(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) Int. Cl.[7]
H04L 29/06

(45) 공고일자 2000년10월16일
(11) 등록번호 10-0268296
(24) 등록일자 2000년07월12일

(21) 출원번호	10-1998-0702879	(65) 공개번호	특1999-0066953
(22) 출원일자	1998년04월20일	(43) 공개일자	1999년08월16일
번역문제출일자	1998년04월20일		
(86) 국제출원번호	PCT/GB 96/00664	(87) 국제공개번호	WO 97/16911
(86) 국제출원일자	1996년03월20일	(87) 국제공개일자	1997년05월09일
(81) 지정국	EP 유럽특허 : 오스트리아 벨기에 스위스 독일 덴마크 스페인 프랑스 영국 그리스 이탈리아 룩셈부르크 모나코 네덜란드 포르투칼 스웨덴 국내특허 : 아일랜드 브라질 캐나다 중국 체코 헝가리 일본 대한민국 폴란드		
(30) 우선권주장	08/551260 1995년10월31일 미국(US)		

(73) 특허권자 인터내셔널 비지네스 머신즈 코포레이션 포만 제프리 엘
미국 10504 뉴욕주 아몬크
(72) 발명자 고어 로버트 세실
미국 텍사스주 78660 플러거빌 세다 레인 504
호프 존 프레데릭
미국 텍사스주 78728 오스틴 에멧 파크웨이 2302
(74) 대리인 김창세, 장성구

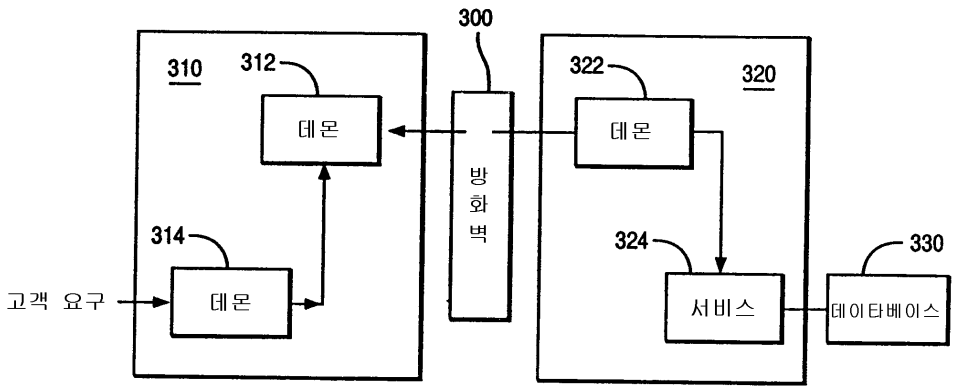
심사관 : 박재일

(54) 보안된 게이트웨이 인터페이스

요약

본 발명에 따른 컴퓨터로 구현된 방법, 유일하게 프로그래밍된 컴퓨터 시스템, 컴퓨터 판독가능한 프로그램 수단을 내포하는 제품은, 보안 방화벽(300)을 침해하지 않으면서, 외부 네트워크(310)상의 고객이, 내부 네트워크(320)상의 내부 비지니스 자원(330)을 이용하여 인증된 비지니스 트랜잭션을 개시할 수 있도록 해준다. 특히, 이 방법은, 내부 컴퓨터 시스템(320) 및 외부 컴퓨터 시스템(310)간의 보안 방화벽(300)을 침해하지 않으면서 외부 컴퓨터 시스템(310)이 내부 자원(330)을 이용하여 트랜잭션 요구(2)를 개시할 수 있도록 내부 컴퓨터 시스템(320)을 조작한다. 이 방법은, 내부 컴퓨터 시스템(320) 및 외부 컴퓨터 시스템(310)간에 내부 컴퓨터 시스템(320)에 의해 개시되는 접속을 인증해서, 인증된 접속을 설정하는 제 1 단계를 포함한다. 제 2 단계에서는, 외부 컴퓨터 시스템(310)에 의해 수신된 트랜잭션 요구(2)를 외부 컴퓨터 시스템(310)에 의해 호출한다. 트랜잭션 요구(2)를 호출하는 것에 응답하여, 제 3 단계에서는, 트랜잭션 요구(2)와, 트랜잭션 요구(2)를 실행하기 위한 인자, 및 프로세스 환경 변수를 포함하는 스트링을 외부 컴퓨터 시스템(310)에 의해 생성한다. 제 4 단계에서는, 인증된 접속을 통해 내부 컴퓨터 시스템(320)으로 스트링을 외부 컴퓨터 시스템(310)에 의해 전송한다. 제 5 단계에서는, 트랜잭션 요구(2)를 내부 컴퓨터 시스템(320)에 의해 검증한다. 제 6 단계에서는, 원래의 프로세스 환경을 내부 컴퓨터 시스템(320)에 의해 재생한다. 마지막 단계에서는, 트랜잭션 요구(2)를 내부 컴퓨터 시스템(320)에 의해 실행해서 출력을 생성한다.

대표도



명세서

기술분야

본 발명은 네트워크에서 이용되는 보안된 게이트웨이 인터페이스(secured gateway interfaces)에 관한 것으로, 특히 외부 네트워크 사용자가 보안 방화벽(security firewalls)을 침해하지 않으면서 내부 자원을 이용하여 인증된 트랜잭션을 개시할 수 있도록 하는 보안 게이트웨이 인터페이스에 관한 것이지만, 이에 제한되는 것은 아니다.

배경 정보 및 관련 분야의 기술

도 1은, 외부 네트워크(예를 들어, 인터넷)상의 임의의 사용자로부터의 사용자/고객 요구(2)를 프로세싱하는 외부 서버(4)(예를 들어, 하이퍼텍스트 전송 프로토콜 데몬(hypertext transfer protocol daemon, HTTPD))와, 내부 네크워크상의 임의의 사용자(예를 들어, 특정 회사를 위해 일하는 임의의 사용자)로부터의 요구를 프로세싱하는 내부 서버(7) 및 내부 데이타베이스(8)를 구비하는 종래의 보안된 게이트웨이 인터페이스(SGI)(9)를 도시한다. SGI(9)는 내부 데이타베이스(8)상의 내부 트랜잭션을 외부에서 개시하는 것을 방지하기 위한 방화벽(6)도 또한 포함한다. 따라서 방화벽(6)은, 외부 고객이 내부 네트워크(즉, 내부 서버(7) 및 내부 데이타베이스(8))에 직접 접속하는 것을 방지한다. 이 제한으로 인해, 제품 및 서비스 구매 요구와 같은 유효한 트랜잭션을 못하게 되는데, 이는 단지 고객이 외부 네트워크로부터 내부 트랜잭션을 개시할 수 없다는 이유 때문이다. 이러한 종래의 방화벽은, 1994년 9월에 발행된 미국의 IEEE 통신 매거진, vol.32, no.9의 페이지 50-57에 에스 엠 벨로윈(S. M. Bellowin)등에 의해 기술되어 있다.

전술한 문제점을 해결하기 위해, 종래에는 외부에서 들어오는 트래픽(inbound traffic)에 대해 특정 포트(예를 들면, 포트(84))를 개방하였다. 그러나, 이 해결책에 의하면 내부 네트워크가 외부의 침해에 노출될 것이 분명하다. 다른 해결책은, 요구되는 모든 자원(예를 들면, 데이타베이스(8))을 외부 서버(4)상에 위치시키는 것이다. 그러나, 이 해결책은 계속해서 내부 트랜잭션의 실행을 막는다. 또한, 외부 서버(4)가 요구되는 모든 자원을 보유하기 위한 충분한 저장 능력이 없거나, 혹은 자원이 비밀을 요하는 것이어서(예를 들어, 고객 데이터) 외부 서버상에 위치시킬 수 없는 이유로 인해, 제공될 수 있는 서비스가 제한된다.

따라서, 보안 방화벽을 침해하지 않으면서, 고객이 내부 비지니스 자원을 이용하여 인증된 비지니스 트랜잭션을 할 수 있도록 하는 기술이 강력히 요구되고 있다.

발명의 개시

따라서, 컴퓨터로 구현된 방법과, 유일하게(uniquely) 프로그래밍된 컴퓨터 시스템과, 컴퓨터로 판독가능한 프로그램 수단을 포함한 제품은, 외부 네트워크상의 고객이 보안 방화벽을 침해하지 않으면서 내부 네트워크상의 내부 비지니스 자원을 이용하여 인증된 비지니스 트랜잭션을 할 수 있게 해준다.

특히 이 방법은, 외부 컴퓨터 시스템이 내부 컴퓨터 시스템과 외부 컴퓨터 시스템간의 보안 방화벽을 침해하지 않으면서 내부 자원을 이용하여 트랜잭션 요구를 개시할 수 있도록 내부 컴퓨터 시스템을 조작한다. 이 방법은, 내부 컴퓨터 시스템과 외부 컴퓨터 시스템간에 내부 컴퓨터 시스템에 의해 개시되는 접속을 인증하여 인증된 접속을 설정하게 되는 제 1 단계를 포함한다. 제 2 단계에서는, 외부 컴퓨터 시스템에 의해 수신된 트랜잭션 요구를, 외부 컴퓨터 시스템에 의해 호출한다. 트랜잭션 요구의 호출에 응답하여, 제 3 단계에서는, 트랜잭션 요구와, 트랜잭션 요구를 실행하기 위한 프로세스 환경 변수를 포함하는 스트링을, 외부 컴퓨터에 의해 생성한다. 제 4 단계에서는, 인증된 접속을 통해 스트링을 내부 컴퓨터 시스템으로 외부 컴퓨터 시스템에 의해 전송한다. 제 5 단계에서는, 트랜잭션 요구를 내부 컴퓨터시스템에 의해 검증한다. 제 6 단계에서는, 원래의 프로세스 환경을 내부 컴퓨터 시스템에 의해 재생한다. 마지막 단계에서는, 트랜잭션 요구를 내부 컴퓨터 시스템에 의해 수행하여 출력을 생성한다.

따라서, 본 발명의 목적은, 사용자 및 실질적인 트랜잭션 프로그램에 투명한 보안된 게이트웨이 인터페이스를 제공하는 것이다.

본 발명의 다른 목적은, 사용자가 방화벽을 통해 내부 자원을 이용하여 내부 네트워크에 대한 트랜잭션을 유효하게 개시할 수 있도록 하는 것이다.

본 발명의 또다른 목적은, 사용자가 인증된 트랜잭션의 유효 세트만을 개시할 수 있도록 하는 것이다.

본 발명의 또다른 목적은, 외부 컴퓨터 시스템이 사용자로부터 트랜잭션 요구를 받기 전에 내부 컴퓨터 시스템 및 외부 컴퓨터 시스템간의 접속을 확실하게 인증하는 것이다.

본 발명의 또다른 목적은, 트랜잭션 프로그램을 이를 수정할 필요없이 방화벽 내부에 저장하는 것이다.

도면의 간단한 설명

도 1은 본 발명을 구현하는데 사용되는 전형적인 네트워크 시스템의 블럭도.

도 2는 본 발명을 구현하기 위한 대표적인 하드웨어 구성을 나타내는 도면.

도 3은 바람직한 실시예에 따른 보안된 게이트웨이 인터페이스(SGI)의 블럭도.

도 4는 상기 도 3에 도시된 SGI에 대한 보다 상세한 프로세스 흐름도.

발명의 상세한 설명

본 발명의 바람직한 실시예는, 외부 사용자/고객이 보안 방화벽을 침해하지 않으면서 내부 비지니스 자원을 이용하여 인증된 비지니스 트랜잭션을 개시하도록 내부 컴퓨터 시스템을 조작하는, 컴퓨터로 구현된 방법, 유일하게 프로그래밍된 컴퓨터 시스템, 및 상세한 논리를 내포하는 메모리를 포함한다.

본 발명은 도 2에 도시된 컴퓨터 시스템상에서 수행된다. 컴퓨터 시스템(100)은, 프로세싱하기 위한 IBM사의 PowerPC 601 또는 Intel사의 486 마이크로프로세서와 같은 중앙 처리 장치(CPU)(10), 캐시(15), RAM(14), ROM(16), 및 비휘발성 RAM(NVRAM)(32)을 구비한다. I/O 어댑터(18)에 의해 제어되는 하나 이상의 디스크(20)는 장기간의 저장 능력을 제공한다. 테이프, CD-ROM, 및 WORM 드라이브를 포함하여 다양한 다른 저장 매체가 사용될 수 있다. 데이터 또는 컴퓨터 프로세스 인스트럭션을 저장하는데 이동가능한 저장 매체도 또한 제공될 수 있다. (IBM 및 PowerPC는 IBM사의 상표이며, Intel은 Intel사의 상표이다).

Sun Solaris, Microsoft의 Windows NT, IBM의 OS/2, 또는 Apple의 System 7과 같은 임의의 적절한 운영 체제를 갖는 데스크탑 컴퓨터로부터의 인스트럭션 및 데이터는 RAM(14)으로부터 CPU(10)를 제어한다. 따라서, 데스크탑 컴퓨터는 RAM(14)으로부터 실행된다. 그러나, 바람직한 실시예에서, IBM RISC System/6000은 AIX 운영 체제를 실행하는데, 이 AIX 운영 체제는 IBM사가 구현한 UNIX 운영 체제이다. 그러나, 전술한 바와 같이, 본 기술 분야에 통상의 지식을 가진 자라면, 본 발명을 구현하는 데에 그 밖의 다른 하드웨어 플랫폼 및 운영 체제가 사용될 수 있음을 쉽게 알 것이다. (Solaris는 선 마이크로시스템사의 상표이며, Windows NT는 마이크로소프트사의 상표이고, System 7은 애플 컴퓨터사의 상표이며, OS/2와 RISC System/6000과 AIX는 IBM사의 상표이다. UNIX는, 미국 및 다른 나라에서 등록된 상표로서, X/Open 주식회사에 배타적으로 허여되어 있다.)

사용자는 사용자 인터페이스 어댑터(22)에 의해 제어되는 I/O 장치(즉, 사용자 제어 장치)를 통해 컴퓨터 시스템(100)과 통신한다. 디스플레이(38)는 사용자에게 정보를 디스플레이하며, 키보드(24), 포인팅 장치(26), 및 스피커(28)는 사용자로 하여금 컴퓨터 시스템을 조작할 수 있게 해준다. 통신 어댑터(34)는 이 컴퓨터 시스템과, 네트워크에 접속된 다른 프로세싱 유닛간의 통신을 제어한다. 디스플레이 어댑터(36)는 이 컴퓨터 시스템과 디스플레이(38)간의 통신을 제어한다.

도 3은 바람직한 실시예에 따른 보안된 게이트웨이 인터페이스(SGI)의 블럭도 및 프로세스 흐름도이다. SGI는 한 쌍의 서버(310, 320)상에 상주하며, 각 서버는 컴퓨터 시스템(100)상에서 구현된다(도 2 참조). 외부 서버(310)는 방화벽(300)의 외부에 상주하며, 내부 서버(320)는 방화벽(300)의 내부에 상주한다. 방화벽(300)은, 외부 트랜잭션이 방화벽을 통과해서 내부 서버(320)에 접근하는 것을 방지하는 임의의 적절한 전형적인 방화벽을 이용하여 구현된다. 바람직한 실시예에서, 방화벽(300)은 네트워크 라우터(network router)(예를 들어, Cisco 라우터)이다. 그러나, 본 기술 분야에 통상의 지식을 가진 자라면, 방화벽(300)이 내부 서버(320)내에 상주할 수 있음을 쉽게 알 것이다.

외부 서버(310)는 인터넷과 같은 외부 네트워크상에서 사용자/고객과의 통신을 관리한다. 그러나, 본 기술 분야에 통상의 지식을 가진 자라면, 임의의 공중 또는 사설 네트워크상에서는 SNA 또는 X.25와 같은 어떠한 타입의 통신 프로토콜도 사용될 수 있음을 알 것이다. 내부 서버(320)는, 내부 기업 정보 네트워크와 같은 내부 네트워크상에서 내부 자원(예를 들어, 데이타베이스(330))에 대한 통신을 관리한다. 외부 서버(310)는 외부 데몬(312)을 가동시키고, 내부 서버(320)는 내부 데몬(322)을 가동시킴으로써, 방화벽(300)을 통한 통신이 가능하게 된다. 데몬은 장시간 작동하는 컴퓨터 프로그램으로서, 외부 이벤트를 기다리며, 이러한 이벤트가 발생될 때마다 사전정의된 일련의 동작을 수행한다. 데몬은 서비스 요구를 기다리고 있다가, 요구가 발생하면 이를 수행한다. 외부 서버(310)는 데몬(314)도 또한 가동시키는데, 이 데몬(314)은 외부 네트워크로부터 서비스 요구를 기다린다. 내부 서버(320)는 원하는 내부 트랜잭션을 수행하기 위한 서비스 프로그램(324)을 포함한다. 서비스 프로그램(324) 및 내부 데이타베이스(330)는 비지니스 트랜잭션(이하 상세히 기술됨)을 구현하는 컴퓨터 프로그램 세트를 나타낸다.

도 4는 상기 도 3에 도시된 SGI의 프로세스 흐름을 좀 더 상세하게 나타낸 도면이다. 외부 서버(310)는 임의의 적절한 전형적인 통신 프로토콜 데몬(31), cgi-bin(415), sgi_client 루틴(416), 및 외부 데몬(312)을 포함한다. 외부 데몬(312)은 sgi_client 루틴(416) 및 내부 데몬(322)과 통신하기 위한 클라이언트/서버 소프트웨어를 구비한다. sgi_client 루틴(416)은 외부 데몬(312)과 통신하기 위한 클라이언트/서버 소프트웨어를 구비한다. cgi-bin(415)은 데몬(314)에 의해 수행되는 소프트웨어의 디렉토리이다. 특히, 이 예에서, cgi-bin(415)은 sgi_client 루틴(416)(이하에 상세히 기술됨)과 통신하기 위한 특별한 펄 스크립트(special perl script)인 example.pl(462)을 포함한다. 바람직한 실시예에서, 데몬(314)은 전형적인 하이퍼텍스트 전송 프로토콜 데몬(httpd)(통상 웹 서버로도 또한 알려져 있음)이다.

내부 서버(320)는 내부 데몬(322), 서비스 프로그램(324), 및 cgi-bin(426)을 포함한다. 서비스 프로그램(324)은 외부 데몬(312) 및 내부 데몬(322)과 통신하며, cgi-bin 루틴(예를 들어, example.pl(480))을 실행한다. 이 예에서, example.pl(480)은 사용자/고객을 인증하고 비지니스 트랜잭션을 수행하기 위해 내부 기업 데이타베이스(예를 들어, 도 3의 회사 데이타베이스(330))와 통신한다.

고객/사용자가 (410)에서 트랜잭션을 성공적으로 요구할 수 있기 전에, 내부 서버(320) 및 외부 서버(310)는 적절하게 접속되어야 한다. 이를 위해, 외부 운영 체제는 외부 데몬(312)을 실행시키며, 외부 서버(310)상에서 통신 포트와, 파일시스템(도시하지 않음)에 상주하는 패스워드 파일의 위치를 식별한다. 그 후, 외부 데몬(312)은 패스워드 파일로부터 8개의 문자 패스워드를 판독하고, 식별된 통신 포트에 소켓(socket)을 생성하며, 그 소켓에서 내부 데몬(322)으로부터의 접속 호출을 기다린다. 따라서, 외부 데몬(312)은 서버의 역할을 하는 것으로 간주되며, (430)에서, 클라이언트의 역할을 하는 것으로 간주되

는 내부 데몬(322)으로부터의 접속 호출을 기다린다. 또한, 외부 데몬(312)은 제 2 포트(데몬(312) 통신 포트+1)에 소켓을 생성하며, (432)에서 sgi_client 루틴(416)으로부터 접속 시도를 기다린다(이하 상세히 기술됨).

내부 운영 체제는 내부 데몬(322)을 실행시키며, 외부 데몬(312)에 내부 데몬(322)을 접속하기 위한 통신 포트, 외부 서버(310)의 호스트네임, 내부 서버(320)내의 파일 시스템(도시하지 않음)에 상주하는 패스워드 파일의 위치, 내부 서버(326)내의 파일 시스템(도시하지 않음)에 상주하는 유효한 서비스 파일의 위치를 식별한다. 그 후, 내부 데몬(322)은 패스워드 파일로부터의 8개의 문자 패스워드를 판독하고, 서비스 파일을 판독하여 유효한 서비스의 테이블을 메모리에 저장하고, 식별된 통신 포트에 소켓을 생성하며, (450)에서 방화벽(300)을 통하여 외부 데몬(312)으로 향하는 표준 접속 호출을 발생시키는데, 이러한 호출은 (430)에서 수신된다. 이 접속은 내부 서버로부터 개시되고 있는 것이기 때문에, 방화벽(300)은 이 접속을 허용한다.

내부 데몬(322) 및 외부 데몬(312)이 접속된 후, 내부 데몬(322) 및 외부 데몬(312)은 서로를 적절하게 인증해야 한다. 이를 위해, 내부 데몬(322)은 내부 운영 체제를 호출하여 현재의 타임 스탬프를 검색하고, 이 타임 스탬프를 외부 데몬(312)으로 전송하고, 회신되는 인증 스트링을 기다린다. 외부 데몬(312)은 이 타임 스탬프를 수신하고, 내부 데몬(322)에 의해 제공된 이 타임 스탬프로 8개의 문자 패스워드를 변조(mangling)(이하에 기술됨)함으로써 인증 스트링을 생성하고, 이 변조된 문자 스트링을 표준 UNIX 암호 커맨드(또는 DES와 같은 임의의 적절한 암호화 알고리즘)로 암호화한 후, (431)에서 이 인증 스트링을 내부 데몬(322)으로 전송한다. 이하의 C 코드는 8개의 문자 패스워드를 타임 스탬프로 변조하는 프로세스를 나타낸다. 이 "create_auth" 코드는 세 개의 인자(arguments)를 필요로 하는데, 그 첫 번째는 타임 스탬프(즉, auth_time)이며, 두 번째는 패스워드(즉, 패스워드에 대한 포인터인 "cred")이며, 세 번째는 생성된 인증 스트링을 저장하기 위한 버퍼이다.

```
int create_auth  (time_t, char * cred, char * p)
{
        char     buf[9];   /* temporary buffer */
        int      i;

        bzero  (buf, sizeof(buf));  /* clear buffer */
        strcpy  (buf, cred);        /* load buffer with password */

        /* mangle each character of the buffer */

        for (i = 0;i < 8;i++)  {
                buf[i] ^= (auth_time & 0177); /*logically ANDing timestamp, then
                                                exclusive ORing result with each
                                          character in buffer; with each
                                      iteration
                                                modifying the timestamp */
                auth_time >>=4;  /* bit wise move timestamp */
         {
         for (i = 0;i < 8;i++)
                 if   (buf[i] == 0)  /* since a valid character string cannot contain
                                        a NULL, chang all NULLS to 1 */
                         buf[i] = 1;

         strcpy  (p,  crpyt  (buf, "aa") + 2);  /* encrypt buffer using aa for
                                                   the key */
                                                /* skip first two characters of
                                                   encryption result (which is the
                                                   key aa) */
                                                /* copy the encryption result to user
                                                   supplied buffer pointed to by P */

           return 0;
    }
```

마찬가지로, 내부 데몬(322)은 타임 스탬프로 패스워드를 변조하고, 이를 암호화하고, 이를 외부 데몬(312)에 의해 제공되는 인증 스트링과 비교한다. 인증 스트링들이 일치할 경우, 프로세스는 되돌아가며, 외부 데몬(312)은 마찬가지로 내부 데몬(322)을 인증한다(즉, 외부 운영 체제로부터 새로운 타임 스탬프를 획득하고, 이 타임 스탬프를 내부 데몬(322)에 전송하며, 내부 데몬(322)은 패스워드를 새로운 타임 스탬프로 변조하고, 이를 암호화하며, 확인을 위해 이를 외부 데몬(312)에 도로 전송한다).

이 인증 프로세스는 외부 및 내부 데몬(312, 322) 모두에 의해 알려진 8개의 문자 패스워드, 타임 스탬프에 의해 무작위로 추출된 문자 변조 함수와, 암호화 프로세스를 사용한다. 변조 함수 때문에, 전술한 프로세스는 각 인증 및 모든 트랜잭션마다 상이한 암호화된 인증 스트링을 제공하게 된다. 이렇게 하면, 포착된 인증 스트링이 어떠한 후속 트랜잭션에 대해서도 쓸모없게 되기 때문에 침해당할 염려가 상당히 감소된다.

내부 데몬(322) 및 외부 데몬(312)이 서로 인증된 후, 이전에 클라이언트로 작용한 내부 데몬(322)은 이제 서버의 역할을 맡게 되고, 외부 데몬(312)이 (453)에서 서비스 스트링을 제공하기를 (452)에서 기다린다. 외부 데몬(312)은 제 2 지정 포트에 다른 소켓을 생성하며, (432)에서 sgi_client 루틴(416)으로부터의 접속 시도를 기다린다. 따라서, 외부 데몬(312)은, 내부 데몬(322)에 대해서는 의사 클라이언트로서, 그리고 sgi_client 루틴(416)에 대해서는 서버로서 이중 역할(정보는 이들 사이에서 전달됨)을 하게 되는 것으로 간주된다.

이제, 데몬(314)은 고객 요구(410)를 받아들일 준비가 되어 있다. 고객 요구는, 예를 들어 특정 증권 또는 금융 시장에 대한 리서치 정보를 구매하는 트랜잭션일 수 있다. (410)에서, 고객은, http 클라이언트 애플리케이션 사용자 인터페이스를 가동시키고 있는 고객의 시스템상에서 특정 아이콘 또는 강조된(highlighted) 문구를 클릭함으로써, "http://external_server/cgi-bin/example.pl?stock1+stock2"로 표

현되는 트랜잭션 요구를 실행할 것을 결정한다. 전형적으로, http 클라이언트 사용자 인터페이스는 사용자에게, 청구서 작성 정보(예를 들어, 신용 카드 번호) 뿐만 아니라 상세한 트랜잭션 정보(예를 들어, 어떤 증권 또는 금융 시장인지에 대한 정보)를 요구한다. 또한, 요구된 서비스가 인증된 사용자에게만 제공되는 경우, 사용자 식별 번호(userid) 및 패스워드를 입력할 것을 사용자에게 요구할 수도 있다.

전송된 사용자 입력의 포맷은 트랜잭션을 구현하는데 사용되는 하이퍼텍스트 마크업 언어(HTML) 형식의 타입에 따라 달라진다. 전형적으로, 두 가지 타입의 HTML 형식이 존재하며, "GET" 형식은 모든 사용자 입력을 커맨드 라인상에 위치시킨다. 따라서, stock1, stock2, 및 그 밖의 다른 임의의 사용자 입력은 커맨드 라인, 즉 .../cgi-bin/example.pl?stock1+stock2+chargecardnumber+expdate의 일부가 될 수 있다.

그러나, 커맨드 라인이 순전히 텍스트로 네트워크를 통하여 전송될 것이기 때문에, 네트워크를 통하여 고객의 신용 카드 번호 및 만기일을 전송하는 것은 적합하지 않다. 따라서, 신용 카드 번호 및 만기일이 네트워크를 통해 안전하게 전송되도록, 암호화를 수반한 "PUT" 타입의 HTML 형식이 사용된다. 이러한 모든 정보를 제공한 후, http 클라이언트 어플리케이션은 (410)에서 http를 통해 외부 서버(310)로 요구를 전송한다.

(460)에서, 데몬(314)은 통상적으로 공지되고 내장된 HTTP 인증 기법(예를 들어, 표준 UNIX 암호 커맨드로 고객의 패스워드를 암호화하고, 그 결과를 데몬(314)에 상주하는 http 패스워드 파일내의 패스워드 엔트리와 비교하는 기법)에 따라 고객의 패스워드를 인증한다. 사용자 식별 번호 및 패스워드가 유효할 경우, (461)에서, 데몬(314)은 "PUT" 형식을 인식하고, 자동적으로 문자 스트림에 대한 암호해독을 실행하며, 적절한 UNIX 프로세스 환경을 생성한다. 데몬(314)은, PATH, USERNAME, LOGNAME, 및 AUTHTYPE 변수를 포함하여 표준 UNIX 프로세스 환경을 생성하기 위한 전형적이고 통상적으로 알려진 http 구성 파일(도시하지 않음)을 구비한다. 나중에, httpsvc(470)는 (471)에서 이 프로세스 환경을 재생한다(이하 기술됨). 프로세스 환경이 일단 생성되면, 데몬(314)은 example.pl(462)(이는 cgi-bin(415)내에 상주해야 함)을 실행시켜서, 임의의 요구된 인자(예를 들어, stock1 및 stock2)와, 사용자 입력을 example.pl(462)의 표준 입력 스트림에 전송한다.

example.pl(462)이 cgi-bin(415)에 상주한다고 가정하고, 방화벽(300)이 존재하지 않을 경우, example.pl(462)은 내부 데이타베이스(330)(도 3 참조)와 직접 통신하고, 원하는 트랜잭션을 수행할 것이다. 그러나, 방화벽(300)이 존재하여, example.pl(462)이 내부 데이타베이스(330)와 직접 통신할 수 없기 때문에, example.pl(462)은 실질적인 트랜잭션 프로그램이 아니다. 오히려, 실질적인 트랜잭션 프로그램은 방화벽(300)의 내부에 존재하는 cgi-bin(426)내에 example.pl(480)로서 상주한다. 따라서, cgi-bin(415)은, cgi-bin(426)에 상주하는 실질적인 트랜잭션 프로그램을 실행시킬 수 있는 것과 동일한 커맨드를 사용하여 수행되는 "특별한" 펄 스크립트(예를 들어, example.pl(426))를 포함한다. 이와는 달리, 외부 서버(310)가 많은 서비스를 제공하고, 이들 서비스 각각이 동일한 방식으로 sgi_client 루틴(416)을 호출할 것을 "특별한" 펄 스크립트에게 요구할 경우, example.pl(462)은 cgi-bin(415)에 상주하는 단일 펄 스크립트에 대한 기호 링크(즉, 간접적인 파일명 참조)일 수 있다. 중요한 점은 고객이 이용가능한 요구가 각각 cgi-bin(426)에 상주하는 펄 스크립트 및 대응하는 트랜잭션 프로그램과 cgi-bin(426)으로 제한된다는 것이다.

펄 스크립트 example.pl(462)은 데몬(314)으로부터 전달되어 온 모든 인자를 프로세스 환경내에 위치시키며(예를 들면, SGIARG1=stock1;SGIARG2=stock2), 그 명칭(호출시 사용되는 명칭, 이 경우 example.pl)을 프로세스 환경내에 위치시키고(예를 들면, SGICMD=example.pl), (프로세스 환경 변수를 덤프(dump)하는) UNIX env 커맨드를 실행시키며, 최종적으로 모든 프로세스 환경 변수를 헤더 스트링에 위치시킨다. 이제, 헤더 스트링은 예를 들면 이하와 같이 나타난다.

"PATH=/bin:/usr/bin□AUTHTYPE=PEM□USERNAME=JohnDoe□SGIARG1=stock1□SGIARG2=stock2W=nSGICMD=example.pl"

다음에, (463)에서, 펄 스크립트 example.pl(462)은 외부 운영 체제를 호출하여, 지정 제 2 포트(데몬(312) 통신 포트+1)를 검색하고 sgi_client 루틴(416)을 실행시켜서, 요구된 서비스 타입(예를 들어, httpsvc), 지정된 제 2 포트, 외부 서버 호스트네임, 헤더 스트링, 및 고객의 사용자 식별 번호를 전송한다. example.pl(462)은 또한, 표준 입력으로서 임의의 표준 입력 문자 스트림(예를 들어, 사용자 입력의 텍스트)을 sgi_client 루틴(416)에 전송한다. 나중에, (469)에서, example.pl(462)은 sgi_client 루틴(416)으로부터 수신된 임의의 출력을 데몬(314)으로 전송할 것이다.

(463)에서, 전송되어 온 정보를 이용하여 sgi_client 루틴(416)이 실행될 때, sgi_client 루틴(416)은 외부 데몬(312)으로의 인증된 접속을 설정한다. 이를 위해, (417)에서, sgi_client 루틴(416)은 외부 서버(310)에 상주하는 개인 클라이언트 패스워드 파일(도시하지 않음)로부터 8개의 문자 패스워드를 판독하며, 지정된 제 2 포트에서 외부 데몬(312)으로의 접속을 설정하는데, 이 외부 데몬은 (432)에서 제 2 소켓 접속으로부터 기다리고 있다. (433)에서, 외부 데몬(312)은 자기 자신을 복제하고, 이를 실행시킨다(예를 들어, UNIX 프로세스 fork). 부모(parent) 프로세스는 자식(child) 프로세스에 대한 소켓 접속을 제공하며, 내부 데몬(322)으로부터의 다른 호출을 기다리기 위해 (430)으로 복귀한다.

(434)에서, 자식 프로세스는 sgi_client 루틴(416)을 인증한다. 이를 위해, 외부 데몬(312)은 외부 서버(310)에 상주하는 개인 클라이언트 패스워드 파일(도시하지 않음)로부터 8개의 문자 패스워드도 또한 판독한다. 외부 데몬(312)은 외부 운영 체제를 호출하여, 현재의 타임 스탬프를 검색하고, (432)에서 이 타임 스탬프를 sgi_client 루틴(416)으로 전송하고, 회신되는 인증 스트링을 기다린다. sgi_client 루틴(416)은 이 타임 스탬프를 수신하고, 외부 데몬(312)에 의해 제공된 이 타임 스탬프로 8개의 문자 패스워드를 변조함으로써 인증 스트링을 생성하고, 이 변조된 문자 스트링을 표준 UNIX 암호 커맨드로 암호화한 후, (434)에서 최종적인 인증 스트링을 외부 데몬(312)으로 전송한다. 마찬가지로, 외부 데몬(312)은 패스워드를 이 타임 스탬프로 변조하고, 이를 암호화하며, 이를 sgi_client 루틴(416)에 의해 제공된 인증 스트링과 비교한다. 인증 스트링들이 일치할 경우, sgi_client 루틴(416)이 인증된다.

(419)에서, 인증이 성공될 경우, sgi_client 루틴(416)은 요구된 서비스의 타입을 외부 데몬(312)으로 전송한다. 이 예에서, sgi_client 루틴(416)이 HTTP 데몬(314)에 의해 간접적으로 호출되었기 때문에,

sgi_client 루틴(416)은 항상 HTTP 서비스를 요구한다. 특별한 펄 스크립트(즉, example.pl(462))는, 요구된 서비스가 "httpsvc"임을 나타내는 인자를 이용하여 sgi_client 루틴(416)을 이전에 실행시켰다. 그 후, (435)에서 외부 데몬(312)은 "httpsvc" 서비스 요구를 내부 데몬(322)에 전송한다.

(452)에서, 내부 데몬(322)은 외부 데몬(312)으로부터 수신될 서비스 요구를 기다린다. (453)에서, 내부 데몬(322)은 외부 데몬(312)으로부터 서비스 요구를 수신하며, 자신의 이미지를 복제하고, 이를 실행시킨다(예를 들어, UNIX 프로세스 fork). 부모 프로세스는 자식 프로세스에 대한 네트워크 소켓 접속을 제공하며, 외부 데몬(312)에 대한 다른 접속을 개시하기 위해 (450)으로 복귀한다. (454)에서, 자식 프로세스는 메모리내의 테이블에 상주하는 유효한 실행가능 서비스(예를 들어, httpsvc)의 리스트 및 이들 서비스에 대한 완전한 디렉토리 경로를 가지고서 요구된 서비스를 확인한다. 요구된 서비스가 유효 서비스의 리스트내에 있지 않을 경우, 이는 거부될 것이다. 따라서, 인증되지 않은 사용자가 외부 데몬(312)을 통해 내부 데몬(322)에 접근을 허용받았다 하더라도, 그 사용자는 유효 서비스의 리스트내에 존재하는 서비스로 제한될 것이다.

서비스 요구가 유효할 경우, (455)에서 내부 데몬(322)은 UNIX exec 커맨드를 호출하고(즉, 새로운 서비스 프로그램을 자신에 중첩시켜, 요구된 서비스를 실행함), httpsvc(470)에 대한 네트워크 소켓 접속을 제공한다. httpsvc(470)는 하나의 추가적인 환경 변수를 프로세스 환경에 부가하는데, 이는 외부 데몬(312)의 명칭이다. SGI는 추가적인 환경 변수를 부가해서, 필요할 경우, http 데몬(314)이 아니라, SGI가 example.pl(480)을 실행시켰는지를 example.pl(480)이 판단할 수 있게 한다.

보조 자료(side note)로서, 외부 데몬(312), 내부 데몬(322), sgi_client 루틴(416), 및 httpsvc(470)는 각각 계정(accounting) 및 에러 로그 파일을 갖는다. 이들 각각은, 에러 및 계정 로그에 위치될 정보의 양을 상이하게 하는 디버깅 및 추적 인자를 갖는다. 또한, 추적 인자가 sgi_client 루틴(416)에 의해 설정될 경우, 외부 데몬(312), 내부 데몬(322), 및 httpsvc(470)는, 각각이 원래 실행되었을 때 어떻게 추적이 설정되었는지에 관계없이, 각각의 에러 로그파일내에서 그 특정 트랜잭션을 모두 추적할 것이다.

(436)에서, 외부 데몬(312)은 이전에 생성된 헤더를, (471)에 수신되는 서비스 프로그램(324)으로 전송한다. 이에 따라, 서비스 프로그램(324)은 (원래의 프로세스 환경 변수를 포함하는) 헤더를 변수=값 스트링으로 파싱(parsing)하며, example.pl(462)에서 정의된 원래의 프로세스 환경을 재생한다. 서비스 프로그램(324)은 헤더 변수 SGICMD=example.pl로부터 cgi-bin(426)내에서 호출할 적절한 프로그램을 결정하고, example.pl(480)과 통신하기 위한 통신 채널(예를 들어, 파이프)을 생성하며, (472)에서 example.pl(480)을 호출한다. (437)에서, 외부 데몬(312)은 표준 입력 문자 스트림(예를 들어, 텍스트)을 서비스 프로그램(324)으로 전송한다. (473)에서, 서비스 프로그램(324)은 텍스트를 example.pl(480)의 표준 입력으로 전송한다.

이 시점에서, 서비스 프로그램(324)이 (471)에서 (본래 (462)에서 생성되었던) 원래의 프로세스 환경을 재생했기 때문에, (비록, 선택사양적으로 SGI가 httpsvc(470)에 의해 헤더에 부가된 추가적인 환경 변수로부터 이를 호출했다고 결정할 수 있다해도) example.pl(480)은 SGI가 아니라 http 데몬(314)에 의해 (472)에서 자신이 실행되고 있다고 믿는다. 따라서, SGI는 고객, http 데몬(314), 및 example.pl(480)에 상주하는 실질적인 트랜잭션 프로그램 모두에게 투명하다. 따라서, http 데몬(314) 및 example.pl(480)에 상주하는 트랜잭션 프로그램 모두 변경될 필요가 없다.

이제 모든 정보가 존재하므로, (481)에서 example.pl(480)은 데이타베이스(330)상에서 내부 트랜잭션을 실행하게 된다. (성공적인지 여부에 관계없이) 일단 트랜잭션이 완료되면, (481)에서 트랜잭션의 출력이 고객에게 리턴된다. (482)에서, example.pl(480)은 트랜잭션의 출력을 수신하며, 이를 서비스 프로그램(324)의 파이프(474)에 전송한다. (474)에서, 서비스 프로그램(324)은 이 출력을 외부 데몬(312)에 전송한다. (438)에서, 외부 데몬(312)은 이 출력을 sgi_client 루틴(416)에 전송한다. (464)에서, sgi_client 루틴(416)은 이 출력을 특별한 펄 스크립트 example.pl(462)에 전송한다. (465)에서, example.pl(462)은 이 출력을 데몬(314)에 전송한다. (466)에서, 데몬(314)은 이 출력을 고객에게 전송한다.

따라서, 고객이 개시한 트랜잭션은 (454)에서의 확인, (481)에서의 프로세싱과, 최종적으로 (466)에서 고객에게 리턴되는 출력을 위하여, 데몬(314)에서 외부 데몬(312)으로, 외부 데몬(312)에서 내부 데몬(322)으로 안전하게 전송될 수 있다. 고객 요구 및 텍스트는 방화벽(300)을 통해 내부 트랜잭션 프로세싱에서 이용될 수 있는데, 이러한 프로세싱은 모두 SGI의 완벽한 제어하에 있으나 고객에게는 완전히 투명하다. 내부 데몬(322)은 (451)에서 인증을 수행하고, (454)에서 서비스가 외부 네트워크에서 이용가능할 것을 엄격히 강요하며, 선택사양적으로 (481)에서 사용자 인증을 수행하기 때문에, 외부 서버(310)와의 타협은 아주 최소한의 내부 보안 위험을 야기하지만, 결코 내부 네트워크와 타협하지는 않는다.

이 특정 실시예를 이용하여, 현재의 http 서버는, 현재의 cgi-bin 커맨드에 대한 변경을 거의 하지 않고 SGI를 구현할 수 있다. SGI는 완전히 은닉되며, 심지어 복잡한 http 서버까지도 자동적으로 지원할 것이다. 현재의 http 서버에 대한 변경을 거의 하지 않으면서 비지니스 트랜잭션에 대해 추가적인 보안 및 지원을 부여할 수 있다. 외부 네트워크에 이용가능한 트랜잭션(example.pl과 같은 프로그램)은 각각 cgi-bin(415) 및 cgi-bin(426)에 상주하는 펄 스크립트 및 트랜잭션 프로그램으로 제한되고, 내부 서버(320)가 통상적으로 엄격한 기업의 통제하에 있어서 내부 개발자(developer)에 의해 쉽게 변경되지 않기 때문에, SGI는 내부 개발자가 기업의 재검토 및 허가없이 내부 트랜잭션을 외부 고객이 이용하도록 하는 것도 또한 어렵게 한다.

본 발명의 변형에서, 대안적인 실시예는 sgi_client 루틴(416) 및 외부 데몬(314)의 기능을 데몬(314)에 합체시킨다. 이는 더 나은 성능을 제공할 것이지만, httpd 구현의 소유 및 개선을 합체시키는 것을 어렵게 만든다.

(57) 청구의 범위

청구항 1

내부 컴퓨터 시스템 및 외부 컴퓨터 시스템간의 보안 방화벽(a security firewall)(300)을 침해하지 않으면서, 상기 외부 컴퓨터 시스템(310)이 내부 자원(330)을 이용하여 트랜잭션 요구(2)를 개시할 수 있도록 상기 내부 컴퓨터 시스템(320)을 조작하는 방법에 있어서,

① 상기 내부 컴퓨터 시스템 및 상기 외부 컴퓨터 시스템간에 상기 내부 컴퓨터 시스템에 의해 개시되는 접속을 인증(authenticating)(451)해서, 인증된 접속을 설정하게 되는 단계와,

② 상기 외부 컴퓨터 시스템에 의해 수신된 트랜잭션 요구를 상기 외부 컴퓨터 시스템에 의해 호출(461)하는 단계와,

③ 상기 트랜잭션 요구의 호출에 응답하여, 프로세스 환경 변수를 포함하는 원래의 프로세스 환경을 상기 외부 컴퓨터 시스템에 의해 생성(462)하고, 상기 트랜잭션 요구 및 상기 트랜잭션 요구를 실행하기 위한 상기 프로세스 환경 변수를 포함하는 스트링을 생성하는 단계와,

④ 상기 인증된 접속을 통해 상기 내부 컴퓨터 시스템으로 상기 스트링을 상기 외부 컴퓨터 시스템에 의해 전송(435, 436, 437)하는 단계와,

⑤ 상기 트랜잭션 요구를 상기 내부 컴퓨터 시스템에 의해 검증(454)하는 단계와,

⑥ 상기 원래의 프로세스 환경을 상기 내부 컴퓨터 시스템에 의해 재생(471)하는 단계와,

⑦ 상기 트랜잭션 요구를 상기 내부 컴퓨터 시스템에 의해 실행(472)해서, 출력을 생성하는(482) 단계를 포함하는 것을 특징으로 하는 내부 컴퓨터 시스템 조작 방법.

청구항 2

제 1 항에 있어서,

(a) 제 1 패스워드 및 제 1 통신 포트를 상기 외부 컴퓨터 시스템(310)에 의해 판독(312)하는 단계와,

(b) 상기 제 1 통신 포트에 제 1 소켓(a first socket)을 상기 외부 컴퓨터 시스템에 의해 생성(312)하고, 상기 내부 컴퓨터 시스템(320)으로부터의 접속 호출을 상기 제 1 소켓에서 기다리는 단계와,

(c) 제 2 패스워드 및 제 2 통신 포트를 상기 내부 컴퓨터 시스템에 의해 판독(322)하는 단계와,

(d) 상기 제 2 통신 포트에 제 2 소켓을 상기 내부 컴퓨터 시스템에 의해 생성(322)하고, 상기 제 2 소켓을 통해 상기 외부 컴퓨터 시스템으로 접속 호출을 전송하여서, 접속을 설정하게 되는 단계를 더 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 3

제 2 항에 있어서,

상기 인증(451) 단계는,

(e) 상기 제 2 소켓을 통해 상기 외부 시스템(310)으로 유일한 타임 스탬프(a unique timestamp)를 상기 내부 컴퓨터 시스템(320)에 의해 전송하는 단계와,

(f) 상기 제 1 패스워드를 상기 수신된 타임 스탬프로 상기 외부 컴퓨터 시스템에 의해 변조(mangling)하는 단계와,

(g) 상기 변조된 제 1 패스워드를 암호화 알고리즘으로 암호화하여 제 1 패스워드 스트링을 생성하는 단계와,

(h) 상기 제 1 패스워드 스트링을 상기 내부 컴퓨터 시스템으로 상기 외부 컴퓨터 시스템에 의해 전송하는 단계와,

(i) 상기 제 2 패스워드를 사용하여 상기 내부 컴퓨터 시스템에 의해 상기 단계 (f) 내지 (g)를 반복해서, 제 2 패스워드 스트링을 생성하는 단계와,

(j) 상기 제 1 패스워드 스트링과 상기 제 2 패스워드 스트링을 상기 내부 컴퓨터 시스템에 의해 비교하는 단계를 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 4

제 3 항에 있어서,

상기 변조 단계는,

① 상기 타임 스탬프를 16진수 0177과 논리적으로 AND하여 유일한 결과를 생성하는 단계와,

② 상기 유일한 결과를 상기 제 1 패스워드의 각 문자와 논리적으로 배타적 OR해서, 상기 변조된 제 1 패스워드를 생성하는 단계를 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 5

제 3 항에 있어서,

상기 암호화 단계는,

상기 변조된 제 2 패스워드의 문자 각각을 키(a key)로 암호화해서, 상기 패스워드 스트링을 생성하는 단계를 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 6

제 1 항에 있어서,

상기 호출(461) 단계는,

① 상기 트랜잭션 요구(2)를 상기 외부 컴퓨터 시스템(310)으로 외부 네트워크에 의해 전송하는 단계 - 상기 트랜잭션 요구는 트랜잭션 프로그램을 실행하기 위한, 입력 데이터, 인자, 및 커맨드를 포함함 - 와,

② 상기 트랜잭션 요구를 상기 외부 컴퓨터 시스템에 의해 수신하는 것에 응답하여, 상기 프로세스 환경 변수를 포함하는 프로세스 환경을 제 1 데몬(a first daemon)(312)에 의해 정의하는 단계를 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 7

제 6 항에 있어서,

① 상기 트랜잭션 요구(2)를 상기 외부 컴퓨터 시스템(310)에 의해 호출하는 것에 응답하여, 상기 커맨드를 호출(461)하는 단계와,

② 상기 커맨드를 호출하는 것에 응답하여, 스크립트(a script)를 실행(462)시키고, 상기 사용자 입력 데이터, 인자, 및 트랜잭션 요구를 상기 스크립트로 전송하는 단계와,

③ 상기 스트링을 상기 스크립트에 의해 생성(462)하는 단계 - 상기 스트링은 상기 트랜잭션 요구를 실행하기 위한, 상기 커맨드, 인자, 및 상기 프로세스 환경 변수를 포함함 - 를 더 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 8

제 7 항에 있어서,

상기 외부 컴퓨터 시스템(310)에 상주하는 클라이언트 루틴(a client routine)(463)을 상기 스크립트에 의해 호출하고, 상기 사용자 입력 데이터와, 상기 외부 컴퓨터 시스템에 상주하는 제 2 데몬(314)에 접속시키기 위한 제 3 통신 포트와, 트랜잭션 요구(2)의 타입을 식별하는 식별자와, 상기 스트링을 상기 클라이언트 루틴으로 전송하는 단계를 더 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 9

제 8 항에 있어서,

① 호출을 상기 스크립트에 의해 수신하는 것에 응답하여, 상기 클라이언트 루틴을 상기 제 2 데몬(314)에 의해 인증(460)하는 단계와,

② 상기 제 2 데몬에 의해 포크(fork)하고, 자식 프로세스(a child process)로 제 3 소켓 접속을 전달하여, 부모 프로세스(a parent process)가 상기 내부 컴퓨터 시스템(320)으로부터 호출을 상기 제 1 소켓 접속에서 기다리는 단계와,

③ 상기 클라이언트 루틴을 인증하는 것에 응답하여, 상기 트랜잭션 요구의 타입을 상기 내부 컴퓨터 시스템에 상주하는 제 3 데몬(322)으로 상기 자식 프로세스에 의해 전송하는 단계를 더 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 10

제 9 항에 있어서,

상기 검증 단계는,

① 상기 외부 컴퓨터 시스템상의 메모리에 저장된 유효 서비스 테이블을 상기 제 3 데몬(322)에 의해 판독(453)하는 단계와,

② 상기 자식 프로세스로부터 수신된 상기 트랜잭션 요구의 타입을 상기 유효 서비스 테이블과 비교(454)하는 단계 - 상기 타입이 상기 유효 서비스 테이블내에 발견될 경우, 상기 트랜잭션 요구가 검증됨 - 를 포함하는 내부 컴퓨터 시스템 조작 방법.

청구항 11

내부 컴퓨터 시스템 및 외부 컴퓨터 시스템간의 보안 방화벽(300)을 침해하지 않으면서, 상기 외부 컴퓨터 시스템(310)이 내부 자원(330)을 이용하여 트랜잭션 요구(2)를 개시할 수 있도록 상기 내부 컴퓨터 시스템(320)을 조작하는 유일하게 프로그래밍된 시스템(a uniquely programmed system)에 있어서,

① 상기 내부 컴퓨터 시스템 및 상기 외부 컴퓨터 시스템간에 상기 내부 컴퓨터 시스템에 의해 개시되는 접속을 인증(451)해서, 인증된 접속을 설정하는 수단(322)과,

② 상기 외부 컴퓨터 시스템에 의해 수신된 트랜잭션 요구를 상기 외부 컴퓨터 시스템에 의해 호출(461)하는 수단(314)과,

③ 상기 트랜잭션 요구의 호출에 응답하여, 프로세스 환경 변수를 포함하는 원래의 프로세스 환경을 상기 외부 컴퓨터 시스템에 의해 생성(462)하는 수단(415)과, 상기 트랜잭션 요구와, 상기 트랜잭션 요구를 실행하기 위한, 상기 인자와, 상기 프로세스 환경 변수를 포함하는 스트링을 생성하는 수단과,

④ 상기 인증된 접속을 통해 상기 내부 컴퓨터 시스템으로 상기 스트링을 상기 외부 컴퓨터 시스템에 의해 전송(435, 436, 437)하는 수단(312)과,

⑤ 상기 트랜잭션 요구를 상기 내부 컴퓨터 시스템에 의해 검증(454)하는 수단(322)과,

⑥ 상기 원래의 프로세스 환경을 상기 내부 컴퓨터 시스템에 의해 재생(431)하는 수단(324)과,

⑦ 상기 트랜잭션 요구를 상기 내부 컴퓨터 시스템에 의해 실행(432)해서, 출력을 생성하는 수단(324)을 포함하는 것을 특징으로 하는 내부 컴퓨터 시스템 조작을 행하는 유일하게 프로그래밍된 시스템.

청구항 12

내부 컴퓨터 시스템 및 외부 컴퓨터 시스템간의 보안 방화벽(300)을 침해하지 않으면서, 상기 내부 컴퓨터 시스템으로 하여금 상기 외부 컴퓨터 시스템(310)이 내부 자원을 이용하여 트랜잭션 요구(2)를 개시할 수 있도록 하게 하는 컴퓨터로 판독가능한 프로그램 코드 수단을 그 안에 구비한 컴퓨터로 이용가능한 매체(a computer usable medium)를 포함하는 것을 특징으로 하는 제품으로서, 상기 제품내의 상기 컴퓨터로 판독가능한 프로그램 코드 수단은,

① 상기 내부 컴퓨터 시스템 및 상기 외부 컴퓨터 시스템간에 상기 내부 컴퓨터 시스템에 의해 개시되는 접속을 인증(451)해서, 인증된 접속을 설정하는 컴퓨터로 판독가능한 프로그램 수단(322)과,

② 상기 외부 컴퓨터 시스템에 의해 수신된 트랜잭션 요구를 상기 외부 컴퓨터 시스템에 의해 호출(461)하는 컴퓨터로 판독가능한 프로그램 수단(314)과,

③ 상기 트랜잭션 요구의 호출에 응답하여, 프로세스 환경 변수를 포함하는 원래의 프로세스 환경을 상기 외부 컴퓨터 시스템에 의해 생성(462)하는 컴퓨터로 판독가능한 프로그램 수단(415)과, 상기 트랜잭션 요구, 및 상기 트랜잭션 요구를 실행하기 위한 상기 프로세스 환경 변수를 포함하는 스트링을 생성하는 컴퓨터로 판독가능한 프로그램 수단과,

④ 상기 인증된 접속을 통해 상기 내부 컴퓨터 시스템으로 상기 스트링을 상기 외부 컴퓨터 시스템에 의해 전송(435, 436, 437)하는 컴퓨터로 판독가능한 프로그램 수단(312)과,

⑤ 상기 트랜잭션 요구를 상기 내부 컴퓨터 시스템에 의해 검증(454)하는 컴퓨터로 판독가능한 프로그램 수단(322)과,

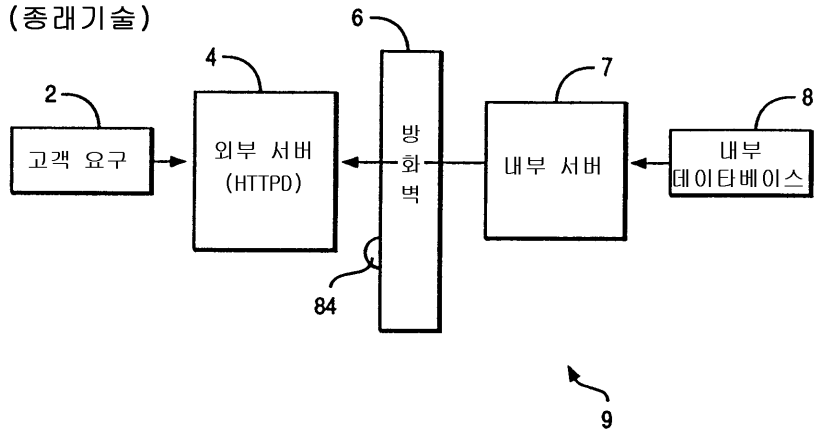
⑥ 상기 원래의 프로세스 환경을 상기 내부 컴퓨터 시스템에 의해 재생(471)하는 컴퓨터로 판독가능한 프로그램 수단(324)과,

⑦ 상기 트랜잭션 요구를 상기 내부 컴퓨터 시스템에 의해 실행(472)해서, 출력을 생성하는 컴퓨터로 판독가능한 프로그램 수단(324)을 포함하는 것을 특징으로 하는 제품.

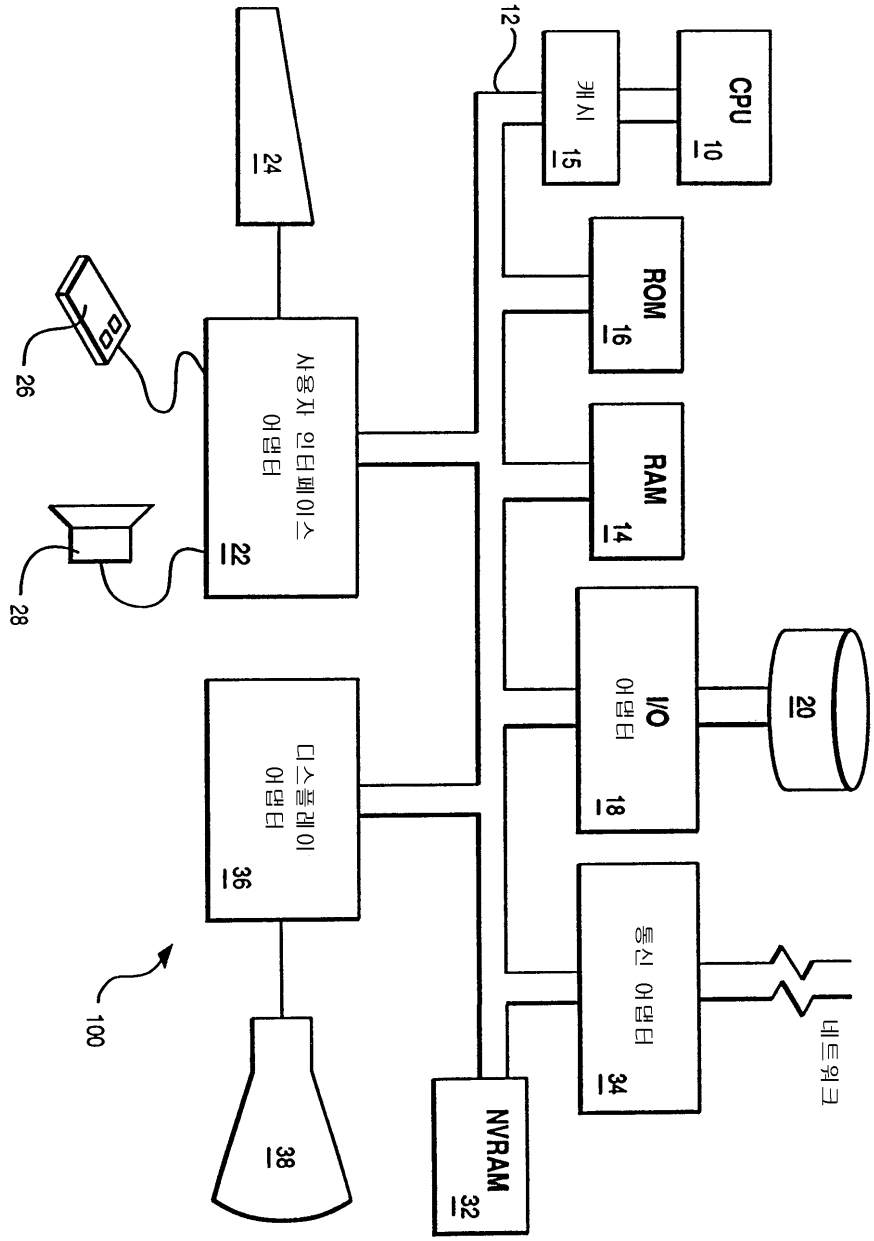
도면

도면1

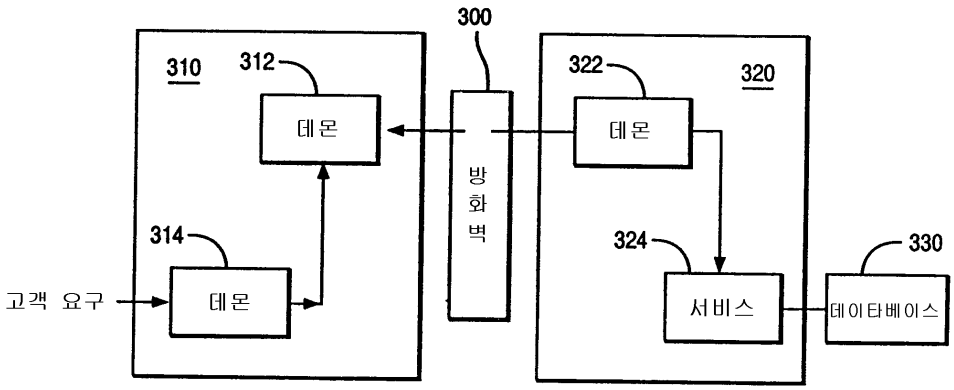
(종래기술)


2
고객 요구
4
외부 서버
(HTTPD)
6
방화벽
84
7
내부 서버
8
내부
데이타베이스
9

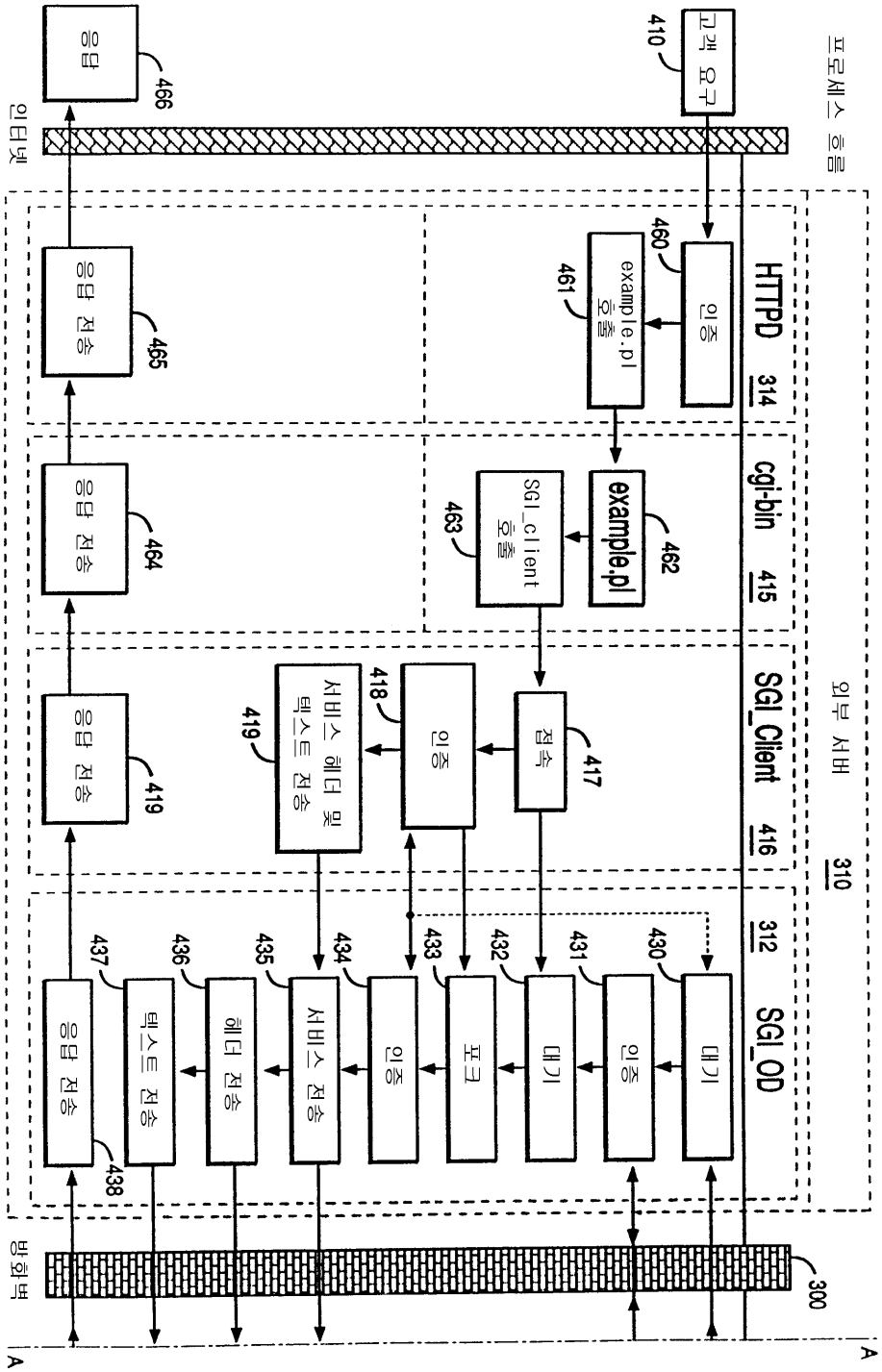
도면2


12
캐시
15
CPU
10
24
ROM
16
26
사용자 인터페이스
어댑터
22
RAM
14
28
I/O
어댑터
18
20
디스플레이
어댑터
36
100
통신 어댑터
34
네트워크
38
NVRAM
32

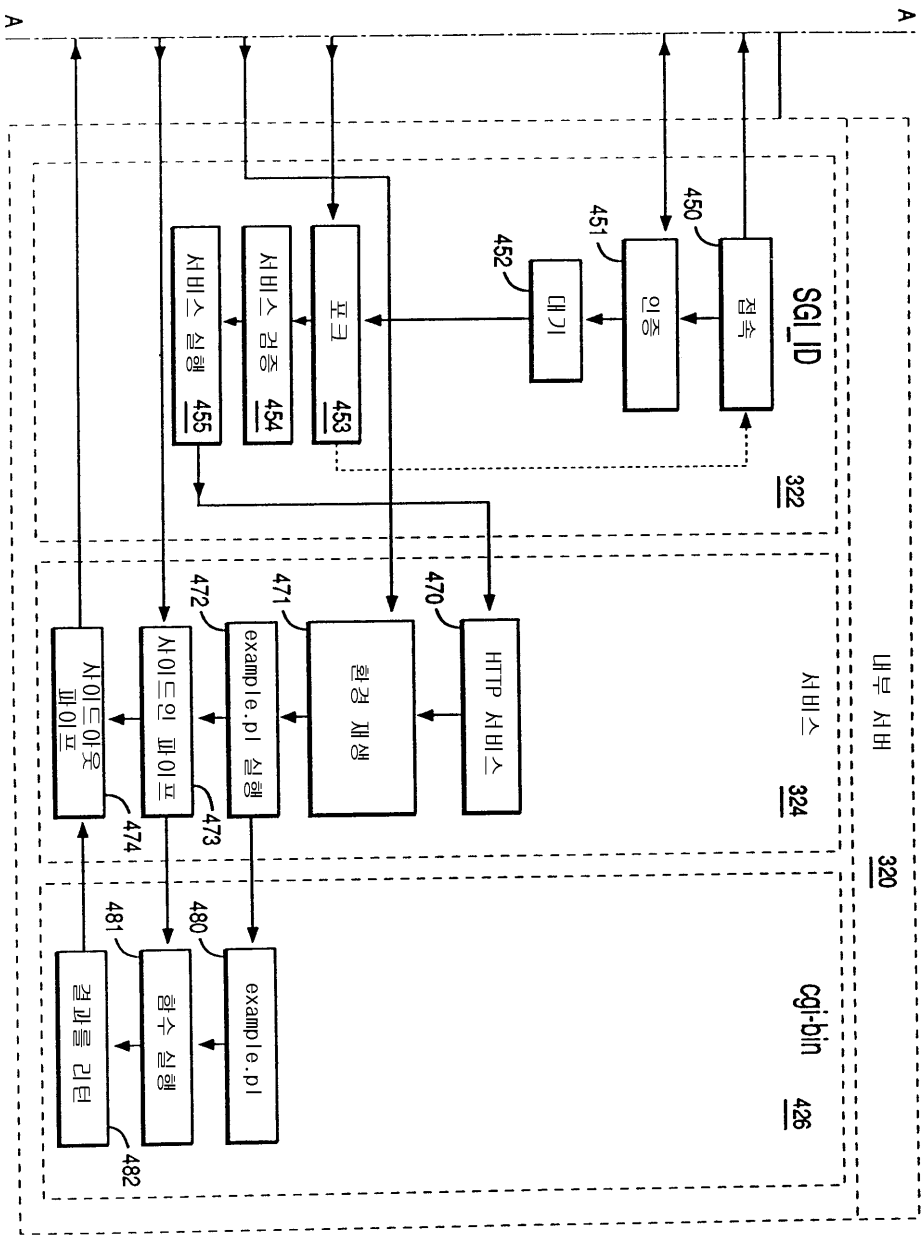
도면3



도면4a


프로세스 흐름
인터넷
방화벽
외부 서버
310
HTTPD
314
cgi-bin
415
SGI_Client
416
SGI_OD
312
고객 요구
410
인증
460
example.pl 호출
461
example.pl
462
SGI_client 호출
463
접속
417
인증
418
서비스 헤더 및 텍스트 전송
419
응답 전송
464
응답 전송
465
응답
466
대기
430
인증
431
대기
432
포크
433
인증
434
서비스 전송
435
헤더 전송
436
텍스트 전송
437
응답 전송
438
300
A
A

도면4b


A
A
내부 서버
320
SGI_ID
322
접속
인증
대기
포크
서비스 검증
서비스 실행
450
451
452
453
454
455
서비스
324
HTTP 서비스
환경 재생
example.pl 실행
사이드인 파이프
사이드아웃 파이프
470
471
472
473
474
cgi-bin
426
example.pl
함수 실행
결과를 리턴
480
481
482