀들을 갖는 유저 패스코드의 길이이다) 사이에서 유저가 사전-선택한 핀은 16 개의 토큰들에 있을 것이다.

[0070] 도 5에 도시된 실시 예에서, 36 개가 아닌(GATE_4 실시예에서는), 26 개가 아닌(GATE_5 실시 예의 경우), 16 개의 토큰들이 생성된다. 따라서, GATE_4 실시예의 경우에, 유저의 패스코드 내의 핀들 중 하나가 16 개의 토큰들 중 하나에 나타날 수 있는 때의 단지 16/36 = 44 %이다. GATE_5 실시예의 경우, 때의 16/26 = 62 %만이 유저의 패스코드의 핀 중 하나가 16 개의 토큰 중 하나에 나타날 것이다. 유저의 패스코드내 모든 핀들이 토큰 테이블에 나타날 가능성이 있으며, 최소 하나의 유저 핀이 토큰 테이블에 존재할 것이라는 보장이 있다. 대부분, 유저의 패스코드에 핀들 중 일부는 토큰들에 누락될 것이고, 일부는 토큰들에 나타날 것이다. 대안 실시예들에서, 유저의 패스코드내 핀들 중 적어도 2개 또는 3개가 토큰 테이블에 존재하도록 변형될 수 있다.

[0071] 이 불확실성이 본 발명을 효과적으로 만든다. 만약 로그인 프로세스가 훔쳐보이거나 인터셉트되면, 해커에게 확

실한 유일한 것은 패스코드의 길이인데, 이는 만약 유저가 패스코드 길이보다 많거나 적은 토큰들을 입력하면 로그인이 실패하기 때문이다. 성공적인 로그인을 초래하지만 보증하지는 않는 유일한 방법은 유저의 패스코드와 동일한 수의 토큰들을 입력하는 것이다.

[0072] 그러나, 해커가 패스코드가 얼마나 긴지를 알더라도, 해커는 개별 핀의 아이덴티티를 결정할 수 없다. 이는 16 개의 토큰들 중 하나에 핀이 반드시 표시되지 않더라도 유저가 성공적으로 로그인 할 수 있기 때문이다. 이것은, 도 6의 토큰 선택 규칙들에 표시된 바와 같이, 유저는 제시되는 토큰들 중 하나에 존재하지 않는 핀에 대한 랜덤 토큰을 선택할 수 있기 때문이다.

[0073] 또한, 비록 유저 패스코드 내의 모든 핀들이 16 개의 토큰들에 나타나더라도, GATE_4 실시 예에서 4 개의 심벌이 있고, GATE_5 실시 예에서 5 개의 심벌이 있기 때문에 해커는 여전이 각각의 토큰내 어느 심벌이 사전-선택된 핀인지를 알 수 없을 것이다. 이러한 불확실성이 본 발명의 시스템 및 방법을 훔쳐보기 및 인터셉션 방지로 만든다.

[0074] 도 6에 나열된 토큰 선택 규칙들에 나타난 바와 같이, 유효한 토큰을 선택하기 위한 규칙들은 다음과 같이 요약될 수 있다 :

[0075] · 유저는 유저의 패스코드내 N 개의 핀들에 대응하는 토큰 테이블에서 N 개의 토큰을 선택해야 한다. 따라서, 만약 유저의 패스코드가 4개의 핀들을 가지면, 유저는 4 개의 토큰들을 선택한다. 유사하게, 만약 유저의 패스코드가 6개의 핀들을 가지면, 유저는 6개의 토큰들을 선택해야 한다.

[0076] · 만약 유저의 핀이 16 토큰들 중 하나에 나타나면, 유저는 핀을 입력하기 위해 해당 토큰을 선택해야 한다.

[0077] · 만약 유저의 패스코드내 핀들 중 하나가 16 개의 토큰들 중 하나에도 없는 경우, 유저는 해당 핀에 대해 16 개의 토큰들 중 하나를 선택해야 한다 (이하 "와일드카드 토큰(wildcard token)"이라고 함).

[0078] 상기에서 논의된 바와 같이, 도 7은 본 발명의 일 실시예에 따른 인증/암호화 모듈(110)에 의해 사용되는 예시적인 토큰 확인 규칙을 리스트한 테이블이다. 이들 규칙들은 유저가 로그인 프로세스에서 선택한 토큰들을 확인하기 위한 것이다. 예를 들어, 규칙들은 유저에 의해 입력된 토큰들의 수가 유저의 패스코드 길이와 같은지 여부를 결정한다. 만약 그렇지 않다면, 유저 로그인은 실패할 것이다. 만약 그렇다면, 규칙들은 유저의 패스코드내 각각의 핀들을 체크하여 16 개의 토큰들 중 하나에 있는지 확인해야 한다. 만약 유저의 패스코드내 핀들 중 하나가 토큰들 중 하나에 없으면, 유저는 랜덤 토큰을 선택해야 한다. 만약 유저의 패스코드내 핀이 토큰들 중 하나에 나타나지 않으면, 유저는 해당 토큰을 선택해야 한다.

[0079] 도 8a는 본 발명의 일 실시예에 따른,도 3의 단계 (310)에서 유저 ID를 입력하기 위한 비어있는 등록 스크린의 샘플 스크린 샷이다. 도면은 유저가 유저 ID를 입력하기 전에 GATE_4 실시예의 등록 스크린이 어떻게 나타날 수 있는지의 예를 도시한다.

[0080] 도 8b는 본 발명의 일 실시예에 따른 시스템 (100)에 의해 제공된 유저 ID를 신설하기 위한 등록 프로세스의 실행의 프레임의 샘플 스크린샷이다. 유저가 도 3의 등록 (유저 ID 신설) 프로세스를 다음과 같이 진행할 때 GATE_4 실시예 스크린이 어떻게 보이는지에 대한 일례를 도시한다 :

[0081] · 유저는 새로운 유저 ID : "admin" (G4 (206))를 입력하고, 그런 다음 "이용 가능성 체크" 버튼 (G4 (208))을 클릭한다. 시스템은 ID "admin"가 이미 그것의 메모리에 있는지 여부를 조사하기 위해 체크한다 (도 3의 단계 (320)). 만약 그렇다면, : “유저 ID 가 이미 존재하니, 현존하는 패스코드를 덮어쓰기(overwrite)를 원하십니까 ?” 라고 묻는 다이어로그 (미도시)를 디스플레이 할 것이다. 만약 유저가 구 패스코드를 덮어쓰기를 원하지 않으면, 프로세스는 다이어로그를 닫고 유저가 다른 유저 ID를 기다린다. 만약 유저 ID "admin"이 존재하지 않거나 또는 만약 존재하지만 유저가 현존하는 패스코드에 덮어 쓰기를 원하면, 시스템은 G4 (212), G4 (222), G4 (232) 및 G4 (242)에 버튼들을 인에이블할 것이고, 이의 각각은 도 2a에 도시 된 바와 같이 미리 정의된 차원으로부터 36개의 심벌들을 가진다. 예를 들어 G4 (212)는 첫번째 차원의 전체 36 개 숫자들을 포함하고, 해당 심벌들은 G4 (210)에는 도시된 바와 같이 "[1]"(상단 왼쪽)위치의 토큰에 표시될 것이다 (숫자는 1부터 36까지이다). G4 (222)는 "Ⓐ"로부터 "?"까지 36개의 심벌들을 가지며 그것들은 "[2]" (G4 (220) : 상단 오른쪽) 위치에서 임의의 토큰에 표시될 것이다. G4 (232)는 "○"로부터 "°F"까지 36개의 심벌들 가지며, 이는 토큰내 "[3]" (G4 (230) : 하단 왼쪽)에 표시될 것이고, G4 (242)는 "+" 로부터 "₪"까지 36개의 심벌들을 가지며 그것들은 "[4]" (G4 (240) : 하단 오른쪽) 위치에서 임의의 토큰에 표시될 것이다. 프로세스의 이 스테이지에서,

시스템은 G4 (212), G4 (222), G4 (232) 및 G4 (242)에 상기의 버튼들을 인에이블할 것이고, 유저는 그것들 중 임의의 것을 클릭할 수 있다. 비교로서, 도 8a에서 유저가 아직 유저 ID를 입력하지 않았기 때문에 해당 버튼들이 인에이블(enable)되지 않고 흐리게 보인다. 유저 ID 없이, 유저는 패스코드를 신설하는 것이 허용되지 않는다.

[0082] · 유저는 제 1 핀을 선택하기 위해서 G4 (222)에 심벌들 중에 "ⓘ" 를 클릭하고, 그것은 G4 (250) (도 3의 단계 (350))에 나타난다.

[0083] · 유저는 제 2 핀을 선택하기 위해서 G4 (232)에 심벌들 중에 "♥" 를 클릭하고, 그것은 G4 (252) (도 3의 단계 (350))에 나타난다.

[0084] · 유저는 제 3 핀을 선택하기 위해서 G4 (212)에 심벌들 중에 "2" 를 클릭하고, 그것은 G4 (254) (도 3의 단계 (350))에 나타난다.

[0085] · 유저는 제 4 핀을 선택하기 위해서 G4 (242)에 심벌들 중에 "✉" 를 클릭하고, 그것은 G4 (256) (도 3의 단계 (350))에 나타난다.

[0086] · 이 예에서 유저가 패스코드내 4 개의 핀들을 갖도록 선택하여서, 패스코드 길이는 4이다.

[0087] · 이 예에서 위치들 G4 (258) 및 G4 (260)은 공백으로 남겨진다.

[0088] · 유저는 그런 다음 "Save" 버튼 - G4 (270) (도 3의 단계 (370))를 클릭함으로써 유저 ID 신설 (등록) 프로세스를 마친다. 시스템은 유저 ID "admin" 와 패스코드 "ⓘ ♥ 2 ✉ "를 메모리에 저장할 것이다 (도 3의 단계 (380)).

[0089] 상기에서 논의된 바와 같이, 차원 옵션들은 도 8b의 예제에 도시된 심벌들에 제한되지 않는다. 차원 옵션들은 임의의 다른 심벌들을 포함할 수 있다. 상기에서 설명된 예시적인 실시예에서 4개의 유저 핀들이 있지만, 그러나, 본 발명의 범위내에 해당하는 임의 개수의 핀들이 사용될 수 있다. 만약 핀들의 수가 너무 작으면, 패스코드는 너무 취약할 것이다. 만약 핀들의 수가 너무 많으면, 유저는 패스코드를 기억할 수 없다. 따라서, 선호되는 길이는 4 핀과 6 핀 사이이다.

[0090] 도 9a는 유저가 임의의 정보를 입력하기 전에 본 발명의 일 실시예에 따른 시스템 (100)에 의해 제공된 로그인 스크린의 실행의 프레임의 샘플 스크린샷이다. 도 9a는 도 9b에 비교로서 사용된다.

[0091] 도 9b는 본 발명의 일 실시예에 따른 시스템 (100)에 의해 실행되는 로그인 프로세스의 실행의 프레임의 샘플 스크린샷이다. 유저가 도 4의 로그인 프로세스를 다음과 같이 진행할 때 GATE_4 실시예 스크린이 어떻게 보이는지에 대한 일례를 도시한다 :

[0092] · 유저는 유저 ID : "admin" (G4 (306)) (도 4의 단계 (410))를 입력한다.

[0093] · 유저는 "Enter" 버튼 (G4 (309))을 클릭한다. 시스템은 id "admin"이 이미 그것의 메모리에 있는지 (도 4의 단계 (420)를 조사하기 위해 체크할 것이고, 만약 존재하지 않으면 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력해 주세요 " 라는 메시지 (미도시)를 표시 할 것이다. 만약 그것이 존재하면, 시스템은 4 x 4 테이블 (G4 (320))을 디스플레이할 것이다. 테이블을 더 잘 설명하기 위해, 로우(row)들은 바람직하게는 A, B, C, D와 같이 상단으로부터 바닥으로 마킹된다. 컬럼(column)들은 또한 바람직하게는 1, 2, 3, 4와 같이 왼쪽에서 오른쪽으로 마킹된다. 이 테이블에 토큰들은 도 5에 설명된 규칙들에 따라 생성된다.

[0094] · 도 8b로부터 유저 ID "admin"과 연관된 패스 코드는 "ⓘ ♥ 2 ✉ "임을 알기 때문에, 유저는 테이블의 16 개의 토큰들을 검토하여 제 1 핀:"ⓘ"를 발견해야 한다. 심벌 ⓘ 은 두번째 차원에 속하기 때문에, 이 예제에서 임의의 토큰의 상단 오른쪽 위치에 나타나고, 따라서 유저는 단지 ⓘ 가 존재하는지를 조사하기 위해 각각의 토큰의 상단 오른쪽 위치를 스캔할 필요가 있다. 이 예에서, 그것은 토큰 D2에 있다. 이 스크린샷은 데모 모드 (demo mode)로 취해진 것이고, 데모 모드에서 프로그램은 프로세스를 더 잘 이해하기 위해 유저에 매칭 심벌을 하일라이트(highlight)한다. 실제로, 이것은 하이라이트될 필요는 없다. 이 예시에서, D2에 ⓘ 가 하이라이트된

다. 그것은 D2 토큰에서 발견되기 때문에, 유저는 도 6에 설명된 규칙들에 따라 이 토큰을 클릭해야 한다.

[0095] · 유저가 D2를 클릭한 후에, 토큰은 패스 코드의 제 1 위치 (G4 (350))로 복사된다 (도 4의 단계 (440)).

[0096] · 유저 패스코드내 제 2 핀은 : "♥"이고, 이 심벌은 세번째 차원에 속한다. 이 예에서, 세번째 차원 심벌들은 임의의 토큰의 하단 좌측에 표시된다. 따라서, 유저는 단지 각각의 16 토큰들의 하단 좌측을 찾아볼 필요가 있다. 이 예에서, 그것은 토큰 D3에 있다. 따라서, 유저는 D3 (도 4의 단계 (440))을 클릭해야 한다. 그것이 이 예시에서 하이라이트된다.

[0097] · 유저가 토큰 D3(도 4의 단계(440))을 클릭한 후에, 토큰은 패스코드의 제 2 위치로 복사된다(G4 (353)).

[0098] · 패스코드내 제 3 핀은 "2"이고, 그것은 첫번째 차원에 속한다. 따라서, 이 예제에서, 유저는 단지 각각의 16 토큰들의 상단 왼쪽 위치를 찾아볼 필요가 있다. 이 예에서, 그것이 존재하지 않는다. 토큰 선택 규칙들에 따라 (도 6), 유저는 해당 핀의 위치에 대하여 와일드카드 토큰 (임의의 토큰)을 선택할 수 있고 그리고 선택해야 한다. 이 예에서, 유저는 무작위적으로 토큰 C3를 클릭한다. 해당 토큰은 세번째 핀 위치 - G4 (356)로 복사된다.

[0099] · 제 4 의 마지막 핀은 " ✉ "이고, 이 심벌은 네번째 차원에 속한다. 이 예에서, 네번째 차원 심벌들은 임의의 토큰의 하단 오른쪽에 표시된다. 따라서, 유저는 단지 각각의 16 토큰들의 하단 오른쪽을 체크할 필요가 있다. 이 예에서, 그것은 토큰 B3에 있어서, 유저는 B3 (도 4의 단계 (440))을 클릭해야 한다. 이 예시에서, 그것이 하이라이트된다. 유저가 B3를 클릭한 후에, 토큰은 패스 코드의 네번째 위치 (G4 (359))로 복사된다.

[0100] · 패스코드에 단지 4개의 핀들이 있기 때문에, 위치들 G4 (362) 및 G4 (365)은 공백으로 남겨지고, 그것들은 공백으로 남아있게 된다. 만약 유저가 하나 또는 둘 모두의 위치들에 토큰을 입력하면, 시스템은 유저 액세스를 거부할 것인데, 이는 유저는 원래의 4개 핀들보다 더 많은 토큰들을 입력하였기 때문이다 (도 4의 단계 (450)).

[0101] · 유저가 4개의 토큰들을 모두 입력하면 유저는 "Login"(G4 (370)) 버튼을 클릭하여 유저가 토큰 선택 프로세스를 완료했다는 것을 시스템에 알리고, 시스템은 입력된 토큰들이 도 7에 설명된 규칙들에 따라 유효한지 확인할 것이다 (도 4의 단계 (450)).

[0102] · 이 예제에서, 유저가 입력한 토큰들이 유효하고, 시스템은 "로그인 성공"메시지 (G4 (380))를 표시하고 유저 액세스를 승인한다 (도 4의 단계 (460)).

[0103] 도 9c는 본 발명의 일 실시예에 따른 실패한 로그인 프로세스에 대하여 시스템 (100)에 의해 실행되는 로그인 프로세스의 실행의 프레임의 샘플 스크린샷이다. 유저가 실패한 로그인 프로세스(도 4) 를 다음과 같이 진행할 때 GATE_4 실시예 스크린이 어떻게 보이는지에 대한 일례를 도시한다 :

[0104] · 유저는 유저 ID : "admin" (G4 (307)) (도 4의 단계 (410))를 입력한다.

[0105] · 유저는 "Enter" 버튼 (G4 (310))을 클릭한다. 시스템은 id "admin"이 이미 그것의 메모리에 있는지 (도 4의 단계 (420)를 조사하기 위해 체크할 것이고, 만약 존재하지 않으면 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력해 주세요 " 라는 메시지 (미도시)를 표시 할 것이다. 만약 그것이 존재하면, 시스템은 4 x 4 테이블 (G4 (321))을 디스플레이할 것이다. 테이블을 더 잘 설명하기 위해, 로우(row)들은 바람직하게는 A, B, C, D와 같이 상단으로부터 바닥으로 마킹된다. 컬럼(column)들은 또한 바람직하게는 1, 2, 3, 4와 같이 왼쪽에서 오른쪽으로 마킹된다. 이 테이블에 토큰들은 도 5에 설명된 규칙들에 따라 생성된다.

[0106] · 도 8b로부터 유저 ID "admin"과 연관된 패스 코드는 " ⓘ ♥ 2 ✉ "임을 알기 때문에, 유저는 테이블의 16 개의 토큰들을 검토하여 제 1 핀:" ⓘ "를 발견해야 한다. 심벌 ⓘ 은 두번째 차원에 속하기 때문에, 이 예제에서 임의의 토큰의 상단 오른쪽 위치에 나타나고, 따라서 유저는 단지 ⓘ 가 존재하는지를 조사하기 위해 각각의 토큰의 상단 오른쪽 부분을 스캔할 필요가 있다. 이 예제에서, 그것은 A1 토큰에 있다. 유저는 도 6에 설명된 규칙들에 따라 이 토큰을 클릭해야 한다.

[0107] · 유저가 A1를 클릭한 후에, 토큰은 패스 코드의 제 1 위치 (G4 (351))로 복사된다 (도 4의 단계 (440)).

[0108] · 유저 패스코드내 제 2 핀은 : "♥"이고, 이 심벌은 세번째 차원에 속한다. 이 예에서, 세번째 차원 심벌들은 임의의 토큰의 하단 좌측에 표시된다. 따라서, 유저는 단지 각각의 16 토큰들의 하단 좌측을 찾아볼 필요가 있다. 이 예에서, 그것은 토큰 D1에 있다. 따라서, 유저는 D1 (도 4의 단계 (440))을 클릭해야 한다. 그러나, 이

예에서 유저는 이 토큰을 클릭하지 않고, 대신에, B4를 클릭했다. 이것은 틀린 토큰이고, 시스템은 이를 에러로 기록할 것이고 유저 액세스를 거부할 것이다.

[0109] · 유저가 토큰 B4(도 4의 단계(440))을 클릭한 후에, 토큰은 패스코드의 제 2 위치로 복사된다(G4 (354)).

[0110] · 패스코드내 제 3 핀은 "2"이고, 그것은 첫번째 차원에 속한다. 따라서, 이 예제에서, 유저는 단지 각각의 16 토큰들의 상단 왼쪽 위치를 찾아볼 필요가 있다. 이 예에서, 그것이 존재하지 않는다. 토큰 선택 규칙들에 따라 (도 6), 유저는 해당 핀의 위치에 대하여 와일드카드 토큰 (임의의 토큰)을 선택할 수 있고 그리고 선택해야 한다. 이 예에서, 유저는 무작위적으로 토큰 C4를 클릭한다. 해당 토큰은 세번째 핀 위치 - G4 (357)로 복사된다.

[0111] · 제 4 의 마지막 핀은 " ✉ "이고, 이 심벌은 네번째 차원에 속한다. 이 예에서, 네번째 차원 심벌들은 임의의 토큰의 하단 오른쪽에 표시된다. 따라서, 유저는 단지 각각의 16 토큰들의 하단 오른쪽을 체크할 필요가 있다. 이 예에서, 그것은 토큰 A2에 있어서, 유저는 A2 (도 4의 단계 (440))을 클릭해야 한다. 유저가 A2를 클릭한 후에, 토큰은 패스 코드의 네번째 위치 (G4 (360))로 복사된다.

[0112] · 패스코드에 단지 4개의 핀들이 있기 때문에, 위치들 G4 (363) 및 G4 (366)은 공백으로 남겨지고, 그것들은 공백으로 남아있게 된다. 만약 유저가 하나 또는 둘 모두의 위치들에 토큰을 입력하면, 시스템은 유저 액세스를 거부할 것인데, 이는 유저는 원래의 4개 핀들보다 더 많은 토큰들을 입력하였기 때문이다 (도 4의 단계 (450)).

[0113] · 유저가 4개의 토큰들을 모두 입력하면 유저는 "Login"(G4 (371)) 버튼을 클릭하여 유저가 토큰 선택 프로세스를 완료했다는 것을 시스템에 알리고, 시스템은 입력된 토큰들이 도 7에 설명된 규칙들에 따라 유효한지 확인할 것이다 (도 4의 단계 (450)).

[0114] · 이 예제에서, 유저가 입력한 토큰들이 유효하지 않고, 시스템은 "로그인 실패"메시지 (G4 (381))를 표시하고 유저에게 액세스를 거부한다 (도 4의 단계 (470)).

[0115] 도 9d는 본 발명의 일 실시예에 따른 다른 실패한 로그인 프로세스에 대하여 시스템 (100)에 의해 실행되는 로그인 프로세스의 실행의 프레임의 샘플 스크린샷이다. 유저가 실패한 로그인 프로세스(도 4) 를 다음과 같이 진행할 때 GATE_4 실시예 스크린이 어떻게 보이는지에 대한 일례를 도시한다 :

[0116] · 유저는 유저 ID : "admin" (G4 (308)) (도 4의 단계 (410))를 입력한다.

[0117] · 유저는 "Enter" 버튼 (G4 (311))을 클릭한다. 시스템은 id "admin"이 이미 그것의 메모리에 있는지 (도 4의 단계 (420)를 조사하기 위해 체크할 것이고, 만약 존재하지 않으면 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력해 주세요 " 라는 메시지 (미도시)를 표시 할 것이다. 만약 그것이 존재하면, 시스템은 4 x 4 테이블 (G4 (322))을 디스플레이할 것이다. 테이블을 더 잘 설명하기 위해, 로우(row)들은 바람직하게는 A, B, C, D와 같이 상단으로부터 바닥으로 마킹된다. 컬럼(column)들은 또한 바람직하게는 1, 2, 3, 4와 같이 왼쪽에서 오른쪽으로 마킹된다. 이 테이블에 토큰들은 도 5에 설명된 규칙들에 따라 생성된다.

[0118] · 도 8b로부터 유저 ID "admin"과 연관된 패스 코드는 " ⓘ ♥ 2 ✉ "임을 알기 때문에, 유저는 테이블의 16 개의 토큰들을 검토하여 제 1 핀:" ⓘ "를 발견해야 한다. 심벌 ⓘ 은 두번째 차원에 속하기 때문에, 이 예제에서 임의의 토큰의 상단 오른쪽 위치에 나타나고, 따라서 유저는 단지 ⓘ 가 존재하는지를 조사하기 위해 각각의 토큰의 상단 오른쪽 부분을 스캔할 필요가 있다. 이 예제에서, 그것은 B2 토큰에 있다. 유저는 도 6에 설명된 규칙들에 따라 이 토큰을 클릭해야 한다.

[0119] · 유저가 B2를 클릭한 후에, 토큰은 패스 코드의 제 1 위치 (G4 (352))로 복사된다 (도 4의 단계 (440)).

[0120] · 유저 패스코드내 제 2 핀은 : "♥"이고, 이 심벌은 세번째 차원에 속한다. 이 예에서, 세번째 차원 심벌들은 임의의 토큰의 하단 좌측에 표시된다. 따라서, 유저는 단지 각각의 16 토큰들의 하단 좌측을 찾아볼 필요가 있다. 이 예에서, 그것이 존재하지 않는다. 토큰 선택 규칙들에 따라 (도 6), 유저는 해당 핀의 위치에 대하여 와일드카드 토큰 (임의의 토큰)을 선택할 수 있고 그리고 선택해야 한다. 이 예에서, 유저는 무작위적으로 토큰 A4를 클릭한다. 해당 토큰은 두번째 핀 위치 - G4 (355)로 복사된다.

[0121] · 패스코드내 제 3 핀은 "2"이고, 그것은 첫번째 차원에 속한다. 따라서, 이 예제에서, 유저는 단지 각각의 16 토큰들의 상단 왼쪽 위치를 찾아볼 필요가 있다. 이 예에서, 그것은 토큰 B3에 있다. 이 예제에서, 유저는 B3를

클릭하고 해당 토큰은 세번째 핀 위치 - G4 (358)로 복사된다.

[0122] · 제 4 의 마지막 핀은 " ✉ "이고, 이 심벌은 네번째 차원에 속한다. 이 예에서, 네번째 차원 심벌들은 임의의 토큰의 하단 오른쪽에 표시된다. 따라서, 유저는 단지 각각의 16 토큰들의 하단 오른쪽을 체크할 필요가 있다. 이 예에서, 그것은 토큰 D1에 있어서, 유저는 D1 (도 4의 단계 (440))을 클릭해야 한다. 유저가 D1를 클릭한 후에, 토큰은 패스 코드의 네번째 위치 (G4 (361))로 복사된다.

[0123] · 패스코드에 단지 4개의 핀들이 있기 때문에, 위치들 G4 (364) 및 G4 (367)은 공백으로 남겨져야 한다. 이 예에서, 유저는 잉여의 토큰 D2를 입력하였고, 따라서 시스템은 유저에게 액세스를 거부할 것인데, 이는 유저가 원래의 4개의 핀들보다 더 많이 입력했기 때문이다.

[0124] · 유저가 5개의 토큰들을 모두 입력하면 유저는 "Login"(G4 (372)) 버튼을 클릭하여 유저가 토큰 선택 프로세스를 완료했다는 것을 시스템에 알리고, 시스템은 입력된 토큰들이 도 7에 설명된 규칙들에 따라 유효한지 확인할 것이다 (도 4의 단계 (450)).

[0125] · 이 예제에서, 유저는 너무 많은 토큰들을 입력하였고, 시스템은 "로그인 실패"메시지 (G4 (382))를 표시하고 유저에게 액세스를 거부한다 (도 4의 단계 (470)).

[0126] 도 10a는 텍스트 포맷에 GATE_4 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 그것은 프로그램들이 시작할 때 비어있는 스크린을 반영한다. 이 이미지는 아래의 도면들 10b, 10c 및 10d에 대한 비교의 베이시스로서 사용된다. 이 스크린은 해당 장면에 나중에 무엇이 발생하는지에 대한 설명으로만 도시된다. 실제 유저 로그인 동안에 도시되지 않는다.

[0127] 도 10b는 텍스트 포맷에 GATE_4 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 도 9b의 유저 로그인 프로세스가 일어날 때 나중 장면에 무엇이 발생하는지를 도시하고 도 4의 프로세스 흐름이 샘플 실시예에서와 유사하게 보이는 것을 도시한다. 이것은 단지 데모이고 실제 유저 로그인 동안에 도시되지 않는다. 도 7 에 도시된 토큰 확인 규칙들을 시각적으로 예시하기 위해 사용된다.

[0128] 3 컬럼들이 있다 : "클라이언트측"(좌측), "네트워크 연결"(중간), 및 "서버측"(우측). 프로세스는 유저가 유저 ID을 입력한 때 클라이언트측에서부터 시작하고, 그런 다음 정보가 네트워크 연결을 통하여 서버측으로 전달된다. 서버는 16 토큰들을 생성하고 그것들을 네트워크로 전달하고, 그런 다음 토큰들이 클라이언트측으로 전달된다.

[0129] 유저는 도 6 에 도시된 토큰 선택 규칙들에 따라 토큰들을 선택한다. 선택된 토큰들은 네트워크로 전달되고, 그런 다음 확인하기 위해 서버측으로 전달되고 액세스 승인 또는 거부의 결과가 네트워크를 통해 클라이언트측으로 전달된다. 프로세스 플로우는 도 10b에 화살표들로 마킹된다. 보다 상세한 설명이 이하에 제공된다.

[0130] · 유저는 유저 ID : "admin" (G4 (406))를 입력한다.

[0131] · 유저는 "Enter" 버튼 (G4 (409))을 클릭한다.

[0132] · 유저 ID "admin"(G4 (406))이 클라이언트측 (G4 (411))에 표시되고 네트워크 연결 (G4 (413))로 전달되고 (G4 (421)), 그런 다음 다시 서버측(G4 (415))으로 전달된다(G4 (422)).

[0133] · 서버측에서 시스템은 그것의 메모리를 체크하여 유저 ID "admin"가 존재하는지를 확인하여, 만약 그렇지 않은 경우 시스템은 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력하십시오"(미도시) 메시지를 표시할 것이다. 도 8b에서의 동일한 예제가 사용되고, 메모리내 패스코드는 : " ⓘ ♥ 2 ✉ "이고, 시스템은 그것을 메모리 (G4 (417))에서 찾는다.

[0134] · 시스템은 도 5 에 도시된 토큰 생성 규칙들에 따라 16개의 토큰들 (G4 (423))을 생성한다.

[0135] · 16개의 토큰들이 네트워크로 전달된다(G4 (424)).

[0136] · 네트워크는 토큰들을 클라이언트측으로 전달한다(G4 (425)).

[0137] · 16개의 토큰들은 4 x 4 테이블 (도 9b에 G4 (320))로 도 9b에 도시된 바와 같이, 유저 로그인 스크린상에 디스플레이된다.

[0138] · 유저는 4개의 토큰들 : G4 (350), G4 (353), G4 (356) 및 G4 (359)을 선택한다(G4 (426)).

[0139] · 4개의 유저 선택된 토큰들은 유저가 "Login" (도 9b에 G4 (370))을 클릭한 후에 네트워크로 전달된다(G4 (427)).

[0140] · 4개의 유저 선택된 토큰들은 그런다음 서버측으로 전달된다(G4 (428)).

[0141] · 서버측에서, 시스템은 모든 4 토큰들을 하나씩 : C11, C12, C13 및 C14 체크하고, 이 예에서 그것들은 모두 정확하다.

[0142] · 상기의 성공한 로그인 (G4 (429))의 결과가 네트워크로 전달된다(G4 (430)).

[0143] · 네트워크는 클라이언트측으로 결과를 전달하고 (G4 (431)) 도 9b의 G4 (380)에 도시된 메시지를 디스플레이한다(G4 (432)).

[0144] 도 10c는 텍스트 포맷에 GATE_4 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 도 9c의 유저 로그인 프로세스가 일어날 때 나중에 장면에 무엇이 발생하는지를 도시하고 도 4의 프로세스 흐름이 샘플 실시예에서와 유사하게 보이는 것을 도시한다. 이것은 단지 데모이고 실제 유저 로그인 동안에 도시되지 않는다. 도 7 에 도시된 토큰 확인 규칙들을 시각적으로 예시하기 위해 사용된다.

[0145] 3 컬럼들이 있다 : "클라이언트측"(좌측), "네트워크 연결"(중간), 및 "서버측"(우측). 프로세스는 유저가 유저 ID을 입력한 때 클라이언트측에서부터 시작하고, 그런 다음 정보가 네트워크 연결을 통하여 서버측으로 전달된다. 서버는 16 토큰들을 생성하고 그것들을 네트워크로 전달하고, 그런 다음 토큰들이 클라이언트측으로 전달된다.

[0146] 유저는 도 6 에 도시된 토큰 선택 규칙들에 따라 토큰들을 선택한다. 선택된 토큰들은 네트워크로 전달되고, 그런 다음 확인하기 위해 서버측으로 전달되고 액세스 승인 또는 거부의 결과가 네트워크를 통해 클라이언트측으로 전달된다.프로세스 플로우는 도 10c에 화살표들로 마킹된다. 보다 상세한 설명이 이하에 제공된다.

[0147] · 유저는 유저 ID : "admin" (G4 (506))를 입력한다.

[0148] · 유저는 "Enter" 버튼 (G4 (509))을 클릭한다.

[0149] · 유저 ID "admin"(G4 (506))이 클라이언트측 (G4 (511))에 표시되고 네트워크 연결 (G4 (513))로 전달되고 (G4 (521)), 그런 다음 다시 서버측(G4 (515))으로 전달된다(G4 (522)).

[0150] · 서버측에서 시스템은 그것의 메모리를 체크하여 유저 ID "admin"가 존재하는지를 확인하여, 만약 그렇지 않은 경우 시스템은 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력하십시오"(미도시) 메시지를 표시할 것이다. 도 8b에서의 동일한 예제가 사용되고, 메모리내 패스코드는 : "ⓘ ♥ 2 ✉"이고, 시스템은 그것을 메모리 (G4 (517))에서 찾는다.

[0151] · 시스템은 도 5 에 도시된 토큰 생성 규칙들에 따라 16개의 토큰들 (G4 (523))을 생성한다.

[0152] · 16개의 토큰들이 네트워크로 전달된다(G4 (524)).

[0153] · 네트워크는 토큰들을 클라이언트측으로 전달한다(G4 (525)).

[0154] · 16개의 토큰들은 4 x 4 테이블 (도 9c에 G4 (321))로 도 9c에 도시된 바와 같이, 유저 로그인 스크린상에 디스플레이된다.

[0155] · 유저는 4개의 토큰들 : G4 (351), G4 (354), G4 (357) 및 G4 (360)을 선택한다(G4 (526)).

[0156] · 4개의 유저 선택된 토큰들은 유저가 "Login" (도 9c에 G4 (371))을 클릭한 후에 네트워크로 전달된다(G4 (527)).

[0157] · 4개의 유저 선택된 토큰들은 그런다음 서버측으로 전달된다(G4 (528)).

[0158] · 서버측에서, 시스템은 4 토큰들 전부를 하나씩: C21, C22, C23 및 C24 체크하고, 이 예에서 두번째 토큰이 부정확하다 (왜냐하면 제 2 핀 “♥”이 D1 토큰에 존재하지 않기 때문인데, 유저가 틀린 B4 토큰을 선택했다. 따라서 결과는 실패한 로그인이다).

[0159] · 상기의 실패한 로그인 (G4 (529))의 결과가 네트워크로 전달된다(G4 (530)).

[0160] · 네트워크는 클라이언트측으로 결과를 전달하고 (G4 (531)) 도 9c의 G4 (381)에 도시된 메시지를 디스플레이

한다(G4 (532)).

[0161] 도 10d는 텍스트 포맷에 GATE_4 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 도 9d의 유저 로그인 프로세스가 일어날 때 나중에 장면에 무엇이 발생하는지를 도시하고 도 4의 프로세스 흐름이 샘플 실시예에서와 유사하게 보이는 것을 도시한다. 이것은 단지 데모이고 실제 유저 로그인 동안에 도시되지 않는다. 도 7 에 도시된 토큰 확인 규칙들을 시각적으로 예시하기 위해 사용된다.

[0162] 3 컬럼들이 있다 : "클라이언트측"(좌측), "네트워크 연결"(중간), 및 "서버측"(우측). 프로세스는 유저가 유저 ID을 입력한 때 클라이언트측에서부터 시작하고, 그런 다음 정보가 네트워크 연결을 통하여 서버측으로 전달된다. 서버는 16 토큰들을 생성하고 그것들을 네트워크로 전달하고, 그런 다음 토큰들이 클라이언트측으로 전달된다.

[0163] 유저는 도 6 에 도시된 토큰 선택 규칙들에 따라 토큰들을 선택한다. 선택된 토큰들은 네트워크로 전달되고, 그런 다음 확인하기 위해 서버측으로 전달되고 액세스 승인 또는 거부의 결과가 네트워크를 통해 클라이언트측으로 전달된다. 프로세스 플로우는 도 10d에 화살표들로 마킹된다. 보다 상세한 설명이 이하에 제공된다.

[0164] · 유저는 유저 ID : "admin" (G4 (606))를 입력한다.

[0165] · 유저는 "Enter" 버튼 (G4 (609))을 클릭한다.

[0166] · 유저 ID "admin"(G4 (606))이 클라이언트측 (G4 (611))에 표시되고 네트워크 연결 (G4 (613))로 전달되고 (G4 (621)), 그런 다음 다시 서버측(G4 (615))으로 전달된다(G4 (622)).

[0167] · 서버측에서 시스템은 그것의 메모리를 체크하여 유저 ID "admin"가 존재하는지를 확인하여, 만약 그렇지 않은 경우 시스템은 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력하십시오"(미도시) 메시지를 표시할 것이다. 도 8b에서의 동일한 예제가 사용되고, 메모리내 패스코드는 : "ⓘ ♥ 2 ✉"이고, 시스템은 그것을 메모리 (G4 (617))에서 찾는다.

[0168] · 시스템은 도 5 에 도시된 토큰 생성 규칙들에 따라 16개의 토큰들 (G4 (623))을 생성한다.

[0169] · 16개의 토큰들이 네트워크로 전달된다(G4 (624)).

[0170] · 네트워크는 토큰들을 클라이언트측으로 전달한다(G4 (625)).

[0171] · 16개의 토큰들은 4 x 4 테이블 (도 9d에 G4 (322))로 도 9d에 도시된 바와 같이, 유저 로그인 스크린상에 디스플레이된다.

[0172] · 유저는 5개의 토큰들 : G4 (352), G4 (355), G4 (358), G4 (361) 및 G4 (364)을 선택한다(G4 (626)).

[0173] · 5개의 유저 선택된 토큰들은 유저가 "Login" (도 9d에 G4 (372))을 클릭한 후에 네트워크로 전달된다(G4 (627)).

[0174] · 5개의 유저 선택된 토큰들은 그런다음 서버측으로 전달된다(G4 (628)).

[0175] · 서버측에서, 시스템은 5개의 토큰들 전부를 하나씩 : C31, C32, C33, C34 및 C35 체크하여, 이 예제에서 다섯번째 토큰이 부정확하고 (왜냐하면 패스코드는 단지 4개의 핀들을 가지지만, 그러나 유저가 다섯번째 토큰을 입력했고, 그것이 틀렸다. 따라서 결과는 실패한 로그인이다).

[0176] · 상기의 실패한 로그인 (G4 (629))의 결과가 네트워크로 전달된다(G4 (630)).

[0177] · 네트워크는 클라이언트측으로 결과를 전달하고 (G4 (631)) 도 9d의 G4 (382)에 도시된 메시지를 디스플레이한다(G4 (632)).

[0178] 도 11a는 GATE_5 실시예에 유저 ID 신설(등록) 프로세스의 샘플 스크린샷이다. 그것은 프로그램들이 시작할 때 비어있는 스크린을 반영한다. 이 이미지는 아래의 도 11b에 대한 비교의 베이시스로서 사용된다.

[0179] 도 11b는 GATE_5 실시예에 유저 ID 신설(등록) 프로세스의 샘플 스크린샷이다. 심벌들의 각각의 차원이 토큰내에 위치된다는 것을 도시한다. 그것은 또한 어떻게 유저가 새로운 유저 ID를 신설하고, 어떻게 유저가 선택하고 및 패스코드와 관련된 유저 ID와 관련된 패스코드를 형성하는 핀들을 저장하는지를 반영한다. 프로세스는 아래와 같이 진행한다:

[0180] · 유저는 새로운 유저 ID : "admin" (G5 (206))를 입력하고, 그런 다음 "이용 가능성 체크" 버튼 (G5 (208))

을 클릭한다. 시스템은 id "admin"가 그것의 메모리에 이미 있는지를 보기 위해 체크하고, 만약 그렇다면, : " 유저 ID 가 이미 존재하니, 현존하는 패스코드를 덮어쓰기(overwrite)를 원하십니까 ? 라고 묻는 다이어로그(미도시)를 디스플레이 할 것이다. 만약 유저가 구 패스코드를 덮어쓰기를 원하지 않으면, 프로세스는 다이어로그를 닫고 유저가 다른 유저 ID를 기다린다. 만약 유저 ID "admin"이 존재하지 않거나 또는 만약 존재하지만 유저가 현존하는 패스코드에 덮어 쓰기를 원하면, 시스템은 G5 (212), G5 (222), G5 (232), G5 (242) 및 G5 (248)에 버튼들을 인에이블할 것이고, 이의 각각은 도 2b에 설명된 미리 정의된 차원으로부터 26개의 심벌들을 가진다.

[0181] 예를 들어 G5 (212)는 첫번째 차원의 전체 26 개 심벌들을 포함하고, 해당 심벌들은 G5 (210)에는 도시된 바와 같이 "[1]"(상단 왼쪽)위치의 토큰에 표시될 것이다. 26 심벌들은 "Ⓐ"로부터 "Ⓩ"까지이다. G5 (222)는 "α"로부터 "ϟ"까지 26개의 심벌들을 도시하고, 그것들은 두번째 차원으로부터 오고 그것들은 "[2]" (G5 (220) : 상단 오른쪽) 위치에서 임의의 토큰에 표시될 것이다. G5 (232)는 1에서 26까지 26 개의 숫자들을 표시하며, 그것들은 토큰의 "[3]"(중간에 G5 (230)) 위치에 나타나며 G5 (242)는 "○"에서 "≠"까지 26 개의 심벌들을 표시하며, 그것들은 네번째 차원으로부터 왔고 그것들은 "[4]"(G5 (240) : 왼쪽 하단) 위치에 임의의 토큰에 표시될 것이다. G5 (248)는 "+"로부터 "₪"까지 26개의 심벌들을 도시하고, 그것들은 다섯번째 차원으로부터 오고 그것들은 "[5]" (G5 (246) : 하단 오른쪽) 위치에서 임의의 토큰에 표시될 것이다.

[0182] 프로세스의 이 때에, 시스템은 G5 (212), G5 (222), G5 (232), G5 (242) 및 G5 (248)에 상기의 버튼들을 인에이블할 것이고, 유저는 그것들 중 임의의 것을 클릭할 수 있다. 비교로서, 도 11a에서 유저가 아직 임의의 유저 ID를 입력하지 않았기 때문에 해당 버튼들이 인에이블(enable)되지 않고 흐리게 보인다. 유저 ID 없이는, 유저가 임의의 핀을 선택하는 것이 허용되지 않는다.

[0183] · 유저는 제 1 핀을 선택하기 위해서 G5 (248)에 심벌들 중에 "$" 를 클릭하고, 그것은 G5 (250)에 나타난다.

[0184] · 유저는 제 2 핀을 선택하기 위해서 G5 (242)에 심벌들 중에 "=" 를 클릭하고, 그것은 G5 (252)에 나타난다.

[0185] · 유저는 제 3 핀을 선택하기 위해서 G5 (212)에 심벌들 중에 "Ⓜ" 를 클릭하고, 그것은 G5 (254)에 나타난다.

[0186] · 유저는 제 4 핀을 선택하기 위해서 G5 (212)에 심벌들 중에 "©" 를 클릭하고, 그것은 G5 (256)에 나타난다.

[0187] · 유저는 제 5 핀을 선택하기 위해서 G5 (232)에 심벌들 중에 "2" 를 클릭하고, 그것은 G5 (258)에 나타난다.

[0188] · 유저는 제 6 핀을 선택하기 위해서 G5 (242)에 심벌들 중에 "☺" 를 클릭하고, 그것은 G5 (260)에 나타난다.

[0189] · 이 예에서 유저가 패스코드내 6 개의 핀들을 갖도록 선택하여서, 그의 패스코드 길이는 6이다.

[0190] · 유저는 그런 다음 "Save" 버튼 - G5 (270)를 클릭함으로써 유저 ID 신설 (등록) 프로세스를 마친다. 시스템은 유저 ID "admin" 와 패스코드 "$=Ⓜ©2☺" 메모리에 저장할 것이다.

[0191] 도 12a는 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 그것은 프로그램들이 시작할 때 비어있는 스크린을 반영한다. 이 이미지는 아래의 도면들 12b, 12c 및 12d에 대한 비교의 베이시스로서 사용된다.

[0192] 도 12b는 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 유저가 패스코드를 선택하는 16 개의 토큰으로 이루어진 4 x 4 테이블을 도시한다. 그것은 어떻게 토큰 선택 프로세스가 진행하고 유저가 선택한 토큰들에 심벌들을 하이라이트하고 유저의 패스코드의 일부로서 그것들에 유저 핀들을 갖는지를 반영한다. 또한 성공한 로그인이 어떤 것인지를 보여준다. 이 예제 프로세스는 도 11b의 예제를 따르고 따라서 동일한 패스코드가 사용될 것이다. 프로세스는 아래와 같이 진행한다:

[0193] · 유저는 새로운 유저 ID : "admin" (G5 (306))를 입력한다.

[0194] · 유저는 "Enter" 버튼 (G5 (309))을 클릭한다. 시스템은 id "admin"이 이미 그것의 메모리에 있는지를 조사하기 위해 체크할 것이고, 만약 존재하지 않으면 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력해 주세요" 라는 메시지 (미도시)를 표시 할 것이다. 만약 그것이 존재하면, 시스템은 4 x 4 테이블 (G4 (320))을 디스플레이할 것이고, 테이블을 더 잘 설명하기 위해서, 로우들은 위에서부터 아래로: A,B,C,D로 마킹된다. 컬럼(column)들은 또한 1, 2, 3, 4로 왼쪽에서 오른쪽으로 마킹된다. 이 테이블에 토큰들은 도 5에 설명된 규칙들에 따라 생성된다.

[0195] · 도 11b로부터 유저 ID "admin"과 연관된 패스 코드는 "$=Ⓜ©2☺" 임을 알기 때문에, 유저는 테이블의 16 개의 토큰들을 검토하여 제 1 핀:"$"를 발견해야 한다.

[0196] 심벌 $은 다섯번째 차원에 속하기 때문에, 임의의 토큰의 하단 오른쪽 위치에만 나타날 것이고, 따라서 유저는 단지 $가 존재하는지를 조사하기 위해 각각의 토큰의 하단 오른쪽 부분을 스캔할 필요가 있다. 이 예에서, 그것은 토큰 B4에 있고, 스크린은 데모 모드로 취해진 것이고, 데모 모드에서 프로그램은 프로세스를 더 잘 이해하기 위해 유저에 매칭 심벌을 하일라이트한다. 실제로, 그것은 하이라이트되지 않을 것이다. 우리의 경우에, B4에 $는 하이라이트되고, 그것은 B4 토큰에서 발견되기 때문에, 유저는 도 6에 설명된 규칙들에 따라 이 토큰을 클릭해야 한다.

[0197] · 유저가 B4를 클릭한 후에, 토큰은 패스 코드의 제 1 위치 (G5 (350))로 복사된다.

[0198] · 유저 패스코드의 두 번째 핀은 "="이며, 이 심벌은 네번째 차원에 속하며 네번재 차원 심벌들은 임의의 토큰의 하단 좌측에만 나타나고, 따라서 유저는 각각의 16개의 토큰들의 하단 좌측만 볼 필요가 있고, 이 경우에 그것은 토큰 D2에 있고, 따라서 유저는 D2를 클릭해야 한다. 그것은 스크린샷에서 하이라이트된다.

[0199] · 유저가 D2를 클릭한 후에, 토큰은 패스 코드의 제 2 위치 (G5 (353))로 복사된다.

[0200] · 패스코드의 제 3 핀은 "Ⓜ"이며 그것은 첫 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 상단 왼쪽 위치만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 D4에 있고 따라서 유저는 D4를 클릭해야 한다. 그것은 스크린 샷에서 하이라이트된다.

[0201] · 유저가 D4를 클릭한 후에, 토큰은 패스 코드의 제 3 위치 (G5 (356))로 복사된다.

[0202] · 제 4 핀은 "©"이며, 이 심벌은 첫번째 차원에 속하며, 첫번째 차원 심벌들은 임의의 토큰의 상단 좌측에만 나타나고, 따라서 유저는 각각의 16 개의 토큰들의 상단 좌측만을 확인하면 되고, 이 경우에 그것은 존재하지 않고, 토큰 선택 규칙들에 따라, 유저는 해당 핀의 위치에 와일드 카드 토큰 (임의의 토큰)을 선택할 수 있고, 해야만 하고, 따라서 유저는 랜덤 토큰 A4를 클릭하고, 해당 토큰은 네번째 핀 위치(G5 (359))로 복사된다.

[0203] · 패스코드의 제 5 핀은 "2"이며 그것은 세 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 중심만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 D2에 있고 따라서 유저는 D2를 클릭해야 한다. 그것은 스크린샷에서 하이라이트된다. 유저가 클릭한 후에, 토큰은 패스 코드의 제 5 위치 (G5 (362))로 복사된다.

[0204] · 제 6 마지막 핀은 "☺"이며, 이 심벌은 네번째 차원에 속하며 네번재 차원 심벌들은 임의의 토큰의 하단 좌측에만 나타나고, 따라서 유저는 각각의 16개의 토큰들의 하단 좌측만 체크할 필요가 있고, 이 경우에 그것은 토큰 A4에 있고, 따라서 유저는 A4를 클릭해야 한다. 그것은 또한 데모 프로그램에 의해 하이라이트된다. 유저가 클릭한 후에, 토큰은 패스 코드의 제 6 마지막 위치 (G5 (365))로 복사된다.

[0205] · 유저가 6개의 토큰들을 모두 입력하면 유저는 "Login"(G5 (370)) 버튼을 클릭하여 유저가 토큰 선택 프로세스를 완료했다는 것을 시스템에 알리고, 시스템은 입력된 토큰들이 도 7에 설명된 규칙들에 따라 유효한지 확인할 것이다.

[0206] · 이 예제에서, 유저가 입력한 토큰들이 유효하고, 시스템은 "로그인 성공"메시지를 표시하고 유저 액세스를 승인한다 (G5 (380)).

[0207] · 이 예에서, G5 (353) 및 G5 (362)의 토큰들이 동일하므로 G5 (359) 및 G5 (365)의 토큰도 마찬가지이고, 우

연의 일치인데, 이런 상황이 꽤 자주 일어날 수 있다. 이것은 패스코드를 추측하려는 사람을 매우 혼란스럽게 할 수 있다.

[0208] 도 12c는 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 유저가 패스코드를 선택하는 16 개의 토큰으로 이루어진 4 x 4 테이블을 도시한다. 그것은 토큰 선택 프로세스가 어떻게 작용하는지를 반영한다. 그것은 또한 3개의 틀린 핀들을 갖는 실패한 로그인이 어떤 것인지를 도시한다. 이 예제 프로세스는 도 11b의 예제를 따르고 따라서 동일한 패스코드가 사용될 것이다. 프로세스는 아래와 같이 진행한다:

[0209] · 유저는 유저 ID : "admin" (G5 (307))를 입력한다.

[0210] · 유저는 "Enter" 버튼 (G5 (310))을 클릭한다. 시스템은 id "admin"이 이미 그것의 메모리에 있는지를 조사하기 위해 체크할 것이고, 만약 존재하지 않으면 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력해 주세요" 라는 메시지 (미도시)를 표시 할 것이다. 만약 그것이 존재하면, 시스템은 4 x 4 테이블 (G4 (321))을 디스플레이할 것이고, 테이블을 더 잘 설명하기 위해서, 로우들은 위에서부터 아래로: A,B,C,D로 마킹된다. 컬럼(column)들은 또한 1, 2, 3, 4로 왼쪽에서 오른쪽으로 마킹된다. 이 테이블에 토큰들은 도 5에 설명된 규칙들에 따라 생성된다.

[0211] · 도 11b로부터 유저 ID "admin"과 연관된 패스 코드는 " $= Ⓜ ©2 ☺" 임을 알기 때문에, 유저는 테이블의 16 개의 토큰들을 검토하여 제 1 핀:"$"를 발견해야 한다.

[0212] 심벌 $은 다섯번째 차원에 속하기 때문에, 임의의 토큰의 하단 오른쪽 위치에만 나타날 것이고, 따라서 유저는 단지 $가 존재하는지를 조사하기 위해 각각의 토큰의 하단 오른쪽 부분을 스캔할 필요가 있다. 이 예제에서, 그것은 A2 토큰에 있고, 유저는 도 6에 설명된 규칙들에 따라 이 토큰을 클릭해야 한다.

[0213] · 유저가 A2를 클릭한 후에, 토큰은 패스 코드의 제 1 위치 (G5 (351))로 복사된다.

[0214] · 유저 패스코드내 제 2 핀은 "="이며, 이 심벌은 네번째 차원에 속하며, 네번째 차원 심벌들은 임의의 토큰의 하단 좌측에만 나타나고, 따라서 유저는 각각의 16 개의 토큰들의 하단 좌측만을 확인하면 되고, 이 경우에 그것은 존재하지 않고, 토큰 선택 규칙들에 따라, 유저는 해당 핀의 위치에 와일드 카드 토큰 (임의의 토큰)을 선택할 수 있고, 해야만 하고, 따라서 유저는 랜덤 토큰 A3를 클릭한다.

[0215] · 유저가 A3를 클릭한 후에, 토큰은 패스 코드의 제 2 위치 (G5 (354))로 복사된다.

[0216] · 패스코드의 제 3 핀은 "Ⓜ"이며 그것은 첫 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 상단 왼쪽 위치만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 C1에 있고 유저는 C1를 클릭해야 한다. 이 경우에 유저는 C1를 클릭한다.

[0217] · 유저가 C1를 클릭한 후에, 토큰은 패스 코드의 제 3 위치 (G5 (357))로 복사된다.

[0218] · 제 4 핀은 "©"이며 이 심벌은 첫 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 상단 왼쪽 위치만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 D2에 있고 유저는 D2를 클릭해야 한다. 이 경우에 유저는 D2를 클릭하지 않고, 대신 유저는 C2를 클릭했다. 이것은 틀렸고, 시스템은 유저 액세스를 거부할 것이다.

[0219] · 유저가 C2를 클릭한 후에, 토큰은 패스 코드의 네번째 위치 (G5 (360))로 복사된다.

[0220] · 패스코드의 제 5 핀은 "2"이며 그것은 세 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 중심만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 C3에 있고 도 6에 토큰 선택 규칙들에 따라, 유저는 이 토큰을 선택해야하지만, 그러나 이 예제에서, 유저는 대신에 토큰 D2를 선택했고, 이것은 틀렸고, 시스템은 체크할 것이고, 통지할 것이다.

[0221] · 유저가 틀린 토큰 B2를 클릭한 후에, 그것은 패스코드의 다섯번째 위치 (G5 (363))로 복사된다.

[0222] · 제 6 마지막 핀은 "☺"이며, 이 심벌은 네번째 차원에 속하며 네번재 차원 심벌들은 임의의 토큰의 하단 좌측에만 나타나고, 따라서 유저는 각각의 16개의 토큰들의 하단 좌측만 체크할 필요가 있고, 이 경우에 그것은 토큰 D1에 있고, 따라서 유저는 D1를 클릭해야 한다. 이 경우에 유저는 D1를 클릭하지 않았고, 대신 유저는 토

큰 C4를 클릭했고, 이것은 틀렸고 시스템은 그것을 통지할 것이다.

[0223] · 유저가 틀린 토큰 C4를 클릭한 후에, 그것은 패스코드의 여섯번째 위치 (G5 (366))로 복사된다.

[0224] · 유저가 6개의 토큰들을 모두 입력하면 유저는 "Login"(G5 (371)) 버튼을 클릭하여 유저가 토큰 선택 프로세스를 완료했다는 것을 시스템에 알리고, 시스템은 입력된 토큰들이 도 7에 설명된 규칙들에 따라 유효한지 확인할 것이다.

[0225] · 이 예제에서, 유저가 입력한 토큰들은 유효하지 않고, 시스템은 "로그인 실패"메시지를 표시하고 유저 액세스를 거부한다 (G5 (381)).

[0226] 도 12d는 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 유저가 패스코드를 선택하는 16 개의 토큰으로 이루어진 4 x 4 테이블을 도시한다. 그것은 토큰 선택 프로세스가 어떻게 작용하는지를 반영한다. 그것은 또한 누락 핀들을 갖는 실패한 로그인이 어떤 것인지를 도시한다. 이 예제 프로세스는 도 11b의 예제를 따르고 따라서 동일한 패스코드가 사용될 것이다. 프로세스는 아래와 같이 진행한다:

[0227] · 유저는 유저 ID : "admin" (G5 (308))를 입력한다.

[0228] · 유저는 "Enter" 버튼 (G5 (311))을 클릭한다. 시스템은 id "admin"이 이미 그것의 메모리에 있는지를 조사하기 위해 체크할 것이고, 만약 존재하지 않으면 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력해 주세요" 라는 메시지 (미도시)를 표시 할 것이다. 만약 그것이 존재하면, 시스템은 4 x 4 테이블 (G4 (322))을 디스플레이할 것이고, 테이블을 더 잘 설명하기 위해서, 로우들은 위에서부터 아래로: A,B,C,D로 마킹된다. 컬럼(column)들은 또한 1, 2, 3, 4로 왼쪽에서 오른쪽으로 마킹된다. 이 테이블에 토큰들은 도 5에 설명된 규칙들에 따라 생성된다.

[0229] · 도 11b로부터 유저 ID "admin"과 연관된 패스 코드는 " $= Ⓜ ©2 ☺" 임을 알기 때문에, 유저는 테이블의 16 개의 토큰들을 검토하여 제 1 핀:"$"를 발견해야 한다.

[0230] 심벌 $은 다섯번째 차원에 속하기 때문에, 임의의 토큰의 하단 오른쪽 위치에만 나타날 것이고, 유저는 단지 $가 존재하는지를 조사하기 위해 각각의 토큰의 하단 오른쪽 부분을 스캔할 필요가 있다. 이 예제에서, 그것은 존재하지 않고, 유저는 해당 핀의 위치에 대하여 와일드카드 토큰 (임의의 토큰)을 선택할 수 있고 그리고 선택해야 하여서 유저는 랜덤 토큰 D3를 클릭한다.

[0231] · 유저가 D3를 클릭한 후에, 토큰은 패스 코드의 제 1 위치 (G5 (352))로 복사된다.

[0232] · 유저 패스코드의 두 번째 핀은 "="이며, 이 심벌은 네번째 차원에 속하며 네번재 차원 심벌들은 임의의 토큰의 하단 좌측에만 나타나고, 따라서 유저는 각각가의 16개의 토큰들의 하단 좌측만 볼 필요가 있고, 이 경우에 그것은 토큰 C1에 있고, 유저는 이 토큰을 클릭해야 한다.

[0233] · 유저가 C1를 클릭한 후에, 토큰은 패스 코드의 제 2 위치 (G5 (355))로 복사된다.

[0234] · 패스코드의 제 3 핀은 "Ⓜ"이며 그것은 첫 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 상단 왼쪽 위치만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 A1에 있고 따라서 유저는 A1를 클릭해야 한다. 이 경우에 유저는 A1를 클릭한다.

[0235] · 유저가 A1를 클릭한 후에, 토큰은 패스 코드의 제 3 위치 (G5 (358))로 복사된다.

[0236] · 제 4 핀은 "©"이며 이 심벌은 첫 번째 차원에 속하고, 따라서 유저는 각각의 16 개의 토큰들의 상단 왼쪽 위치만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 D3에 있고 유저는 D3를 클릭해야 한다. 이 경우에 유저는 D3를 클릭했다.

[0237] · 유저가 D3를 클릭한 후에, 토큰은 패스 코드의 네번째 위치 (G5 (361))로 복사된다.

[0238] · 패스코드의 제 5 핀은 "2"이며 그것은 세 번째 차원에 속하고, 유저는 각각의 16 개의 토큰들의 중심만을 살펴볼 필요가 있고, 이 경우에 그것은 토큰 C4에 있고 도 6에 토큰 선택 규칙들에 따라, 이 예제에서 유저는 토큰 C4를 선택했다.

[0239] · 유저가 토큰 C4를 클릭한 후에, 그것은 패스코드의 다섯번째 위치 (G5 (364))로 복사된다.

[0240] · 제 6 마지막 핀은 "☺"이며, 이 심벌은 네번째 차원에 속하며 네번재 차원 심벌들은 임의의 토큰의 하단 좌측에만 나타나고, 따라서 유저는 각각의 16개의 토큰들의 하단 좌측만 체크할 필요가 있고, 이 경우에 그것은 토큰 C2에 있고, 따라서 유저는 C2를 클릭해야 한다. 그러나 이 경우에 유저는 C2를 클릭하지 않았고, 대신 유저는 마지막 위치를 공백으로 남겨두었고, 단지 5개의 핀들을 입력했다. 이것은 틀렸다.

[0241] · 유저가 상기의 5개의 토큰들을 입력한 후에 유저는 "Login"(G5 (372)) 버튼을 클릭하여 유저가 토큰 선택 프로세스를 완료했다는 것을 시스템에 알리고, 시스템은 입력된 토큰들이 도 7에 설명된 규칙들에 따라 유효한지 확인할 것이다.

[0242] · 이 예에서, 유저가 입력한 토큰들은 유효하지 않은데, 왜냐하면 원래의 패스코드는 6 핀들을 갖기 때문인데, 하지만 이 예제에서 유저는 단지 5개의 토큰들을 입력했고, 따라서 액세스를 위한 유저 요청은 거부되었고 시스템은 "로그인 실패" 메시지를 디스플레이했다(G5 (382)).

[0243] 도 13a는 텍스트 포맷에 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 그것은 프로그램들이 시작할 때 비어있는 스크린을 반영한다. 이 이미지는 아래의 도면들 13b, 13c 및 13d에 대한 비교의 베이시스로서 사용된다. 이 스크린은 해당 장면에 나중에 발생하는 것에 대한 설명으로만 도시된다. 실제 유저 로그인 동안에 도시되지 않는다.

[0244] 도 13b는 텍스트 포맷에 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 도 12b의 유저 로그인 프로세스가 일어날 때 나중 장면에 무엇이 발생하는지를 도시하고 도 4의 프로세스 흐름이 샘플 실시예에서와 유사하게 보이는 것을 도시한다. 이것은 단지 데모이고 실제 유저 로그인 동안에 도시되지 않는다. 도 7 에 도시된 토큰 확인 규칙들을 시각적으로 예시하기 위해 사용된다.

[0245] 3 컬럼들이 있다 : "클라이언트측"(좌측), "네트워크 연결"(중간), 및 "서버측"(우측). 프로세스는 유저가 유저 ID을 입력한 때 클라이언트측에서부터 시작하고, 그런 다음 정보가 네트워크 연결을 통하여 서버측으로 전달된다. 서버는 16개의토큰들을 생성하고 그것들을 네트워크로 전달하고, 그런 다음 토큰들이 클라이언트측으로 전달된다.

[0246] 유저는 도 6 에 도시된 토큰 선택 규칙들에 따라 토큰들을 선택하고, 그런 다음 선택된 토큰들은 네트워크로 전달되고 그런 다음 확인하기 위해서 서버측으로 전달된다. 액세스 승인 또는 거부의 결과가 네트워크를 통해 클라이언트측으로 전달된다. 프로세스 플로우는 도 13b에 화살표들로 마킹된다. 프로세스는 아래와 같이 진행한다:

[0247] · 유저는 유저 ID "admin" (G5 (406))를 입력한다.

[0248] · 유저는 "Enter" (G5 (409))을 클릭한다.

[0249] · 유저 ID "admin"(G5 (406))이 클라이언트측 (G5 (411))에 표시되고 네트워크 연결 (G5 (413))로 전달되고 (G5 (421), 그런 다음 다시 서버측(G5 (415))으로 전달된다(G5 (422)).

[0250] · 서버측에서 시스템은 그것의 메모리를 체크하여 유저 ID "admin"가 존재하는지를 확인하여, 만약 그렇지 않은 경우 시스템은 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력하십시오"(미도시) 메시지를 표시할 것이다. 도 11b에서의 동일한 예제가 사용되고, 따라서 메모리내 패스코드는 : " $= Ⓜ ©2 ☺" 이고, 시스템은 그것을 메모리 (G5 (417))에서 찾는다.

[0251] · 시스템은 도 5 에 따라 16개의 토큰들 (G5 (423))을 생성한다.

[0252] · 16개의 토큰들이 네트워크로 전달된다(G5 (424)).

[0253] · 네트워크는 토큰들을 클라이언트측으로 전달한다(G5 (425)).

[0254] · 16개의 토큰들은 4 x 4 테이블 (G5 (320))로 도 12b에 도시된 바와 같이, 유저 로그인 스크린상에 디스플레이된다.

[0255] · 유저는 6개의 토큰들 : G5 (350), G5 (353), G5 (356), G5 (359), G5 (362) 및 G5 (365)을 선택한다(G5 (426)).

[0256] · 6개의 유저 선택된 토큰들은 유저가 도 12b에 "Login" (G5 (370))을 클릭한 후에 네트워크로 전달된다(G5 (427)).

[0257] · 6개의 유저 선택된 토큰들은 그런다음 서버측으로 전달된다(G5 (428)).

[0258] · 서버측에서, 시스템은 모든 6개의 토큰들을 하나씩 : K11, K12, K13, K14, K15 및 K16 체크하고, 이 예에서 그것들은 모두 정확하다.

[0259] · 상기의 성공한 로그인 (G5 (429))의 결과가 네트워크로 전달된다(G5 (430)).

[0260] · 네트워크는 클라이언트측으로 결과를 전달하고 (G5 (431)) 도 12b의 G5 (380)에 도시된 메시지를 디스플레이한다(G5 (432)).

[0261] 도 13c는 텍스트 포맷에 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 도 12c의 유저 로그인 프로세스가 일어날 때 나중 장면에 무엇이 발생하는지를 도시하고 도 4의 프로세스 흐름이 샘플 실시예에서와 유사하게 보이는 것을 도시한다. 이것은 단지 데모이고 실제 유저 로그인 동안에 도시되지 않는다. 도 7 에 도시된 토큰 확인 규칙들을 시각적으로 예시하기 위해 사용된다.

[0262] 3 컬럼들이 있다 : "클라이언트측"(좌측), "네트워크 연결"(중간), 및 "서버측"(우측). 프로세스는 유저가 유저 ID을 입력한 때 클라이언트측에서부터 시작하고, 그런 다음 정보가 네트워크 연결을 통하여 서버측으로 전달된다. 서버는 16개의토큰들을 생성하고 그것들을 네트워크로 전달하고, 그런 다음 토큰들이 클라이언트측으로 전달된다.

[0263] 유저는 도 6 에 도시된 토큰 선택 규칙들에 따라 토큰들을 선택한다. 선택된 토큰들은 네트워크로 전달되고, 그런 다음 확인하기 위해 서버측으로 전달되고 액세스 승인 또는 거부의 결과가 네트워크를 통해 클라이언트측으로 전달된다. 프로세스 플로우는 도 13c에 화살표들로 마킹된다. 보다 상세한 설명이 이하에 제공된다.

[0264] · 유저는 유저 ID : "admin" (G5 (506))를 입력한다.

[0265] · 유저는 "Enter" 버튼 (G5 (509))을 클릭한다.

[0266] · 유저 ID "admin"(G5 (506))이 클라이언트측 (G5 (511))에 표시되고 네트워크 연결 (G5 (513))로 전달되고 (G5 (521), 그런 다음 다시 서버측(G5 (515))으로 전달된다(G5 (522)).

[0267] · 서버측에서 시스템은 그것의 메모리를 체크하여 유저 ID "admin"가 존재하는지를 확인하여, 만약 그렇지 않은 경우 시스템은 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력하십시오"(미도시) 메시지를 표시할 것이다. 도 11b에서의 동일한 예제가 사용되고, 메모리내 패스코드는 : " $= Ⓜ ©2 ☺" 이고, 시스템은 그것을 메모리 (G5 (517))에서 찾는다.

[0268] · 시스템은 도 5 에 도시된 토큰 생성 규칙들에 따라 16개의 토큰들 (G5 (523))을 생성한다.

[0269] · 16개의 토큰들이 네트워크로 전달된다(G5 (524)).

[0270] · 네트워크는 토큰들을 클라이언트측으로 전달한다(G5 (525)).

[0271] · 16개의 토큰들은 4 x 4 테이블 (도 12c에 G5 (321))로 도 12c에 도시된 바와 같이, 유저 로그인 스크린상에 디스플레이된다.

[0272] · 유저는 6개의 토큰들 : G5 (351), G5 (354), G5 (357), G5 (360), G5 (363) 및 G5 (366)을 선택한다(G5 (526)).

[0273] · 6개의 유저 선택된 토큰들은 유저가 "Login" (도 12c에 G5 (371))을 클릭한 후에 네트워크로 전달된다(G5 (527)).

[0274] · 6개의 유저 선택된 토큰들은 그런다음 서버측으로 전달된다(G5 (528)).

[0275] · 서버측에서, 시스템은 모든 6개의 토큰들을 하나씩 : K21, K22, K23, K24, K25 및 K26체크한다. 이 예제에서 마지막 3 개의 선택된 토큰들이 부정확하다 (유저는 개별적으로 네번째, 다섯번째 및 여섯번째 토큰들에 대하여 D2 토큰, C3 토큰 및 D1 토큰을 선택해야 할 필요가 있지만, 대신 C2 토큰, B2 토큰 및 C4 토큰을 선택했고, 이는 틀렸다. 따라서 결과는 실패한 로그인이다).

[0276] · 상기의 실패한 로그인 (G5 (529))의 결과가 네트워크로 전달된다(G5 (530)).

[0277] · 네트워크는 클라이언트측으로 결과를 전달하고 (G5 (531)) 도 12c의 G5 (381)에 도시된 메시지를 디스플레이한다(G5 (532)).

[0278] 도 13d는 텍스트 포맷에 GATE_5 실시예에 유저 로그인 프로세스의 샘플 스크린샷이다. 도 12c의 유저 로그인 프로세스가 일어날 때 나중 장면에 무엇이 발생하는지를 도시하고 도 4의 프로세스 흐름이 샘플 실시예에서와 유사하게 보이는 것을 도시한다. 이것은 단지 데모이고 실제 유저 로그인 동안에 도시되지 않는다. 도 7 에 도시된 토큰 확인 규칙들을 시각적으로 예시하기 위해 사용된다.

[0279] 3 컬럼들이 있다 : "클라이언트측"(좌측), "네트워크 연결"(중간), 및 "서버측"(우측). 프로세스는 유저가 유저 ID을 입력한 때 클라이언트측에서부터 시작하고, 그런 다음 정보가 네트워크 연결을 통하여 서버측으로 전달된다. 서버는 16개의토큰들을 생성하고 그것들을 네트워크로 전달하고, 그런 다음 토큰들이 클라이언트측으로 전달된다.

[0280] 유저는 도 6 에 도시된 토큰 선택 규칙들에 따라 토큰들을 선택하고, 그런 다음 선택된 토큰들은 네트워크로 전달되고 그런 다음 확인하기 위해서 서버측으로 전달된다. 액세스 승인 또는 거부의 결과가 네트워크를 통해 클라이언트측으로 전달된다. 프로세스 플로우는 도 13d에 화살표들로 마킹된다. 프로세스는 아래와 같이 진행한다:

[0281] · 유저는 유저 ID "admin" (G5 (606))를 입력한다.

[0282] · 유저는 "Enter" (G5 (609))을 클릭한다.

[0283] · 유저 ID "admin"(G5 (606))이 클라이언트측 (G5 (611))에 표시되고 네트워크 연결 (G5 (613))로 전달되고 (G5 (621), 그런 다음 다시 서버측(G5 (615))으로 전달된다(G5 (622)).

[0284] · 서버측에서 시스템은 그것의 메모리를 체크하여 유저 ID "admin"가 존재하는지를 확인하여, 만약 그렇지 않은 경우 시스템은 "유저 ID가 존재하지 않습니다, 유효한 유저 ID를 입력하십시오"(미도시) 메시지를 표시할 것이다. 도 11b에서의 동일한 예제가 사용되고, 따라서, 메모리내 패스코드는 : " $= Ⓜ ©2 ☺" 이고, 시스템은 그것을 메모리 (G5 (617))에서 찾는다.

[0285] · 시스템은 도 5 의 토큰 생성 규칙들에 따라 16개의 토큰들 (G5 (623))을 생성한다.

[0286] · 16개의 토큰들이 네트워크로 전달된다(G5 (624)).

[0287] · 네트워크는 토큰들을 클라이언트측으로 전달한다(G5 (625)).

[0288] · 16개의 토큰들은 4 x 4 테이블 (G5 (322))로 도 12d에 도시된 바와 같이, 유저 로그인 스크린상에 디스플레이된다.

[0289] · 유저는 5 개의 토큰 G5 (352), G5 (355), G5 (358), G5 (361) 및 G5 (364)를 선택한다 (G5 (626)). G5 (367)를 통지하고, 그것은 "[☺] 입력 누락"이라고 표시되고 그것은 시스템이 "☺ "심벌을 마지막 핀으로 기대하고 있고 따라서 여섯 번째 토큰이 있어야하지만 여섯번째 토큰의 입력이 누락되었다는 것을 의미하고, 이것은 에러이고, 시스템은 유저 액세스를 거부한다.

[0290] · 5개의 유저 선택된 토큰들은 유저가 도 12d에 "Login" (G5 (372))을 클릭한 후에 네트워크로 전달된다(G5 (627)).

[0291] · 5개의 유저 선택된 토큰들은 그런다음 서버측으로 전달된다(G5 (628)).

[0292] · 서버측에서, 시스템은 5개의 토큰들 모두 하나씩 : K31, K32, K33, K34 및 K35 체크하여, 이 예에서 5 개의 토큰이 모두 정확하지만 여섯 번째 토큰이 누락되었고 (유저가 C2 를 클릭하지 않았고, 대신 유저가 마지막 위치를 비워두고 5 개의 핀들만 입력했다), 따라서 K36은 에러를 나타내는 [x] 마크가 표시된다. 이것은 틀렸다. 따라서 결과는 실패한 로그인이다.

[0293] · 상기의 실패한 로그인 (G5 (629))의 결과가 네트워크로 전달된다(G5 (630)).

[0294] · 네트워크는 클라이언트측으로 결과를 전달하고 (G5 (631)) 도 12d의 G5 (382)에 도시된 메시지를 디스플레이

한다(G5 (632)).

[0295] 이 방법은 패스코드를 훨씬 더 쉽게 추측 할 수 있도록 하기 위해 다음 특징부를 사용하도록 확장 될 수 있다 : 특정 수의 핀을 "은닉(hidden)"으로 지정하여, 해당 은닉된 핀들이 선택 테이블에 나타나면, 그것들은 선택되지 않는다. 대신에, 유저는 이들 핀들을 갖지 않는 임의의 다른 토큰을 선택하고, 해당 핀들을 갖는 토큰들을 피해야 한다.

[0296] 따라서, 예를 들어, 만약 유저가 패스코드 “123($#)456” 을 가지면, 패스코드의 길이는 8이고, 중간에 2개의 핀들은 "(" 와 ")" 사이에 도시된 핀들은 은닉된다. 핀들 1, 2, 3, 4, 5, 6에 대하여 상기의 규칙들을 따른다. "$"#"의 경우, "은닉된 핀 규칙(hidden-pin rules)"을 따르고 : 이는 : 만약 임의의 16 개의 토큰들 중 어느 것도 표시되지 않는 경우, 유저는 그것들의 해당 위치에 와일드 카드 토큰을 선택할 수 있고, 선택해야 하지만, 그러나 만약에 그것들중 임의의 것이 선택 테이블에 16 개의 토큰들 중 하나에 나타나면, 유저는 해당 핀을 피하고 임의의 다른 15 개 토큰들 중 하나를 선택해야 한다.

[0297] 핀을 은닉된 핀으로 만들기 위해, 패스코드 신설 (등록) 스크린의 각각의 핀 아래에있는 체크란이 디스플레이될 수 있고, 유저가 핀 아래의 박스를 체크하면, 해당 핀은 은닉된 핀이되고, 토큰 확인 프로세스 동안에, 상기의 은닉된 핀 규칙을 사용하여 유저 로그인을 확인한다.

[0298] 본 발명은 임의의 통신 프로세스에 또한 사용될 수 있어서, 메시지와 함께 일부 토큰을 갖는 키 및 16 개의 토큰들 [도 5에 설명된 토큰 생성 규칙들에 따라 발신자 패스코드로 생성]을 갖는 선택 테이블을 부착한다. 만약 해당 키가 해당 테이블에 대응하여 유효하면, 그러면 해당 메시지는 참된 메시지(true message)이다. 만약 키가 선택 테이블에 대응하여 유효하지 않으면, 그러면 부착된 메시지는 거짓 메시지(false message)이다. 참된 메시지를 유지시키고 거짓 메시지를 드랍(drop) 시킴으로써, 최종 [원래의] 정확한 메시지를 획득할 것이다. 수신자는 메시지를 복호화하기 위해 동일한 패스코드를 사용하고, 따라서 프로세스는 이하의 예들과 같이 진행할 수 있다:

[0299] · 메시지_1 : 나는 7 pm에 집에 갈 것이다 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효하지 않은 키]

[0300] --> 메시지를 폐기한다

[0301] 메시지_2 : 나는 3 pm에 집에 갈 것이다 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> 나는 3 pm에 집에 갈 것이다

[0302] 메시지_3 : 우리는 3일 공격을 포기할 것이다 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효하지 않은 키] --> 메시지를 폐기한다

[0303] 메시지_4 : 우리는 3일 정오에 공격할 것이다 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> 우리는 3일 정오에 공격할 것이다

[0304] 따라서, 올바른 최종 메시지는 : ‘나는 3 pm에 집에 갈 것이다’. ‘우리는 3일 정오에 공격할 것이다’ 이다.

[0305] · 메시지_1 : 나는 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> 나는

[0306] 메시지_2 : - 것이다 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> - 것이다

[0307] 메시지_3 : 하지 않을 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효하지 않은 키] --> 메시지를 폐기한다

[0308] 메시지_4 : -간다 [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> 간다

[0309] 따라서, 올바른 최종 메시지는 : ‘나는 갈 것이다’ 이다.

[0310] · 메시지_1 : u [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효하지 않은 키] --> 메시지를 폐기한다

[0311] 메시지_2 : n [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효하지 않은 키] --> 메시지를 폐기한다

[0312] 메시지_3 : t [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> t

[0313] 메시지_4 : r [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> r

[0314] 메시지_5 : u [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> u

[0315] 메시지_6 : e [+ 16개의 토큰들을 갖는 토큰 테이블 + 유효한 키] ==> e

[0316] 따라서, 올바른 최종 메시지는 : ‘ true’ 이다.

[0317] 또한, 본 발명은 전자 통신에서 신뢰할 수 있는 엔티티로 위장함으로써 종종 악의적 인 이유로 유저 이름, 패스워드 및 신용 카드 세부사항들과 같은 민감한 정보를 획득하려고 시도하는 피싱 (phishing)을 무력화 할 수 있다. 본 발명에서, 유저와 서버 사이에서 전달되는 모든 정보는 토큰들의 형태이고, 각각의 토큰은 다수의 심벌들을 가지므로, 어떠한 명확한 유저 핀도 제공되지 않는다.

[0318] 도 14는 GATE_4 실시예를 사용하는 메시지 암호화 프로세스의 샘플 스크린샷이다. 어떻게 평문 텍스트 메시지가 발신자 패스코드로 암호화되고 암호화된 메시지가 어떤 것인지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0319] · 메시지 발신자는 G4 (700)에서 평문 텍스트 메시지 "secret" 를 입력한다.

[0320] · 발신자는 G4 (702)에서 패스코드 "123"를 입력하고 "Encrypt" 버튼[G4 (703)]을 클릭한다.

[0321] · 원래의 메시지 "secret"는 일부 랜덤 필러 문자(filler character)들과 혼합되고 G4 (704)에 도시된 바와 같이 아래의 결과 메시지 "sLeQWNcrMfYeMtHQr" 가 된다.

[0322] · 수신자는 메시지를 복호화하기 위해 수신자 측과 동일한 패스코드 "123" [G4 (706)]를 사용할 것이다.

[0323] 도 15a는 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 어떻게 암호화된 원래의 메시지가 수신자 패스코드로 성공적으로 복호화될 수 있는지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0324] · 결과 메시지 [G4 (704)]내 문자는 4 x 4 토큰 테이블과 부착되고, 테이블내 각각의 토큰은 4 심벌들을 가지며, 메시지내 각각의 문자는 또한 "key"에 부착되고, 키는 그 안에 일부 토큰들을 가지며, 키안에 토큰들의 수는 2에서부터 6까지의 범위에 이를 수 있다.

[0325] · 수신자는 메시지를 복호화하기 위해 동일한 패스코드 "123" [G4 (706)]를 사용하고, 복호화된 메시지는 하이라이트된 문자들로서 G4 (704)에 도시된다. 결과 복호화된 메시지는 "secret" [G4 (708)]이다.

[0326] · 도 15a는 각각의 문자에 대하여 그것이 어떤 것인지의 일 예를 도시한다. 스크린샷에서, 메시지내 제 1 문자 "s" [G4 (710)]가 디스플레이되고 일 예로서 유저가 [G4 (710)]을 클릭한 후에, G4 (712)는 현재 디스플레이하는 문자가 "s"인 것을 보여준다. 이 문자에 부착된 4 x 4 토큰 테이블이 테이블 G4 (714)에 도시된다.

[0327] · "s" 에 부착된 키 토큰들이 G4 (720), G4 (722) 및 G4 (724)로서 또한 도시된다.

[0328] · 메시지내 필러 문자들: L, Q, W, N, M, f, Y, M, H, Q 및 r은 유효하지 않는 토큰 테이블들 및 키들과 의도적으로 부착되고, 따라서 그것들은 수신자 측에서 유효하지 않을 것이다.

[0329] · 도시된 예제에서, 유저는 G4 (704)에 각각의 문자를 클릭할 수 있고 그것의 컨텐츠 및 키 토큰들이 보여지고, 그런 다음 문자가 유효한지를 보기 위해서 "Check" 버튼 [G4 (726)]을 클릭한다. 스크린샷에서, 문자 "s"는 유효하고 체크는 성공하였다 [G4 (730)]는 것을 도시한다.

[0330] 도 15b는 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 언제 도 15a에 도시된 프로세스가 일어나고 어떻게 메시지가 수신자 측에서 확인되는지의 장면들이 나중에 어떻게 일어나는지를 도시한다. 프로세스는 아래와 같이 진행한다:

[0331] · 메시지 문자 "s" [G4 (750)]가 발신자 패스코드 "123" [G4 (752)]로 암호화되고 16 토큰들 [G4 (754)] 및 일부 토큰들 [G4 (720), G4 (722) 및 G4 (724)]을 갖는 키와 부착된다. 이 정보는 네트워크 [G4 (756)]로, 그런 다음 수신자 [G4 (758)]로 발송된다.

[0332] · 수신자 측에서, 메시지를 디코딩하기 위해 동일한 패스코드 "123" [G4 (760)]가 사용된다. 키 토큰들은 확인 프로세스 G4 (764), G4 (766)를 거쳐[G4 (762)], 각각의 키 토큰을 체크한다. C51, C52 및 C53로부터, 그것들이 모든 유효하다는 것을 알 수 있고, 따라서 최종 결론은 도 15a에 도시된 바와 같이 [G4 (730)] 메시지가 유효하다[G4 (770)]는 것에 도달한다 [G4 (768)].

[0333] 도 16a는 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 어떻게 암호화된 필러 메시지가 수신자 패스코드로 복호화될 수 있고 유효하지 않은 정보로 인식될 수 있는지의 일 예를 도시한다. 프로세

스는 아래와 같이 진행한다:

[0334] · 도 16a는 원래의 메시지의 일부가 아닌 각각의 필러 문자에 대하여 그것이 어떤 것인지의 일 예를 도시한다. 유저가 그것을 클릭한 후에 메시지 [G4 (704)]내 제 2 문자 "L" [G4 (711)]의 컨텐츠를 보여준다. G4 (713)는 문자가 "L"이다는 것을 보여준다. 이 문자에 부착된 16개의 토큰들의 4 x 4 테이블이 G4 (715)에 도시된다. "L"에 부착된 4 개의 키 토큰들은 G4 (721), G4 (723), G4 (725) 및 G4 (727)로 도시된다.

[0335] · 문자 "L" 은 필러 문자이고 원래의 메시지의 일부가 아니기 때문에, 발신자는 확인되지 않을 키 및 4 x 4 토큰 테이블과 그것을 의도적으로 부착하였다. 발신자 패스코드 [G4 (702)] 및 수신자 패스코드 [G4 (706)]가 동일하고 3개의 핀들 "1", "2" 및 "3"을 가지고 있는 것을 명확히 알 수 있고, 따라서 유효한 키는 3 개의 토큰들보다 많지도 적지도 않아야 한다. 이 예에서, 그것은 4개의 키 토큰들을 가지며, 따라서 그것을 유효하지 않고 문자 "L"은 무시되어야 하고 최종 복호화된 메시지의 부분이 아닐 것이다.

[0336] · 스크린샷에서, 유저가 "Check" 버튼 G4 (726)를 클릭한 때, 그것은 확인 프로세스가 실패하였다는 것을 보여준다 [G4 (731)].

[0337] 도 16b는 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 언제 도 16a에 도시된 프로세스가 일어나고 어떻게 필러 메시지가 수신자 측에서 유효하지 않다는 장면들이 나중에 어떻게 일어나는지를 도시한다. 프로세스는 아래와 같이 진행한다:

[0338] · 메시지 문자 "L" [G4 (751)]가 발신자 패스코드 "123" [G4 (752)]로 암호화되고 16 개의 토큰들 [G4 (755)] 및 일부 토큰들 [G4 (721), G4 (723), G4 (725) 및 G4 (727)]을 갖는 키와 부착된다. 상기의 정보는 네트워크 [G4 (757)]로, 그런 다음 수신자 [G4 (759)]로 발송된다.

[0339] · 수신자 측에서, 메시지를 디코딩하기 위해 동일한 패스코드 "123" [G4 (760)]가 사용된다. 키 토큰들은 확인 프로세스 G4 (7645), G4 (767)를 거쳐[G4 (763)], C61, C62, C63 및 C64로부터 각각의 키 토큰을 체큰한다. 마지막 2개의 키 토큰들 유효하지 않다는 것을 알 수 있다.

[0340] · 세번째 패스코드 "3" 은 세번째 토큰 [G4 (790)]에 나타나고, 그것이 선택되어야 한다. 그러나, 여덟번째 토큰 [G4 (792)]이 선택되었고 세번째 키 토큰 G4 (725)에 나타났다. 이것은 부정확하다.

[0341] · 발신자 패스코드 가 수신자 패스코드와 같고 3개의 핀들을 갖지만, 그러나 부착된 키가 4 개의 토큰들을 가진다. 최종 토큰 [G4 (727)] 또한 유효하지 않다.

[0342] · 최종 결론은 도 16a에 도시된 바와 같이 [G4 (731)] 메시지가 유효하지 않다[G4 (771)]는 것에 도달한다 [G4 (769)].

[0343] 도 17는 GATE_4 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 어떻게 암호화된 원래의 메시지가 발신자 패스코드와 다른 수신자 패스코드에 의해 성공적으로 복호화되지 않을 수 있는지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0344] · 유저는 G4 (700)에 평문 텍스트 메시지 "secret"를 타이핑하고, 그런 다음 패스코드 "123" [G4 (702)]를 입력하고 "Encrypt" 버튼 [G4 (703)]을 클릭한다.

[0345] · 메시지가 암호화되고, 발송되고 수신되어 : "sLeQWNcrMfYeMtHQr" [G4 (705)]로서 G4 (705)에 나타났다.

[0346] · 수신자는 패스코드 "123" [G4 (702)]로 암호화된 발신자로부터 수신된 메시지를 복호화하기 위해 패스코드 "567" [G4 (707)]를 사용한다.

[0347] · 복호화된 메시지가 G4 (705):"ecrQ"에 하이라이트된다.

[0348] · 결과 메시지는 "ecrQ" [G4 (709)]로서 도시된다.

[0349] · 결과 메시지는 발신자로부터의 원래의 메시지: "secret" [G4 (700)]과 다른데, 이는 수신자가 메시지를 복호화하기 위해 상이한 패스코드를 사용했기 때문이다.

[0350] 도 18은 GATE_5 실시예를 사용하는 메시지 암호화 프로세스의 샘플 스크린샷이다. 어떻게 평문 텍스트 메시지가 발신자 패스코드로 암호화되고 암호화된 메시지가 어떤 것인지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0351] · 메시지 발신자는 G5 (700)에서 평문 텍스트 메시지 "FYEO" 를 입력한다.

[0352] · 발신자는 G5 (702)에서 패스코드 "123"를 입력하고 "Encrypt" 버튼[G5 (703)]을 클릭한다.

[0353] · 원래의 메시지 "FYEO"는 일부 랜덤 필러 문자들과 혼합되고 아래의 결과 메시지 "FlPRojcYnEbAO" [G5 (704)]가 된다.

[0354] · 수신자는 메시지를 복호화하기 위해 수신자 측과 동일한 패스코드 "123" [G5 (706)]를 사용할 것이다.

[0355] 도 19a는 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 어떻게 암호화된 원래의 메시지가 수신자 패스코드로 성공적으로 복호화될 수 있는지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0356] · 수신된 메시지 [G5 (704)]내 각각의 문자는 4 x 4 토큰 테이블과 부착되고, 테이블내 각각의 토큰은 5개의 심벌들을 가지며, 메시지내 각각의 문자는 또한 "key"에 부착되고, 키는 그 안에 일부 토큰들을 가지며, 키안에 토큰들의 수는 2에서부터 6까지의 범위에 이를 수 있다.

[0357] · 수신자는 메시지를 디코딩하기 위해 동일한 패스코드 "123" [G5 (706)]를 사용하고, 복호화된 메시지는 하이라이트된 문자들로서 G5 (704)에 도시된다. 결과 복호화된 메시지는 "FYEO" [G5 (708)]이다.

[0358] · 도 19a는 각각의 문자에 대하여 그것이 어떤 것인지의 일 예를 도시한다. 스크린샷에서, 메시지[G5 (710)]내 제 1 문자 "F" 가 일 예로서 디스플레이된다. G5 (712)는 현재 디스플레이 문자가 "F"인 것을 보여주고,이 문자에 부착된 4 x 4 토큰 테이블이 테이블 G5 (714)에 도시된다.

[0359] · "F" 에 부착된 키 토큰들이 G5 (720), G5 (722) 및 G5 (724)로서 또한 도시된다.

[0360] · 메시지내 필러 문자들: l, P, R, o, j, c, n, b 및 A는 유효하지 않는 토큰 테이블들 및 키들과 의도적으로 부착되고, 따라서 그것들은 수신자 측에서 유효하지 않을 것이다.

[0361] · 이 예제에서, 유저는 G5 (704)에 각각의 문자를 클릭할 수 있고 그것의 컨텐츠 및 키 토큰들이 보여지고, 그런 다음 문자가 유효한지를 보기 위해서 "Check" 버튼 [G5 (726)]을 클릭한다. 스크린샷에서, 문자 "F"는 유효하고 체크는 성공하였다 [G5 (730)]는 것을 도시한다.

[0362] 도 19b는 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 언제 도 19a에 도시된 프로세스가 일어나고 어떻게 메시지가 수신자 측에서 확인되는지의 장면들이 나중에 어떻게 일어나는지를 도시한다. 프로세스는 아래와 같이 진행한다:

[0363] · 메시지 문자 "F" [G5 (750)]가 발신자 패스코드 "123" [G5 (752)]로 암호화되고 16 개의 토큰들 [G5 (754)]의 4 x 4 테이블 및 일부 토큰들 [G5 (720), G5 (722) 및 G5 (724)]을 갖는 키와 부착된다. 이 정보는 네트워크 [G5 (756)]로, 그런 다음 수신자 [G5 (758)]로 발송된다.

[0364] · 수신자 측에서, 메시지를 복호화하기 위해 동일한 패스코드 "123" [G5 (760)]가 사용한다. 키 토큰들은 확인 프로세스 G5 (764), G5 (766)를 거쳐[G5 (762)], 각각의 키 토큰을 체크한다. K51, K52 및 K53로부터, 그것들이 모든 유효하다는 것을 알 수 있고, 따라서 최종 결론은 도 19a에 도시된 바와 같이 [G5 (730)] 메시지가 유효하다[G5 (770)]는 것에 도달한다 [G5 (768)].

[0365] 도 20a는 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 어떻게 암호화된 필러 메시지가 수신자 패스코드로 복호화될 수 있고 유효하지 않은 정보로 인식될 수 있는지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0366] · 도 20a는 원래의 메시지의 일부가 아닌 각각의 필러 문자에 대하여 그것이 어떤 것인지의 일 예를 도시한다. 메시지 [G5 (704)]내 제 3 문자 "P" [G5 (711)]의 컨텐츠를 보여준다. G5(713)은 문자가 "P"인 것을 보여주고, 이 문자에 부착된 16개의 토큰들의 4 x 4 테이블이 G5 (715)에 도시된다. "P"에 부착된 3개의 키 토큰들은 G5 (721), G5 (723) 및 G5 (725)로 도시된다.

[0367] · 문자 "P"는 필러 문자이고 원래의 메시지의 일부가 아니고, 발신자는 유효하지 않은 4 x 4 토큰 테이블 및 키와 그것을 의도적으로 부착하였기 때문에, 따라서 발신자 패스코드 [G5 (702)] 및 수신자 패스코드 [G5 (706)] 가 동일하고 및 3 개의 핀들 "1", "2" 및 "3"을 갖는 것을 분명히 알 수 있다. 패스코드내 제 1 핀 "1"이 4 x 4 테이블 [G5 (716)]에 마지막 토큰에 표시되고, 해당 토큰이 제 1 키 토큰으로 선택되어야 한다. 그러나, 테이블내 제 2 토큰 [G5 (718)]이 선택되었고, 제 1 키 토큰 위치 G5 (721)에 도시된다. 이것을 틀렸고 이 메시지는 유효하지 않을 것이다.

[0368] · 스크린샷에서, 유저가 "Check" 버튼 G5 (726)를 클릭한 때, 그것은 이 문자 "P"에 대하여 확인 프로세스가 실패하였다는 것을 보여준다 [G5 (731)].

[0369] 도 20b는 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 언제 도 20a에 도시된 프로세스가 일어나고 어떻게 필러 메시지가 수신자 측에서 유효하지 않다는 장면들이 나중에 어떻게 일어나는지를 도시한다. 프로세스는 아래와 같이 진행한다:

[0370] · 메시지 문자 "P" [G5 (751)]가 발신자 패스코드 "123" [G5 (752)]로 암호화되고 16 개의 토큰들 [G5 (755)]의 4 x 4 테이블 및 일부 토큰들 [G5 (721), G5 (723) 및 G5 (725)]을 갖는 키와 부착된다. 이 정보는 네트워크 [G5 (757)]로, 그런 다음 수신자 [G5 (759)]로 발송된다.

[0371] · 수신자 측에서, 메시지를 복호화하기 위해 동일한 패스코드 "123" [G5 (760)]가 사용한다. 키 토큰들은 확인 프로세스 G5 (765), G5 (767)를 거쳐[G5 (763)], 각각의 키 토큰을 체크한다. K61, K62 및 K63로부터 첫번째 토큰이 유효하지 않다는 것을 알 수 있다.

[0372] · 첫번째 패스코드 "1" 은 마지막 토큰 [G5 (790)]에 나타나고, 그것이 선택되어야 한다. 그러나, 두번째 토큰 [G5 (792)]이 선택되었고 첫번째 키 토큰 위치 G5 (721)에 나타났다. 따라서 그것은 유효하지 않다.

[0373] · 최종 결론은 도 20a에 도시된 바와 같이 [G5 (731)] 메시지가 유효하지 않다[G5 (771)]는 것에 도달한다 [G5 (769)].

[0374] 도 21는 GATE_5 실시예를 사용하는 메시지 복호화 프로세스의 샘플 스크린샷이다. 어떻게 암호화된 원래의 메시지가 발신자 패스코드와 다른 수신자 패스코드에 의해 성공적으로 복호화되지 않을 수 있는지의 일 예를 도시한다. 프로세스는 아래와 같이 진행한다:

[0375] · 유저는 G5 (700)에 평문 텍스트 메시지 "FYEO"를 타이핑하고, 그런 다음 패스코드 "123" [G5 (702)]를 입력하고 "Encrypt" 버튼 [G5 (703)]을 클릭한다.

[0376] · 메시지가 암호화되고, 발송되고 수신되어 : "F1PRojcYnEbAO"[G5 (705)]로서 G5 (705)에 나타났다.

[0377] · 수신자는 패스코드 "123" [G5 (702)]로 암호화된 발신자로부터 수신된 메시지를 복호화하기 위해 패스코드 "680" [G5 (707)]를 사용한다.

[0378] · 복호화된 메시지가 G5 (705): "nE"에 하이라이트된다.

[0379] · 결과 메시지는 "nE" [G5 (709)]로서 도시된다.

[0380] · 결과 메시지는 발신자로부터의 원래의 메시지: "FYEO" [G5 (700)]과 다른데, 이는 수신자가 메시지를 복호화하기 위해 상이한 패스코드를 사용했기 때문이다.

[0381] 각각의 유효한 메시지에 대한 16개의 토큰들을 갖는 4 x 4 테이블에 대응하는 유효한 키 토큰들을 생성하기 위해서 이하의 단계들이 바람직하게 패스코드내 각각의 핀에 대하여 사용된다 :

[0382] · 모든 16 개의 토큰들은: (a) 만약 핀이 토큰에서 찾아지면, 해당 토큰을 선택하고; 및 (b) 만약 핀이 임의의 토큰에 없으면, 테이블내 16개의 토큰들으로부터 랜덤 토큰을 선택한다.

[0383] 각각의 유효하지 않은 메시지에 대한 16개의 토큰들을 갖는 4 x 4 테이블에 대응하는 유효하지 않은 키 토큰들을 생성하기 위해서 이하의 단계들이 바람직하게 사용된다 :

[0384] <1> 불리언 "Done_Fixing" 을 거짓으로 설정한다

[0385] <2> 모든 16 개의 토큰들에 거쳐, 패스코드내 각각의 핀에 대하여 아래의 단계들 <3> 및 <4>을 수행한다

[0386] <3> <A> 만약 핀이 토큰에서 발견되면:

[0387] (1) 만약 Done_Fixing이 거짓과 같으면, 의도적으로 틀린 토큰을 선택한 것을 제외하고 임의의 다른 토큰을 선택하고, Done_Fixing을 참으로 설정한다.

[0388] (2) 만약 Done_Fixing이 참과 같으면, 해당 토큰을 선택한다.

[0389] <B> 만약 핀이 임의의 토큰에 없으면, 16개 로부터 랜덤 토큰을 선택한다.

[0390] <4> 상기에서 생성된 키 토큰을 벡터로 저장한다.

[0391] <5> -1 내지 1의 범위내 랜덤 번호 N 를 발생시킨다

[0392] <A> 만약 N = -1이면, 벡터로부터 마지막 키 토큰을 삭제한다.

[0393] <B> 만약 N = 0이면, 아무것도 하지 않는다.

[0394] <C> 만약 N = 1 이고 유저 핀 길이 <6이면, 테이블내 16을으로부터 벡터에 랜덤 토큰을 추가한다.

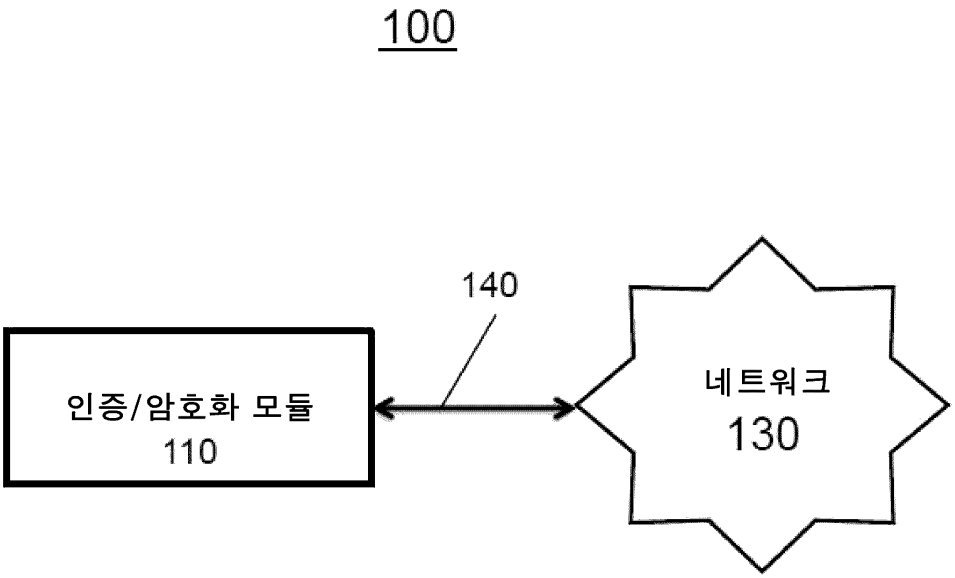
[0395] <6> 벡터내 토큰들이 최종 키 토큰들일 것이다.

[0396] 앞에서의 실시예들 및 장점들은 단지 예시이고, 본 발명의 제한으로서 이해되지 않아야 한다. 본 발명의 설명은 예시적인 것으로 의도되고, 청구항의 범위를 제한하지 않는다. 많은 대안들, 수정예들 및 변형예들이 당해 기술분야의 통상의 기술자들에 명확해질 것이다. 다양한 변화들이 이하의 청구항들에 정의된 본 발명의 취지 및 범위에서 벗어나지 않고서 이루어질 수 있다.

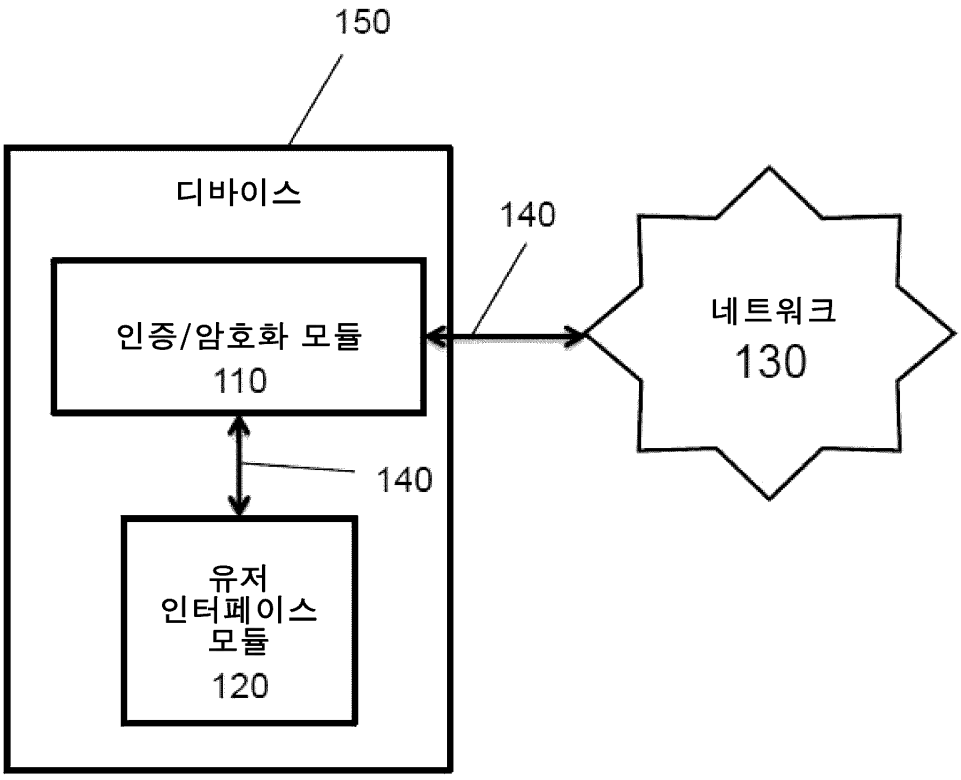
[0397] 예를 들어, 비록 본 발명은 4 차원 및 5 차원의 심벌들이 개별적으로 사용되는 GATE_4 및 GATE_5 실시예와 관련하여 설명되었지만, (단지 하나의 차원만 포함하는) 임의 개수의 차원이 사용될 수 있고 또한 본 발명의 범위내에 해당한다. 일반적으로, 각각의 토큰이 하나 초과의 심벌을 갖는 한, 임의 개수의 차원으로 분류 된 임의 개수의 심벌들이 사용될 수 있다. 또한, 전술한 GATE_4 및 GATE_5 실시예 뿐만 아니라 관련된 스크린샷은 설명을위한 것이지 본 발명의 범위를 제한하는 것은 아니다.

도면

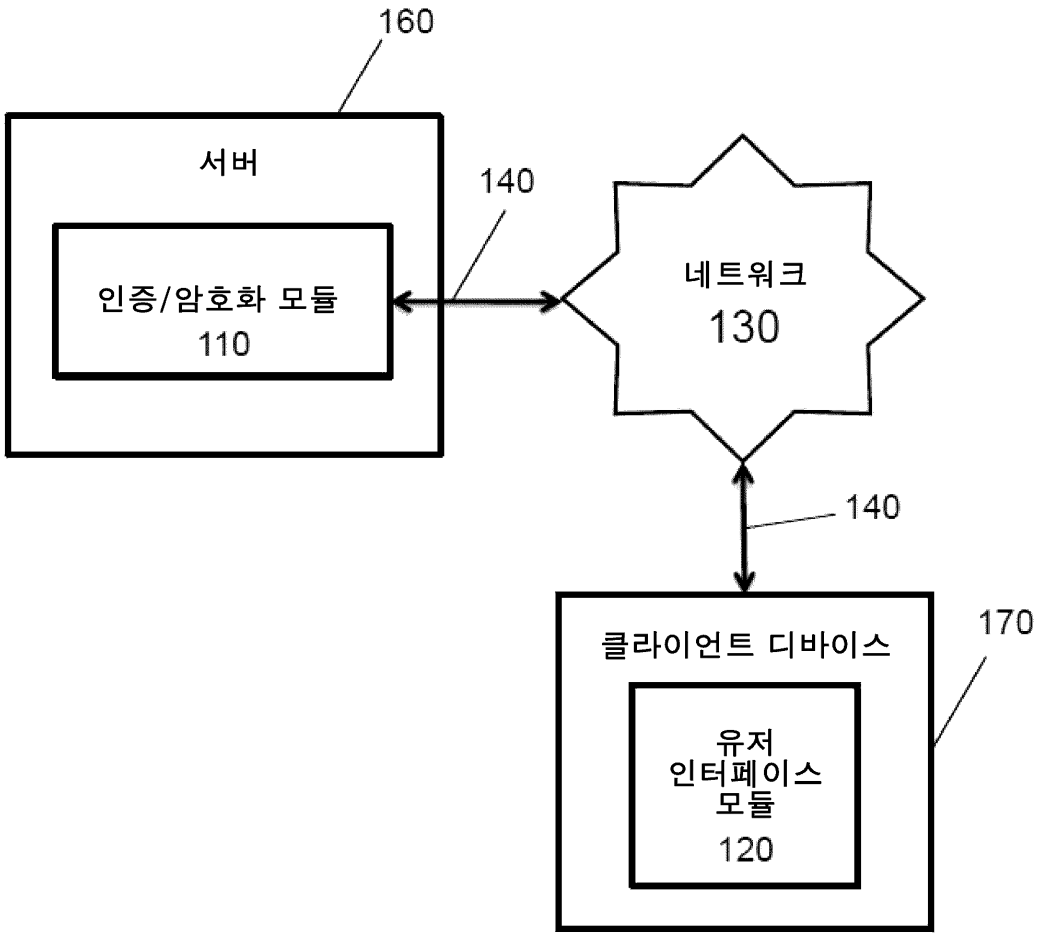
도면1a



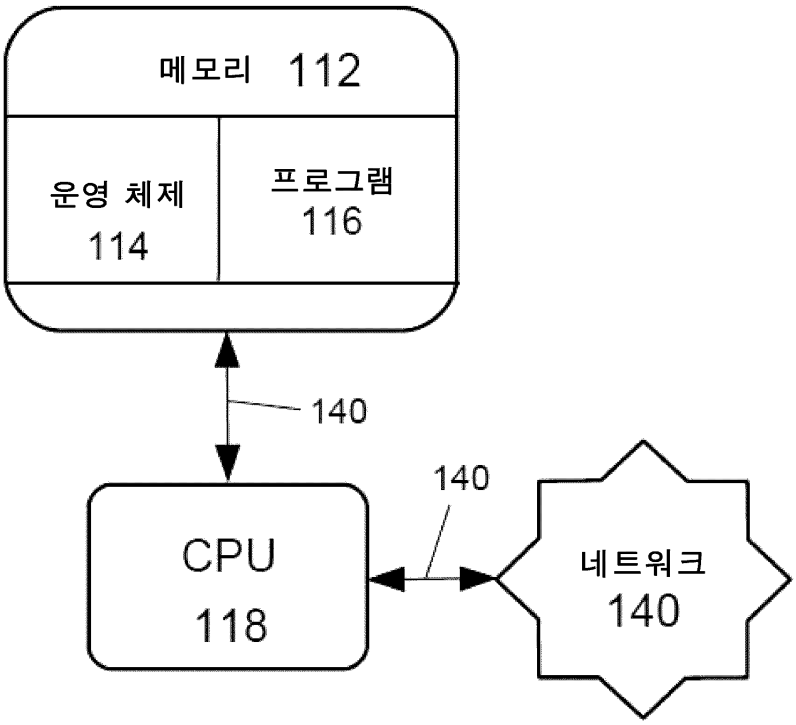
도면1b



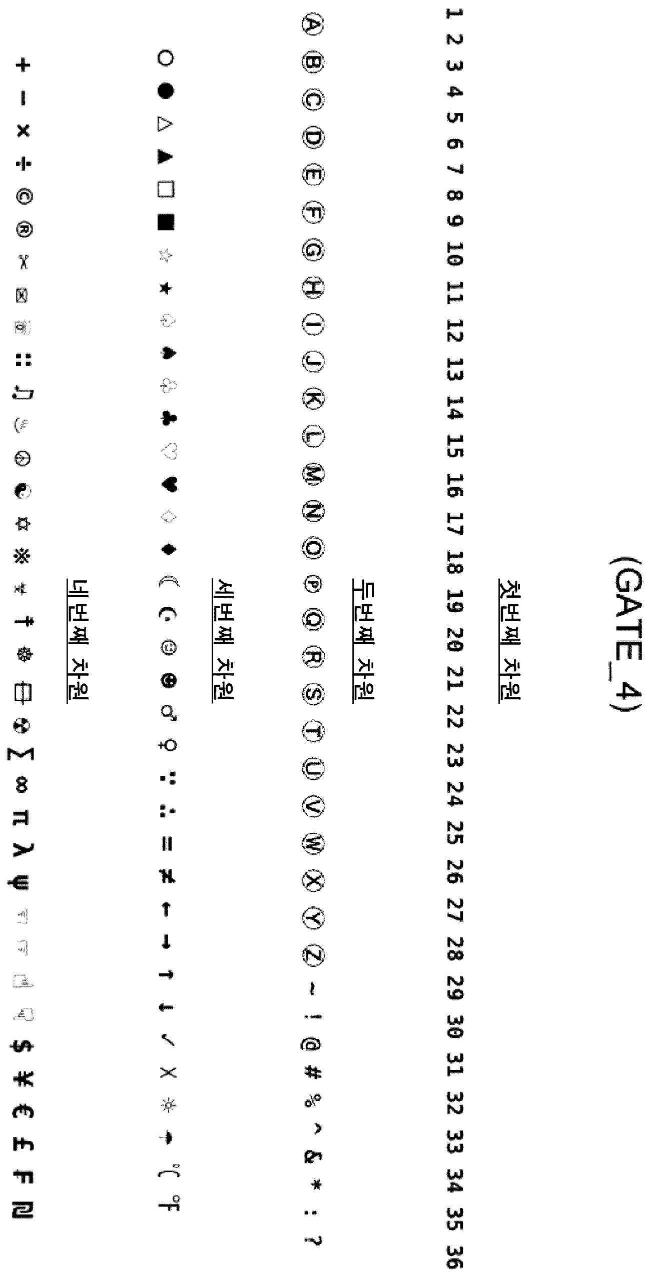
도면1c


160
서버
인증/암호화 모듈
110
140
네트워크
130
140
클라이언트 디바이스
170
유저
인터페이스
모듈
120

도면1d



도면2a


(GATE_4)
첫번째 차원
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
두번째 차원
Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ Ⓕ Ⓖ Ⓗ Ⓘ Ⓙ Ⓚ Ⓛ Ⓜ Ⓝ Ⓞ Ⓟ Ⓠ Ⓡ Ⓢ Ⓣ Ⓤ Ⓥ Ⓦ Ⓧ Ⓨ Ⓩ ~ ! @ # % ^ & * : ?
세번째 차원
네번째 차원

도면2b

(GATE_5)

첫번째 차원

Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ Ⓕ Ⓖ Ⓗ Ⓘ Ⓙ Ⓚ Ⓛ Ⓜ Ⓝ Ⓞ Ⓟ Ⓠ Ⓡ Ⓢ Ⓣ Ⓤ Ⓥ Ⓦ Ⓧ Ⓨ Ⓩ

두번째 차원

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

세번째 차원

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

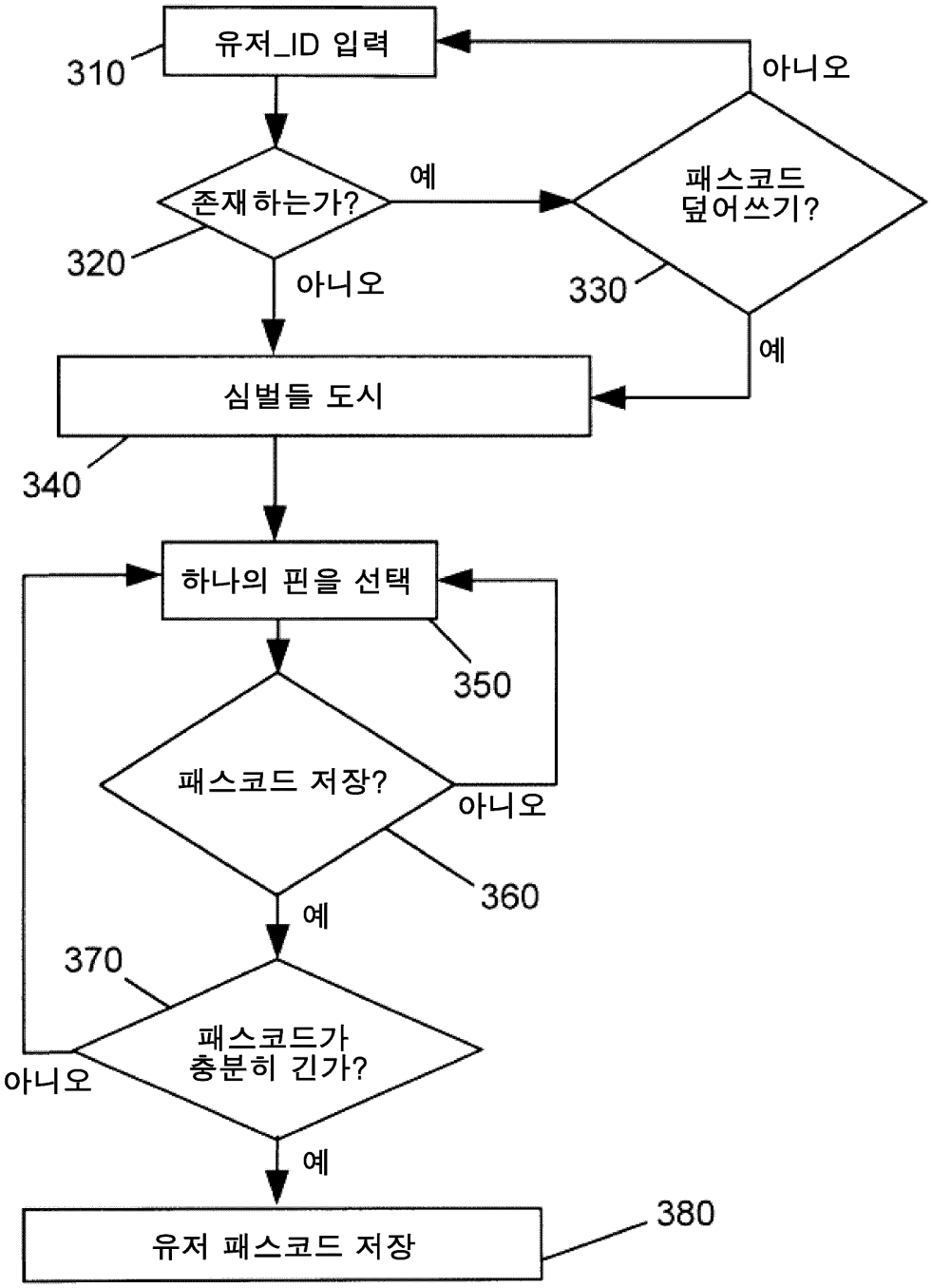
네번째 차원

○ ● △ ▲ □ ■ ☆ ★ ♤ ♠ ♧ ♣ ♡ ♥ ◇ ◆ ☺ ☻ ♂ ♀ ➔ ☼ ∵ ∴ = ≠

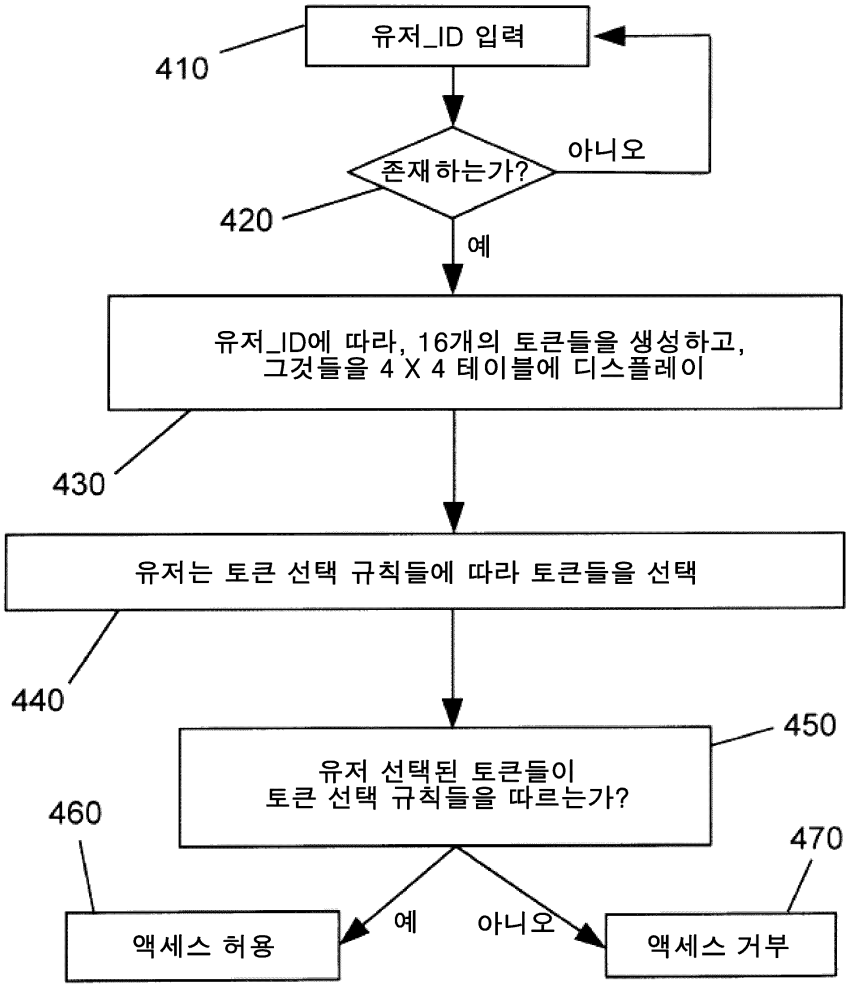
다섯번째 차원

+ − × ÷ ↑ → ↑ ↓ ☓ ⊠ ☸ ∷ ♫ ☮ ☯ ✡ ☾ ℉ ⋆ 日 ☢ $ ¥ € £ F ₪

도면3


유저_ID 입력
310
존재하는가?
320
예
아니오
패스코드
덮어쓰기?
330
아니오
예
심벌들 도시
340
하나의 핀을 선택
350
패스코드 저장?
아니오
360
예
370
패스코드가
충분히 긴가?
아니오
예
유저 패스코드 저장
380

도면4


유저_ID 입력
410
존재하는가?
아니오
420
예
유저_ID에 따라, 16개의 토큰들을 생성하고,
그것들을 4 X 4 테이블에 디스플레이
430
유저는 토큰 선택 규칙들에 따라 토큰들을 선택
440
450
유저 선택된 토큰들이
토큰 선택 규칙들을 따르는가?
460
470
예
아니오
액세스 허용
액세스 거부

도면5

(토큰 생성 규칙들)

[(예시를 위해, GATE_4 실시예가 사용된다(도 2a 참조)]

각각의 차원으로부터 하나.
예를 들어 : 토큰_1(2 Ⓩ ♡ *), 토큰_2(30 ? ♠ ⅄), …. , 토큰16(16 @ ✓ ☎)
적어도 하나의 유저 핀*이 16개의 토큰들 중 하나에, 가능한 많이, 심지어 그것들의 전부가 있어야 한다.
상기의 심벌들의 순서들은, 첫번째 것은 항상 첫번째 _Dim으로부터, 두번째 것은 두번째_Dim으로부터 …인 것에 주목한다.

규칙들 및 단계들	예제 결과들
[1] 4 벡터들을 생성하고, 각각은 도2에 설명된 네개의 차원들 중 하나의 차원으로부터 36 심벌들을 함유한다 : GATE_4를 위한 이용가능한_심벌들.매번 심벌은 벡터로부터 제거되고, 그것의 사이즈는 하나씩 감소할 것이다. 벡터들로부터 심벌들을 제거하는 이유는 중복을 피하기 위한 것이다. 상기의 벡터들을 차원 벡터들 : V_1, V_2, V_3, V_4로 나타낸다. 각각의 토큰 생성 단계는 4개의 상기 벡터들의 각각으로부터 하나의 남은 심벌을 제거할 것이며, 어떠한 두개의 토큰들도 같은 심벌들을 갖지 않을 것이다.	[1] V_1(1, 2, … 35, 36) V_2(Ⓐ, Ⓑ, … ;, ?) V_3(○, ●, … ℃, ℉) V_4(+, −, … F, ₪)
[2] 로그인 프로세스 동안에 유저에 의해 입력된 유저_ID를 획득한다.	[2] admin
[3] 등록 동안에 메모리에 저장된 유저_ID로부터 유저 패스코드**를 획득한다.	[3] ⓘ ♥ 2 ✉
[4] 유저 패스코드로부터 User_Pin_Vector ***로 랜덤 핀을 저장한다.	[4] ♥
[5] User_Pin_Show_Up_Location에 랜덤 숫자{1에서부터 16까지}를 저장한다.	[5] 7
[6] (i=1; i<=16; i++)에 대하여, 단계[7] 및 [8]을 수행하여 16개의 토큰들을 생성한다.	[6]
[7] 비어있는 토큰 : 4개의 심벌들을 보유하기 위한 벡터를 생성한다	[7] 토큰_i()
[8] (j=1; j<=4; j++)에 대하여, 단계[8.1] 또는 [8.2]을 수행하여 각각의 차원으로부터 하나의 심벌을 토큰에 추가한다. 단계 [8.1]은 적어도 하나의 유저 핀이 사용되는 것을 확실히 한다.	[8] 4개의 심벌들을 그것에 추가한다.
[8.1] 만약 i 가 USER_Pin_Show_Up_Location과 같고, User_Pin_Vector 로부터 ♥가 여전히 V_i 에 있으면, V_i로부터 그것을 제거하고 그것을 현재 토큰에 추가한다.	[8.1] 토큰_7 (15 ~ ♥ ☞)
[8.2] 그렇지 않으면, 하나의 랜덤 심벌을 V_i 로부터 제거하여 현재 토큰에 추가한다. V_i의 사이즈는 해당 심벌을 제거한 후에 하나씩 감소할 것이다.	[8.2] 토큰_i(8 Ⓕ ★ ⊟)
[9] 상기 단계들 후에, 적어도 하나의 유저 핀과 16개의 토큰들을 가질 것이다.	[9] 토큰_1, …., 토큰_16

주석 : * 핀 – 패스코드**내 각각의 심벌은 핀이다. 예를 들어, 이하의 패스코드 :ⓘ♥ 2✉에서 ⓘ는 첫번째 핀이고, ⓘ은 네번째 핀이다.

** 패스코드 – 상식의 패스워드와 유사하지만, 그러나 각각의 차원에 선택들로부터 특정 심벌들을 함유할 수 있다 [예를 들어, ⓘ♥ 2✉], 차원들은 도면들 2a 및 2b에 설명된다.

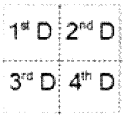
*** User_Pin_Vector – 컴퓨터 언어 자바에서 벡터 이는 임의의 유형의 임의의 개수의 엘리먼트들, 이 경우 심벌을 함유할 수 있다.

도면6

(토큰 선택 규칙들)

[1] 유저는 4 x 4 테이블에 16개의 토큰들을 볼 것이고, 각각의 토큰은 그 안에 4개의 심벌들을 가질 것이고, 토큰(2 Ⓩ ♡ ÷)은 이것처럼 보일 것이다 :

각각의 차원은 토큰내 고정된 위치를 가지며, 상단 좌측은 첫번째 차원 심벌을 위한 것이고, 상단 우측은 두번째 차원 심벌들을 위한 것이고, 하단 좌측은 세번째 심벌들을 위한 것아고, 하단 우측은 네번째 차원 심벌들을 위한 것이다.



고정된 위치는 유저가 빠르게 심벌들을 토큰내에 위치시키는 것을 도울 것이다.

[2] 유저 핀들을 함유하는 토큰들을 선택하기 위해서 유저 패스코드의 순서 예를 들어, Ⓘ ♥ 2 ✉ 의 순서를 따라야 한다. 예를 들어, 우측 테이블에 16 토큰들을 가지며, 그것들 중 적어도 하나는 유저 핀을 함유하며, 유저는 토큰들을 선택하기 위해서 아래의 단계들을 따를 수 있다.

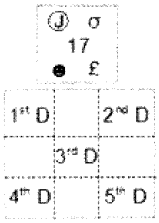
	1	2	3	4
A	13 & ☺ ✉	34 Ⓜ ♢ ♫	29 Ⓟ → ☢	28 ! ✓ ÷
B	3 Ⓠ ← Ⓡ	36 Ⓥ = ☯	6 : ↓ π	19 Ⓦ X +
C	20 Ⓩ ☞	26 Ⓙ ℉ €	17 Ⓢ ♤ ×	12 Ⓒ ♡ ₪
D	32 Ⓚ ♠ $	35 Ⓕ ☆ ☸	16 Ⓑ ● ☜	10 Ⓤ ℃ ☎

더 쉬운 참조를 위하여, 로우들 : A, B, C, D 및 컬럼들 : 1, 2, 3, 4로 명명한다.

[3] 유저의 첫번째 핀은 두번째 차원에 속하는 Ⓘ이기 때문에, 각각의 토큰의 상단 우측 코너에서 그것을 검색할 수 있는데, 16개 중 임의에 없기 때문에, 그것의 위치에서 임의의 토큰을 선택할 수 있고 선택하여야 한다.

따라서, A2 : (34 Ⓜ ♢ ♫)를 선택할 수 있다. 두번째 유저 핀 : ♥ 과 동일하고, 왜냐하면, 그것이 누락되었기 때문이고, D3 : (16 Ⓑ ● ☜)를 선택할 수 있다. 세번째 유저 핀 2가 또한 누락되고, 따라서 B1 : (3 Ⓠ ← Ⓡ)을 선택할 것이다. 그러나 네번째 유저 핀 ✉은 토큰 A1에 있고, 유저는 그것을 유효한 것으로 선택해야 하고, 따라서 A1 : (13 & ☺ ✉)을 선택할 것이다. 따라서, 결국에 유저는 이하의 4개의 토큰들을 선택하였다 : (34 Ⓜ ♢ ♫), (16 Ⓑ ● ☜), (3 Ⓠ ← Ⓡ), 및 (13 & ☺ ✉), 이들 4개의 토큰들은 확인을 위해 서버로 발송될 것이다.

GATE_5에 대하여 : 프로세스는 GATE_4와 같고, 단지 하나의 더 많은 차원을 추가한다. 샘플 토큰 및 다섯번째 차원의 위치가 오른쪽에 도시된다. 이제 토큰은 이것과 같을 수 있다 : (Ⓙ σ 17 ● £).



각각의 테이블은 16토큰들을 가지며, 그러나 각각의 토큰은 5 심벌들을 함유한다 – 도 2b로부터 다섯번째 차원의 각각으로부터 하나.

도면7

(토큰 확인 규칙들)

GATE_4에 대하여 : 도6의 예제로 계속.

[1] 유저 핀들을 함유하는 토큰들을 선택하기 위해서 유저는 유저 패스코드 ⓘ♥2✉ 의 순서를 따라야 한다.

[2] 원래의 16개의 토큰들이 오른쪽 테이블에 도시된다.

[3] 유저가 선택한 4개의 토큰들은 : (34 Ⓜ ♢ ♫), (16 Ⓑ ● ☜), (3 Ⓠ ← Ⓡ) 및 (13 & ☺ ✉)이다.

[4] 확인 프로세스는 그것들을 하나씩 체크할 것이며, 만약 그것들중 하나가 실패하면, 유저 로그인 요청은 거부될 것이다.

[5] 만약 유저가 패스코드내 핀 카운트보다 더 많거나 또는 더 적은 토큰들을 선택하면, 로그인 요청은 또한 거부될 것이다

[6] 샘플 구현예에서, 유저 패스코드들은 6개의 핀들까지 일 수 있다.

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

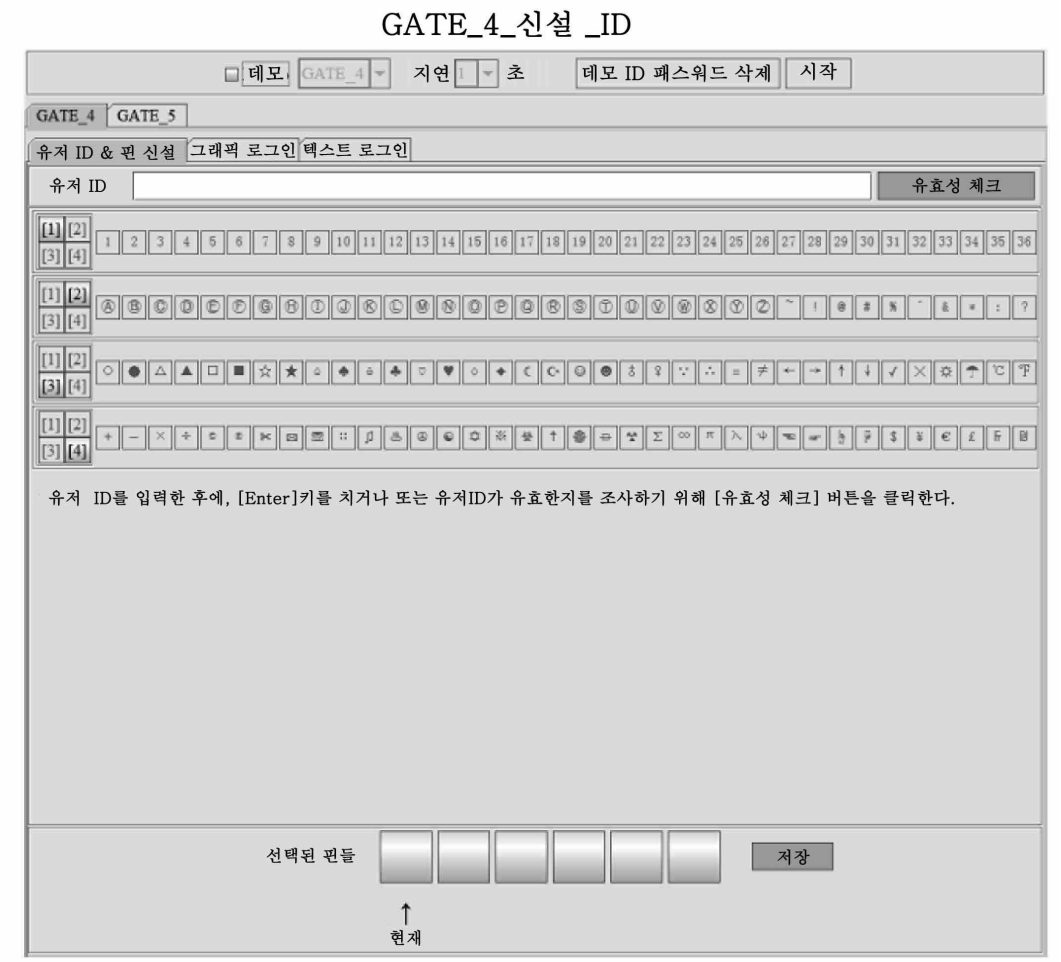
[7] 유저 첫번째 핀이 ⓘ이기 때문에, 16개의 토큰들 전부에 모든 심벌을 살펴보고, 심벌 ⓘ가 존재하는지를 본다. 그것이 존재하지 않기 때문에, 유저는 그 위치에 와일드-카드 토큰을 선택할 수 있고, 선택해야 하고, 그리고 예제에서, 유저의 첫번째 선택된 토큰 (34 Ⓜ ♢ ♫)은 유효하다.

[8] 유저의 두번째 핀 ♥ 또한 16개의 토큰들 전부에서 누락되고, 따라서 유저가 선택한 두번째 토큰 : (16 Ⓑ ● ☜) 또한 유효하고, 유저가 선택한 세번째 토큰 (3 Ⓠ ← Ⓡ)도 같다.

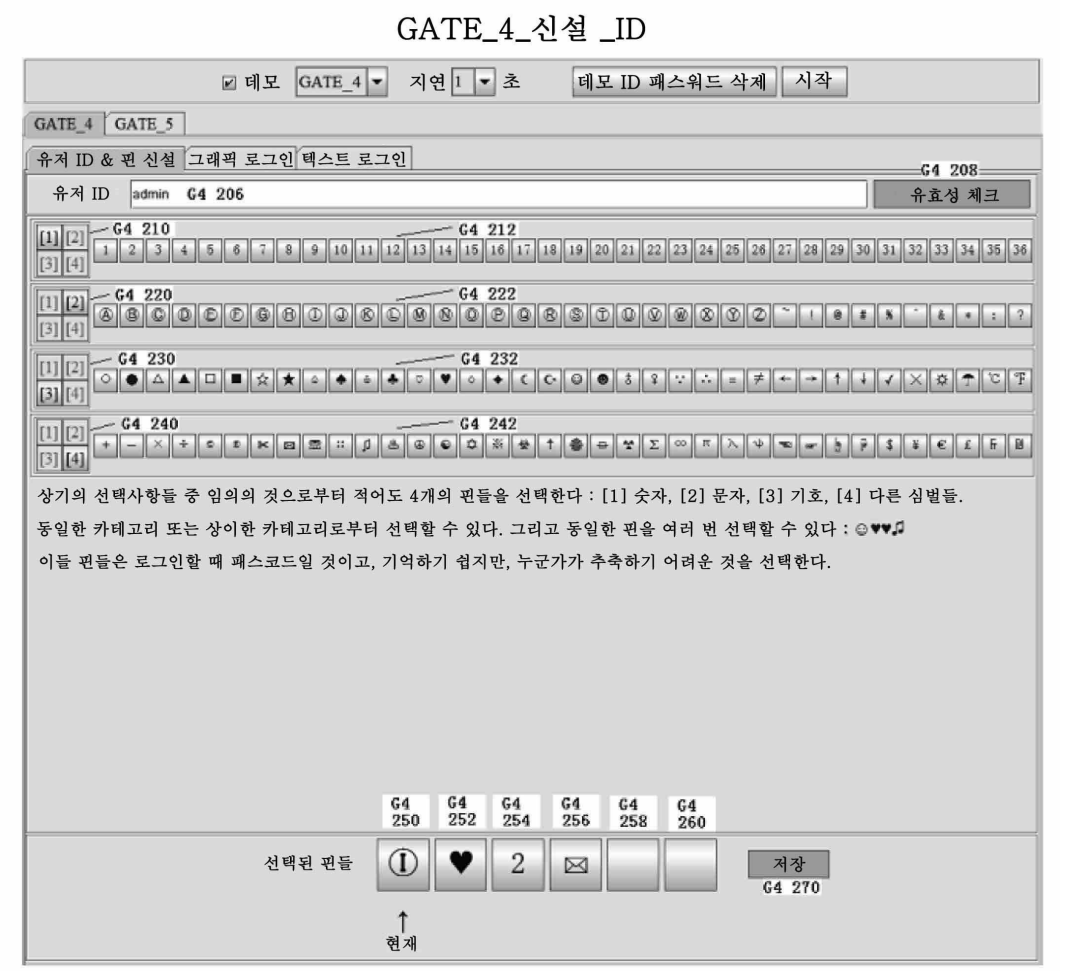
[9] 유저의 네번째 핀은 심벌 ✉이고, 상기의 16개의 토큰들 전부를 살펴볼 때, 그것은 A1 토큰에 있는 것을 볼 수 있고, 따라서 유효한 것으로 하기 위해서, 유저는 A1를 선택해야하고, 선택할 수 있고 예제에서, 네번째 마지막 토큰으로서 (13 & ☺ ✉)을 선택하였다. 원래의 패스코드에 핀 카운트 (4)보다 더 많거나 더 작지 않고, 따라서 유저 로그인 요청이 승인된다.

GATE_5에 대하여 : 프로세스는 같고, 단지 하나의 더 많은 차원을 추가한다.

도면8a


GATE_4_신설 _ID
데모
GATE_4
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
유효성 체크
유저 ID를 입력한 후에, [Enter]키를 치거나 또는 유저ID가 유효한지를 조사하기 위해 [유효성 체크] 버튼을 클릭한다.
선택된 핀들
저장
현재

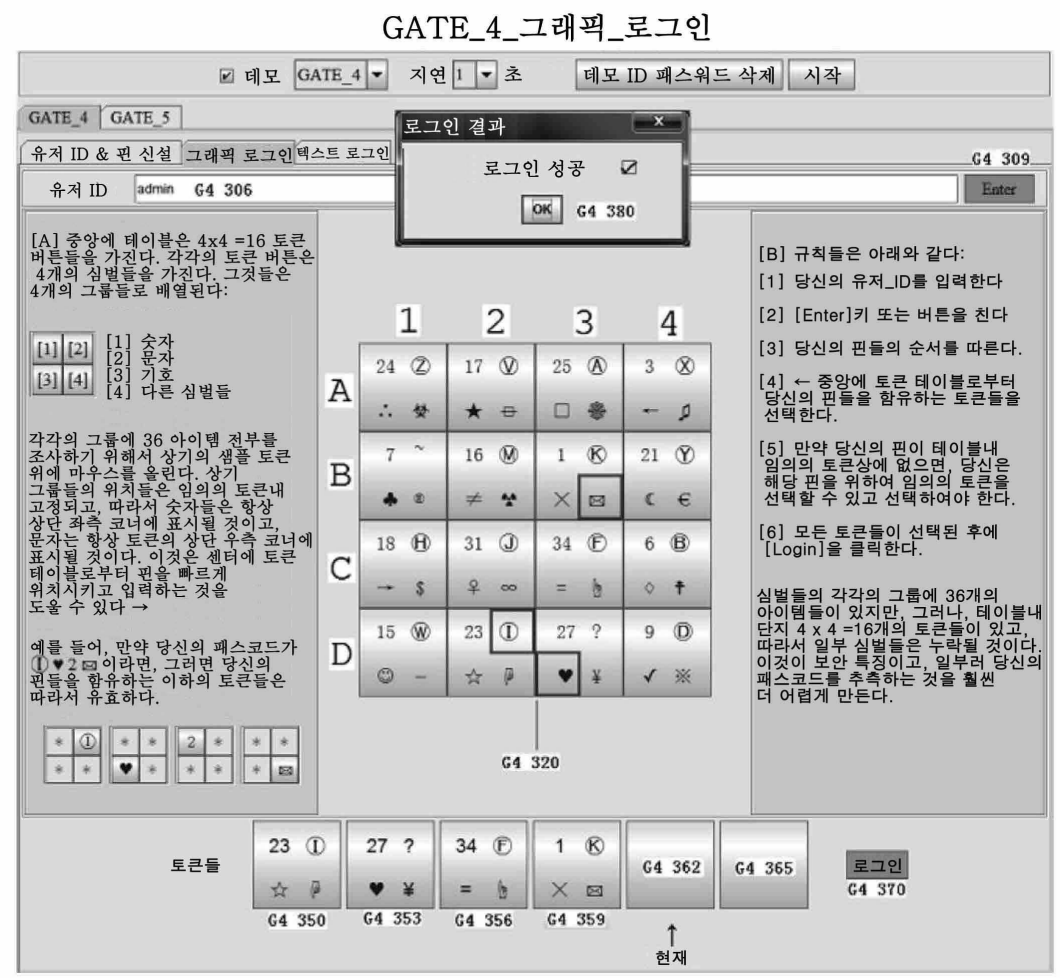
도면8b


GATE_4_신설 _ID
데모
GATE_4
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
admin G4 206
G4 208
유효성 체크
G4 210
G4 212
G4 220
G4 222
G4 230
G4 232
G4 240
G4 242
상기의 선택사항들 중 임의의 것으로부터 적어도 4개의 핀들을 선택한다 : [1] 숫자, [2] 문자, [3] 기호, [4] 다른 심벌들.
동일한 카테고리 또는 상이한 카테고리로부터 선택할 수 있다. 그리고 동일한 핀을 여러 번 선택할 수 있다 : ☺♥♥♫
이들 핀들은 로그인할 때 패스코드일 것이고, 기억하기 쉽지만, 누군가가 추측하기 어려운 것을 선택한다.
G4 250
G4 252
G4 254
G4 256
G4 258
G4 260
선택된 핀들
2
저장
G4 270
현재

도면9a


GATE_4_그래픽_로그인
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
Enter
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 4개의 심벌들을 가진다. 그것들은 4개의 그룹들로 배열된다:
[1] [2]
[3] [4]
[1] 숫자
[2] 문자
[3] 기호
[4] 다른 심벌들
각각의 그룹에 36 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 숫자들은 항상 상단 좌측 코너에 표시될 것이고, 문자는 항상 토큰의 상단 우측 코너에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ① ♥ 2 ✉ 이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
[B] 규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 36개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
로그인
현재

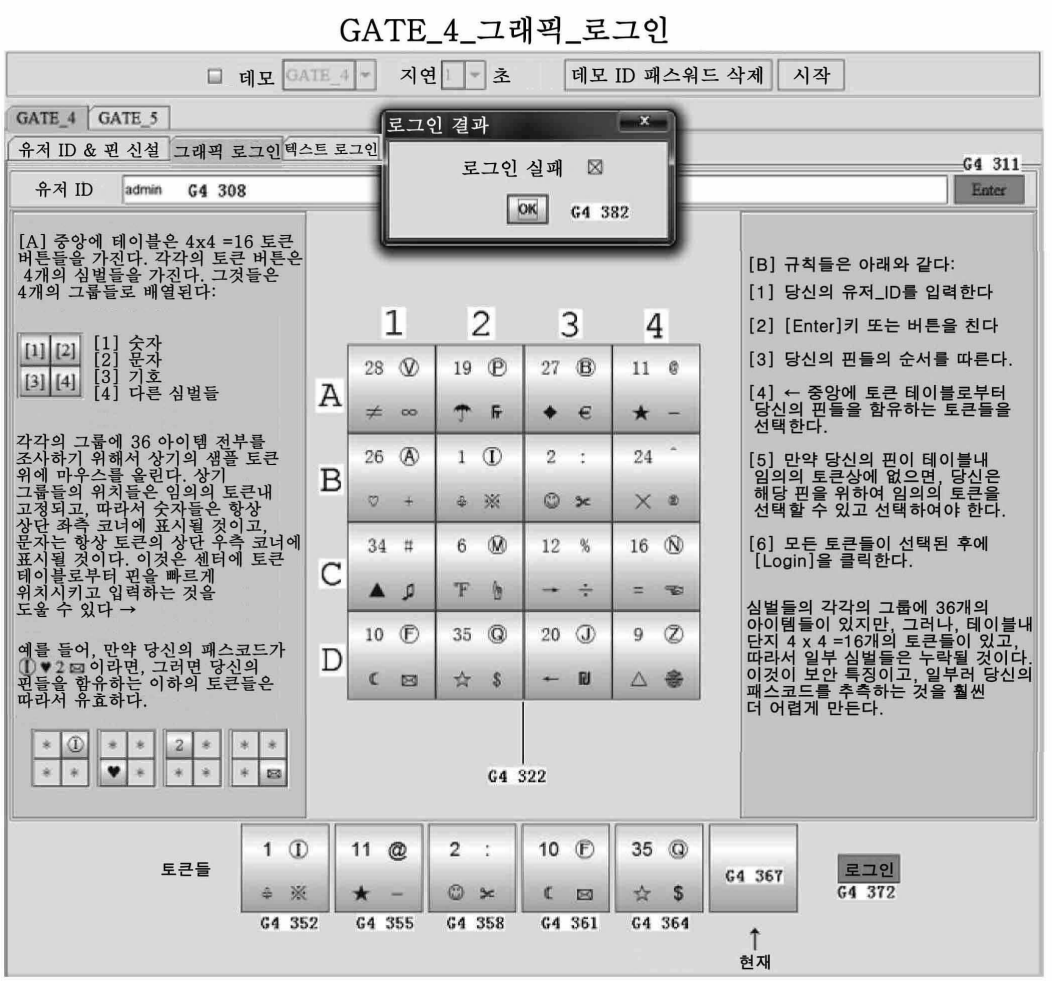
도면9b


GATE_4_그래픽_로그인
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
로그인 결과
로그인 성공
OK
G4 380
G4 309
유저 ID
admin
G4 306
Enter
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 4개의 심벌들을 가진다. 그것들은 4개의 그룹들로 배열된다:
[1] 숫자
[2] 문자
[3] 기호
[4] 다른 심벌들
각각의 그룹에 36 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 숫자들은 항상 상단 좌측 코너에 표시될 것이고, 문자는 항상 토큰의 상단 우측 코너에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ①♥2✉이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
1
2
3
4
A
B
C
D
G4 320
[B] 규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 36개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
G4 350
G4 353
G4 356
G4 359
G4 362
G4 365
현재
로그인
G4 370

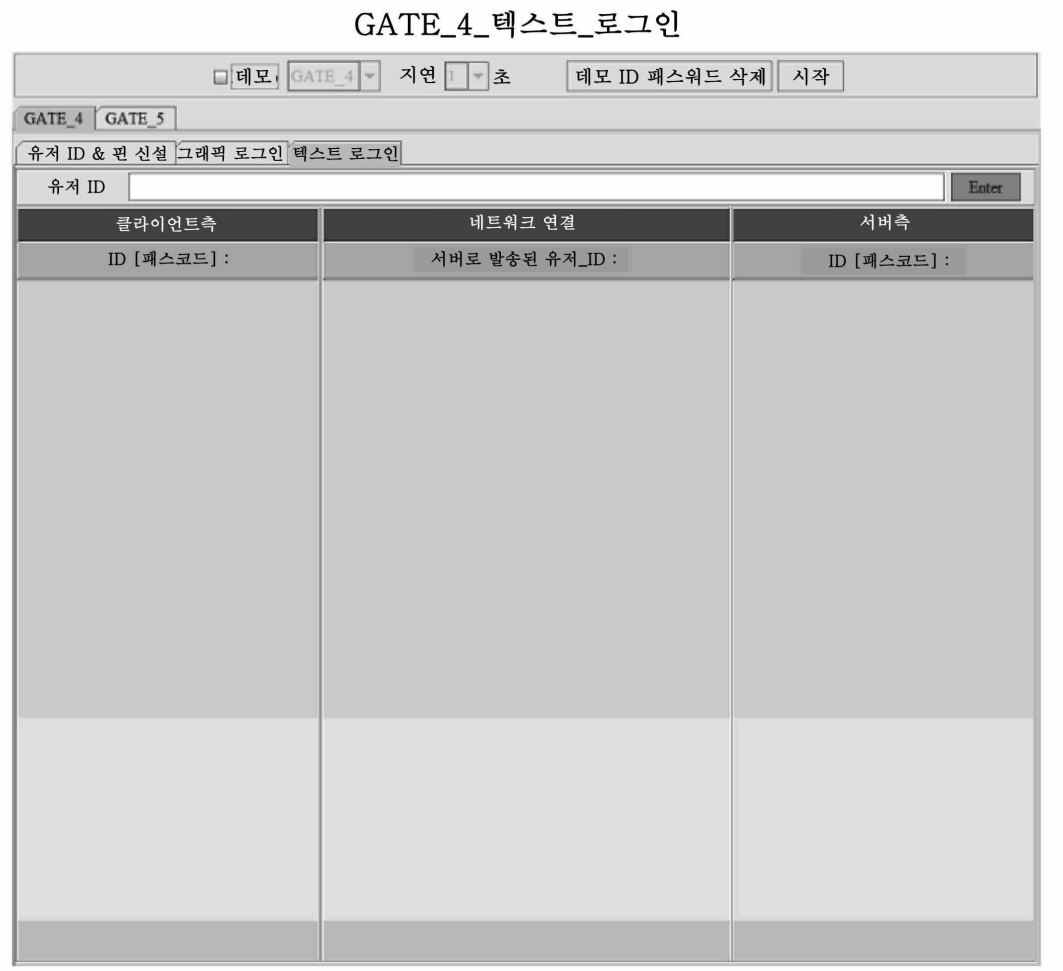
도면9c


GATE_4_그래픽_로그인
테모
GATE_5
지연
초
테모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G4 310
유저 ID
admin G4 307
Enter
로그인 결과
로그인 실패
OK
G4 381
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 4개의 심벌들을 가진다. 그것들은 4개의 그룹들로 배열된다:
[1] 숫자
[2] 문자
[3] 기호
[4] 다른 심벌들
각각의 그룹에 36 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 숫자들은 항상 상단 좌측 코너에 표시될 것이고, 문자는 항상 토큰의 상단 우측 코너에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ①♥2✉이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
1
2
3
4
A
B
C
D
G4 321
[B] 규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 36개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
G4 351
G4 354
G4 357
G4 360
G4 363
G4 366
현재
로그인
G4 371

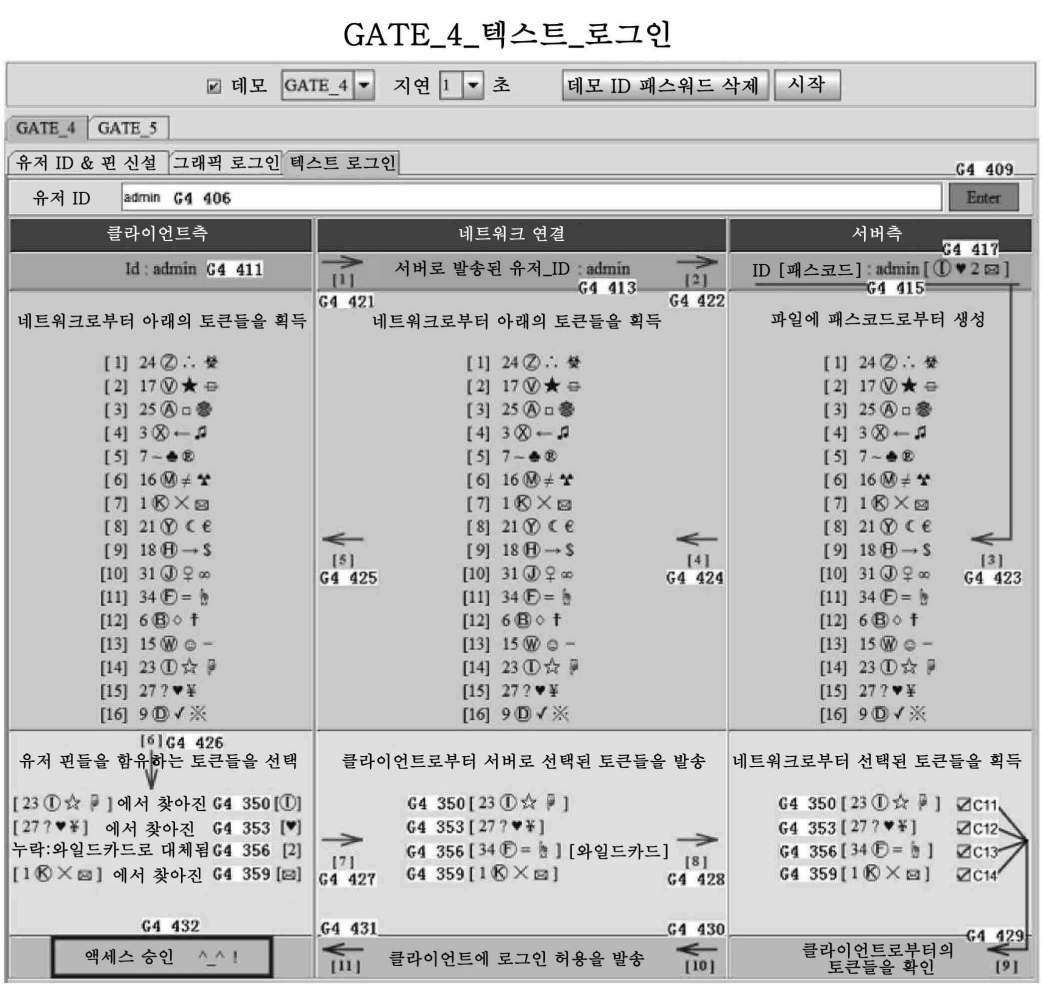
도면9d


GATE_4_그래픽_로그인
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
로그인 결과
로그인 실패
OK
G4 382
G4 311
Enter
유저 ID
admin
G4 308
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 4개의 심벌들을 가진다. 그것들은 4개의 그룹들로 배열된다:
[1] 숫자
[2] 문자
[3] 기호
[4] 다른 심벌들
각각의 그룹에 36 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 숫자들은 항상 상단 좌측 코너에 표시될 것이고, 문자는 항상 토큰의 상단 우측 코너에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ⓘ♥2✉이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
1 2 3 4
A
B
C
D
G4 322
[B] 규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 36개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
G4 352
G4 355
G4 358
G4 361
G4 364
G4 367
현재
로그인
G4 372

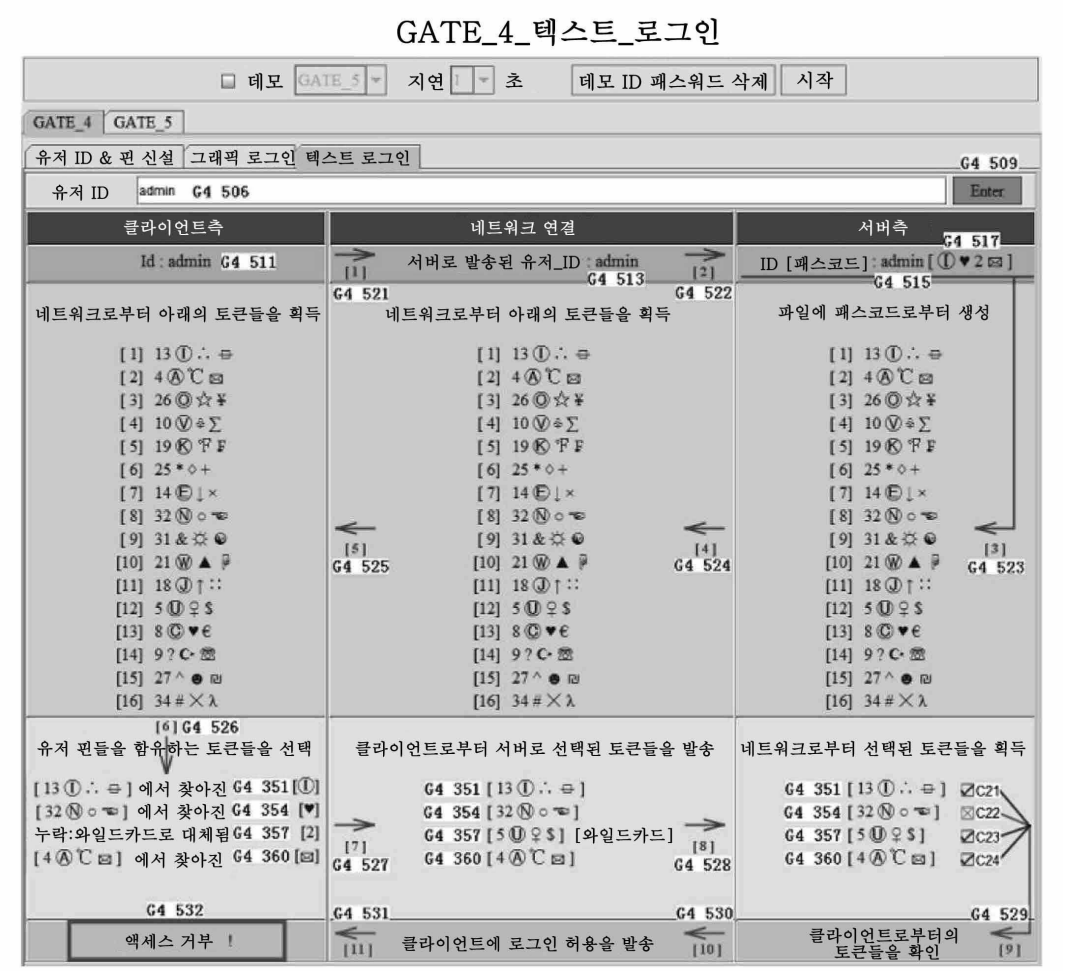
도면10a


GATE_4_텍스트_로그인
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
Enter
클라이언트측
네트워크 연결
서버측
ID [패스코드] :
서버로 발송된 유저_ID :
ID [패스코드] :

도면10b


GATE_4_텍스트_로그인
데모
GATE_4
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G4 409
유저 ID
admin G4 406
Enter
클라이언트측
네트워크 연결
서버측
Id : admin G4 411
서버로 발송된 유저_ID : admin
G4 413
ID [패스코드] : admin [ⒾG4 417
G4 415
[1]
[2]
G4 421
G4 422
네트워크로부터 아래의 토큰들을 획득
네트워크로부터 아래의 토큰들을 획득
파일에 패스코드로부터 생성
[1] 24 Ⓩ ∴
[2] 17 Ⓥ ★
[3] 25 Ⓐ
[4] 3 Ⓧ
[5] 7 ~ ♣
[6] 16 Ⓜ ≠
[7] 1 Ⓚ ×
[8] 21 Ⓨ ☾ €
[9] 18 Ⓗ → $
[10] 31 Ⓙ ♀ ∞
[11] 34 Ⓕ =
[12] 6 Ⓑ ◇
[13] 15 Ⓦ ☺ −
[14] 23 Ⓘ ☆
[15] 27 ? ♥ ¥
[16] 9 Ⓓ ✓ ※
[5]
G4 425
[4]
G4 424
[3]
G4 423
[6] G4 426
유저 핀들을 함유하는 토큰들을 선택
클라이언트로부터 서버로 선택된 토큰들을 발송
네트워크로부터 선택된 토큰들을 획득
[23 Ⓘ ☆] 에서 찾아진 G4 350 [Ⓘ]
[27 ? ♥ ¥] 에서 찾아진 G4 353 [♥]
누락:와일드카드로 대체됨G4 356 [2]
[1 Ⓚ ×] 에서 찾아진 G4 359 [ ]
G4 350 [23 Ⓘ ☆]
G4 353 [27 ? ♥ ¥]
G4 356 [34 Ⓕ =] [와일드카드]
G4 359 [1 Ⓚ ×]
G4 350 [23 Ⓘ ☆] C11
G4 353 [27 ? ♥ ¥] C12
G4 356 [34 Ⓕ =] C13
G4 359 [1 Ⓚ ×] C14
[7]
G4 427
[8]
G4 428
G4 432
G4 431
G4 430
G4 429
액세스 승인 ^_^ !
클라이언트에 로그인 허용을 발송
클라이언트로부터의
토큰들을 확인
[11]
[10]
[9]

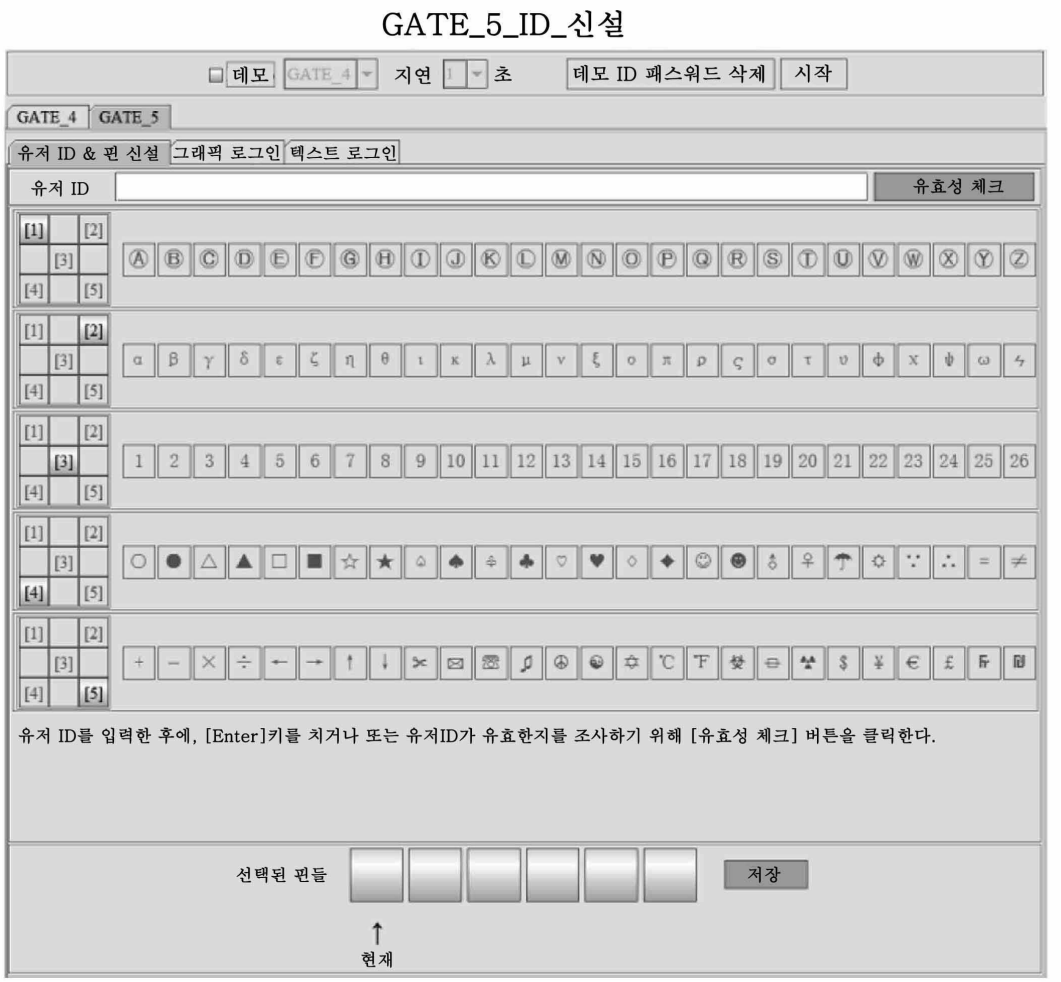
도면10c


GATE_4_텍스트_로그인
데모
GATE_5
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G4 509
유저 ID
admin G4 506
Enter
클라이언트측
네트워크 연결
서버측
G4 517
Id : admin G4 511
[1]
서버로 발송된 유저_ID : admin
G4 513
[2]
G4 521
G4 522
ID [패스코드] : admin [ⒾV2✉]
G4 515
네트워크로부터 아래의 토큰들을 획득
네트워크로부터 아래의 토큰들을 획득
파일에 패스코드로부터 생성
[1] 13ⒾV2✉
[2] 4ⒶC✉
[3] 26ⓄV¥
[4] 10VΣ
[5] 19KF
[6] 25*◇+
[7] 14E↓×
[8] 32N○
[9] 31&V
[10] 21WA
[11] 18J↑
[12] 5U♀$
[13] 8C♥€
[14] 9?C
[15] 27^●
[16] 34#×λ
[5]
G4 525
[4]
G4 524
[3]
G4 523
[6] G4 526
유저 핀들을 함유하는 토큰들을 선택
클라이언트로부터 서버로 선택된 토큰들을 발송
네트워크로부터 선택된 토큰들을 획득
[13ⒾV2✉] 에서 찾아진 G4 351 [Ⓘ]
[32N○] 에서 찾아진 G4 354 [♥]
누락:와일드카드로 대체됨 G4 357 [2]
[4ⒶC✉] 에서 찾아진 G4 360 [✉]
G4 351 [13ⒾV2✉]
G4 354 [32N○]
G4 357 [5U♀$] [와일드카드]
G4 360 [4ⒶC✉]
G4 351 [13ⒾV2✉] C21
G4 354 [32N○] C22
G4 357 [5U♀$] C23
G4 360 [4ⒶC✉] C24
[7]
G4 527
[8]
G4 528
G4 532
G4 531
G4 530
G4 529
액세스 거부 !
클라이언트에 로그인 허용을 발송
클라이언트로부터의 토큰들을 확인
[11]
[10]
[9]

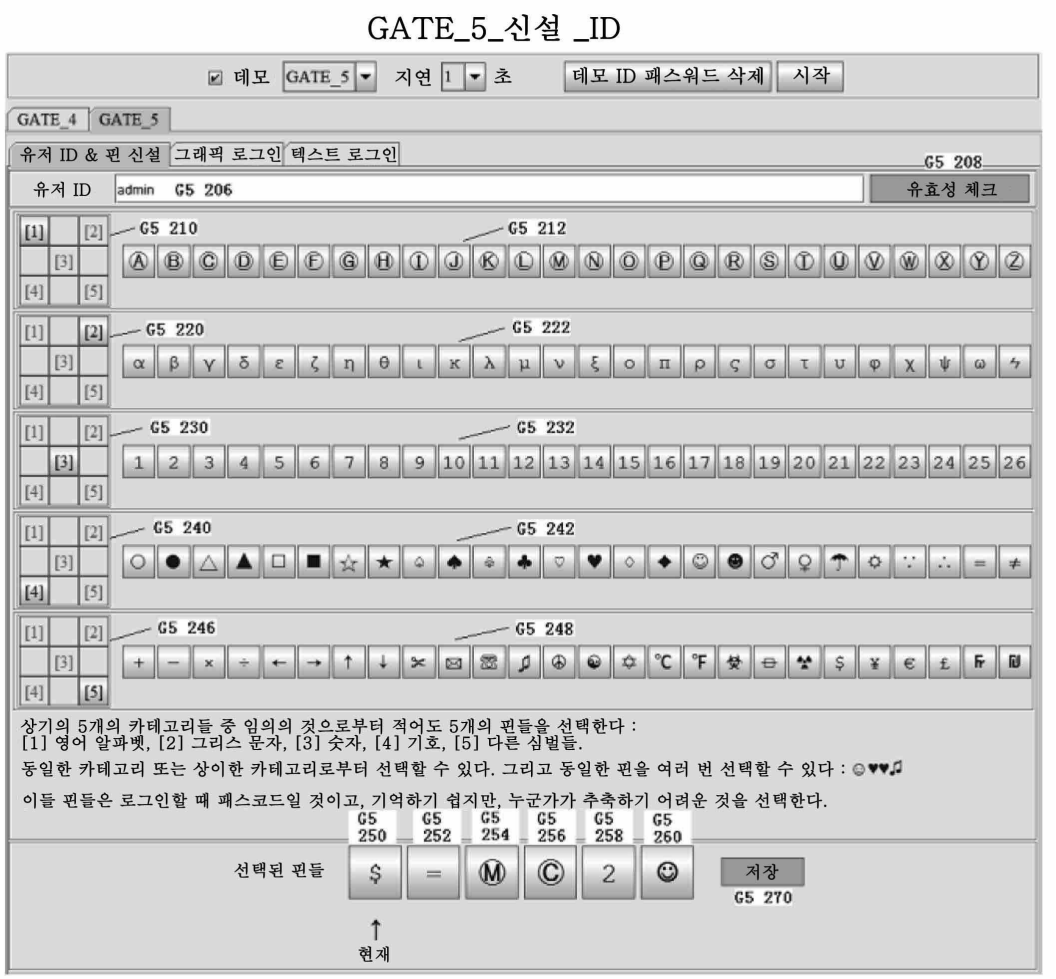
도면10d


GATE_4_텍스트_로그인
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G4 609
유저 ID
admin G4 606
Enter
클라이언트측
네트워크 연결
서버측
Id : admin G4 611
서버로 발송된 유저_ID : admin
G4 613
G4 617
ID [패스코드] : admin [ⓘ♥2✉]
G4 615
G4 621
G4 622
네트워크로부터 아래의 토큰들을 획득
네트워크로부터 아래의 토큰들을 획득
파일에 패스코드로부터 생성
[1] 28 Ⓥ ≠ ∞
[2] 19 Ⓟ ⍑ F
[3] 27 Ⓑ ♦ €
[4] 11 @ ★ −
[5] 26 Ⓐ ♡ +
[6] 1 ⓘ ♤ ※
[7] 2 : ☺ ✂
[8] 24 ^ ✕ ®
[9] 34 # ▲ ♫
[10] 6 Ⓜ ℉
[11] 12 % → ÷
[12] 16 Ⓝ = ☜
[13] 10 Ⓕ ☾ ✉
[14] 35 Ⓠ ☆ $
[15] 20 Ⓙ ← ₪
[16] 9 Ⓩ △
G4 625
G4 624
G4 623
[6] G4 626
유저 핀들을 함유하는 토큰들을 선택
클라이언트로부터 서버로 선택된 토큰들을 발송
네트워크로부터 선택된 토큰들을 획득
[1 ⓘ ♤ ※] 에서 찾아진 G4 352 [ⓘ]
누락:와일드카드로 대체됨 G4 355 [♥]
[2 : ☺ ✂] 에서 찾아진 G4 358 [2]
[10 Ⓕ ☾ ✉] 에서 찾아진 G4 361 [✉]
[35 Ⓠ ☆ $] 너무 많은 핀 G4 364 [_?_]
G4 352 [1 ⓘ ♤ ※]
G4 355 [11 @ ★ −] [와일드카드]
G4 358 [2 : ☺ ✂]
G4 361 [10 Ⓕ ☾ ✉]
G4 364 [35 Ⓠ ☆ $] [유효하지 않음]
G4 352 [1 ⓘ ♤ ※] C31
G4 355 [11 @ ★ −] C32
G4 358 [2 : ☺ ✂] C33
G4 361 [10 Ⓕ ☾ ✉] C34
G4 364 [35 Ⓠ ☆ $] C35
[7]
G4 627
[8]
G4 628
G4 632
G4 631
G4 630
G4 629
액세스 거부 !
클라이언트에 로그인 허용을 발송
클라이언트로부터의 토큰들을 확인
[11]
[10]
[9]

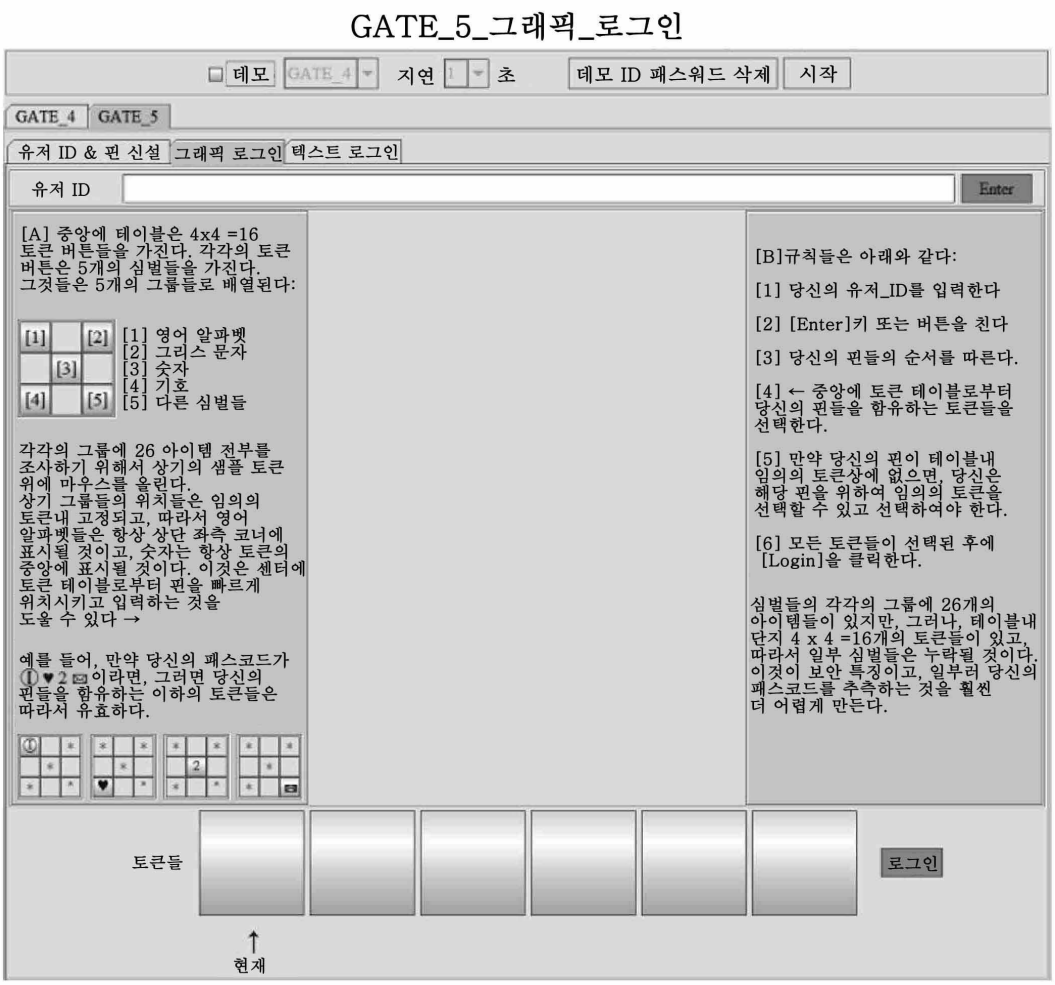
도면11a


GATE_5_ID_신설
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
유효성 체크
유저 ID를 입력한 후에, [Enter]키를 치거나 또는 유저ID가 유효한지를 조사하기 위해 [유효성 체크] 버튼을 클릭한다.
선택된 핀들
저장
현재

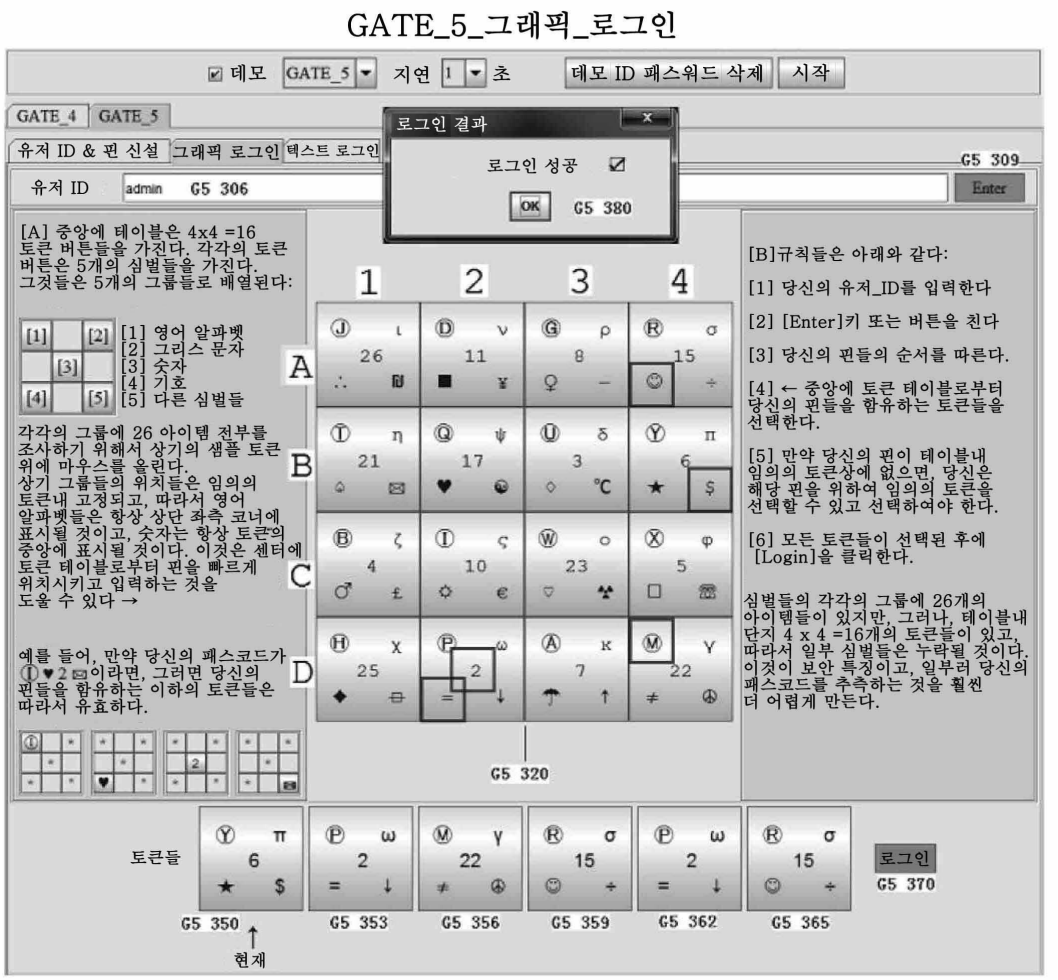
도면11b


GATE_5_신설 _ID
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G5 208
유저 ID
admin
G5 206
유효성 체크
G5 210
G5 212
G5 220
G5 222
G5 230
G5 232
G5 240
G5 242
G5 246
G5 248
상기의 5개의 카테고리들 중 임의의 것으로부터 적어도 5개의 핀들을 선택한다 :
[1] 영어 알파벳, [2] 그리스 문자, [3] 숫자, [4] 기호, [5] 다른 심벌들.
동일한 카테고리 또는 상이한 카테고리로부터 선택할 수 있다. 그리고 동일한 핀을 여러 번 선택할 수 있다 : ☺♥♥♫
이들 핀들은 로그인할 때 패스코드일 것이고, 기억하기 쉽지만, 누군가가 추측하기 어려운 것을 선택한다.
선택된 핀들
G5 250
G5 252
G5 254
G5 256
G5 258
G5 260
$
=
Ⓜ
©
2
☺
저장
G5 270
현재

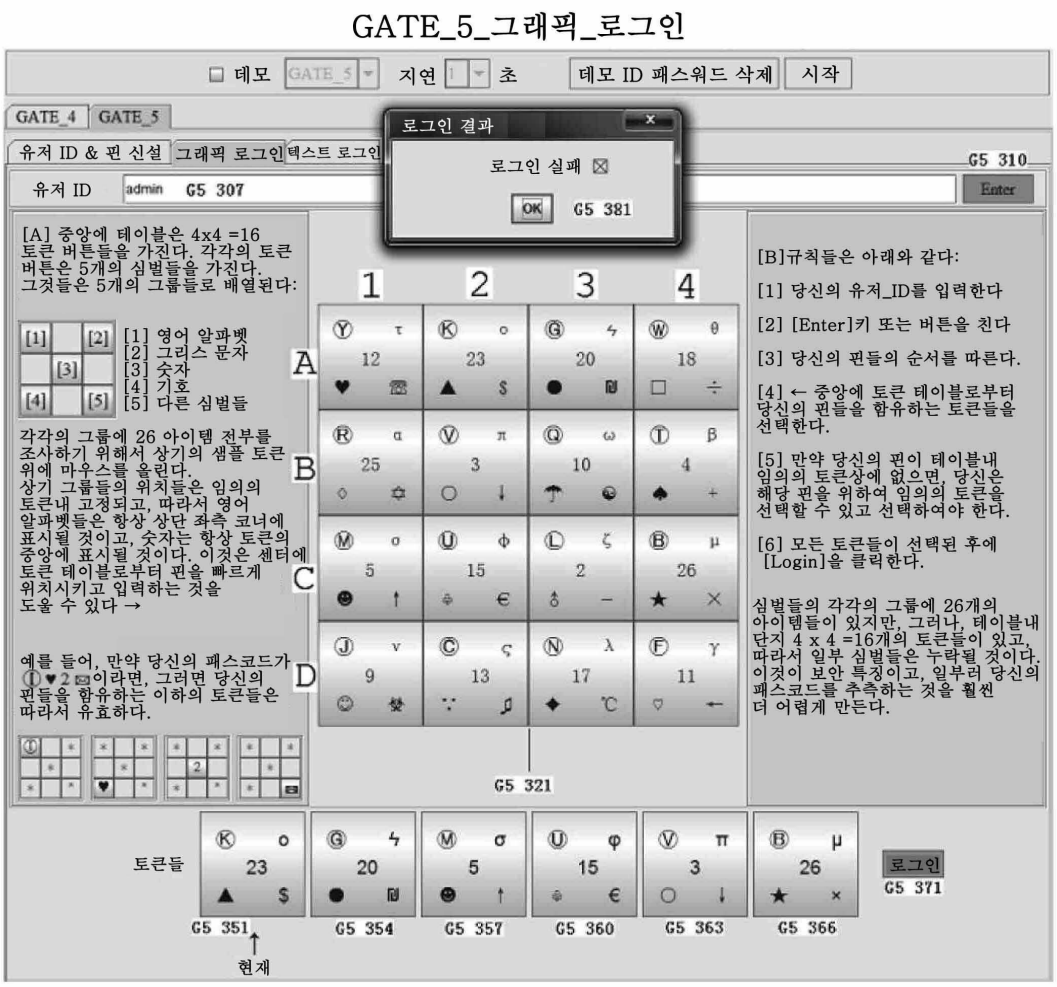
도면12a


GATE_5_그래픽_로그인
데모
GATE_4
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
Enter
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 5개의 심벌들을 가진다. 그것들은 5개의 그룹들로 배열된다:
[1] [2] [3] [4] [5]
[1] 영어 알파벳
[2] 그리스 문자
[3] 숫자
[4] 기호
[5] 다른 심벌들
각각의 그룹에 26 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 영어 알파벳들은 항상 상단 좌측 코너에 표시될 것이고, 숫자는 항상 토큰의 중앙에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
[B]규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 26개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
로그인
현재

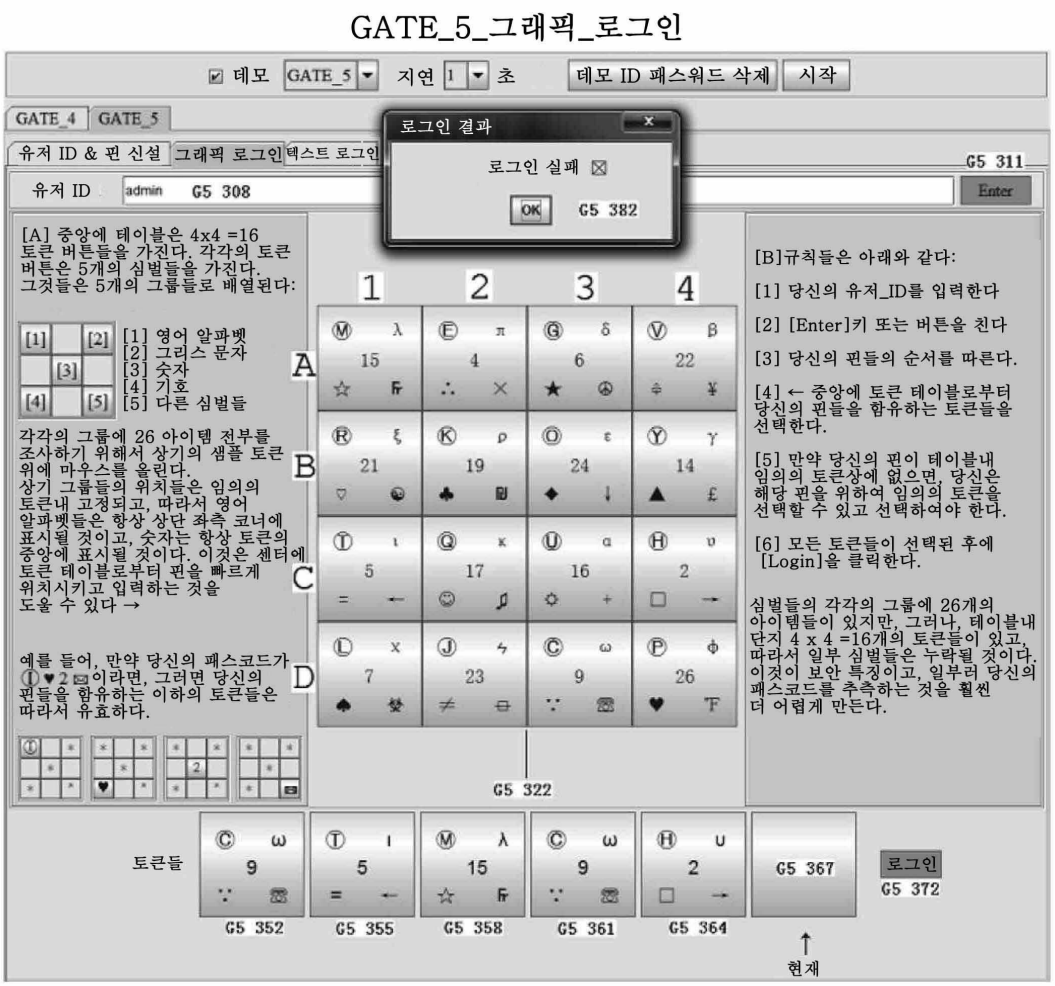
도면12b


GATE_5_그래픽_로그인
데모
GATE_5
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
로그인 결과
로그인 성공
OK
G5 380
G5 309
유저 ID
admin
G5 306
Enter
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 5개의 심벌들을 가진다. 그것들은 5개의 그룹들로 배열된다:
[1] 영어 알파벳
[2] 그리스 문자
[3] 숫자
[4] 기호
[5] 다른 심벌들
각각의 그룹에 26 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 영어 알파벳들은 항상 상단 좌측 코너에 표시될 것이고, 숫자는 항상 토큰의 중앙에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ⓘ♥2✉이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
1
2
3
4
A
B
C
D
G5 320
[B]규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 26개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
G5 350
현재
G5 353
G5 356
G5 359
G5 362
G5 365
로그인
G5 370

도면12c


GATE_5_그래픽_로그인
데모
GATE_5
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
로그인 결과
로그인 실패
OK
G5 381
G5 310
유저 ID
admin
G5 307
Enter
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 5개의 심벌들을 가진다. 그것들은 5개의 그룹들로 배열된다:
[1] 영어 알파벳
[2] 그리스 문자
[3] 숫자
[4] 기호
[5] 다른 심벌들
각각의 그룹에 26 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다.
상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 영어 알파벳들은 항상 상단 좌측 코너에 표시될 것이고, 숫자는 항상 토큰의 중앙에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ⓘ♥2✉이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
1
2
3
4
A
B
C
D
G5 321
[B]규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 26개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
로그인
G5 371
G5 351
G5 354
G5 357
G5 360
G5 363
G5 366
현재

도면12d


GATE_5_그래픽_로그인
데모
GATE_5
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G5 311
유저 ID
admin
G5 308
Enter
로그인 결과
로그인 실패
OK
G5 382
[A] 중앙에 테이블은 4x4 =16 토큰 버튼들을 가진다. 각각의 토큰 버튼은 5개의 심벌들을 가진다. 그것들은 5개의 그룹들로 배열된다:
[1] 영어 알파벳
[2] 그리스 문자
[3] 숫자
[4] 기호
[5] 다른 심벌들
각각의 그룹에 26 아이템 전부를 조사하기 위해서 상기의 샘플 토큰 위에 마우스를 올린다. 상기 그룹들의 위치들은 임의의 토큰내 고정되고, 따라서 영어 알파벳들은 항상 상단 좌측 코너에 표시될 것이고, 숫자는 항상 토큰의 중앙에 표시될 것이다. 이것은 센터에 토큰 테이블로부터 핀을 빠르게 위치시키고 입력하는 것을 도울 수 있다 →
예를 들어, 만약 당신의 패스코드가 ⓘ♥2🖂이라면, 그러면 당신의 핀들을 함유하는 이하의 토큰들은 따라서 유효하다.
1
2
3
4
A
B
C
D
G5 322
[B]규칙들은 아래와 같다:
[1] 당신의 유저_ID를 입력한다
[2] [Enter]키 또는 버튼을 친다
[3] 당신의 핀들의 순서를 따른다.
[4] ← 중앙에 토큰 테이블로부터 당신의 핀들을 함유하는 토큰들을 선택한다.
[5] 만약 당신의 핀이 테이블내 임의의 토큰상에 없으면, 당신은 해당 핀을 위하여 임의의 토큰을 선택할 수 있고 선택하여야 한다.
[6] 모든 토큰들이 선택된 후에 [Login]을 클릭한다.
심벌들의 각각의 그룹에 26개의 아이템들이 있지만, 그러나, 테이블내 단지 4 x 4 =16개의 토큰들이 있고, 따라서 일부 심벌들은 누락될 것이다. 이것이 보안 특징이고, 일부러 당신의 패스코드를 추측하는 것을 훨씬 더 어렵게 만든다.
토큰들
G5 352
G5 355
G5 358
G5 361
G5 364
G5 367
현재
로그인
G5 372

도면13a


GATE_5_텍스트_로그인
데모
GATE_4
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
유저 ID
Enter
클라이언트측
네트워크 연결
서버측
ID [패스코드] :
서버로 발송된 유저_ID :
ID [패스코드] :
클라이언트로 로그인 허용을 발송
클라이언트로부터의 토큰들을 확인

도면13b


GATE_5_텍스트_로그인
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G5 409
유저 ID
admin G5 406
Enter
클라이언트측
네트워크 연결
서버측
Id : admin G5 411
서버로 발송된 유저_ID : admin
G5 417
ID [패스코드] : admin [$ = Ⓜ Ⓒ 2 ☺]
G5 421
G5 413
G5 422
G5 415
네트워크로부터 아래의 토큰들을 획득
네트워크는 정보를 서버로부터 클라이언트 전달
파일에 패스코드로부터 생성
[1] Ⓙ ι 26 ∴ ₪
[2] Ⓓ ν 11 ■ ¥
[3] Ⓖ ρ 8 ♀ −
[4] Ⓡ σ 15 ☺ ÷
[5] Ⓣ η 21 ♤ ✉
[6] Ⓠ ψ 17 ♥ ☯
[7] Ⓤ δ 3 ◇ ℃
[8] Ⓨ π 6 ★ $
[9] Ⓑ ζ 4 ♂ £
[10] Ⓘ ς 10 ☼ €
[11] Ⓦ ο 23 ♡ ☢
[12] Ⓧ φ 5 □ ▩
[13] Ⓗ χ 25 ♦ ⊟
[14] Ⓟ ω 2 = ↓
[15] Ⓐ κ 7 ⯪ ↑
[16] Ⓜ γ 22 ≠ ☮
[5]
G5 425
[4]
G5 424
[3]
G5 423
[6] G5 426
유저 핀들을 함유하는 토큰들을 선택
클라이언트로부터 서버로 선택된 토큰들을 발송
네트워크로부터 선택된 토큰들을 획득
[Ⓨ π 6 ★ $] 에서 찾아진 G5 350 [$]
[Ⓟ ω 2 = ↓] 에서 찾아진 G5 353 [=]
[Ⓜ γ 22 ≠ ☮] 에서 찾아진 G5 356 [Ⓜ]
누락:와일드카드로 대체됨 G5 359 [Ⓒ]
[Ⓟ ω 2 = ↓] 에서 찾아진 G5 362 [2]
[Ⓡ σ 15 ☺ ÷] 에서 찾아진 G5 365 [☺]
G5 350 [Ⓨ π 6 ★ $]
G5 353 [Ⓟ ω 2 = ↓]
G5 356 [Ⓜ γ 22 ≠ ☮]
G5 359 [Ⓡ σ 15 ☺ ÷][와일드카드]
G5 362 [Ⓟ ω 2 = ↓]
G5 365 [Ⓡ σ 15 ☺ ÷]
G5 350 [Ⓨ π 6 ★ $] K11
G5 353 [Ⓟ ω 2 = ↓] K12
G5 356 [Ⓜ γ 22 ≠ ☮] K13
G5 359 [Ⓡ σ 15 ☺ ÷] K14
G5 362 [Ⓟ ω 2 = ↓] K15
G5 365 [Ⓡ σ 15 ☺ ÷] K16
G5 427
[7]
G5 428
[8]
G5 431
[11]
G5 430
[10]
G5 429
[9]
G5 432
액세스 승인 ^_^ !
클라이언트에 로그인 허용을 발송
클라이언트로부터의
토큰들을 확인

도면13c


GATE_5_텍스트_로그인
데모
GATE_5
지연
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G5 509
유저 ID
admin
G5 506
Enter
클라이언트측
네트워크 연결
서버측
Id : admin
G5 511
서버로 발송된 유저_ID : admin
G5 513
G5 517
ID [패스코드] : admin [$ = Ⓜ Ⓒ 2 ☺]
G5 515
[1]
[2]
G5 521
G5 522
네트워크로부터 아래의 토큰들을 획득
네트워크는 정보를 서버로부터 클라이언트 전달
파일에 패스코드로부터 생성
[1] Ⓨ τ 12 ♥ ☎
[2] Ⓚ o 23 ▲ $
[3] Ⓖ ϟ 20 ● ▣
[4] Ⓦ θ 18 □ ÷
[5] Ⓡ α 25 ◇ ✡
[6] Ⓥ π 3 ○ ↓
[7] Ⓠ ω 10 ☂ ☯
[8] Ⓣ β 4 ♠ +
[9] Ⓜ σ 5 ☻ ↑
[10] Ⓤ φ 15 ♤ €
[11] Ⓛ ζ 2 ♂ −
[12] Ⓑ μ 26 ★ ×
[13] Ⓙ v 9 ☺ ♀
[14] Ⓒ ς 13 ∵ ♬
[15] Ⓝ λ 17 ♦ ℃
[16] Ⓕ γ 11 ♡ ←
[5]
G5 525
[4]
G5 524
[3]
G5 523
[6] G5 526
유저 핀들을 함유하는 토큰들을 선택
클라이언트로부터 서버로 선택된 토큰들을 발송
네트워크로부터 선택된 토큰들을 획득
[Ⓚ o 23 ▲ $] 에서 찾아진 G5 351 [$]
누락:와일드카드로 대체됨G5 354 [=]
[Ⓜ σ 5 ● ↑] 에서 찾아진 G5 357[Ⓜ]
[Ⓤ φ 15 ♤ €] 에서 찾아지지 않은G5 360[Ⓒ]
[Ⓥ π 3 ○ ↓] 에서 찾아지지 않은G5 363 [2]
[Ⓑ μ 26 ★ ×] 에서 찾아지지 않은G5 366[☺]
G5 351[Ⓚ o 23 ▲ $]
G5 354[Ⓖ ϟ 20 ● ▣]
[와일드카드]
G5 357[Ⓜ σ 5 ● ↑]
G5 360[Ⓤ φ 15 ♤ €]
G5 363[Ⓥ π 3 ○ ↓]
G5 366[Ⓑ μ 26 ★ ×]
G5 351[Ⓚ o 23 ▲ $]
G5 354[Ⓖ ϟ 20 ● ▣]
G5 357[Ⓜ σ 5 ● ↑]
G5 360[Ⓤ φ 15 ♤ €]
G5 363[Ⓥ π 3 ○ ↓]
G5 366[Ⓑ μ 26 ★ ×]
K21
K22
K23
K24
K25
K26
[7]
G5 527
[8]
G5 528
G5 531
G5 530
G5 529
G5 532
액세스 거부 !
[11]
클라이언트에 로그인 허용을 발송
[10]
클라이언트로부터의
토큰들을 확인
[9]

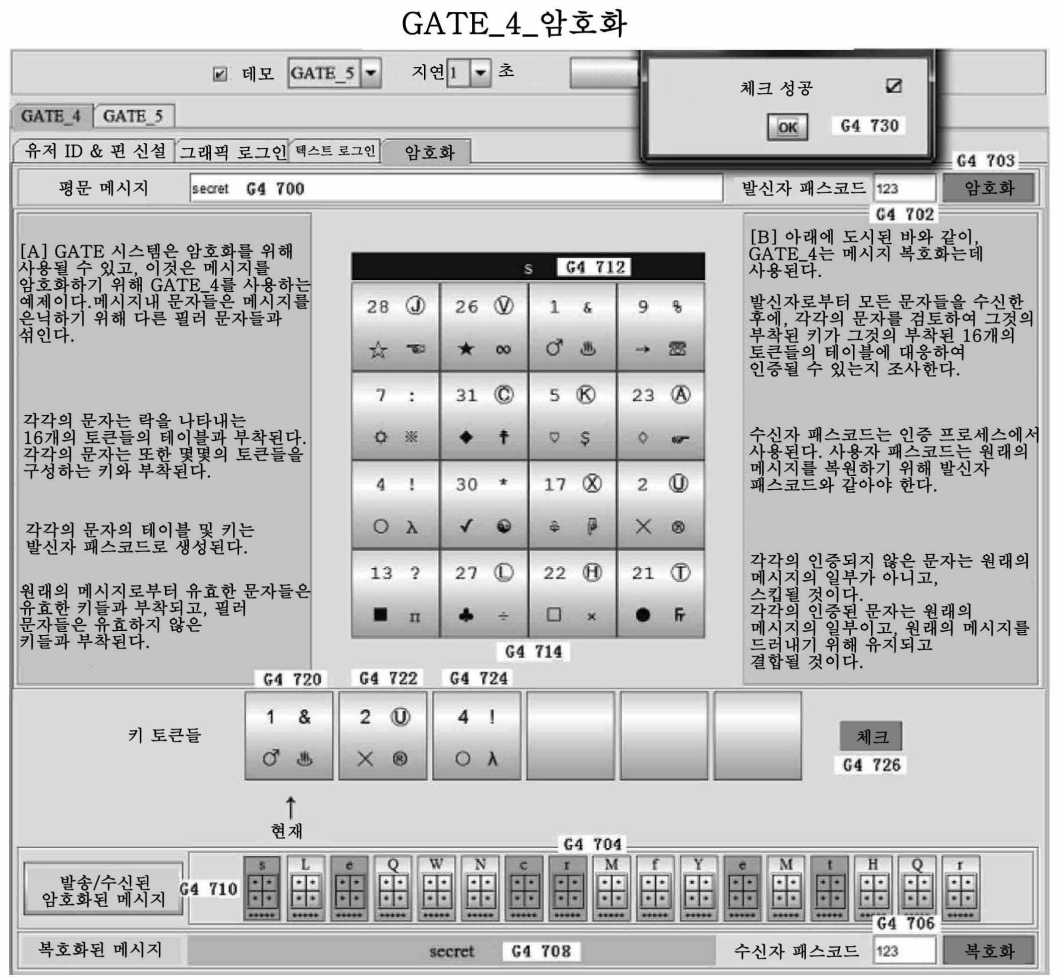
도면13d


GATE_5_텍스트_로그인
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
G5 609
유저 ID
admin
G5 606
Enter
클라이언트측
네트워크 연결
서버측
Id : admin G5 611
서버로 발송된 유저_ID : admin
G5 613
ID [패스코드] : admin [$ = Ⓜ Ⓒ 2 ☺]
G5 615
G5 617
G5 621
G5 622
네트워크로부터 아래의 토큰들을 획득
네트워크는 정보를 서버로부터 클라이언트 전달
파일에 패스코드로부터 생성
[1] Ⓜ λ 15 ☆ ₣
[2] Ⓔ π 4 ∴ ×
[3] Ⓖ δ 6 ★ ☮
[4] Ⓥ β 22 ♤ ¥
[5] Ⓡ ξ 21 ♡ ☯
[6] Ⓚ ρ 19 ♣ ♲
[7] Ⓞ ε 24 ♦ ↓
[8] Ⓨ γ 14 ▲ £
[9] Ⓣ ι 5 = ←
[10] Ⓠ κ 17 ☺ ♫
[11] Ⓤ α 16 ☼ +
[12] Ⓗ υ 2 □ →
[13] Ⓛ χ 7 ♠ ♀
[14] Ⓙ ϟ 23 ≠ ♾
[15] Ⓒ ω 9 ∵ ☎
[16] Ⓟ φ 26 ♥ ₸
[5] G5 625
[4] G5 624
[3] G5 623
[6] G5 626
유저 핀들을 함유하는 토큰들을 선택
클라이언트로부터 서버로 선택된 토큰들을 발송
네트워크로부터 선택된 토큰들을 획득
누락:와일드카드로 대체됨 G5 352 [$]
[Ⓣ ι 5 = ←] 에서 찾아진 G5 355 [=]
[Ⓜ λ 15 ☆ ₣] 에서 찾아진 G5 358 [Ⓜ]
[Ⓒ ω 9 ∵ ☎]에서 찾아진G5 361 [Ⓒ]
[Ⓗ υ 2 □ →] 에서 찾아진 G5 364 [2]
입력 누락 G5 367 [☺]
G5 352 [Ⓒ ω 9 ∵ ☎] [와일드카드]
G5 355 [Ⓣ ι 5 = ←]
G5 358 [Ⓜ λ 15 ☆ ₣]
G5 361 [Ⓒ ω 9 ∵ ☎]
G5 364 [Ⓗ υ 2 □ →]
G5 367 핀 누락
[7] G5 627
[8] G5 628
G5 352 [Ⓒ ω 9 ∵ ☎]
G5 355 [Ⓣ ι 5 = ←]
G5 358 [Ⓜ λ 15 ☆ ₣]
G5 361 [Ⓒ ω 9 ∵ ☎]
G5 364 [Ⓗ υ 2 □ →]
G5 367 ?
K31
K32
K33
K34
K35
K36
G5 629
G5 631
G5 630
G5 632 액세스 거부 !
클라이언트에 로그인 허용을 발송
클라이언트로부터의 토큰들을 확인
[11]
[10]
[9]

도면14


GATE_4_암호화
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G4 702
G4 703
평문 메시지
secret G4 700
발신자 패스코드
123
암호화
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_4를 사용하는 예제이다.메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
[B] 아래에 도시된 바와 같이, GATE_4는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다.
각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
키 토큰들
체크
현재
G4 704
발송/수신된 암호화된 메시지
s L e Q W N c r M f Y e M t H Q r
G4 706
복호화된 메시지
수신자 패스코드
123
복호화

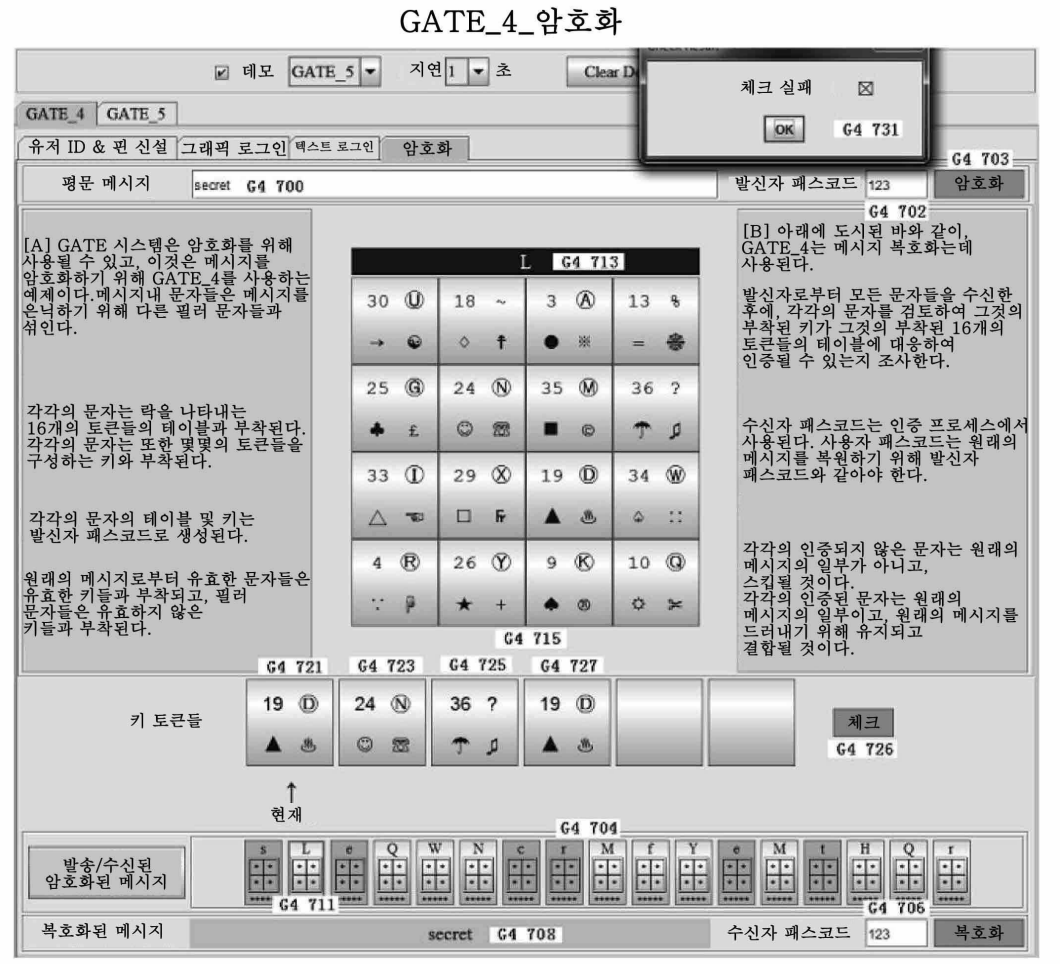
도면15a


GATE_4_암호화
데모
GATE_5
지연 1 초
체크 성공
OK
G4 730
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G4 703
평문 메시지
secret G4 700
발신자 패스코드 123
암호화
G4 702
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_4를 사용하는 예제이다.메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
G4 712
G4 714
[B] 아래에 도시된 바와 같이, GATE_4는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다.
각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
G4 720
G4 722
G4 724
키 토큰들
체크
G4 726
현재
G4 704
발송/수신된 암호화된 메시지
G4 710
G4 706
복호화된 메시지
secret G4 708
수신자 패스코드 123
복호화

도면15b


GATE_4_암호화
데모
GATE_5
지연 1 초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
메시지
G4 750
Enter
수신자측
네트워크 연결
발신자측
패스코드 : 123 G4 760
메시지와 함께 발송된 토큰들
패스코드 : 123 G4 752
네트워크로부터 이하의 토큰들을 획득
네트워크는 발신자로부터 수신자로 정보를 전달
파일에 패스코드로부터 생성
[1] [28, Ⓙ, ☆, ☜]
[2] [26, Ⓥ, ★, ∞]
[3] [1, &, ♂, ♨]
[4] [9, %, →, ☎]
[5] [7, ;, ☼, ※]
[6] [31, Ⓒ, ♦, †]
[7] [5, Ⓚ, ▽, $]
[8] [23, Ⓐ, ◇, ☞]
[9] [4, !, ○, λ]
[10] [30, *, ✓, ☯]
[11] [17, Ⓧ, ♤, ♕]
[12] [2, Ⓤ, ×, ®]
[13] [13, ?, ■, π]
[14] [27, Ⓛ, ♣, ÷]
[15] [22, Ⓗ, □, ×]
[16] [21, Ⓣ, ●, F]
[3] G4 758
[2] G4 756
[1] G4 754
[4] G4 762
키 토큰들을 체크
수신자측 키 토큰들 결과
수신자측 키 토큰들 확인
[1 & ♂ ♨] 에서 찾아진 G4 720 [1]
[2 Ⓤ × ®] 에서 찾아진 G4 722 [2]
누락:와일드카드로 대체됨 G4 724 [3]
G4 720 [1 & ♂ ♨]
G4 722 [2 Ⓤ × ®]
G4 724 [4 ! ○ λ] [와일드카드]
G4 720 [1 & ♂ ♨] C51
G4 722 [2 Ⓤ × ®] C52
G4 724 [4 ! ○ λ] C53
G4 764
[5]
G4 766
[6]
G4 768
[7]
G4 770 메시지 유효 ^_^ !

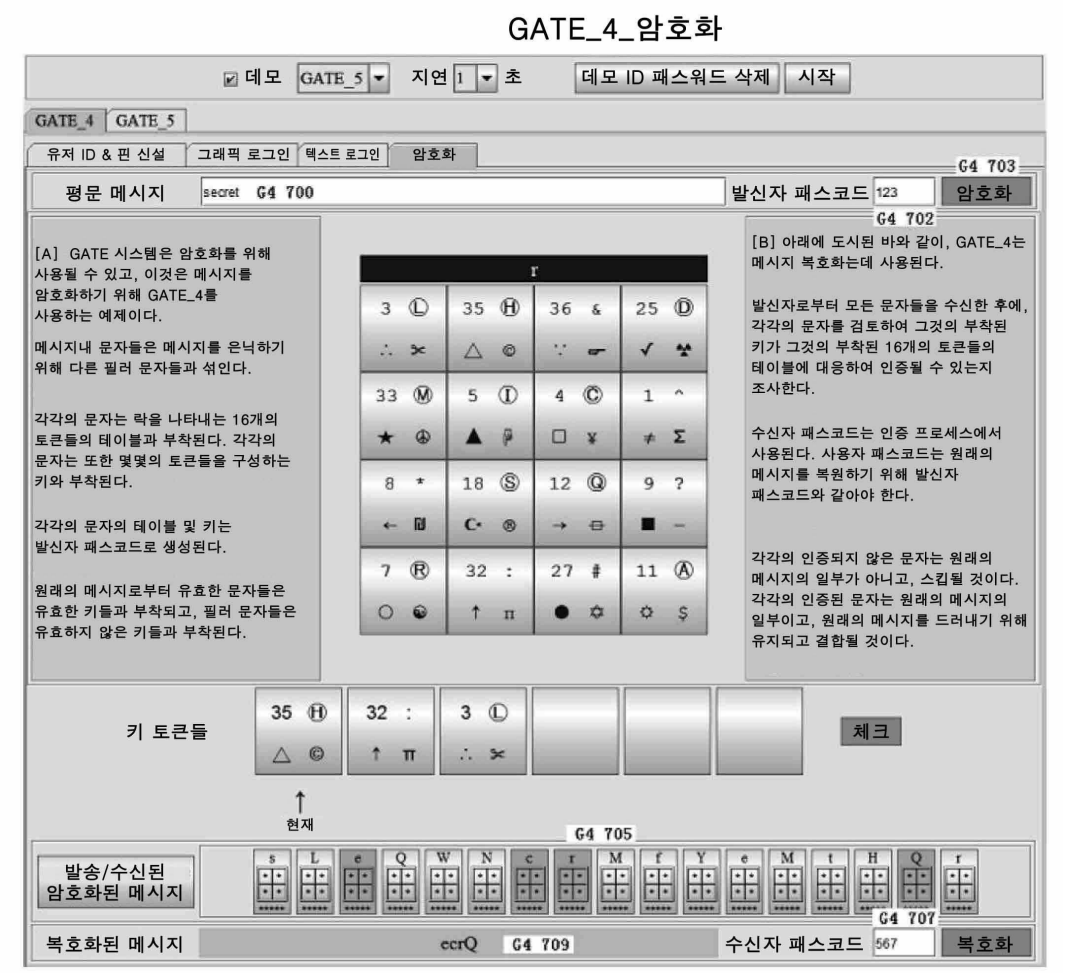
도면16a


GATE_4_암호화
데모
GATE_5
지연 1 초
Clear D
체크 실패
OK
G4 731
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G4 703
평문 메시지
secret G4 700
발신자 패스코드
123
암호화
G4 702
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_4를 사용하는 예제이다.메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
L G4 713
G4 715
[B] 아래에 도시된 바와 같이, GATE_4는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다.
각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
G4 721
G4 723
G4 725
G4 727
키 토큰들
체크
G4 726
현재
G4 704
발송/수신된 암호화된 메시지
G4 711
G4 706
복호화된 메시지
secret G4 708
수신자 패스코드
123
복호화

도면16b


GATE_4_암호화
데모
GATE_5
지연 1 초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
메시지
G4 751
Enter
수신자측
네트워크 연결
발신자측
패스코드 : 123 G4 760
메시지와 함께 발송된 토큰들
패스코드 : 123 G4 752
네트워크로부터 이하의 토큰들을 획득
네트워크는 발신자로부터 수신자로 정보를 전달
파일에 패스코드로부터 생성
G4 790
G4 792
[3] G4 759
[2] G4 757
[1] G4 755
[4] G4 763
키 토큰들을 체크
누락:와일드카드로 대체됨G4 721 [1]
누락:와일드카드로 대체됨G4 723 [2]
[36 ? ☂ ♫]에서 찾아지지 않음 G4 725 [3]
너무 많은 핀[19 Ⓓ ▲ ♨] G4 727[_?_]
수신자측 키 토큰들 결과
G4 765
[5]
G4 721[19 Ⓓ ▲ ♨][와일드카드]
G4 723[24 Ⓝ ☺ ☎][와일드카드]
G4 767
G4 725[36 ? ☂ ♫]
[6]
G4 727[19 Ⓓ ▲ ♨] [유효하지 않음]
수신자측 키 토큰들 확인
G4 721[19 Ⓓ ▲ ♨] C61
G4 723[24 Ⓝ ☺ ☎] C62
G4 725[36 ? ☂ ♫] C63
G4 727[19 Ⓓ ▲ ♨] C64
G4 769
[7]
G4 771 메시지 유효하지않음 !

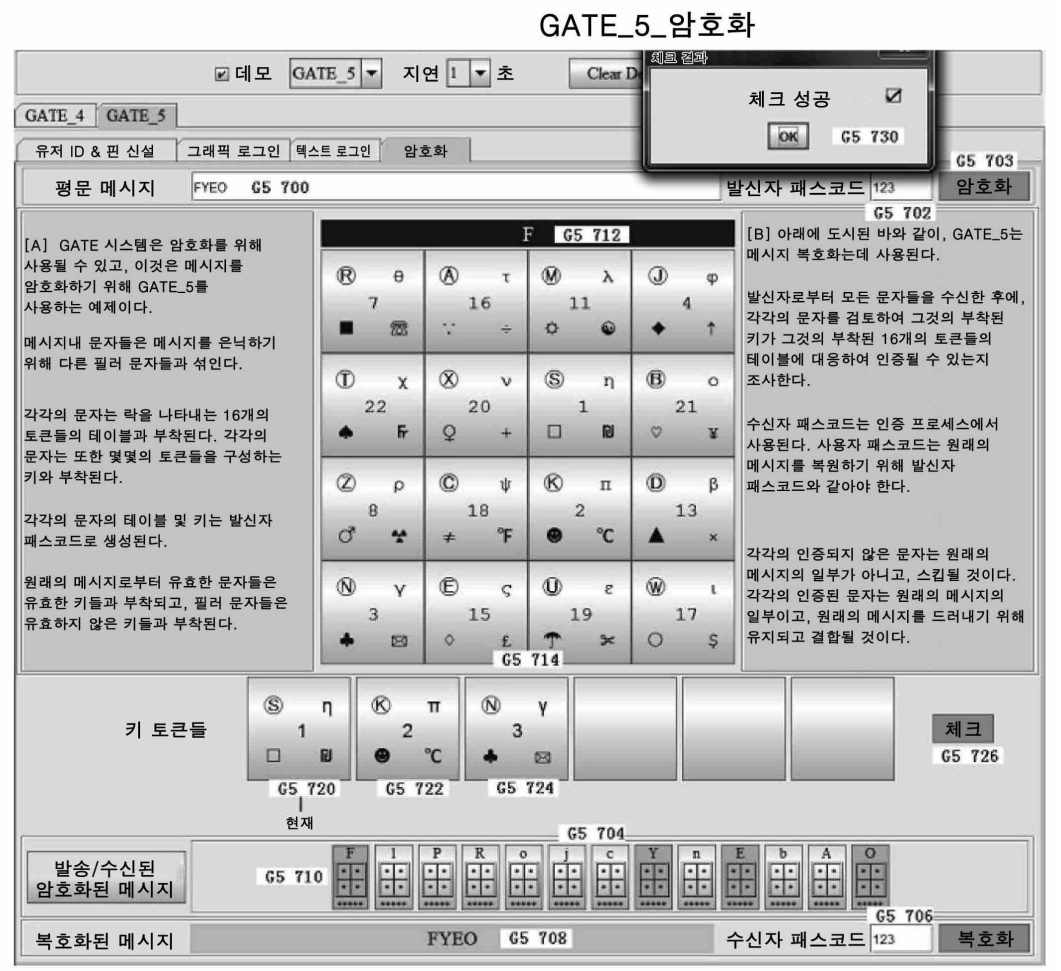
도면17


GATE_4_암호화
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G4 703
평문 메시지
secret G4 700
발신자 패스코드
123
암호화
G4 702
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_4를 사용하는 예제이다.
메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
[B] 아래에 도시된 바와 같이, GATE_4는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다. 각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
키 토큰들
체크
현재
G4 705
발송/수신된 암호화된 메시지
G4 707
복호화된 메시지
ecrQ
G4 709
수신자 패스코드
567
복호화

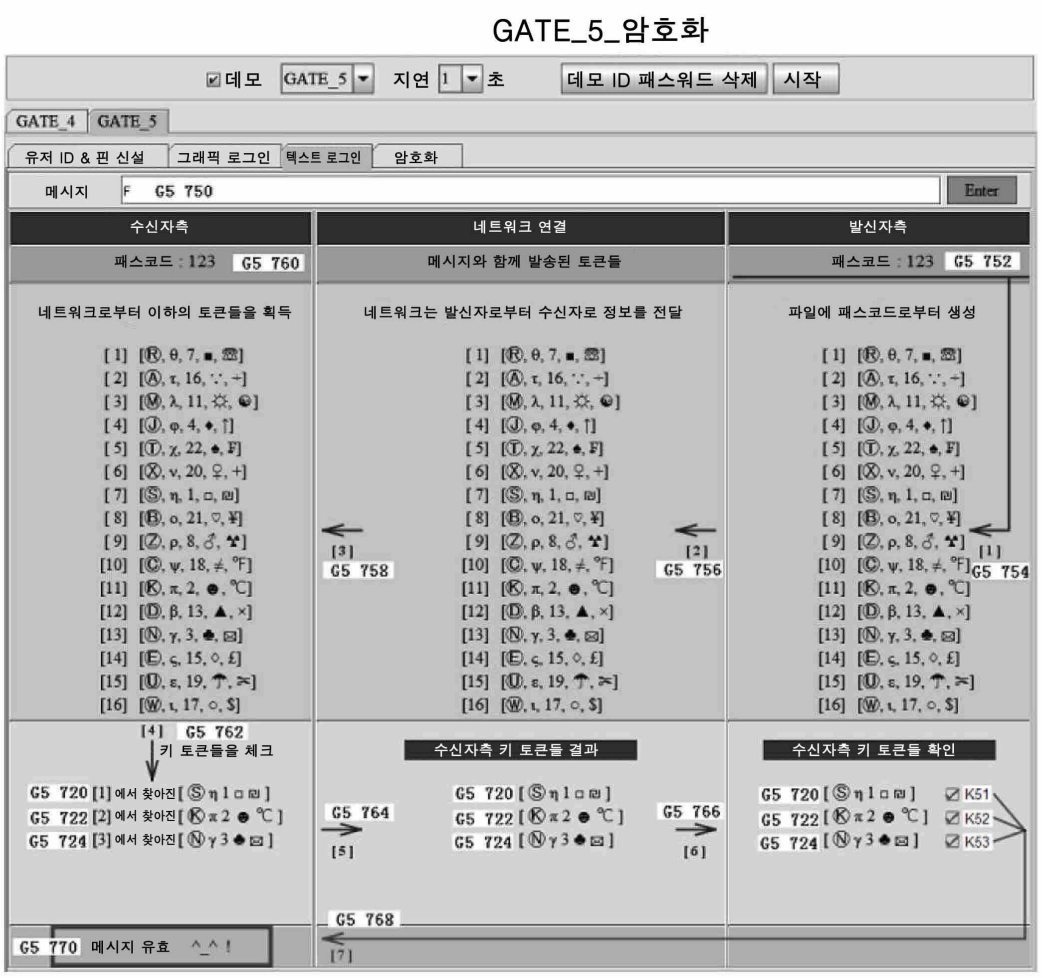
도면18


GATE_5_암호화
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G5 702
G5 703
평문 메시지
FYEO
G5 700
발신자 패스코드
123
암호화
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_5를 사용하는 예제이다.
메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
[B] 아래에 도시된 바와 같이, GATE_5는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다. 각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
키 토큰들
체크
현재
G5 704
발송/수신된 암호화된 메시지
F
l
P
R
o
j
c
Y
n
E
b
A
O
G5 706
복호화된 메시지
수신자 패스코드
123
복호화

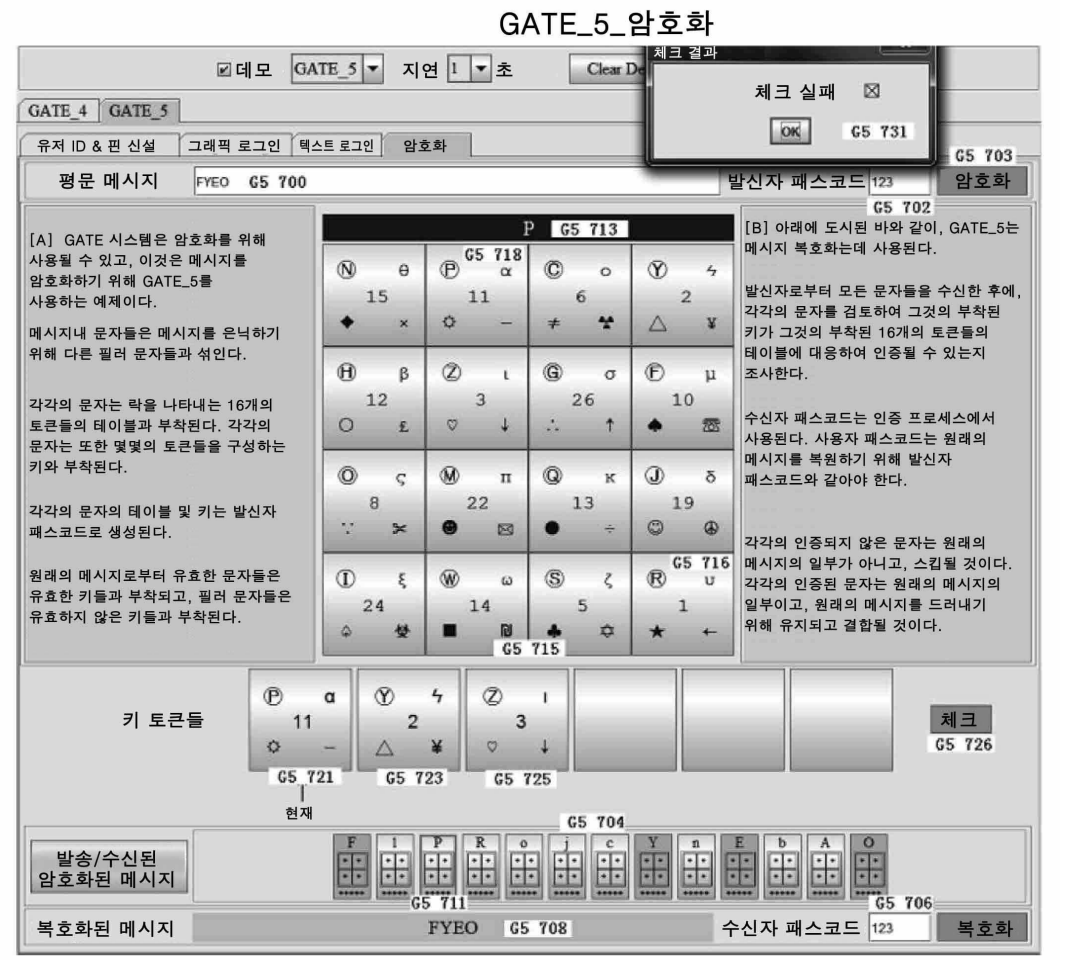
도면19a


GATE_5_암호화
데모
GATE_5
지연
초
Clear D
체크 컴팩
체크 성공
OK
G5 730
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G5 703
평문 메시지
FYEO G5 700
발신자 패스코드
123
암호화
G5 702
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_5를 사용하는 예제이다.
메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
F G5 712
G5 714
[B] 아래에 도시된 바와 같이, GATE_5는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다. 각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
키 토큰들
체크
G5 726
G5 720
G5 722
G5 724
현재
G5 704
발송/수신된 암호화된 메시지
G5 710
G5 706
복호화된 메시지
FYEO G5 708
수신자 패스코드
123
복호화

도면19b


GATE_5_암호화
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
메시지
F G5 750
Enter
수신자측
네트워크 연결
발신자측
패스코드 : 123
G5 760
메시지와 함께 발송된 토큰들
패스코드 : 123
G5 752
네트워크로부터 이하의 토큰들을 획득
네트워크는 발신자로부터 수신자로 정보를 전달
파일에 패스코드로부터 생성
[1] [Ⓡ, θ, 7, ■, ☎]
[2] [Ⓐ, τ, 16, ∵, +]
[3] [Ⓜ, λ, 11, ☼, ☯]
[4] [Ⓙ, φ, 4, ♦, ↑]
[5] [Ⓣ, χ, 22, ♠, ₣]
[6] [Ⓧ, ν, 20, ♀, +]
[7] [Ⓢ, η, 1, □, ₪]
[8] [Ⓑ, ο, 21, ▽, ¥]
[9] [Ⓩ, ρ, 8, ♂, ☢]
[10] [Ⓒ, ψ, 18, ≠, ℉]
[11] [Ⓚ, π, 2, ●, ℃]
[12] [Ⓓ, β, 13, ▲, ×]
[13] [Ⓝ, γ, 3, ♣, ✉]
[14] [Ⓔ, ς, 15, ◇, £]
[15] [Ⓤ, ε, 19, ☂, ✂]
[16] [Ⓦ, ι, 17, ○, $]
[3]
G5 758
[2]
G5 756
[1]
G5 754
[4] G5 762
키 토큰들을 체크
수신자측 키 토큰들 결과
수신자측 키 토큰들 확인
G5 720 [1] 에서 찾아진 [Ⓢ η 1 □ ₪]
G5 722 [2] 에서 찾아진 [Ⓚ π 2 ● ℃]
G5 724 [3] 에서 찾아진 [Ⓝ γ 3 ♣ ✉]
G5 720 [Ⓢ η 1 □ ₪]
G5 722 [Ⓚ π 2 ● ℃]
G5 724 [Ⓝ γ 3 ♣ ✉]
G5 720 [Ⓢ η 1 □ ₪] K51
G5 722 [Ⓚ π 2 ● ℃] K52
G5 724 [Ⓝ γ 3 ♣ ✉] K53
G5 764
[5]
G5 766
[6]
G5 768
[7]
G5 770 메시지 유효 ^_^ !

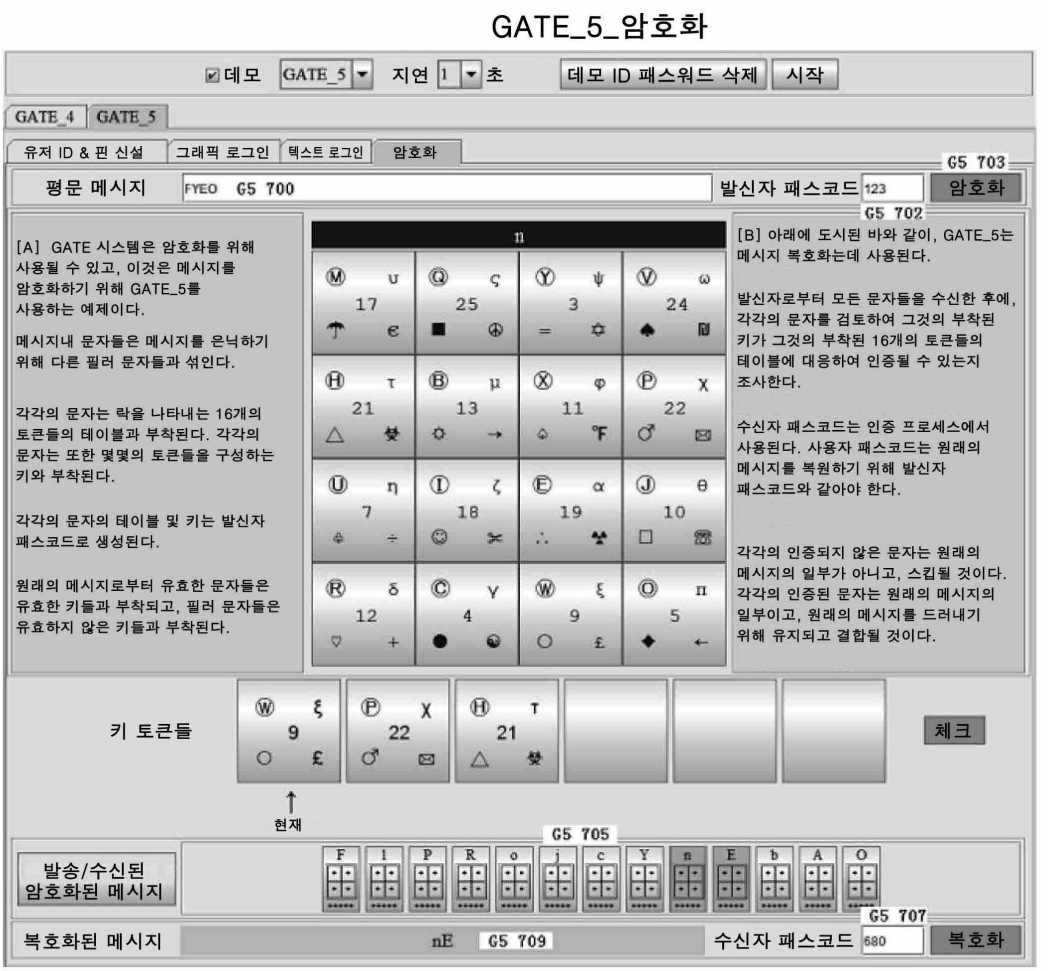
도면20a


GATE_5_암호화
데모
GATE_5
지연
1
초
Clear De
체크 결과
체크 실패
OK
G5 731
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G5 703
평문 메시지
FYEO G5 700
발신자 패스코드
123
암호화
G5 702
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_5를 사용하는 예제이다.
메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
P G5 713
G5 718
G5 716
G5 715
[B] 아래에 도시된 바와 같이, GATE_5는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다. 각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
키 토큰들
G5 721
현재
G5 723
G5 725
체크
G5 726
G5 704
발송/수신된 암호화된 메시지
G5 711
G5 706
복호화된 메시지
FYEO G5 708
수신자 패스코드
123
복호화

도면20b


GATE_5_암호화
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
메시지
P G5 751
Enter
수신자측
네트워크 연결
발신자측
패스코드 : 123 G5 760
메시지와 함께 발송된 토큰들
패스코드 : 123 G5 752
네트워크로부터 이하의 토큰들을 획득
네트워크는 발신자로부터 수신자로 정보를 전달
파일에 패스코드로부터 생성
G5 792
G5 790
[3]
G5 759
[2]
G5 757
[1]
G5 755
[4] G5 763
키 토큰들을 체크
G5 721 [1]에서 찾아지지 않은[Ⓟ α 11 ☼ –]
G5 723 [2] 에서 찾아진[Ⓨ ϟ 2 △ ¥]
G5 725 [3] 에서 찾아진[Ⓩ ι 3 ♡ ↓]
수신자측 키 토큰들 결과
G5 765
[5]
G5 767
[6]
수신자측 키 토큰들 확인
K61
K62
K63
G5 769
[7]
G5 771 메시지 유효하지 않음 !

도면21


GATE_5_암호화
데모
GATE_5
지연
1
초
데모 ID 패스워드 삭제
시작
GATE_4
GATE_5
유저 ID & 핀 신설
그래픽 로그인
텍스트 로그인
암호화
G5 703
평문 메시지
FYEO G5 700
발신자 패스코드
123
암호화
G5 702
[A] GATE 시스템은 암호화를 위해 사용될 수 있고, 이것은 메시지를 암호화하기 위해 GATE_5를 사용하는 예제이다.
메시지내 문자들은 메시지를 은닉하기 위해 다른 필러 문자들과 섞인다.
각각의 문자는 락을 나타내는 16개의 토큰들의 테이블과 부착된다. 각각의 문자는 또한 몇몇의 토큰들을 구성하는 키와 부착된다.
각각의 문자의 테이블 및 키는 발신자 패스코드로 생성된다.
원래의 메시지로부터 유효한 문자들은 유효한 키들과 부착되고, 필러 문자들은 유효하지 않은 키들과 부착된다.
[B] 아래에 도시된 바와 같이, GATE_5는 메시지 복호화는데 사용된다.
발신자로부터 모든 문자들을 수신한 후에, 각각의 문자를 검토하여 그것의 부착된 키가 그것의 부착된 16개의 토큰들의 테이블에 대응하여 인증될 수 있는지 조사한다.
수신자 패스코드는 인증 프로세스에서 사용된다. 사용자 패스코드는 원래의 메시지를 복원하기 위해 발신자 패스코드와 같아야 한다.
각각의 인증되지 않은 문자는 원래의 메시지의 일부가 아니고, 스킵될 것이다. 각각의 인증된 문자는 원래의 메시지의 일부이고, 원래의 메시지를 드러내기 위해 유지되고 결합될 것이다.
키 토큰들
체크
현재
G5 705
발송/수신된 암호화된 메시지
G5 707
복호화된 메시지
nE
G5 709
수신자 패스코드
680
복호화