(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2010년07월09일
(11) 등록번호 10-0969196
(24) 등록일자 2010년07월02일

(51) Int. Cl.
G06F 15/00 (2006.01) *G06F 17/00* (2006.01)
G06F 21/00 (2006.01)
(21) 출원번호 10-2007-0100637
(22) 출원일자 2007년10월05일
심사청구일자 2007년10월05일
(65) 공개번호 10-2009-0035381
(43) 공개일자 2009년04월09일
(56) 선행기술조사문헌
JP2004023662 A
JP10340255 A
KR1020040001829 A
JP2005208996 A
전체 청구항 수 : 총 20 항

(73) 특허권자
인하대학교 산학협력단
인천 남구 용현동 253 인하대학교
(72) 발명자
양대헌
서울 서초구 서초동 삼풍 ATP 3동 405호
맹영재
서울시 광진구 자양1동 226-12호
강전일
충남 공주시 옥룡동 284-1번지
(74) 대리인
특허법인이지

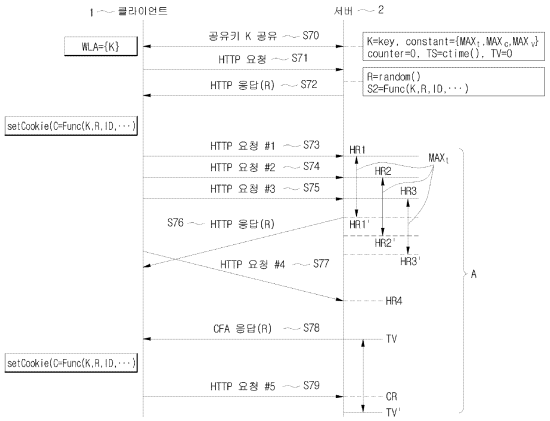
심사관 : 이종익

(54) 웹 환경에서의 안전한 사용자 세션 관리 방법 및 이를수행하는 프로그램이 기록된 기록매체

(57) 요 약

웹 환경에서의 안전한 사용자 세션 관리 방법이 개시된다. 사용자 세션 관리 방법은, 상기 서버가 이전에 생성한 난수와 상기 서버에 저장된 공유키를 이용하여 제1 서버 인증값을 계산하는 단계; 난수를 새로 생성하는 단계; 상기 난수와 상기 서버에 저장된 공유키를 이용하여 제2 서버 인증값을 계산하는 단계; 상기 난수를 포함하는 HTTP 응답(HTTP response)을 상기 클라이언트에 전송하는 단계; 상기 클라이언트로부터 쿠키(cookie)를 포함하는 HTTP 요청(HTTP request)을 전송받는 단계-여기서, 상기 쿠키는 클라이언트 인증값을 포함하고, 상기 클라이언트 인증값은 상기 클라이언트에 저장된 공유키와 상기 클라이언트가 전송받은 HTTP 응답에 포함된 난수를 이용하여 계산됨-; 및 상기 제1 서버 인증값 및 상기 제2 서버 인증값 중 어느 하나와 상기 클라이언트 인증값을 비교하여 상기 클라이언트의 송신자 인증의 성공 또는 실패를 결정하는 단계를 포함한다. 사용자의 행동패턴이나 네트워크 상황에 의한 오류를 허용하고 시간 유연한 송신자 인증이 가능하다.

대 표 도



특허청구의 범위

청구항 1

웹 환경에서 네트워크를 통해 클라이언트와 연결된 서버에서 상기 클라이언트와의 사용자 세션을 관리하는 방법에 있어서,

상기 서버가 이전에 생성한 제1 난수와 상기 서버에 저장된 공유키를 이용하여 제1 서버 인증값을 계산하는 단계;

제2 난수를 새로 생성하는 단계;

상기 제2 난수와 상기 서버에 저장된 공유키를 이용하여 제2 서버 인증값을 계산하는 단계;

상기 제2 난수를 포함하는 HTTP 응답(HTTP response)을 상기 클라이언트에 전송하는 단계;

상기 클라이언트로부터 쿠키(cookie)를 포함하는 HTTP 요청(HTTP request)을 전송받는 단계-여기서, 상기 쿠키는 클라이언트 인증값을 포함하고, 상기 클라이언트 인증값은 상기 클라이언트에 저장된 공유키와 상기 클라이언트가 전송받은 HTTP 응답에 포함된 난수를 이용하여 계산됨-; 및

상기 제1 서버 인증값 및 상기 제2 서버 인증값 중 어느 하나와 상기 클라이언트 인증값을 비교하여 상기 클라이언트의 송신자 인증의 성공 또는 실패를 결정하는 단계를 포함하는 사용자 세션 관리 방법.

청구항 2

제1항에 있어서,

상기 서버는 상기 클라이언트로부터의 사용자 로그인 과정에서 상기 공유키를 상기 클라이언트와 공유하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 3

제1항에 있어서,

상기 결정 단계는,

상기 클라이언트 인증값과 상기 제1 서버 인증값이 동일하고 상기 HTTP 요청의 도착 시점이 직전 HTTP 요청의 도착 시점으로부터 소정 시간 경과 이전인 경우 상기 송신자 인증이 성공한 것으로 결정하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 4

제3항에 있어서,

상기 송신자 인증이 성공한 경우 상기 제2 서버 인증값의 계산에 이용된 난수를 HTTP 응답에 포함시켜 상기 클라이언트에 전송하는 단계를 더 포함하는 사용자 세션 관리 방법.

청구항 5

제1항에 있어서,

상기 결정 단계는,

상기 클라이언트 인증값과 상기 제2 서버 인증값이 동일한 경우 상기 송신자 인증이 성공한 것으로 결정하되,

제3 난수를 새로 생성하는 단계;

상기 제2 서버 인증값을 상기 제1 서버 인증값에 복사하는 단계;

상기 제3 난수 및 상기 서버에 저장된 공유키를 이용하여 상기 제2 서버 인증값을 갱신하는 단계; 및

상기 제3 난수를 HTTP 응답에 포함시켜 상기 클라이언트에 전송하는 단계를 더 포함하는 사용자 세션 관리 방법.

청구항 6

제1항에 있어서,

상기 결정 단계는,

상기 클라이언트 인증값이 상기 제1 서버 인증값 및 상기 제2 서버 인증값과 동일하지 않은 경우 상기 송신자 인증이 실패한 것으로 결정하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 7

제6항에 있어서,

상기 제2 난수와, 상기 클라이언트가 상기 제2 난수 및 상기 클라이언트에 저장된 공유키를 이용하여 계산한 새로운 클라이언트 인증값을 즉시 HTTP 요청하도록 하는 실행코드를 포함하는 HTTP 응답을 상기 클라이언트에 전송하는 단계를 더 포함하는 사용자 세션 관리 방법.

청구항 8

제7항에 있어서,

상기 제2 난수와 상기 실행 코드를 포함하는 HTTP 응답의 전송 단계는 소정 횟수 동안 반복하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 9

제8항에 있어서,

상기 소정 횟수를 반복한 이후에 소정 시간 동안 상기 제2 난수와 상기 실행 코드를 포함하는 HTTP 응답의 전송 단계를 반복하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 10

제9항에 있어서,

상기 소정 시간이 경과한 후에 상기 사용자 세션을 삭제시키는 단계를 더 포함하는 사용자 세션 관리 방법.

청구항 11

웹 환경에서 네트워크를 통해 클라이언트와 연결된 서버에서 상기 클라이언트와의 사용자 세션을 관리하는 방법에 있어서,

상기 서버가 이전에 생성한 제1 난수와 서버 비밀키를 인수로 하는 일방향 함수로 제1 서버 인증값을 계산하는 단계-여기서, 상기 서버 비밀키는 상기 서버에 저장된 공유키와 키 변형 인수를 이용하여 생성됨-;

제2 난수를 새로 생성하는 단계;

상기 제2 난수와 상기 서버 비밀키를 인수로 하는 일방향 함수로 제2 서버 인증값을 계산하는 단계;

상기 제2 난수 및 상기 키 변형 인수를 포함하는 HTTP 응답(HTTP response)을 상기 클라이언트에 전송하는 단계;

상기 클라이언트로부터 쿠키(cookie)를 포함하는 HTTP 요청(HTTP request)을 전송받는 단계-여기서, 상기 쿠키는 클라이언트 인증값을 포함하고, 상기 클라이언트 인증값은 클라이언트 비밀키 및 상기 클라이언트가 전송받

은 HTTP 응답에 포함된 상기 제2 난수를 인수로 하는 일방향 함수로 계산되고, 상기 클라이언트 비밀키는 상기 클라이언트에 저장된 공유키와 상기 키 변형 인수를 이용하여 생성됨-; 및

상기 제1 서버 인증값 및 상기 제2 서버 인증값 중 어느 하나와 상기 클라이언트 인증값을 비교하여 상기 클라이언트의 송신자 인증의 성공 또는 실패를 결정하는 단계를 포함하는 사용자 세션 관리 방법.

청구항 12

삭제

청구항 13

삭제

청구항 14

웹 환경에서 네트워크를 통해 서버와 연결된 클라이언트에서 사용자 세션을 관리하는 방법에 있어서,

상기 클라이언트가 상기 서버와 공유키를 공유하는 단계;

상기 공유키를 웹 브라우저의 문서 객체 모델(DOM: Document Object Model)의 윈도우 객체의 속성 중 윈도우 이름(window.name), 플래시(Flash)의 지역 공유 객체(LSO: Local Shared Object) 또는 액티브X에 저장하는 단계;

상기 서버로부터의 HTTP 응답을 전송받는 단계-상기 HTTP 응답은 상기 서버에서 생성된 세션정보를 포함함-;

상기 세션정보와 상기 공유키를 이용하여 클라이언트 인증값을 계산하는 단계; 및

상기 클라이언트 인증값을 쿠키에 저장하는 단계를 포함하되,

상기 세션정보는 상기 공유키를 저장한 저장 공간에 저장되는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 15

제14항에 있어서,

상기 서버로부터 복수의 HTTP 응답을 전송받는 단계를 더 포함하되,

상기 복수의 HTTP 응답에 포함된 세션정보와 상기 저장 공간에 저장된 세션정보가 동일한 경우 이미 생성된 상기 쿠키를 재사용하고,

다른 경우 상기 저장 공간에 저장된 세션정보를 갱신하고 상기 갱신된 세션정보를 이용하여 상기 클라이언트 인증값을 새로 계산하고 상기 쿠키를 새로 생성하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 16

제14항에 있어서,

소정 시간 이내에 송신자 인증이 필요한 복수의 HTTP 요청시 각 상기 HTTP 요청은 상기 저장 공간에 저장된 세션정보를 이용하여 생성된 클라이언트 인증값을 포함하는 쿠키를 공통으로 포함하여 상기 서버에 전송되는 단계를 더 포함하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 17

제14항에 있어서,

상기 클라이언트 인증값은 상기 세션정보와 상기 공유키를 이용하여 일방향 함수로 계산된 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 18

제14항에 있어서,

상기 세션정보는 난수인 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 19

제14항에 있어서,

상기 세션정보는 난수와 키 변형 인수를 포함하는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 20

제19항에 있어서,

상기 클라이언트 인증값은 클라이언트 비밀키 및 상기 난수를 인수로 하는 일방향 함수로 계산되고,

상기 클라이언트 비밀키는 상기 클라이언트에 저장된 상기 공유키와 상기 키 변형 인수를 이용하여 생성되는 것을 특징으로 하는 사용자 세션 관리 방법.

청구항 21

제1항 내지 제11항 중 어느 한 항의 방법을 컴퓨터에서 실행하기 위한 프로그램을 기록하는 컴퓨터 판독 가능한 기록매체.

청구항 22

제14항 내지 제20항 중 어느 한 항의 방법을 컴퓨터에서 실행하기 위한 프로그램을 기록하는 컴퓨터 판독 가능한 기록매체.

명 세 서

발명의 상세한 설명

기 술 분 야

[0001] 본 발명은 웹 환경에서의 안전한 사용자 세션 관리 방법에 관한 것이다.

배 경 기 술

[0002] 일반적으로 웹 환경은 개방형 환경으로 불특정 다수의 사용자가 접속하여 웹 서비스를 받는다. 웹 서비스에서 세션이란 상태 정보를 유지하지 않는 HTTP(hypertext transfer protocol: 인터넷에서, 웹 서버와 사용자의 인터넷 브라우저 사이에 문서를 전송하기 위해 사용되는 통신 규약) 프로토콜에서 한 사용자가 연속적으로 요청하는 일련의 서비스 요구를 처리하기 위하여 서버(server)와 클라이언트(client)에서 저장하고 주고 받는 정보를 의미한다.

[0003] 웹 환경에서 사용자의 세션을 관리하기 위하여 기존에는 클라이언트가 가진 쿠키(cookie)를 이용하여 사용자를 구분하거나 인증하였다. 일례로, 쿠키에 사용자의 IP, 해쉬된 사용자의 패스워드, 전자 서명을 유지하는 방법이 있다.

[0004] 하지만, 클라이언트가 가진 쿠키는 HTTP 요청(HTTP request) 때마다 서버에 평문 형태로 자동 전송되기 때문에 쿠키에 포함된 정보가 공격자(attacker)에게 노출되는 문제점이 있다. 그리고 IP 스푸핑(spoofing)이나 오프라인 사전단어 공격(offline dictionary attack)에 취약하여 쿠키를 이용하여 사용자의 세션을 보호하는데 근본적인 해결책이 될 수 없는 문제점이 있다.

[0005] 즉, 사용자의 세션 유지에 사용되는 쿠키는 도청(sniffing) 등을 통해 쉽게 노출될 수 있다. 이를 방지하기 위한 액티브X(ActiveX)는 사용자에게 부담이 되거나 편리성을 제공하지 못하는 문제점이 있다. 또한, 액티브X 자체의 설치에 문제가 발생하는 경우에는 웹 서비스에 접근조차 불가능한 경우도 발생한다.

발명의 내용

해결 하고자하는 과제

[0006] 본 발명은 시도-응답(challenge-response) 방식의 인증 알고리즘을 HTTP 프로토콜에 적용하여 사용자의 세션 보호가 가능한 사용자 세션 관리 방법 및 이를 수행하는 프로그램이 기록된 기록매체를 제공한다.

[0007] 또한, 본 발명은 웹 브라우저에서 비밀키를 저장하기 위하여 쿠키가 아닌 다른 저장 공간을 이용함으로써 평문 형태로 자동 전송되는 종래 쿠키의 단점을 극복한 사용자 세션 관리 방법 및 이를 수행하는 프로그램이 기록된 기록매체를 제공한다.

[0008] 또한, 본 발명은 HTTP 프로토콜에 시도-응답 방식을 적용할 경우 사용자의 행동패턴 또는 네트워크의 상황에 따라 공유하는 난수가 동기화되지 않는 경우 공유키의 재검증 루틴을 추가한 오류를 줄일 수 있는 사용자 세션 관리 방법을 제공한다.

[0009] 또한, 본 발명은 문턱 시간을 설정하여 동시다발적인 HTTP 요청에 대해서도 유연하게 응답 가능한 사용자 세션 관리 방법을 제공한다.

과제 해결수단

[0010] 본 발명의 일 측면에 따르면, 웹 환경에서 네트워크를 통해 클라이언트와 연결된 서버에서 상기 클라이언트와의 사용자 세션을 관리하는 방법이 제공된다.

[0011] 사용자 세션 관리 방법은, 상기 서버가 이전에 생성한 난수와 상기 서버에 저장된 공유키를 이용하여 제1 서버 인증값을 계산하는 단계; 난수를 새로 생성하는 단계; 상기 난수와 상기 서버에 저장된 공유키를 이용하여 제2 서버 인증값을 계산하는 단계; 상기 난수를 포함하는 HTTP 응답(HTTP response)을 상기 클라이언트에 전송하는 단계; 상기 클라이언트로부터 쿠키(cookie)를 포함하는 HTTP 요청(HTTP request)을 전송받는 단계-여기서, 상기 쿠키는 클라이언트 인증값을 포함하고, 상기 클라이언트 인증값은 상기 클라이언트에 저장된 공유키와 상기 클라이언트가 전송받은 HTTP 응답에 포함된 난수를 이용하여 계산됨-; 및 상기 제1 서버 인증값 및 상기 제2 서버 인증값 중 어느 하나와 상기 클라이언트 인증값을 비교하여 상기 클라이언트의 송신자 인증의 성공 또는 실패를 결정하는 단계를 포함한다.

[0012] 여기서, 상기 서버는 상기 클라이언트로부터의 사용자 로그인 과정에서 상기 공유키를 상기 클라이언트와 공유할 수 있다.

[0013] 그리고 상기 결정 단계는, 상기 클라이언트 인증값과 상기 제1 서버 인증값이 동일하고 상기 HTTP 요청의 도착 시점이 직전 HTTP 요청의 도착 시점으로부터 소정 시간 경과 이전인 경우 상기 송신자 인증이 성공한 것으로 결정할 수 있다. 상기 송신자 인증이 성공한 경우 상기 제2 서버 인증값의 계산에 이용된 난수를 HTTP 응답에 포함시켜 상기 클라이언트에 전송하는 단계를 더 포함할 수 있다.

[0014] 한편, 상기 결정 단계는, 상기 클라이언트 인증값과 상기 제2 서버 인증값이 동일한 경우 상기 송신자 인증이 성공한 것으로 결정하되, 난수를 새로 생성하는 단계; 상기 제2 서버 인증값을 상기 제1 서버 인증값에 복사하는 단계; 상기 새로 생성된 난수 및 상기 서버에 저장된 공유키를 이용하여 상기 제2 서버 인증값을 갱신하는 단계; 및 상기 새로 생성된 난수를 HTTP 응답에 포함시켜 상기 클라이언트에 전송하는 단계를 더 포함할 수 있다.

[0015] 또한, 상기 결정 단계는, 상기 클라이언트 인증값이 상기 제1 서버 인증값 및 상기 제2 서버 인증값과 동일하지 않은 경우 상기 송신자 인증이 실패한 것으로 결정할 수 있다. 상기 난수와, 상기 클라이언트가 상기 난수 및 상기 클라이언트에 저장된 공유키를 이용하여 계산한 새로운 클라이언트 인증값을 즉시 HTTP 요청하도록 하는 실행코드를 포함하는 HTTP 응답을 상기 클라이언트에 전송하는 단계를 더 포함할 수 있다. 상기 난수와 상기 실행 코드를 포함하는 HTTP 응답의 전송 단계는 소정 횟수 동안 반복할 수 있다. 상기 소정 횟수를 반복한 이후에 소정 시간 동안 상기 난수와 상기 실행 코드를 포함하는 HTTP 응답의 전송 단계를 반복할 수 있다. 상기 소정

시간이 경과한 후에 상기 사용자 세션을 삭제시키는 단계를 더 포함할 수 있다.

[0016] 상기 제1 서버 인증값 및 상기 제2 서버 인증값은 상기 서버에 저장된 공유키와 키 변형 인수를 이용하여 생성된 서버 비밀키 및 상기 난수를 인수로 하는 일방향 함수로 계산될 수 있다. 상기 HTTP 응답을 상기 클라이언트에 전송하는 단계는 상기 키 변형 인수를 상기 HTTP 응답에 더 포함하여 전송할 수 있다. 상기 클라이언트 인증값은 상기 클라이언트에 저장된 공유키와 상기 키 변형 인수를 이용하여 생성된 클라이언트 비밀키 및 상기 난수를 인수로 하는 일방향 함수로 계산될 수 있다.

[0017] 한편, 상술한 사용자 세션 관리 방법은 컴퓨터에 의하여 수행될 수 있으며, 컴퓨터에서 실행하기 위한 프로그램을 기록하는 컴퓨터 판독 가능한 기록매체에 기록되어 서버에서 수행될 수 있다.

[0018] 본 발명의 다른 측면에 따르면, 웹 환경에서 네트워크를 통해 서버와 연결된 클라이언트에서 사용자 세션을 관리하는 방법이 제공된다.

[0019] 사용자 세션 관리 방법은, 상기 클라이언트가 상기 서버와 공유키를 공유하는 단계; 상기 공유키를 웹 브라우저의 문서 객체 모델(DOM: Document Object Model)의 윈도우 객체의 속성 중 윈도우 이름(window.name), 플래시(Flash)의 지역 공유 객체(LSO: Local Shared Object) 또는 액티브X에 저장하는 단계; 상기 서버로부터의 HTTP 응답을 전송받는 단계-상기 HTTP 응답은 상기 서버에서 생성된 세션정보를 포함함-; 상기 세션정보와 상기 공유키를 이용하여 클라이언트 인증값을 계산하는 단계; 및 상기 클라이언트 인증값을 쿠키에 저장하는 단계를 포함하되, 상기 세션정보는 상기 공유키를 저장한 저장 공간에 저장된다.

[0020] 여기서, 상기 서버로부터 복수의 HTTP 응답을 전송받는 단계를 더 포함하되, 상기 복수의 HTTP 응답에 포함된 세션정보와 상기 저장 공간에 저장된 세션정보가 동일한 경우 이미 생성된 상기 쿠키를 재사용하고, 다른 경우 상기 저장 공간에 저장된 세션정보를 갱신하고 상기 갱신된 세션정보를 이용하여 상기 클라이언트 인증값을 새로 계산하고 상기 쿠키를 새로 생성할 수 있다.

[0021] 그리고 소정 시간 이내에 송신자 인증이 필요한 복수의 HTTP 요청시 각 상기 HTTP 요청은 상기 저장 공간에 저장된 세션정보를 이용하여 생성된 클라이언트 인증값을 포함하는 쿠키를 공통으로 포함하여 상기 서버에 전송되는 단계를 더 포함할 수 있다.

[0022] 또한, 상기 클라이언트 인증값은 상기 세션정보와 상기 공유키를 이용하여 일방향 함수로 계산될 수 있다.

[0023] 상기 세션정보는 난수일 수 있다.

[0024] 한편, 상기 세션정보는 난수와 키 변형 인수를 포함할 수 있다. 상기 클라이언트 인증값은 상기 클라이언트에 저장된 상기 공유키 및 상기 키 변형 인수를 이용하여 계산한 클라이언트 비밀키와, 상기 HTTP 응답에 포함된 상기 난수를 인수로 하는 일방향 함수로 계산될 수 있다.

[0025] 한편, 상술한 사용자 세션 관리 방법은 컴퓨터에 의하여 수행될 수 있으며, 컴퓨터에서 실행하기 위한 프로그램을 기록하는 컴퓨터 판독 가능한 기록매체에 기록되어 클라이언트에서 수행될 수 있다.

[0026] 전술한 것 외의 다른 측면, 특징, 이점이 이하의 도면, 특허청구범위 및 발명의 상세한 설명으로부터 명확해질 것이다.

효 과

[0027] 본 발명에 따른 사용자 세션 관리 방법 및 이를 수행하는 프로그램이 기록된 기록매체는 시도-응답 방식의 인증 알고리즘을 HTTP 프로토콜에 적용하여 사용자의 세션 보호가 가능하다.

[0028] 또한, 웹 브라우저에서 공유키를 저장하기 위하여 쿠키가 아닌 다른 저장 공간을 이용함으로써 평문 형태로 자동 전송되는 종래 쿠키의 단점 극복이 가능한 효과가 있다.

[0029] 또한, HTTP에서 웹 서비스에 로그인한 사용자와 서버 사이의 채널을 시도-응답 방식의 인증 알고리즘을 이용하여 도청(sniffing), 침입(injection), 위조(forgery) 등의 공격으로부터 안전하게 보호할 수 있다.

[0030] 또한, 액티브X 등의 신규 프로그램을 클라이언트에 배포하지 않고서도 사용자의 세션 보호가 가능하며, 클라이언트의 웹 브라우저가 서버와 계속 통신을 유지하지 않아도 된다. 그리고 상술한 과정을 통해 인증된 쿠키로 서버에서 제공하는 다양한 웹 서비스와 안전하게 세션 공유가 가능하다.

[0031] 또한, HTTP 프로토콜에 시도-응답 방식을 적용할 경우 사용자의 행동패턴 또는 네트워크의 상황에 따라 공유하는 난수가 동기화되지 않는 경우 공유키의 재검증 루틴을 추가하여 오류를 줄일 수 있다.

[0032] 또한, 문턱 시간을 설정하여 동시다발적인 HTTP 요청에 대해서도 유연하게 응답 가능하다.

발명의 실시를 위한 구체적인 내용

[0033] 본 발명은 다양한 변환을 가할 수 있고 여러 가지 실시예를 가질 수 있는 바, 특정 실시예들을 도면에 예시하고 상세한 설명에 상세하게 설명하고자 한다. 그러나, 이는 본 발명을 특정한 실시 형태에 대해 한정하려는 것이 아니며, 본 발명의 사상 및 기술 범위에 포함되는 모든 변환, 균등물 내지 대체물을 포함하는 것으로 이해되어야 한다. 본 발명을 설명함에 있어서 관련된 공지 기술에 대한 구체적인 설명이 본 발명의 요지를 흐릴 수 있다고 판단되는 경우 그 상세한 설명을 생략한다.

[0034] 제1, 제2 등의 용어는 다양한 구성요소들을 설명하는데 사용될 수 있지만, 상기 구성요소들은 상기 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 사용된다.

[0035] 본 출원에서 사용한 용어는 단지 특정한 실시예를 설명하기 위해 사용된 것으로, 본 발명을 한정하려는 의도가 아니다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 출원에서, "포함하다" 또는 "가지다" 등의 용어는 명세서상에 기재된 특징, 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것이 존재함을 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.

[0036] 도 1은 본 발명에 적용되는 시도-응답 프로토콜(challenge-response protocol)의 인증 방법을 설명한 도면이다. 이하에서 비밀키는 클라이언트(1)가 클라이어트 인증값을 계산하고 서버(2)가 서버 인증값을 계산하기 위해 필요한 키이며, 공유키는 클라이언트(1)와 서버(2)가 최초 사용자 로그인 과정에서 공유하는 키를 의미한다.

[0037] 클라이언트(1)와 서버(2) 사이의 비밀키인 K를 클라이언트(1)와 서버(2)가 서로 공유하고 있다(단계 S10). 여기서, 비밀키 K는 공유키에 해당한다. 공유키 K는 안전한 키 교환 프로토콜(예를 들어, PAKE(Password Authenticated Key Exchange) 등)을 이용하여 클라이언트(1)를 통한 서버(2)로의 사용자 로그인 과정에서 공유 가능하다.

[0038] 서버(2)는 난수 R1을 생성하고(단계 S11), 세션 등을 이용하여 난수 R1을 저장한다. 그리고 생성한 난수 R1을 클라이언트(1)에 전송한다(단계 S12). 난수 R1을 클라이언트(1)에 전송하는 과정이 시도(challenge) 과정에 해당한다.

[0039] 클라이언트(1)는 서버(2)로부터 전송받은 난수 R1과 클라이언트(1)가 미리 저장하고 있었던 공유키 K를 이용하여 클라이언트 인증값 C를 계산한다(단계 S13). 난수 R1과 공유키 K를 이용하여 클라이언트 인증값 C를 계산하는 함수는 일방향 함수이다. 일방향 함수는 인수로부터 결과(함수값)를 구하는 것은 간단하지만, 결과로부터 인수를 구하는 것은 어려운 함수를 일컬으며, 이러한 일방향 함수로는 해쉬 함수, 유사 난수 함수 등이 포함된다.

[0040] 클라이언트(1)는 생성된 클라이언트 인증값 C를 서버(2)에 전송한다(단계 S14). 클라이언트 인증값 C를 서버(2)에 전송하는 과정이 응답(response) 과정에 해당한다.

[0041] 서버(2)는 단계 S11에서 난수 R1을 생성한 후 후술할 단계 S16 이전까지의 과정 중에 임의의 시점에서 난수 R1과 서버(2)가 미리 저장하고 있었던 공유키 K를 이용하여 서버 인증값 S를 계산한다(단계 S15). 난수 R1과 공유키 K를 이용하여 서버 인증값 S를 계산하는 함수는 일방향 함수이며, 클라이언트(1)에서 이용한 일방향 함수와 동일한 것이 바람직하다.

[0042] 서버(2)는 클라이언트(1)로부터 전송받은 클라이언트 인증값 C와, 서버(2) 내에서 생성한 서버 인증값 S를 비교한다(단계 S16).

[0043] 비교 결과 클라이언트 인증값 C와 서버 인증값 S가 다른 경우에는 클라이언트(1)가 가지고 있는 공유키 K가 서버(2)가 가지고 있는 공유키 K와 다른 것으로 판단하여 클라이언트(1)의 인증에 실패한 것으로 간주하고(단계 S17a), 사용자 세션이 유효하지 않으므로 사용자 세션을 삭제한다.

[0044] 비교 결과 클라이언트 인증값 C와 서버 인증값 S가 동일한 경우에는 클라이언트(1)가 가지고 있는 공유키 K가 서버(2)가 가지고 있는 공유키 K와 동일한 것으로 판단하여 클라이언트(1)의 인증에 성공한 것으로 간주하고(단계 S17b), 사용자 세션의 유효를 유지시킨다. 그리고 새로운 난수 R2를 생성한다(단계 S18). 그리고 상술한 단계 S12 내지 S16에 설명한 시도-응답 과정을 새로운 난수 R2를 이용하여 반복하게 된다.

[0045] 서버(2)에서 클라이언트(1)로 새로운 난수 R2를 전송(단계 S19)하는 경우 새로운 난수 R2가 클라이언트(1)에 도달하기 이전에 클라이언트(1)로부터 이전에 클라이언트(1)에 전송되었던 난수 R1에 상응하는 클라이언트 인증값 C가 서버(2)로 다시 전송될 수 있다(단계 S14a).

[0046] 이 경우 서버(2)는 새로운 난수 R2를 생성하였는 바, 서버 인증값은 새로운 난수 R2에 상응하는 함수값이고, 클라이언트 인증값은 기존의 난수 R1에 상응하는 함수값이어서 클라이언트(1)와 서버(2)가 공유한 난수가 동기화되지 않을 수 있다.

[0047] 본 발명에서는 상술한 시도-응답 프로토콜을 HTTP 프로토콜에 적용하는 경우 사용자의 행동패턴 또는 네트워크의 상황에 따라 상술한 것과 같이 클라이언트(1)와 서버(2)가 공유하는 난수가 동기화되지 않을 수 있으므로, 이를 대비할 필요가 있다. 여기서, 사용자의 행동패턴은 다수의 웹 브라우저 이용 혹은 동시다발적인 서비스 요청 등이 있을 수 있다. 네트워크 상황은 패킷이 지연되거나 패킷이 손실되는 경우 등을 의미한다.

[0048] 이러한 사용자의 행동패턴 또는 네트워크 상황에 따라 HTTP 요청이나 HTTP 응답이 지연되거나 손실될 수 있다. HTTP 응답이 지연 또는 손실될 경우 서버(2)는 HTTP 응답을 하기 위하여 난수를 갱신하였지만, 클라이언트(1)에서는 이를 받아보지 못하거나 수신이 늦어지는 사이에 갱신되지 않은 난수에 상응하는 HTTP 요청이 전송되어 서버(2)과 클라이언트(1) 사이에 공유된 난수가 동기화되지 않는다.

[0049] 상술한 시도-응답 방식의 목적이 공유키를 검증하기 위함이므로 난수가 동기화되지 않았음을 이유로 한번 검증에 실패한 경우 클라이언트측 스크립트(Client Side Script), 플래시(Flash) 또는 액티브X(ActiveX) 등의 간결한 실행코드만을 클라이언트(1)에 전송하여 공유키를 재검증한다. 공유키를 재검증하는 횟수를 제한하며, 재검증 시도가 소정 횟수를 초과하면 사용자 세션을 삭제하여 로그아웃시킨다. 여기서, 재검증 루틴은 클라이언트(1)의 캐시를 이용하도록 하여 재검증에 부담을 줄이도록 한다.

[0050] 또한, 웹 환경에서는 HTTP 요청이 동시다발적으로 이루어지는 경우가 있다. 하나의 문서 안에 또 다른 문서를 포함하는 것이 그 예이다. 이 경우 HTTP 요청과 HTTP 응답의 순서가 뒤섞일 수 있다. 따라서, 일정 시간 내에 도착한 2번째 HTTP 요청부터는 첫번째 HTTP 요청에 사용된 클라이언트 인증값을 그대로 사용할 수 있도록 하고, 해당 클라이언트 인증값이 유효한 시간을 연장가능하도록 하여 짧은 시간 안에 다수의 HTTP 요청에 대해서 유연하게 응답을 할 수 있도록 한다.

[0051] 이하에서는 상술한 내용이 적용된 웹 환경에서 안전한 사용자 세션 관리 방법을 제공하고자 한다.

[0052] 도 2는 본 발명의 일 실시예에 따른 서버에 포함되는 세션 관리 모듈의 블록구성도이다.

[0053] 도 2를 참조하면, 세션 관리 모듈(20), 난수 생성부(21), 인증값 산출부(22), 인증값 비교부(23), 인증 관리부(24), 시간 분석부(25), 오류허용 분석부(26)가 도시되어 있다.

[0054] 세션 관리 모듈(20)은 웹 서비스를 제공하는 서버(2)에 포함된다. 세션 관리 모듈(20)은 클라이언트(1)로부터의 HTTP 요청에 대한 송신자 인증을 위해 세션을 관리한다.

[0055] 난수 생성부(21)는 클라이언트(1)와 공유하기 위한 난수를 생성한다.

[0056] 인증값 산출부(22)는 생성된 난수와 미리 저장되어 있는 공유키 등을 인수로 하는 미리 정해진 함수로 계산한 결과값인 서버 인증값을 산출한다. 본 발명에서는 시간 간격을 두고 생성된 2개의 난수에 각각 대응하는 제1 서버 인증값과 제2 서버 인증값을 산출한다. 제1 서버 인증값이 먼저 생성된 난수에 상응하는 결과값이고, 제2 서버 인증값이 새로 생성된 난수에 상응하는 결과값이다.

[0057] 인증값 비교부(23)는 인증값 산출부(22)에서 계산한 제1 서버 인증값 또는 제2 서버 인증값과 클라이언트(1)로부터 전송되는 클라이언트 인증값(클라이언트(1)에서 난수와 공유키 등을 인수로 하는 미리 정해진 함수로 계산한 결과값)을 비교한다.

[0058] 인증 관리부(24)는 비교 결과에 따라 HTTP 요청을 한 클라이언트(1)의 송신자 인증 성공 또는 인증 실패가 결정된다. 인증이 실패한 경우 부적합한 클라이언트(1)로부터의 HTTP 요청(예를 들어, 공격자의 HTTP 요청 등)으로 판단하고 세션을 삭제시킨다. 인증이 성공한 경우 적합한 클라이언트(1)로부터의 HTTP 요청으로 판단하고 세션을 유지시킨다.

[0059] 시간 분석부(25)는 클라이언트(1)로부터 소정 시간(예를 들어, 문턱 시간 등) 내에 연속적으로 도착하는 HTTP 요청이 동일한 클라이언트 인증값을 가지고, 클라이언트 인증값이 제1 서버 인증값과 동일한 경우 동시다발적인 서비스 요청으로 판단하여 난수를 갱신하지 않고 HTTP 응답을 제공한다.

[0060] 클라이언트 인증값이 제2 서버 인증값과 동일한 경우에는 먼저 생성된 난수가 적용된 HTTP 요청이 종료되고 이후 새로 생성된 난수가 적용된 HTTP 요청이 수신되는 것으로 판단한다. 이 경우 제2 서버 인증값을 제1 서버 인증값에 복사한다. 그리고 난수를 새로 생성하고 새로운 제2 서버 인증값을 계산하며, 새로 생성한 난수를 클라이언트(1)에 전송하는 HTTP 응답에 포함시킨다.

[0061] HTTP 요청이 클라이언트(1)로부터 소정 시간 내에 연속적으로 도착하지 못하고, 클라이언트 인증값이 제2 서버 인증값과 다른 경우, 해당 HTTP 요청은 공격자에 의한 HTTP 요청인 경우가 대부분이다. 하지만, 상술한 것과 같이 사용자의 행동패턴이나 네트워크 상황에 따라 클라이언트(1)가 서버(2)와 동일한 공유키를 가지고 있음에도 난수의 비동기화로 인한 오류가 발생할 수도 있다.

[0062] 이러한 오류를 허용하기 위하여 오류허용 분석부(26)는 공유키의 재검증 루틴을 이용한다. 소정 횟수 동안 및/또는 소정 시간 동안 클라이언트(1)에 난수를 전송하고 이 난수에 상응하여 계산된 클라이언트 인증값을 즉시 HTTP 요청으로 전송하게 하는 HTTP 응답을 클라이언트(1)에 전송한다. 즉시 전송된 HTTP 요청에 포함된 클라이언트 인증값을 제1 서버 인증값 또는 제2 서버 인증값과 비교하여 클라이언트(1)이 공유키를 가지고 있는지 여부를 재검증한다.

[0063] 도 3 및 도 4를 참조하여 클라이언트(1)와 서버(2)가 공유키를 공유하는 경우 안정하게 사용자 세션을 관리하는 방법을 설명하고, 이후 도 5a 이하 도면을 참조하여 본 발명에 따른 시간에 유연하고 오류를 허용하는 사용자 세션 관리 방법을 설명하기로 한다.

[0064] 도 3은 안전한 사용자 세션 관리 방법, HTTP 요청의 송신자 인증 프로토콜을 설명한 도면이다.

[0065] 클라이언트(1)와 서버(2) 사이의 비밀키 K를 클라이언트(1)와 서버(2)가 서로 공유하고 있다(단계 S30). 즉, 비밀키 K는 공유키에 해당한다. 공유키 K는 안전한 키 교환 프로토콜(예를 들어, PAKE 등)를 이용하여 클라이언트(1)를 통한 서버(2)로의 웹 서비스의 사용자 로그인 과정에서 공유가능하다.

[0066] 공유키 K는 클라이언트(1)의 로그인 단계에서 교환되어 로그아웃될 때까지 웹 브라우저에 저장되어 있어야 한다. 그리고 웹 페이지가 바뀌더라도 지속적으로 유지되어야 한다.

[0067] 이 경우 공유키를 쿠키에 저장하는 경우 클라이언트(1)로부터의 HTTP 요청에 쿠키가 포함되어 네트워크로 전송되므로 공유키가 그대로 노출되는 문제점이 있다.

[0068] 따라서, 클라이언트(1)는 공유키 K를 쿠키가 아닌 별도의 저장 공간에 저장한다. 웹 브라우저의 문서 객체 모델(DOM: Document Object Model)의 윈도우 객체의 속성 중 윈도우 이름(window.name), 플래시(Flash)의 지역 공유 객체(LSO: Local Shared Object), 액티브X 등의 저장 공간이 이용될 수 있다. 이러한 저장 공간에 대해서는 추후 상세히 설명하기로 한다.

[0069] 클라이언트(1)로부터 HTTP 요청(HTTP request)이 전송되면(단계 S31), 서버(2)는 다음과 같이 동작한다.

[0070] 서버(2)는 난수 R을 생성한다(단계 S32). 이와 관련된 의사 코드(Pseudo code)는 하기와 같다.

[0071] R = random()

[0072] 서버(2)는 난수 R과 공유키 K를 이용하여 미리 정해진 함수로 계산한 결과값인 서버 인증값 S를 산출한다(단계 S37). 여기서, 사용자 식별정보인 ID 등이 미리 정해진 함수의 인수로 더 포함될 수 있다. 이와 관련된 의사 코드는 하기와 같다.

[0073] S = Func(K, R, ID, …)

[0074] 여기서, 미리 정해진 함수는 일방향 함수인 것이 바람직하다. 일방향 함수는 인수로부터 결과(함수값)를 구하는 것은 간단하지만, 결과로부터 인수를 구하는 것은 어려운 함수를 일컬으며, 이러한 일방향 함수로는 해쉬 함수, 유사 난수 함수 등이 포함된다.

[0075] 서버(2)는 공유키 K, 난수 R, 서버 인증값 S를 웹 어플리케이션에서 제공하는 세션 또는 데이터베이스에 저장한다. 이와 관련된 의사 코드는 하기와 같다.

[0076] Session = {K, R, S}

[0077] 상술한 단계 S37은 후술한 단계 S32 이후 단계 S38 이전에 임의의 시점에서 서버(2)에서 수행되면 충분하다.

[0078] 서버(2)는 HTTP 응답(HTTP response)을 통해 단계 S32에서 생성된 난수 R을 클라이언트(1)에 전송한다(단계 S33). 난수 R은 세션정보에 해당한다. 여기서, 세션정보는 사용자 세션 인증을 위해 서버(2)가 클라이언트(1)에 제공하는 정보를 의미한다.

[0079] 클라이언트(1)는 전송받은 난수 R과, 소정의 저장 공간에 저장된 공유키 K를 이용하여 미리 정해진 함수로 계산한 결과값인 클라이언트 인증값 C를 산출한다(단계 S34). 여기서, 사용자 식별정보인 ID 등이 미리 정해진 함수의 인수로 더 포함될 수 있다. 여기서, 미리 정해진 함수는 단계 S37에서 서버(2)가 이용한 일방향 함수와 동일해야 한다.

[0080] 이후 클라이언트(1)는 산출된 클라이언트 인증값 C를 쿠키에 저장한다(단계 S35). 이와 관련된 의사 코드는 하기와 같다.

[0081] setCookie(C = Func(K, R, ID, …))

[0082] 클라이언트(1)가 HTTP 요청을 하는 경우 클라이언트 인증값 C를 저장하고 있는 쿠키는 자동으로 포함되어 서버(2)로 전송된다(단계 S36).

[0083] 서버(2)는 쿠키에 포함된 클라이언트 인증값 C와 단계 S37에서 산출된 서버 인증값 S를 비교한다(단계 S38). 비교 결과 두 인증값이 다른 경우 단계 S39로 진행하고, 동일한 경우 단계 S40으로 진행한다.

[0084] 단계 S39에서, 두 인증값이 다르므로 서버(2)는 클라이언트(1)의 송신자 인증이 실패한 것으로 간주하고 서버(2)에 저장된 서버측 세션유지정보(난수 R, 공유키 K 등)를 삭제하고 세션을 삭제한다. 이와 관련된 의사 코드는 하기와 같다.

[0085] Session destroy()

[0086] 그리고 클라이언트(1)에 저장된 클라이언트측 세션유지정보(공유키 K와 난수 R, 쿠키 등)가 모두 NULL로 설정되도록 하는 클라이언트측 스크립트, 플래시 또는 액티브X를 클라이언트(1)에 전송한다. 이 과정에서 사용자는 로그아웃된다.

[0087] 단계 S40에서, 두 인증값이 동일하므로 서버(2)는 클라이언트(1)의 송신자 인증이 성공한 것으로 간주하고, 새로운 난수 R을 생성한다(단계 S41). 그리고 클라이언트(1)가 HTTP 요청을 통해 요청한 서비스에 대한 HTTP 응답에 새로 생성된 난수 R을 첨부하여 클라이언트(1)에 전송한다(단계 S42). 이후 서버(2)는 다른 HTTP 요청을 받게 되면 단계 S38 이하의 과정을 반복하게 된다.

[0088] 클라이언트(1)는 단계 S30에서 교환한 공유키 K를 쿠키가 아닌 별도의 저장 공간에 저장한다. 웹 브라우저의 문서 객체 모델(DOM: Document Object Model)의 윈도우 객체의 속성 중 윈도우 이름(window.name), 플래시(Flash)의 지역 공유 객체(LSO: Local Shared Object), 액티브X 등의 저장 공간이 이용될 수 있다.

[0089] 일 실시예로, 클라이언트(1)에서의 저장 공간은 웹 브라우저의 문서 객체 모델(DOM: Document Object Model)의 윈도우 객체의 속성 중 윈도우 이름(window.name)일 수 있다. 여기서, 문서 객체 모델은 웹 브라우저를 통한 확장성 생성 언어(XML) 문서의 상호 연동을 위한 객체 기반의 문서 모델을 의미한다. 플랫폼과 언어 면에서 중립적인 인터페이스로서 프로그램과 스크립트에 의한 문서의 내용, 구조, 종류의 동적인 접근과 변경이 가능하며, 스크립트나 프로그램 언어에 웹 페이지를 연결해 준다. 웹 페이지를 조작, 생성하기 위해 사용되는 속성(property), 방법(method) 및 이벤트(event)가 객체를 구성하는데, 이러한 객체들은 대부분의 웹 브라우저에서 스크립트 언어를 통해 접근할 수 있다.

[0090] 윈도우 이름(window.name)의 원래 목적은 각각의 브라우저(또는 탭)마다 이름을 부여하고 이 이름을 통해서 브라우저 간에 접근이 가능하도록 하는 것이었다. 브라우저 내의 페이지가 바뀌더라도 브라우저의 이름은 바뀌지 않기 때문에 최근엔 이를 이용하여 페이지 사이에 데이터를 전달하는 용도로 사용하고 있다. 윈도우 이름은 쓰기 가능한 크기도 정해져 있지 않아서 원하는 데이터를 크기 제한 없이 저장할 수 있다는 장점이 있다.

[0091] 다른 실시예로, 클라이언트(1)에서의 저장 공간은 플래시의 지역 공유 객체(LSO: Local Shared Object)일 수 있다. 지역 공유 객체는 플래시 플레이어에서 사용되는 쿠키와 유사한 데이터 엔티티(cookie-like data entity)이다. 플레시 플레이어에서 구동되는 어플리케이션은 스트링(string)이나 숫자(number) 같은 기본 데이터 타입(basic data type)으로 구성되거나 더 복잡한 객체로 구성되는 데이터를 저장(store)하고 검색(retrieve)할 수 있다. 플래시의 지역 공유 객체는 플래시 플러그인(plug-in)이 설치되어 있다면 이용할 수 있는 공간이다. 변수명이 윈도우 이름(window.name)으로 지정된 일 실시예와는 다르게 변수명을 원하는 것으로 설정할 수 있고 쿠키와 같이 도메인별로 데이터를 관리할 수 있기 때문에 보안문제가 어느 정도 완화될 수 있다. 지역 공유 객체의 기본 크기는 100kB이다.

[0092] 또 다른 실시예로, 클라이언트(1)는 액티브X를 이용하여 공유키 K를 저장할 수도 있다. 액티브X는 익스플로러의 경우 이용할 수 있는 플러그인으로, 컴파일된 프로그램을 사용자의 컴퓨터에 설치하여 클라이언트의 자원을 보다 많이 이용할 수 있다는 장점이 있다. 액티브X를 이용할 경우 제작자가 설정한 위치에 데이터를 저장하고 읽을 수 있다. 액티브X와 관련된 기술로 저장 공간의 역할을 할 수 있는 것은 액티브X 도큐먼트, 액티브X 스크립팅 등이 있다. 액티브X 도큐먼트는 HTML과는 무관하게 작성된 것으로, MS워드나 엑셀 파일 등을 의미한다. 액티브 스크립팅은 액티브X 컨트롤이나 자바 애플릿에 포함시킬 수 있는 스크립트 언어로, J스크립트나 VB스크립트가 대표적이다.

[0093] 이러한 저장 공간에 비밀키인 공유키를 저장하고 있는 경우 비밀키가 외부로 노출되는 경우 안전한 세션관리가 어려울 수 있다. 예를 들어, 윈도우 이름은 쓰기 가능한 크기도 정해져 있지 않아서 원하는 데이터를 크기 제한 없이 저장할 수 있다는 장점이 있지만 다른 서비스로 이동하여도 값이 바뀌지 않기 때문에 여기에 비밀키를 저장한다면 보안상에 결점이 발생할 수 있다. 여기서 생길 수 있는 보안 문제의 해결 방법은 비밀키 자체를 저장하지 않고 이를 변형 또는 암호화한 것을 저장하여 윈도우 이름에 저장된 값이 노출되더라도 비밀키가 직접적으로 노출되지 않도록 하는 것이다.

[0094] 도 3을 참조하여 상술한 내용에서는 비밀키와 공유키가 동일한 경우이며, 이하에서는 비밀키와 공유키가 다른 경우를 중심으로 도 4를 참조하여 상세히 설명하기로 한다.

[0095] 도 4는 다른 안전한 사용자 세션 관리 방법, 키 보호모드에서의 HTTP 요청의 송신자 인증 프로토콜을 설명한 도면이다.

[0096] 즉, 키 보호모드는 공유키가 노출되더라도 비밀키는 노출되지 않아 HTTP 요청의 송신자 인증이 가능한 경우를 의미하며, 도 3을 참조하여 설명한 것과는 달리 공유키와 비밀키가 다른 값을 가진다.

[0097] 클라이언트(1)와 서버(2) 사이의 공유키인 K를 클라이언트(1)와 서버(2)가 서로 공유하고 있다(단계 S50). 공유키 K는 안전한 키 교환 프로토콜(예를 들어, PAKE 등)를 이용하여 클라이언트(1)를 통한 서버(2)로의 웹 서비스의 사용자 로그인 과정에서 공유가능하다.

[0098] 공유키 K는 클라이언트(1)의 로그인 단계에서 교환되어 로그아웃될 때까지 웹 브라우저에 저장되어 있어야 한다. 그리고 웹 페이지가 바뀌더라도 지속적으로 유지되어야 한다.

[0099] 이 경우 공유키를 쿠키에 저장하는 경우 클라이언트(1)로부터의 HTTP 요청에 쿠키가 포함되어 네트워크로 전송되므로 공유키가 그대로 노출되는 문제점이 있다.

[0100] 따라서, 클라이언트(1)는 공유키 K를 쿠키가 아닌 별도의 저장 공간에 저장한다. 웹 브라우저의 문서 객체 모델(DOM)의 윈도우 객체의 속성 중 윈도우 이름(window.name), 플래시의 지역 공유 객체(LSO), 액티브X 등의 저장 공간이 이용될 수 있다. 이에 대해서는 앞서 설명하였는바 상세한 설명은 생략하기로 한다.

[0101] 여기서, 공유키 K는 키 보호모드에 적용하기 위하여 하기의 수학식 1과 같은 특징을 가진다.

수학식 1

[0102] K = ① Func1(K1, K2) or ② K1

[0103] Func2(K, K2) = K'

[0104] K' = ① K1 or ② K3

[0105] 여기서, K는 공유키, K'는 비밀키, Func1은 클라이언트(1)에 저장된 값을 통해 비밀키가 직접적으로 노출되지 않도록 키 변형 인수 K2를 이용하여 키를 변형시키는 키 변형 함수, Func2는 공유키 K로부터 키 변형 인수 K2를 이용하여 비밀키 K2를 생성해내는 비밀키 생성 함수이다.

[0106] 클라이언트(1)는 공유키 K만을 저장하고 있으며, 서버(2)는 공유키 K와 키 변형 인수 K2를 저장하고 있다. 이 경우 클라이언트(1)에 저장된 공유키 K 및/또는 키 변형 인수 K2가 외부에 노출되더라도 공유키 K와 키 변형 인수 K2를 이용하여 클라이언트 비밀키 K'를 생성해내므로, 비밀키의 보호가 가능하게 된다.

[0107] 제1 실시예(①)에 따르면, K1이 비밀키이다.

[0108] 비밀키 K1이 노출되지 않도록 키 변형 함수 Func1과 키 변형 인수 K2를 이용하여 변형된 공유키 K를 클라이언트(1)와 서버(2)가 공유한다. 여기서, 비밀키 생성 함수 Func2는 키 변형 함수 Func1을 통해 변형된 키 K로부터 비밀키 K1을 복구해내는 함수이다.

[0109] 따라서, 클라이언트(1)는 공유키 K, 키 변형 인수 K2를 비밀키 생성 함수 Func2로 계산한 결과값인 클라이언트 비밀키 K', 즉 K1을 계산하고, 이를 송신자 인증을 위한 클라이언트측 세션유지정보 생성에 이용한다. 그리고 서버(2)는 공유키 K, 키 변형 인수 K2를 비밀키 생성 함수 Func2로 계산한 결과값인 서버 비밀키 K', 즉 K1을 계산하고, 이를 송신자 인증을 위한 서버측 세션유지정보 생성에 이용한다.

[0110] 제2 실시예(②)에 따르면, K1이 공유키이다.

[0111] 공유키 K가 K1인 경우 비밀키 생성 함수 Func2는 K1과 키 변형 인수 K2를 인수로 하여 새로운 결과값인 K3를 비밀키로 생성해낸다. 따라서, 클라이언트(1)는 공유키 K1, 키 변형 인수 K2를 비밀키 생성 함수 Func2로 계산한 결과값인 클라이언트 비밀키 K', 즉 K3를 계산하고, 이를 송신자 인증을 위한 클라이언트측 세션유지정보 생성에 이용한다. 그리고 서버(2)는 공유키 K1, 키 변형 인수 K2를 비밀키 생성 함수 Func2로 계산한 결과값인 서버 비밀키 K', 즉 K3를 계산하고, 이를 송신자 인증을 위한 서버측 세션유지정보 생성에 이용한다.

[0112] 단계 S50을 통해 서버(2)와 클라이언트(1)는 안전한 키 교환 프로토콜을 통해 공유키 K를 공유함을 가정하고 이하 송신자 인증 방법을 설명하기로 한다.

[0113] 서버(2)는 클라이언트(1)로부터의 HTTP 요청(단계 S51)을 전송받고, 서버(2)는 난수 R을 생성한다(단계 S52). 이와 관련된 의사 코드는 하기와 같다.

[0114] R = random()

[0115] 서버(2)는 난수 R과 서버 비밀키 K'를 이용하여 미리 정해진 함수로 계산한 결과값인 서버 인증값 S를 산출한다(단계 S58). 여기서, 서버 비밀키 K'는 서버(2)가 미리 공유키 K와 키 변형 인수 K2를 인수로 하여 비밀키 생성 함수 Func2로 계산한 결과값이다.

[0116] 여기서, 사용자 식별정보인 ID 등이 미리 정해진 함수의 인수로 더 포함될 수 있다. 이와 관련된 의사 코드는 하기와 같다.

[0117] S = Func(K', R, ID, …)

[0118] 여기서, 미리 정해진 함수는 일방향 함수인 것이 바람직하다. 이러한 일방향 함수로는 해쉬 함수, 유사 난수 함수 등이 포함된다.

[0119] 서버(2)는 공유키 K, 키 변형 인수 K2, 난수 R, 서버 인증값 S를 웹 어플리케이션에서 제공하는 세션 또는 데이터베이스에 저장한다. 이와 관련된 의사 코드는 하기와 같다.

[0120] Session = {K, K2, R, S}

[0121] 상술한 단계 S58은 후술한 단계 S52 이후 단계 S59 이전에 임의의 시점에서 서버(2)에서 수행되면 충분하다.

[0122] 서버(2)는 HTTP 응답을 통해 단계 S52에서 생성된 난수 R과 키 변형 인수 K2를 클라이언트(1)에 전송한다(단계

S53). 난수 R과 키 변형 인수 K2는 세션정보에 해당한다. 여기서, 세션정보는 사용자 세션 인증을 위해 서버(2)가 클라이언트(1)에 제공하는 정보를 의미한다.

[0123] 이 경우 HTTP 응답에는 클라이언트(1)에서 공유키 K와 키 변형 인수 K2를 이용하여 클라이언트 비밀키 K'를 계산하고, 클라이언트 비밀키 K'를 이용하여 클라이언트측 세션유지정보인 클라이언트 인증값 C를 계산하며, 산출된 클라이언트 인증값 C를 쿠키에 저장하도록 하는 실행코드(예를 들어, 자바스크립트 등)가 더 포함될 수 있다.

[0124] 클라이언트(1)는 전송받은 키 변형 인수 K2와 소정의 저장 공간에 저장된 공유키 K를 이용하여 비밀키 생성 함수 Func2로 계산한 클라이언트 비밀키 K'를 산출한다(단계 S54).

[0125] 이후 클라이언트(1)는 전송받은 난수 R과, 단계 S54에서 산출된 클라이언트 비밀키 K'를 이용하여 미리 정해진 함수로 계산한 결과값, 즉 클라이언트측 세션유지정보인 클라이언트 인증값 C를 산출한다(단계 S55). 여기서, 사용자 식별정보인 ID 등이 미리 정해진 함수의 인수로 더 포함될 수 있다. 여기서, 미리 정해진 함수는 단계 S58에서 서버(2)가 이용한 일방향 함수와 동일해야 한다.

[0126] 이후 클라이언트(1)는 산출된 클라이언트 인증값 C를 쿠키에 저장한다(단계 S56). 이와 관련된 의사 코드는 하기와 같다.

[0127] setCookie(C = Func(K', R, ID, …))

[0128] 여기서, 클라이언트 인증값 C를 쿠키에 저장한 직후에 공유키 K를 제외한 나머지는 모두 삭제하여 외부로 노출되지 않도록 한다.

[0129] 클라이언트(1)가 인증이 필요한 HTTP 요청을 하는 경우 새로운 클라이언트측 세션유지정보가 포함되어 서버(2)로 전송된다. 즉, 클라이언트 인증값 C를 저장하고 있는 쿠키가 자동으로 포함되어 서버(2)로 전송된다(단계 S57).

[0130] 서버(2)는 쿠키에 포함된 클라이언트 인증값 C와 단계 S58에서 산출된 서버 인증값 S를 비교한다(단계 S59). 비교 결과 두 인증값이 다른 경우 단계 S60으로 진행하고, 동일한 경우 단계 S61로 진행한다.

[0131] 단계 S60에서, 두 인증값이 다른 경우는 서버 비밀키와 클라이언트 비밀키가 다른 경우로, 서버(2)는 클라이언트(1)의 송신자 인증이 실패한 것으로 간주하고 서버측 세션유지정보(난수 R, 공유키 K, 키 변형인수 K2 등)을 삭제하고 세션을 삭제한다. 이와 관련된 의사 코드는 하기와 같다.

[0132] Session destroy()

[0133] 그리고 클라이언트(1)에 저장된 클라이언트측 세션유지정보(공유키 K, 키 변형 인수 K2, 난수 R, 쿠키 등)가 모두 NULL로 설정되도록 하는 클라이언트측 스크립트, 플래시 또는 액티브X를 클라이언트(1)에 전송한다. 이 과정에서 사용자는 로그아웃된다.

[0134] 단계 S61에서, 두 인증값이 동일한 경우는 서버 비밀키와 클라이언트 비밀키가 동일한 경우로, 서버(2)는 클라이언트(1)의 송신자 인증이 성공한 것으로 간주하고, 새로운 난수 R을 생성한다(단계 S62). 그리고 클라이언트(1)가 HTTP 요청을 통해 요청한 서비스에 대한 HTTP 응답에 새로 생성된 난수 R과 키 변형 인수 K2를 첨부하여 클라이언트(1)에 전송한다(단계 S63). 이후 서버(2)는 다른 HTTP 요청을 받게 되면 단계 S59 이하의 과정을 반복하게 된다.

[0135] 상술한 사용자 세션 관리 방법을 이용하면 사용자의 세션 자체는 보호받을 수 있지만, 여전히 송수신되는 HTTP의 내용은 도청될 수 있다. 도청으로부터 송수신되는 내용을 보호하기 위해서 HTTP의 내용 중 중요한 내용(예를 들어, 인터넷 뱅킹 서비스에서 계좌번호 등)을 클라이언트(1)와 서버(2)가 공유하는 비밀키를 이용하여 암호화하여 전송한다. 여기서, 암호화는 AES(advanced encryption standard)의 암호루틴을 이용한다.

[0136] 서버(2)가 HTTP 응답을 하는 경우 중요한 내용을 비밀키를 이용하여 암호화하고, 클라이언트(1)에서 이를 복호화할 수 있도록 하는 클라이언트측 스크립트, 플래시 또는 액티브X를 함께 클라이언트(1)로 전송한다.

[0137] 클라이언트(1)가 HTTP 요청을 하는 경우 중요한 내용을 비밀키로 암호화할 수 있도록 하는 암호루틴을 서버(2)가 HTTP 응답에 포함하여 전송한다. 이 경우 클라이언트(1)는 HTTP 요청을 생성할 때 암호루틴에 따라 비밀키를 이용하여 HTTP 내용 중 중요한 내용은 암호화가 가능하다.

[0138] 클라이언트(1)와 서버(2)는 비밀키 또는 비밀키를 생성할 수 있는 공유키를 공유하고 있으므로, 암호화된 내용을 복호화하는 것은 어렵지 않음을 당업자는 이해할 것이다.

[0139] 이러한 HTTP 내용 중 일부의 암호화는 종래 SSL(secure sockets layer)을 이용한 것과는 다음과 같은 차이가 있다. SSL을 이용하는 경우 SSL은 전송 계층(Transport layer)이며, HTTP 내용 전체를 암호화하고, 도메인간의 세션 관리가 불가능하였다. 하지만, 상술한 것과 같이 클라이언트(1)와 서버(2)가 공유하고 있는 비밀키를 이용한 암호화는 응용 계층(Application layer)에서 이루어지며, HTTP 내용 중 일부만의 선택적인 암호화가 가능하고 도메인간의 세션 관리가 가능한 장점이 있다.

[0140] 또한, 본 발명에서 윈도우 이름을 공유키의 저장 공간으로 이용하는 경우 자식 창을 사용하는 경우에는 window.name=opener.name과 같은 방법 등을 통해서 부모 창이 저장하고 있는 공유키를 자식 창에서도 공유하도록 한다. 여기서, opener.name이 부모 창의 윈도우 이름이다.

[0141] 상술한 안전한 사용자 세션 관리 방법에 의하는 경우에도 사용자의 행동패턴이나 네트워크 상황에 따른 세션정보(도 3에서는 난수, 도 4에서는 난수와 키 변형 인수)의 비동기화로 인해 서버(2)에서 클라이언트(1)의 공유키 저장 여부 검증이 어려운 바 이하에서는 이를 해결한 사용자 세션 관리 방법을 설명하기로 한다.

[0142] 도 5a는 본 발명의 일 실시예에 따른 안전한 사용자 세션 관리 방법을 설명하기 위한 도면이고, 도 5b는 도 5a의 서버에서 실행되는 의사 코드이다.

[0143] 클라이언트(1)와 서버(2)는 안전한 키 교환 프로토콜(예를 들어, PAKE 등)을 이용하여 공유키 K를 공유한다(단계 S70). 이는 웹 서비스의 사용자 로그인 과정에서 적용된다.

[0144] 여기서, 클라이언트(1)는 공유키 K를 소정의 저장 공간(WLA: 윈도우 이름(window.name), 플래시의 지역 공유 객체(LSO), 액티브X 등)에 저장한다.

[0145] 서버(2)는 공유키 K 이외에 최대 문턱 시간(MAXt), 최대 문턱 카운터(MAXc), 최대 검증 문턱 시간(MAXv)의 상수(constant)와, 카운터(counter), 타임스탬프(TS: timestamp), 검증 타임스탬프(TV)의 변수를 저장한다.

[0146] 각 변수의 초기치는 다음과 같다.

[0147] counter = 0, TS = ctime(), TV = 0

[0148] 여기서, ctime()은 현재 시각을 나타내는 함수이다.

[0149] 여기서, 제1 서버 인증값 S1에는 임의의 값으로 초기화되어 있다. 제1 서버 인증값 S1에는 제2 서버 인증값 S2가 갱신되는 경우 갱신되기 이전의 제2 서버 인증값 S2를 복사한다.

[0150] 클라이언트(1)로부터의 HTTP 요청을 수신하면(단계 S71), 서버(2)는 난수 R를 생성하고, 생성한 난수 R과 서버(2)에 저장된 공유키 K를 이용하여 미리 정해진 함수로 제2 서버 인증값 S2를 계산한다. 미리 정해진 함수는 일방향 함수인 것이 바람직하다. 그리고 사용자 식별정보 ID가 미리 정해진 함수의 인수로 더 포함될 수 있다.

[0151] 서버(2)는 단계 S71에서 생성한 난수 R을 포함하는 HTTP 응답을 클라이언트(1)에 전송한다(단계 S72).

[0152] 서버(2)는 제1 서버 인증값 S1, 제2 서버 인증값 S2 및 제2 서버 인증값 S2에 상응하는 난수 R을 웹 어플리케이션에서 제공하는 세션 또는 데이터베이스에 저장한다.

[0153] 클라이언트(1)는 단계 S72에서 전송된 HTTP 응답을 수신한다. 클라이언트(1)는 수신한 HTTP 응답에 포함된 난수 R이 이미 소정의 저장 공간에 저장되어 있는 난수와 다를 경우에만 수신한 HTTP 응답에 포함된 난수 R과 클라이언트(1)에 저장된 공유키 K를 이용하여 미리 정해진 함수로 클라이언트 인증값 C를 계산한다. 미리 정해진 함수는 서버(2)에서 이용되는 일방향 함수와 동일한 것이 바람직하다. 사용자 식별정보 ID가 미리 정해진 함수의 인수로 더 포함될 수 있다. 그리고 HTTP 응답에 포함된 난수 R을 소정의 저장 공간에 덮어 쓴다.

[0154] 클라이언트 인증값 C는 쿠키에 저장한다. 클라이언트(1)의 HTTP 요청의 헤더에 쿠키는 자동으로 포함되어 서버(2)로 전송된다(단계 S73).

[0155] 서버(2)는 쿠키에 포함된 클라이언트 인증값 C가 서버(2)가 가지고 있는 제1 서버 인증값 S1과 동일하고 ctime()-TS가 최대 문턱 시간(MAXt) 이내라면 시간 유연 루틴인 의사 코드 모듈 T1을 수행한다. 여기서, 최대 문턱 시간(MAXt)은 연속적인 HTTP 응답을 동시다발적인 하나의 HTTP 응답 그룹으로 묶을 수 있는 시간이다.

[0156] 시간 유연 루틴(T1)에서, 타임스탬프(TS)에 현재 시간(ctime())을 저장한다. 이는 이후 연속적인 HTTP 응답이 최대 문턱 시간 이내에 도달하는지 여부를 판단하기 위한 기준이 된다. 그리고 카운터(counter)는 0으로 초기화한다. 카운터는 재검증 루틴(T3)에서 설명하기로 한다. 그리고 클라이언트(1)에 현재 서버(2)가 저장하고 있는 난수 R을 HTTP 응답에 포함시켜 전송한다(단계 S76).

[0157] HTTP 요청 #1(S73), HTTP 요청 #2(S74), HTTP 요청 #3(S75), HTTP 요청 #4(S77)이 동시다발적인 하나의 HTTP 응답 그룹인 것으로 가정한다.

[0158] 이 경우 서버(2)는 HTTP 요청 #1이 서버(2)에 도착한 시점인 HR1으로부터 최대 문턱 시간(MAXt) 이내인 HR1'까지 도착하는 HTTP 요청 #2를 시간 유연 루틴(T1)에 의해 동시다발적인 하나의 HTTP 응답 그룹으로 판단한다.

[0159] 그리고 서버(2)는 HTTP 요청 #2가 서버(2)에 도착한 시점인 HR2로부터 최대 문턱 시간(MAXt) 이내인 HR2'까지 도착하는 HTTP 요청 #3을 시간 유연 루틴(T1)에 의해 동시다발적인 하나의 HTTP 응답 그룹으로 판단한다.

[0160] 하지만, 서버(2)는 HTTP 요청 #3이 서버(2)에 도착한 시점인 HR3으로부터 최대 문턱 시간(MAXt) 이내인 HR3'까지 도착하지 못한 HTTP 요청 #4는 동시다발적인 하나의 HTTP 응답 그룹으로 판단하지 못한다. HTTP 요청 #4는 추후 재검증 루틴(T3)을 통해 재검증이 이루어진다.

[0161] 쿠키에 포함된 클라이언트 인증값 C가 서버(2)가 가지고 있는 제2 서버 인증값 S2와 동일하다면 서버(2)는 갱신 루틴인 의사 코드 모듈 T2를 수행한다. 클라이언트(1)에 제2 서버 인증값 S2에 상응하는 난수가 저장되었으므로, 서버(2)는 새로운 난수를 생성한다. 그리고 제2 서버 인증값 S2를 제1 서버 인증값 S1에 복사한다. 그리고 제2 서버 인증값 S2에는 새로운 난수와 공유키 K를 이용하여 미리 정해진 함수로 계산한 함수값을 저장한다. 그리고 타임스탬프는 현재 시각을 저장하고 카운터는 0으로 초기화한다. 그리고 클라이언트(1)에 현재 서버(2)가 새로 생성한 난수 R을 HTTP 응답에 포함시켜 전송한다.

[0162] 상술한 조건을 모두 만족하지 않는 경우 서버(2)는 재검증 루틴인 의사 코드 모듈 T3를 수행한다. 재검증 루틴(T3)은 반복 루틴(F1), 시간 루틴(F2), 삭제 루틴(F3)을 포함한다.

[0163] 반복 루틴(F1)에서, 서버(2)는 카운터를 1씩 증가시키면서 서버(2)가 저장하고 있는 난수 R과, 클라이언트(1)가 난수 R 및 클라이언트(1)에 저장된 공유키 K를 이용하여 계산한 새로운 클라이언트 인증값 C를 즉시 HTTP 요청하도록 하는 실행코드(예를 들어, 클라이언트측 스크립트, 플래시, 액티브X 등)를 포함하는 HTTP 응답을 클라이언트(1)에 전송한다(단계 S78). 이러한 HTTP 응답을 CFA 응답(CFA response)이라고 정의한다. 검증 타임스탬프(TV)는 현재 시각을 저장한다.

[0164] 클라이언트(1)는 수신한 CFA 응답에 따라 단계 S78에서 수신한 난수 R을 이용하여 계산한 새로운 클라이언트 인증값 C를 쿠키에 저장 또는 HTTP 요청에 get 또는post 형식으로 첨부하여 서버(2)에 전송한다(단계 S79).

[0165] 반복 루틴(F1)에서 송신자 인증이 될 때까지 소정 횟수 반복한다. 여기서, 소정 횟수는 최대 문턱 카운터(MAXc)에 저장된 값이다.

[0166] 소정 횟수를 초과하더라도 서버(2)는 시간 루틴(F2)을 수행하여, ctime()-검증 타임스탬프(TV)가 최대 검증 문턱 시간(MAXv) 이내라면 서버(2)가 저장하고 있는 난수 R과, 클라이언트(1)가 난수 R 및 클라이언트(1)에 저장된 공유키 K를 이용하여 계산한 새로운 클라이언트 인증값 C를 즉시 HTTP 요청하도록 하는 실행코드(예를 들어, 클라이언트측 스크립트, 플래시, 액티브X 등)를 포함하는 HTTP 응답을 클라이언트(1)에 전송한다.

[0167] 반복 루틴(F1)과 시간 루틴(F2)에 의해서도 송신자 인증이 안 되는 경우 최종적으로 송신자 인증이 실패한 것으로 간주하고 사용자 세션을 삭제한다(삭제 루틴(F3)). 그리고 사용자는 로그아웃된다.

[0168] 사용자가 로그아웃되거나 로그아웃하는 경우 서버(2)는 사용자 세션유지정보(공유키 K, 각종 상수들(MAXt, MAXc, MAXv), 각종 변수들(counter, TS, TV), 난수 R, 서버 인증값(S1, S2))를 삭제한다. 그리고 클라이언트(2)의 소정 저장 공간에 저장된 공유키 K, 난수 R, 쿠키 등을 모두 NULL 로 설정하도록 클라이언트(1)에 클라이언트측 스크립트, 플래시 또는 액티브X를 전송한다.

[0169] 도 6a는 본 발명의 다른 실시예에 따른 안전한 사용자 세션 관리 방법을 설명하기 위한 도면이고, 도 6b는 도 6a의 서버에서 실행되는 의사 코드이다.

[0170] 도 5a 및 5b에 도시된 일 실시예는 클라이언트 인증값, 서버 인증값을 생성함에 있어서 공유키 K를 직접 이용하는 것에 비해 도 6a 및 6b에 도시된 다른 실시예는 공유키 K와 키 변형 인수 K2를 이용하여 비밀키 K'를 생성하고, 비밀키 K'를 이용하여 클라이언트 인증값, 서버 인증값을 생성하는 것이 차이가 있다. 키 변형 인수를 이용하는 것은 도 4를 참조하여 상세히 설명하였는 바 상세한 설명은 생략하기로 한다.

[0171] 한편, 상술한 사용자 세션 관리 방법은 컴퓨터 프로그램으로 작성 가능하다. 상기 프로그램을 구성하는 코드들 및 코드 세그먼트들은 당해 분야의 컴퓨터 프로그래머에 의하여 용이하게 추론될 수 있다. 또한, 상기 프로그램은 컴퓨터가 읽을 수 있는 정보저장매체(computer readable media)에 저장되고, 컴퓨터에 의하여 읽혀지고 실행됨으로써 문서 탐색 서비스 제공 방법을 구현한다. 상기 정보저장매체는 자기 기록매체, 광 기록매체, 및 캐리어 웨이브 매체를 포함한다.

[0172] 상기에서는 본 발명에 대하여 그 실시예를 중심으로 설명하였지만, 해당 기술 분야에서 통상의 지식을 가진 자라면 하기의 특허 청구의 범위에 기재된 본 발명의 사상 및 영역으로부터 벗어나지 않는 범위 내에서 본 발명을 다양하게 수정 및 변경시킬 수 있음을 이해할 수 있을 것이다.

도면의 간단한 설명

[0173] 도 1은 본 발명에 적용되는 시도-응답 프로토콜의 인증 방법을 설명한 도면.

[0174] 도 2는 본 발명의 일 실시예에 따른 서버에 포함되는 세션 관리 모듈의 블록구성도.

[0175] 도 3은 안전한 사용자 세션 관리 방법, HTTP 요청의 송신자 인증 프로토콜을 설명한 도면.

[0176] 도 4는 다른 안전한 사용자 세션 관리 방법, 키 보호모드에서의 HTTP 요청의 송신자 인증 프로토콜을 설명한 도면.

[0177] 도 5a는 본 발명의 일 실시예에 따른 안전한 사용자 세션 관리 방법을 설명하기 위한 도면.

[0178] 도 5b는 도 5a의 서버에서 실행되는 의사 코드.

[0179] 도 6a는 본 발명의 다른 실시예에 따른 안전한 사용자 세션 관리 방법을 설명하기 위한 도면.

[0180] 도 6b는 도 6a의 서버에서 실행되는 의사 코드.

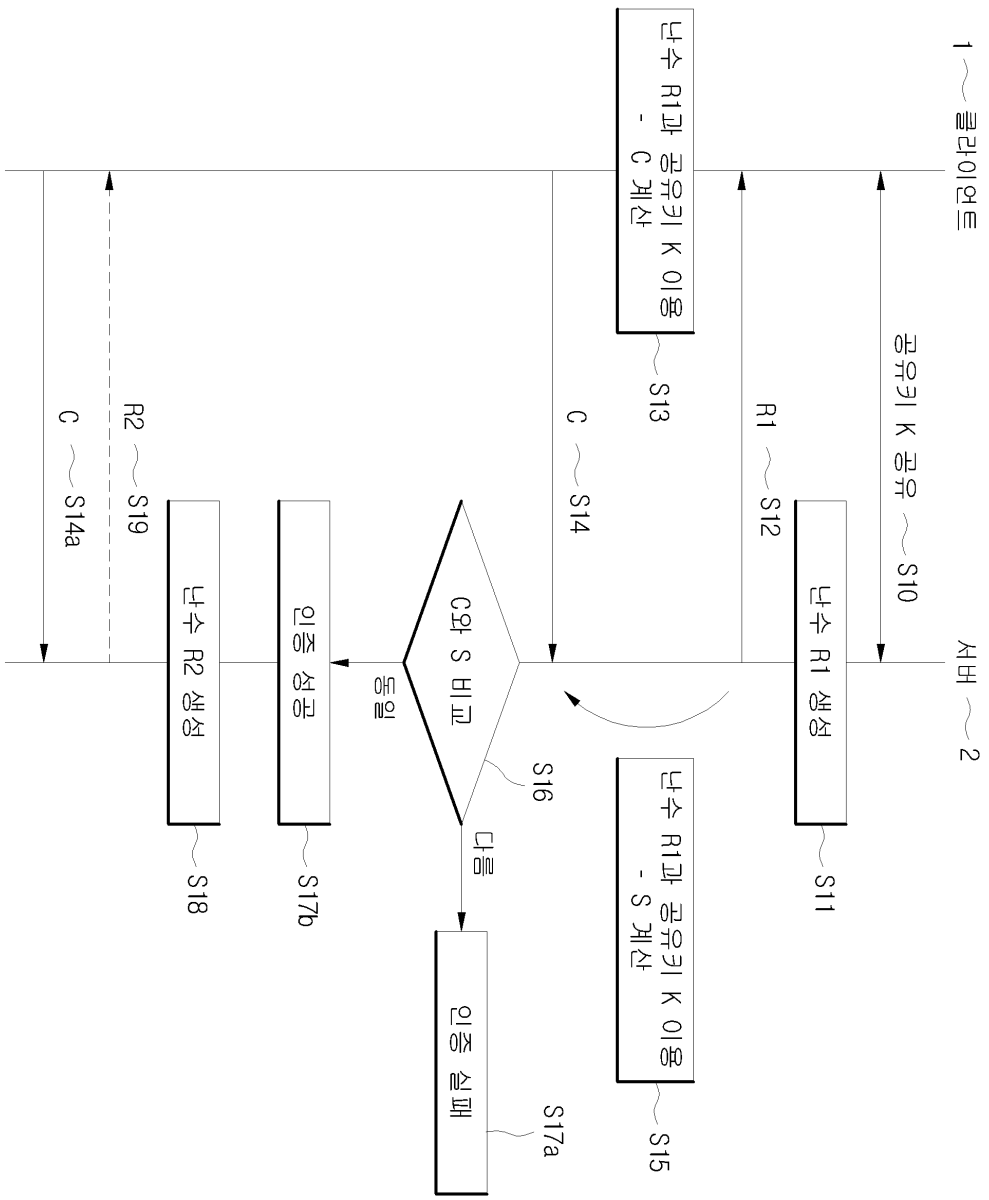
[0181] <도면의 주요부분에 대한 부호의 설명>

[0182] 1: 클라이언트

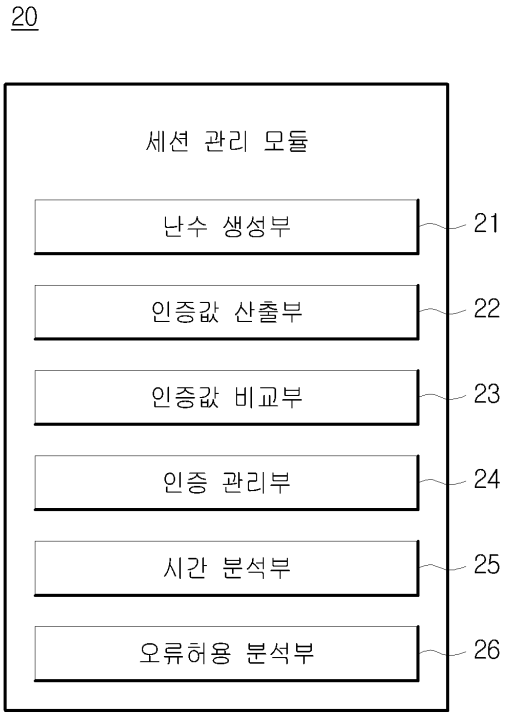
[0183] 2: 서버

도면

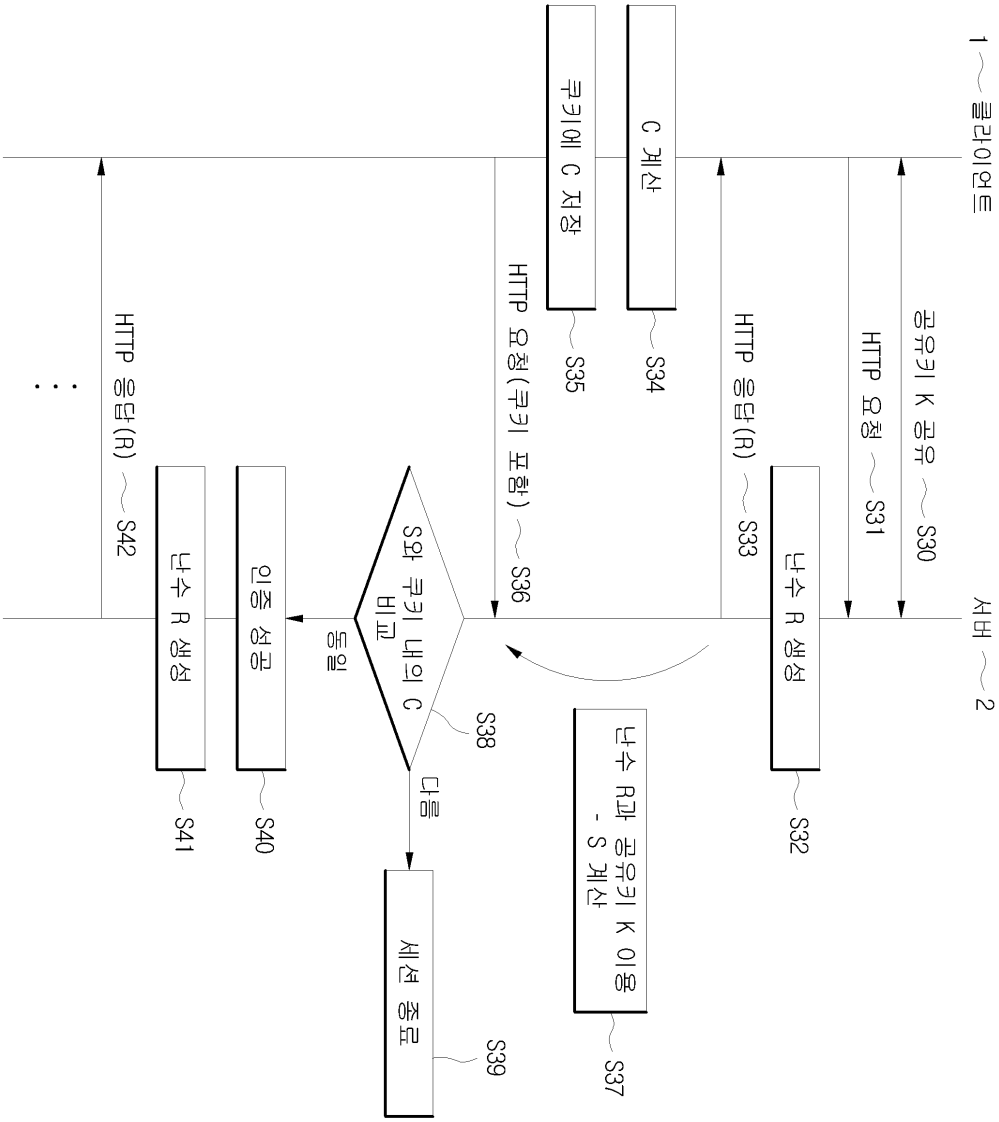
도면1


1 클라이언트
2 서버
공유키 K 공유
S10
난수 R1 생성
S11
R1
S12
난수 R1과 공유키 K 이용
- S 계산
S15
난수 R1과 공유키 K 이용
- C 계산
S13
C
S14
C와 S 비교
S16
동일
다름
인증 성공
S17b
인증 실패
S17a
난수 R2 생성
S18
R2
S19
C
S14a

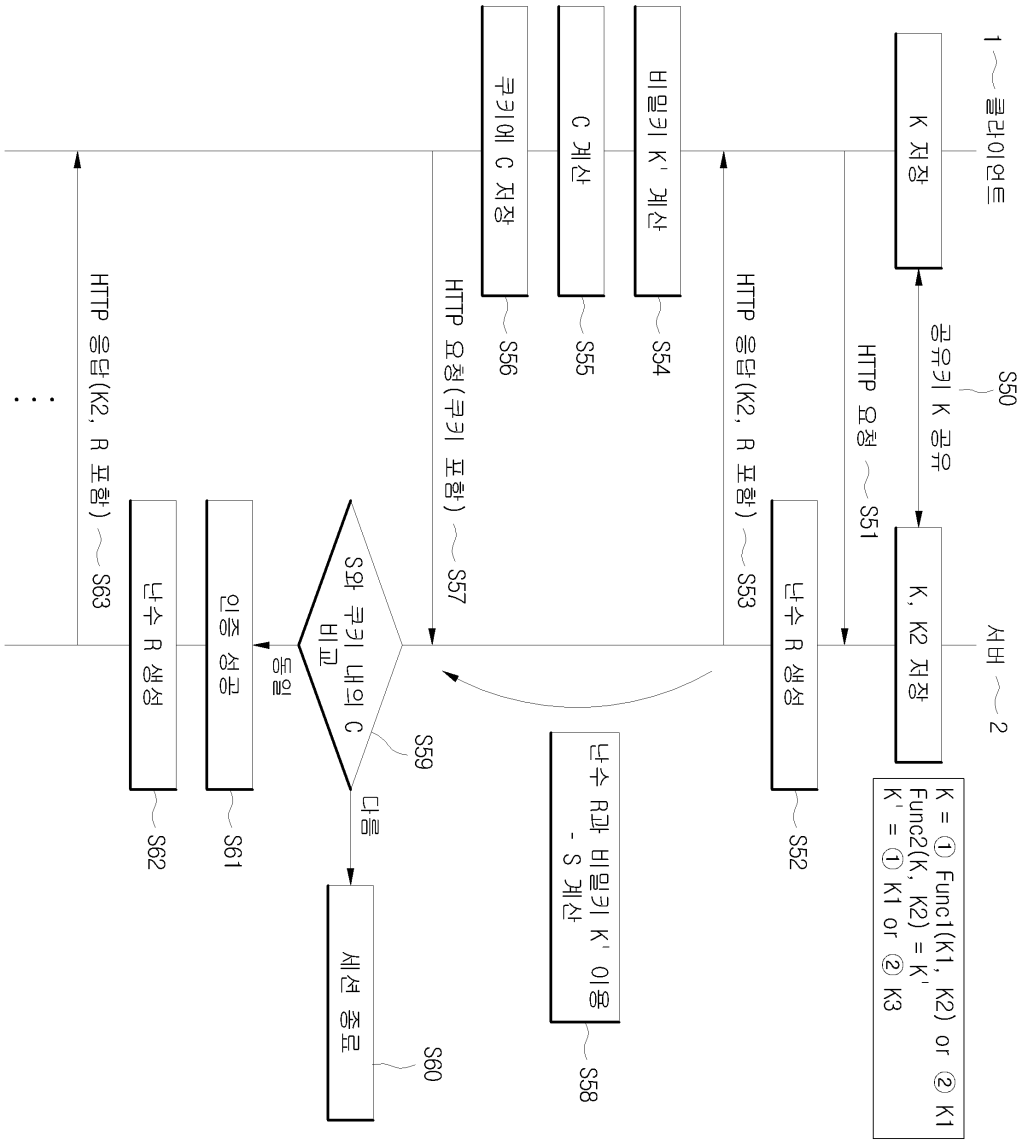
도면2


20
세션 관리 모듈
난수 생성부
21
인증값 산출부
22
인증값 비교부
23
인증 관리부
24
시간 분석부
25
오류허용 분석부
26

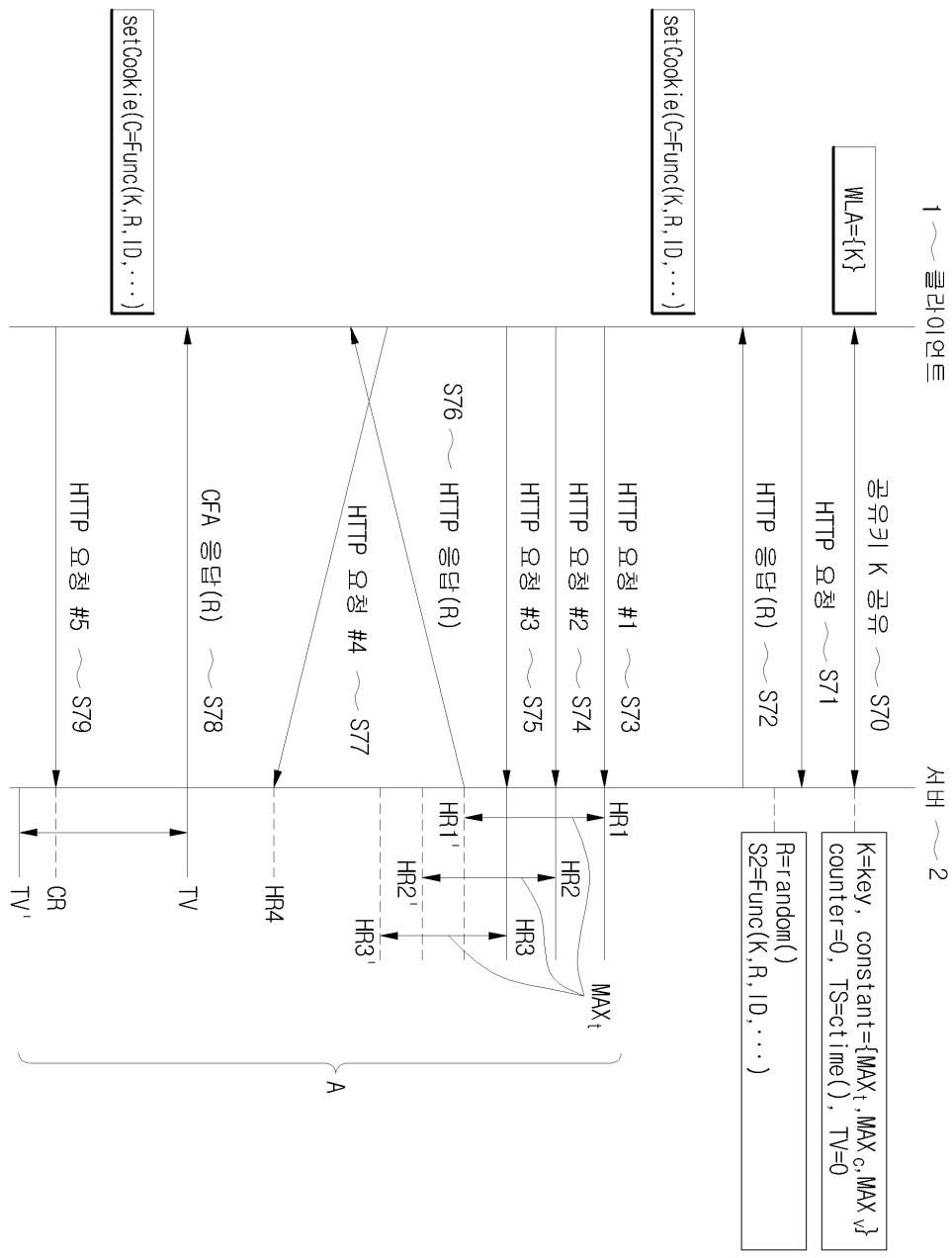
도면3


1 클라이언트
2 서버
공유키 K 공유 S30
HTTP 요청 S31
난수 R 생성 S32
HTTP 응답(R) S33
C 계산 S34
쿠키에 C 저장 S35
HTTP 요청(쿠키 표함) S36
난수 R과 공유키 K 이용 - S 계산 S37
S와 쿠키 내의 C 비교 S38
다름
동일
세션 종료 S39
인증 성공 S40
난수 R 생성 S41
HTTP 응답(R) S42
. . .

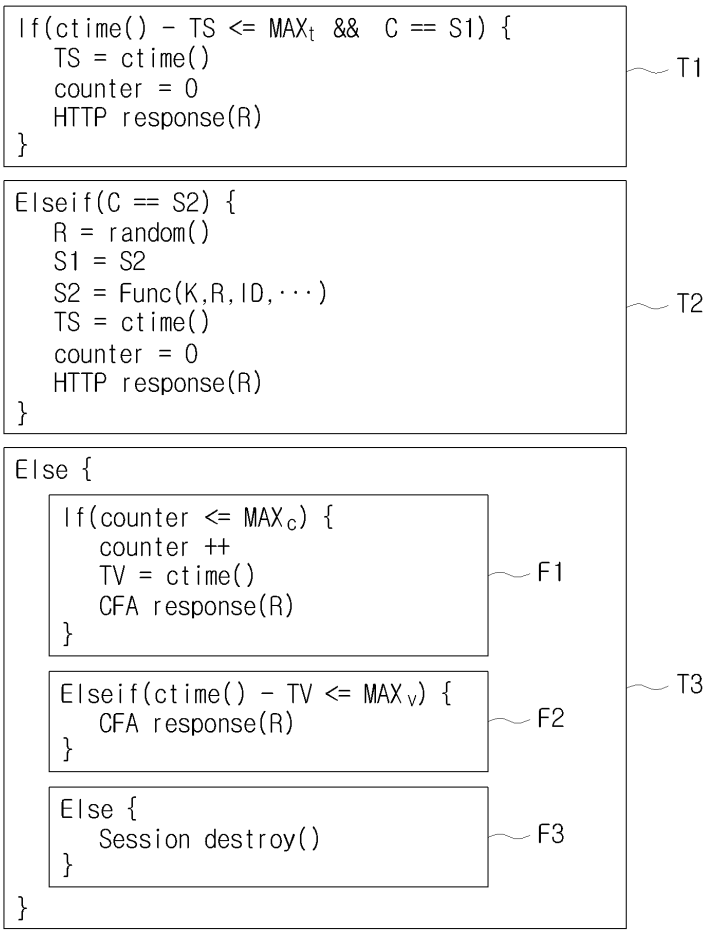
도면4


1 클라이언트
서버 2
K 저장
공유키 K 공유
S50
HTTP 요청
S51
K, K2 저장
K = ① Func1(K1, K2) or ② K1
Func2(K, K2) = K'
K' = ① K1 or ② K3
난수 R 생성
S52
HTTP 응답(K2, R 포함)
S53
비밀키 K' 계산
S54
C 계산
S55
쿠키에 C 저장
S56
HTTP 요청(쿠키 표함)
S57
난수 R과 비밀키 K' 이용
- S 계산
S58
S와 쿠키 내의 C
비교
S59
동일
다를
인증 성공
S61
난수 R 생성
S62
HTTP 응답(K2, R 포함)
S63
세션 종료
S60
...

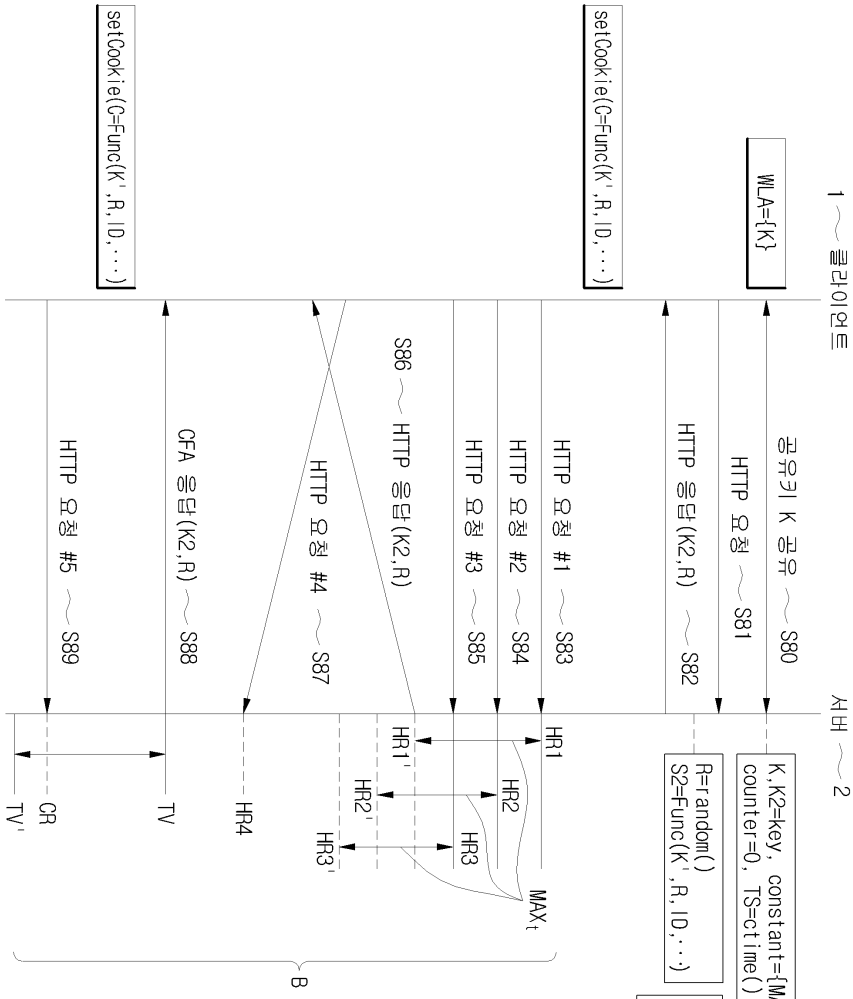
도면5a


1 ~ 클라이언트
서버 ~ 2
WLA={K}
setCookie(C=Func(K,R,ID,···))
setCookie(C=Func(K,R,ID,···))
공유키 K 공유 ~ S70
HTTP 요청 ~ S71
HTTP 응답(R) ~ S72
HTTP 요청 #1 ~ S73
HTTP 요청 #2 ~ S74
HTTP 요청 #3 ~ S75
HTTP 응답(R) S76 ~
HTTP 요청 #4 ~ S77
CFA 응답(R) ~ S78
HTTP 요청 #5 ~ S79
K=key, constant={MAXt,MAXc,MAXv}
counter=0, TS=ctime(), TV=0
R=random()
S2=Func(K,R,ID,···)
HR1
HR2
HR3
MAXt
HR1'
HR2'
HR3'
HR4
TV
CR
TV'
A

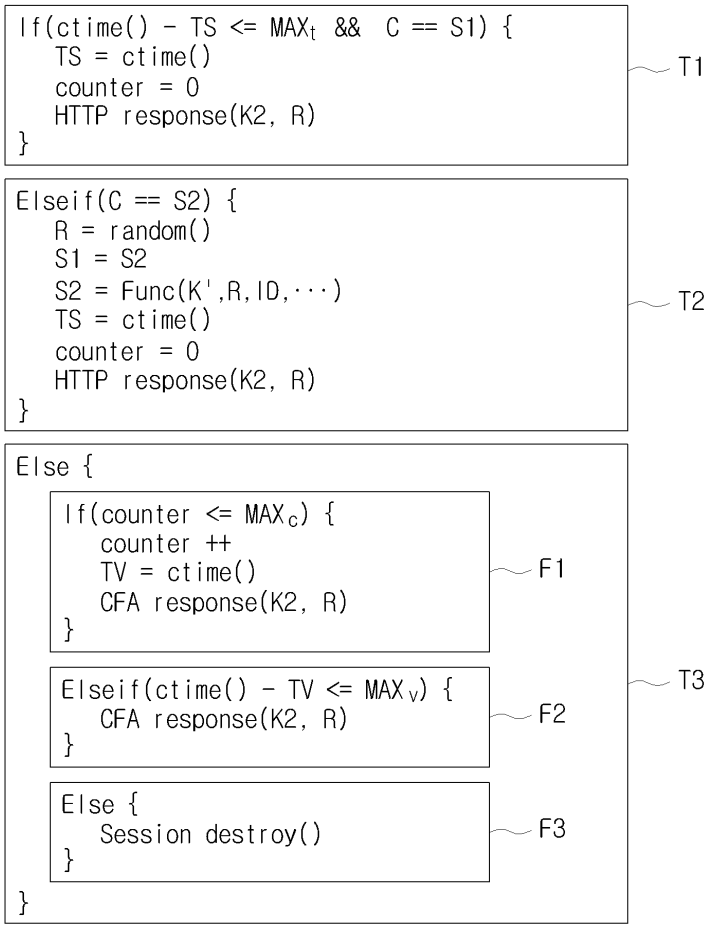
도면5b


If(ctime() - TS <= MAXt && C == S1) {
TS = ctime()
counter = 0
HTTP response(R)
}
T1
Elseif(C == S2) {
R = random()
S1 = S2
S2 = Func(K,R,ID,···)
TS = ctime()
counter = 0
HTTP response(R)
}
T2
Else {
If(counter <= MAXc) {
counter ++
TV = ctime()
CFA response(R)
}
F1
Elseif(ctime() - TV <= MAXv) {
CFA response(R)
}
F2
Else {
Session destroy()
}
F3
T3
}

도면6a


1 ~ 클라이언트
2 ~ 서버
WLA={K}
공유키 K 공유 ~ S80
HTTP 요청 ~ S81
HTTP 응답(K2,R) ~ S82
setCookie(C=Func(K',R,ID,···)
HTTP 요청 #1 ~ S83
HTTP 요청 #2 ~ S84
HTTP 요청 #3 ~ S85
S86 ~ HTTP 응답(K2,R)
HTTP 요청 #4 ~ S87
CFA 응답(K2,R) ~ S88
HTTP 요청 #5 ~ S89
setCookie(C=Func(K',R,ID,···)
K,K2=key, constant={MAXt,MAXc,MAXv}
counter=0, TS=ctime(), TV=0
R=random()
S2=Func(K',R,ID,···)
K = ① Func1(K1, K2) or ② K1
Func2(K, K2) = K'
K' = ① K1 or ② K3
HR1
HR2
HR3
MAXt
HR1'
HR2'
HR3'
HR4
TV
CR
TV'
B

도면6b


If(ctime() - TS <= MAXt && C == S1) {
TS = ctime()
counter = 0
HTTP response(K2, R)
}
T1
Elseif(C == S2) {
R = random()
S1 = S2
S2 = Func(K',R,ID,···)
TS = ctime()
counter = 0
HTTP response(K2, R)
}
T2
Else {
If(counter <= MAXc) {
counter ++
TV = ctime()
CFA response(K2, R)
}
F1
Elseif(ctime() - TV <= MAXv) {
CFA response(K2, R)
}
F2
Else {
Session destroy()
}
F3
}
T3