(12) 특허협력조약에 의하여 공개된 국제출원

(19) 세계지식재산권기구
국제사무국

(43) 국제공개일
2012 년 12 월 27 일 (27.12.2012)

WIPO | PCT

(10) 국제공개번호
WO 2012/176978 A2

(51) 국제특허분류:
H04L 12/22 (2006.01) *G06F 21/20* (2006.01)
H04L 12/26 (2006.01)

(21) 국제출원번호: PCT/KR2012/003215

(22) 국제출원일: 2012 년 4 월 26 일 (26.04.2012)

(25) 출원언어: 한국어

(26) 공개언어: 한국어

(30) 우선권정보:
10-2011-0061162 2011 년 6 월 23 일 (23.06.2011) KR

(71) 출원인 (US 을(를) 제외한 모든 지정국에 대하여): **(주) 잉카인터넷 (INCA INTERNET CO., LTD.)** [KR/KR]; 서울특별시 구로구 구로 3 동 235-2 에이스하이엔드타워 1201 호, 152-740 Seoul (KR).

(72) 발명자; 겸
(75) 발명자/출원인 (US 에 한하여): **고보승 (KO, Bo-Seung)** [KR/KR]; 경기도 광명시 철산 4 동 도덕파크타운 203 동 1702 호, 423-741 Gyeonggi-Do (KR). **김성은 (KIM, Sung-Eun)** [KR/KR]; 경기도 안양시 동안구 호계 2 동 현대홈타운 1 차아파트 101-1603, 431-753 Gyeonggi-Do (KR). **김상훈 (KIM, Sang-Hoon)** [KR/KR]; 서울특별시 동작구 상도동 건영아파트 102 동 1201 호, 156-030 Seoul (KR). **박명수 (PARK, Myung-Su)** [KR/KR]; 서울특별시 강서구 공항동 2-85 번지 세왕빌라 501 호, 157-240 Seoul (KR).

(74) 대리인: **특허법인 우인 (WOOIN PATENT & LAW FIRM)**; 서울특별시 강남구 역삼로 157, 2 층(역삼동, 중평빌딩), 135-925 Seoul (KR).

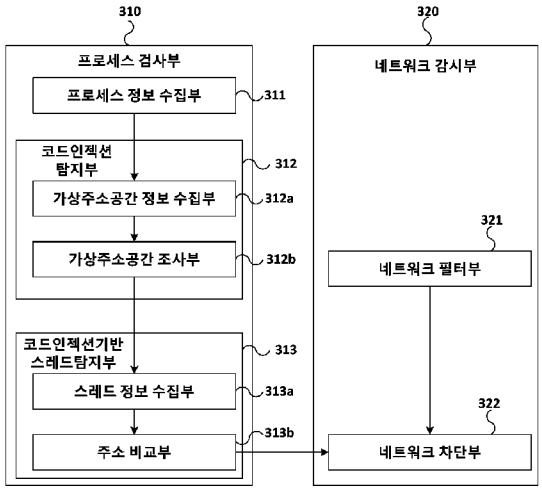
(81) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 국내 권리의 보호를 위하여): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG,

[다음 쪽 계속]

(54) **Title**: NETWORK ACCESS CONTROL SYSTEM AND METHOD

(54) **발명의 명칭** : 네트워크 접근 제어시스템 및 방법

[Fig. 3]



310 ... Process inspecting unit
311 ... Process information collecting unit
312 ... Code injection detecting unit
312a ... Virtual address space information collecting unit
312b ... Virtual address space investigating unit
313 ... Code injection-based thread detecting unit
313a ... Thread information collecting unit
313b ... Address comparing unit
320 ... Network monitoring unit
321 ... Network filter unit
322 ... Network blocking unit

(57) **Abstract**: The present invention relates to a system and method for controlling a network access of a network packet on the basis of a thread which is inserted into a process through code injection. The network access control system according to the present invention comprises: a process inspecting unit for detecting a code injection-based thread included in a process; and a network monitoring unit for performing network filtering so as to detect a network packet having access to a network, and, if a communication subject of the detected network packet is the code injection-based thread, blocking the traffic of the detected network packet.

(57) **요약서**: 이 발명은 프로세스에 코드 인젝션(code injection) 기반으로 삽입된 스레드를 기준으로 네트워크 패킷의 네트워크 접근을 제어하는 시스템 및 방법에 관한 것이다. 이 발명에 따른 네트워크 접근 제어시스템은, 프로세스에 포함된 코드 인젝션 기반 스레드를 탐지하는 프로세스 검사부와, 네트워크 필터링을 수행하여 네트워크에 접근하는 네트워크 패킷을 탐지하고 상기 탐지된 네트워크 패킷의 통신 주체가 상기 코드 인젝션 기반 스레드이면 상기 탐지된 네트워크 패킷의 트래픽이 차단되도록 하는 네트워크 감시부를 포함한다.

WO 2012/176978 A2

SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **지정국** (별도의 표시가 없는 한, 가능한 모든 종류의 역내 권리의 보호를 위하여): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), 유라시아 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 유럽 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

공개:

— 국제조사보고서 없이 공개하며 보고서 접수 후 이를 별도 공개함 (규칙 48.2(g))

명세서

발명의 명칭: 네트워크 접근 제어시스템 및 방법

기술분야

[1] 이 발명은 네트워크 접근 제어시스템 및 방법에 관한 것으로서, 보다 상세하게는 프로세스에 코드 인젝션(code injection) 기반으로 삽입된 스레드를 기준으로 네트워크 접근을 제어하는 시스템 및 방법에 관한 것이다.

배경기술

[2] 인터넷은 TCP/IP(Transmission Control Protocol/Internet Protocol)를 사용하는 네트워크로서, 인터넷 기술이 발달함에 따라 인터넷의 사용 영역이 확장됨으로써 개인 및 일반 기업체에서도 인터넷 망에 접속하여 각종 정보를 취득하고 이를 업무에 이용하고 있다. 그러나, 상기와 같은 정보 취득의 장점과는 반대로 바이러스 등 각종 해킹 프로그램이 인터넷에 난무하게 되고, 인터넷에 접속된 네트워크 망에 바이러스 등이 침입하게 됨에 따라 인터넷을 사용하는 시스템이 공격을 받게 된다.

[3] 이러한 시스템에 대한 불법적인 침해를 예방하기 위한 보안 기술로서, 마이크로소프트(Microsoft) 사에서는 윈도우즈 엑스피(Windows XP) 버전부터 컴퓨터 또는 네트워크를 보호하기 위한 인터넷 연결 방화벽(ICF : Internet Connection Firewall)을 기본적으로 제공하고 있다.

[4] 인터넷 연결 방화벽은 컴퓨터 사용자가 원하지 않는 트래픽이 외부로부터 컴퓨터로 유입되거나, 컴퓨터로부터 외부로 유출되지 않도록 하는 보안기술이다. 이를 위해, 인터넷 연결 방화벽은 통신허용목록을 테이블로 구축하고, 네트워크 필터링 기술을 사용하여 인바운드 패킷(외부로부터 컴퓨터로 유입되는 패킷) 및 아웃바운드 패킷(컴퓨터로부터 외부로 유출되는 패킷)을 추적하여 통신허용목록과 비교한다. 해당 패킷이 통신허용목록에 포함되면 해당 패킷의 트래픽을 허용하고, 해당 패킷이 통신허용목록에 포함되지 않으면 해당 패킷의 트래픽을 차단한다. 이때, 통신허용목록에는 사용자에 의해 통신이 허용된 프로세스의 프로세스 아이디와 경로가 저장된다.

[5] 도 1은 종래의 인터넷 연결 방화벽에서의 네트워크 접근 차단방법을 도시한 동작 흐름도이다.

[6] 인터넷 연결 방화벽은 네트워크 패킷에 대해, 해당 패킷을 송신 또는 수신하는 프로세스의 프로세스 정보(프로세스 아이디와 경로)를 분석하고(S11), 해당 프로세스 아이디와 경로가 통신허용목록에 포함되면(S12), 해당 패킷의 트래픽을 허용한다(S13). 그러나, 해당 프로세스 아이디와 경로가 통신허용목록에 포함되지 않으면(S12), 사용자에게 해당 프로세스가 송신 또는 수신하고자 하는 패킷을 허용할지 말지를 검증받는다(S14). 사용자가 상기 패킷의 전송을 허용하면(S15), 통신허용목록에 해당 프로세스의 프로세스

아이디와 경로를 추가하고(S16) 해당 패킷의 트래픽을 허용한다(S13). 한편, 사용자가 상기 패킷의 전송을 허용하지 않으면(S15), 해당 패킷의 트래픽을 차단한다(S17).

[7] 이러한 종래의 인터넷 연결 방화벽에서의 네트워크 접근 차단방법은, 프로세스 아이디를 기준으로 패킷 트래픽을 허용 또는 차단하기 때문에, 해커가 통신이 허용된 프로세스에 악성코드를 삽입하고, 그 악성코드가 동작하여 패킷 트래픽을 발생할 경우에는 이를 차단할 수 없는 문제점이 있다.

[8] 이를 좀 더 상세하게 설명한다.

[9] 통상적으로 인터넷 익스플로러(iexplore.exe)는 윈도우즈 운영체제(Windows OS) 하에서 널리 사용되는 웹브라우저 프로그램으로서, 아웃바운드 패킷과 인바운드 패킷 등의 트래픽을 수시로 발생한다. 따라서, 인터넷 익스플로러에서 발생된 패킷의 전송에 대해 검증이 요청되면, 사용자는 별 의심없이 허용할 것이며, 그럴 경우 인터넷 익스플로러가 외부로 유출하는 아웃바운드 패킷과 외부에서 인터넷 익스플로러로 유입되는 인바운드 패킷이 모두 사용자의 추가 검증없이 허용될 것이다.

[10] 한편, 트로이 목마(trojan)나 제우스(Zeus)와 같은 프로그램은 프로세스에 코드 인젝션(code injection)을 수행하여 악성 코드를 삽입하고 그 악성 코드로 하여금 스레드를 생성하도록 한다. 최근 제우스 프로그램에 의해 사용자 컴퓨터에 삽입된 악성 코드가 컴퓨터에 저장된 은행 거래 계좌와 비밀번호 등의 금융 정보를 해커에게 유출시켜, 해커가 그 금융 정보를 이용하여 중소기업이나 지방자치단체 은행 계정에 접근, 현금을 인출하는 금융 사고가 발생한 바 있다.

[11] 예컨대, 도 2에 도시된 바와 같이 인터넷 익스플로러에 악성 코드가 삽입되어 그 악성 코드를 기반으로 스레드가 생성되고, 그 악성 스레드(Thread 3)가 악의적으로 정보를 유출하고자 시도한다고 가정하자. 이 경우, 인터넷 연결 방화벽은 프로세스 아이디(즉, 인터넷 익스플로러(iexplore.exe))를 이용하여 통신허용목록을 검색하여 인터넷 익스플로러의 네트워크 접근을 제어한다. 즉, 인터넷 익스플로러가 통신허용목록에 포함되어 있으면 그 악성 스레드에 의한 악의적인 정보 유출을 허용하고, 인터넷 익스플로러가 통신허용목록에 포함되어 있지 않으면 그 악성 스레드에 의한 악의적인 정보 유출을 차단한다. 물론, 인터넷 연결 방화벽은 인터넷 익스플로러의 다른 정상 스레드를 통한 정상적인 패킷에 대해서는 프로세스 아이디를 기준으로 그 네트워크 접근을 제어한다.

[12] 따라서, 사용자가 인터넷 익스플로러에 대해 네트워크 접근을 허용하면, 다른 정상 스레드를 통한 정상적인 패킷 전송이 원활하게 이루어져 사용자가 불편없이 인터넷에 접속할 수 있지만, 인터넷 익스플로러에 포함된 악성 스레드에 의한 네트워크 접근도 모두 허용되기 때문에 보안이 취약해지는 문제점이 있다.

[13] 그렇다고 사용자가 인터넷 익스플로러에 대해 네트워크 접근을 허용하지

않으면, 악성 스레드에 의한 악의적인 정보 유출을 방지할 수는 있지만, 다른 정상 스레드를 통한 정상적인 패킷 전송도 모두 차단되기 때문에 사용자가 인터넷을 사용하는데 큰 불편함이 따른다.

발명의 상세한 설명

기술적 과제

[14] 상술한 종래기술의 문제점을 해결하기 위하여 안출된 이 발명의 목적은, 프로세스 아이디뿐만 아니라 스레드 아이디를 기반으로 네트워크 패킷의 트래픽을 허용 또는 차단하여 사용자의 편의성과 보안 강도가 향상되는 네트워크 접근 제어시스템 및 방법을 제공하기 위한 것이다.

과제 해결 수단

[15] 이 발명에 따른 네트워크 접근 제어시스템은, 프로세스에 포함된 코드 인젝션 기반 스레드를 탐지하는 프로세스 검사부와, 네트워크 필터링을 수행하여 네트워크에 접근하는 네트워크 패킷을 탐지하고 상기 탐지된 네트워크 패킷의 통신 주체가 상기 코드 인젝션 기반 스레드이면 상기 탐지된 네트워크 패킷의 트래픽이 차단되도록 하는 네트워크 감시부를 포함한 것을 특징으로 한다.

[16] 또한, 이 발명에 따른 네트워크 접근 제어방법은, 네트워크 접근 제어시스템이 컴퓨터에서 실행되는 프로세스 정보를 수집하는 제1단계와, 상기 네트워크 접근 제어시스템이 상기 수집된 프로세스에 포함된 코드 인젝션된 가상주소공간 영역을 탐지하는 제2단계와, 상기 네트워크 접근 제어시스템이 상기 코드 인젝션된 가상주소공간에 코드 인젝션 기반 스레드가 존재하는지를 탐지하는 제3단계와, 상기 네트워크 접근 제어시스템이 네트워크 필터링하여 네트워크 패킷 트래픽을 수집하는 제4단계와, 상기 네트워크 접근 제어시스템이 상기 코드 인젝션 기반 스레드 정보를 이용하여 상기 네트워크 패킷의 트래픽을 허용 또는 차단하는 제5단계를 포함하는 것을 특징으로 한다.

발명의 효과

[17] 이상과 같이 이 발명에 따르면, 프로세스에 코드 인젝션으로 삽입된 악성코드에 의해 생성된 악성 스레드에 의한 악의적인 정보 유출을 실시간으로 차단할 수 있는 효과가 있다. 또한, 동일 프로세스라 하더라도 스레드별로 허용/차단 여부를 결정하여 정상 스레드에 의한 정상적인 네트워크 접근은 허용하고 악성 스레드에 의한 악의적인 네트워크 접근은 차단하여 사용자의 편의성과 보안 강도가 향상되는 효과가 있다.

도면의 간단한 설명

[18] 도 1은 종래의 인터넷 연결 방화벽에서의 네트워크 접근 차단방법을 도시한 동작 흐름도이다.

[19] 도 2는 종래기술에 따른 네트워크 접근 차단방법의 문제점을 설명하기 위하여 도시한 도면이다.

[20] 도 3은 이 발명에 따른 네트워크 접근 제어시스템의 구성 블록도이다.

[21] 도 4는 임의의 프로세스의 배드(Vad) 트리 구조를 도시한 일 예시도이다.

[22] 도 5는 도 4의 프로세스의 배드(Vad) 트리 구조를 표현한 메모리 구조를 도시한 도면이다.

[23] 도 6은 이 발명에 따른 네트워크 접근 제어방법을 도시한 동작 흐름도이다.

[24] 도 7은 상기 도 6에서 임의의 한 프로세스의 코드 인젝션된 가상주소공간 영역을 탐지하는 단계(S62)를 상세하게 도시한 동작 흐름도이다.

[25] 도 8은 도 6의 단계 S62에서 탐지된 코드 인젝션된 가상주소공간을 기반으로 한 스레드 즉, 코드 인젝션 기반 스레드를 탐지하는 단계(S63)를 상세하게 도시한 동작 흐름도이다.

[26] 도 9는 도 6의 단계 S63에서 탐지된 코드 인젝션 기반 스레드 정보를 이용하여 네트워크 패킷을 제어하는 단계(S65)를 상세하게 도시한 동작 흐름도이다.

[27] [부호의 설명]

[28] 310 : 프로세스 검사부 311 : 프로세스 정보 수집부

[29] 312 : 코드 인젝션 탐지부 313 : 코드 인젝션 기반 스레드 탐지부

[30] 320 : 네트워크 감시부 321 : 네트워크 필터부

[31] 322 : 네트워크 차단부

[32]

발명의 실시를 위한 최선의 형태

[33] 이하, 첨부된 도면을 참조하여 이 발명에 따른 네트워크 접근 제어시스템 및 방법을 보다 상세하게 설명한다.

[34] 도 3은 이 발명에 따른 네트워크 접근 제어시스템의 구성 블록도이다.

[35] 이 발명에 따른 네트워크 접근 제어시스템은, 프로세스에 코드 인젝션 기반 스레드를 탐지하는 프로세스 검사부(310)와, 네트워크 필터링을 수행하여 네트워크에 접근하는 네트워크 패킷을 탐지하고 상기 탐지된 네트워크 패킷의 통신 주체가 상기 코드 인젝션 기반 스레드이면 상기 탐지된 네트워크 패킷의 전송이 차단되도록 하는 네트워크 감시부(320)를 포함한다.

[36] 프로세스 검사부(310)는 컴퓨터에서 실행중인 프로세스 정보를 수집하는 프로세스 정보 수집부(311)와, 상기 프로세스 정보 수집부에서 수집된 프로세스에 코드 인젝션된 영역을 탐지하는 코드 인젝션 탐지부(312)와, 상기 코드 인젝션된 영역을 기반으로 생성된 스레드를 탐지하는 코드 인젝션 기반 스레드 탐지부(313)를 포함한다.

[37] 네트워크 감시부(320)는 네트워크 필터링을 수행하여 네트워크 패킷을 탐지하는 네트워크 필터부(321)와, 상기 네트워크 패킷의 통신(송신 또는 수신)하는 주체의 프로세스 아이디 및 스레드 아이디를 기준으로 상기 네트워크 패킷의 전송을 허용 또는 차단하는 네트워크 차단부(322)를 포함한다.

[38] 프로세스 정보 수집부(311)는 컴퓨터에서 실행중인 프로세스에 대해 프로세스별로 프로세스구조체를 수집함으로써 프로세스 정보를 수집한다.

프로세스 정보 수집부(311)가 프로세스별 프로세스 정보를 수집하는 방법으로는, psapi.dll의 EnumProcesses 함수를 사용하여 프로세스를 열거하고 각 프로세스별 프로세스구조체를 수집하는 방법, ZwQuerySystemInformation 함수를 사용하여 프로세스 정보를 획득하여 프로세스별 프로세스구조체를 수집하는 방법, 커널 전역변수인 PspCidTable을 조사하여 프로세스별 프로세스구조체(EProcess)를 수집하는 방법, 커널 전역변수인 PspCidTable을 조사하여 스레드구조체(EThread)를 수집한 후 스레드구조체(EThread)의 프로세스 멤버를 통해서 프로세구조체(EProcess)를 수집하는 방법, 임의의 프로세스의 프로세스구조체(EProcess)의 핸들테이블(HandleTable)을 검사하여 나머지 프로세스별 프로세스구조체(EProcess)를 수집하는 방법, 실행된 프로세스를 관리하는 csrss.exe의 내부 프로세스 정보항목을 수집하는 방법, 프로세스구조체(EProcess)의 메모리 상주 패턴으로 메모리스캔을 통해 프로세스별 프로세스구조체(EProcess)를 수집하는 방법, 스레드구조체(EThread)의 메모리 상주 패턴으로 메모리스캔을 통해 스레드별 스레드구조체(EThread)를 수집한 후 수집된 스레드구조체(EThread)의 프로세스 멤버를 통해서 프로세스별 프로세스구조체(EProcess)를 수집하는 방법 등 다양한 방법이 포함된다.

[39] 코드 인젝션 탐지부(312)는 프로세스 정보 수집부(311)에서 수집된 각 프로세스에서 코드 인젝션된 영역을 탐지한다. 코드 인젝션 탐지부(312)는 프로세스에 할당된 가상주소공간 정보를 수집하는 가상주소공간 정보 수집부(312a)와, 수집된 가상주소공간에 코드 인젝션 여부를 조사하는 가상주소공간 조사부(312b)를 포함한다.

[40] 가상주소공간 정보 수집부(312a)가 프로세스에 할당된 가상주소공간 정보를 수집하는 방법으로는, 커널레벨(Kernel Level)에서 프로세스구조체(EProcess)의 배드루트(VadRoot) 멤버를 검사하는 방법과, 유저레벨(User Level)에서 VirtualQueryEx 함수 사용하는 방법이 있다.

[41] 가상주소공간 정보 수집부(312a)가 프로세스구조체(EProcess)의 배드루트(VadRoot) 멤버를 검사하여 프로세스에 할당된 가상주소공간 정보를 수집하는 방법을 상세하게 설명한다. 여기서, 배드(Vad)란 가상주소기술자(Virtual Address Descriptor)로서, 통상적으로 하나의 프로세스에는 다수의 가상주소공간들이 할당되는데, 각각의 배드(Vad)는 프로세스에 할당된 각각의 가상주소공간을 표현한다.

[42] 프로세스 정보 수집부(311)에서 수집된 프로세스구조체(EProcess)는 커널레벨(Kernel Level)에서 관리하는 객체로서, 프로세스의 내부 정보를 포함하며, 이중 배드루트(VadRoot) 멤버가 포함된다. 이 배드루트(VadRoot)는 트리 형태의 자료구조로서, 프로세스에 할당된 메모리 주소 정보를 관리한다.

[43] 도 4는 임의의 프로세스의 배드(Vad) 트리 구조를 도시한 일 예시도이다. 도 4에서 각 사각 블록은 하나의 배드(Vad)에 해당되며, 가상주소공간의

가상페이지번호(VPN; Virtual Page Number)의 시작주소(StartVPN) 및 끝주소(EndVPN)와, 메모리타입(Private/Mapped Exe/Mapped)과, 프로텍트(protect) 플래그 값(READWRITE/READONLY/EXE_WRITE_COPY)가 기재되어 있다. 가상주소공간 정보 수집부(312a)는 프로세스구조체(EProcess)의 배드루트(VadRoot)를 기반으로, 프로세스에 할당된 가상주소공간에 어떤 가상주소 범위(시작주소와 끝주소)가 할당되어 있는지, 그리고 그 속성(메모리타입, 프로텍트 플래그값)을 확인하며 수집할 수 있다. 도 5는 도 4의 프로세스의 배드(Vad) 트리 구조를 표현한 메모리 구조를 도시한 도면이다.

[44] 다음, 가상주소공간 정보 수집부(312a)가 유저레벨(User Level)에서 VirtualQueryEx 함수를 사용하여 프로세스에 할당된 가상주소공간 정보를 수집하는 방법을 상세하게 설명한다. 유저레벨(User Level)에서 VirtualQueryEx 함수를 사용하면, 쿼리한 가상주소에 대한 정보를 얻을 수 있는데, VirtualQueryEx 함수의 쿼리 결과가 MEMORY_BASIC_INFORMATION 구조체 형태로 얻어진다. 이 MEMORY_BASIC_INFORMATION 구조체는 앞서 설명한 배드루트(VadRoot)를 통해서 얻어지는 정보와 유사하며, 프로세스에 할당된 가상주소공간별로 해당 가상주소공간에 할당된 가상 주소 범위(시작주소와 끝주소)와, 그 속성(메모리타입, 프로텍트 플래그값)을 확인할 수 있다. VirtualQueryEx 함수의 자세한 사용법은 아래 링크에서 확인할 수 있다.

[45] http://msdn.microsoft.com/en-us/library/aa366907(v=vs.85).aspx

[46] http://msdn.microsoft.com/en-us/library/ms810627.aspx

[47] 가상주소공간 조사부(312b)는 가상주소공간 정보 수집부(312a)에서 수집한 각 가상주소공간의 속성 정보를 이용하여 해당 가상주소공간에 코드 인젝션(code injection)이 되어 있는지를 조사한다. 프로세스에 정상적으로 삽입되어 로드되는 DLL 파일의 경우, 그 DLL 파일에 할당된 가상주소공간의 메모리타입은 매핑 타입(Mapped Type)이고, 프로텍트 플래그값은 실행가능(executable)이다. 한편, 프로세스에 코드 인젝션된 경우, 그 코드 인젝션된 가상주소공간의 메모리타입은 프라이빗 타입(Private Type)이고 프로텍트 플래그값은 실행가능(executable)이다.

[48] 따라서, 가상주소공간 조사부(312b)는 프로세스에 할당된 가상주소공간들 중 메모리타입이 매핑 타입이 아니라 프라이빗 타입(Private type)이고, 프로텍트 플래그값이 실행가능(executable)이면, 해당 가상주소공간을 코드 인젝션 영역으로 판단한다. 가상주소공간 조사부(312b)는 코드 인젝션 영역으로 판단된 가상주소공간의 가상 주소 범위(시작주소와 끝주소) 정보를 코드 인젝션 기반 스레드 탐지부(313)에게 전달한다.

[49] 코드 인젝션 기반 스레드 탐지부(313)는 프로세스에서 실행되는 모든 스레드들에 대해 정보를 수집하는 스레드 정보 수집부(313a)와, 상기 수집된 스레드의 시작 주소값(Win32StartAddress)과 상기 가상주소공간 조사부(312b)에서 제공된 코드 인젝션된 가상주소공간의 가상 주소 범위를

비교하여 상기 수집된 스레드의 시작 주소값이 상기 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하면 상기 코드 인젝션된 가상주소공간에 코드 인젝션 기반 스레드가 존재하는 것으로 판단하는 주소 비교부(313b)를 포함한다.

[50] 스레드 정보 수집부(313a)가 프로세스에서 실행되는 모든 스레드들에 대한 정보를 수집하는 방법으로는, 프로세스구조체(EProcess)의 스레드리스트헤드(ThreadListHead)를 검사하는 방법과, ZwQuerySystemInformation 함수를 사용하는 방법 등 다양한 방법들이 포함된다.

[51] 먼저, 프로세스구조체(EProcess)의 스레드리스트헤드를 검사하는 방법을 설명한다. 프로세스구조체(EProcess)는 앞서 언급했듯이 커널 레벨(Kernel Level)에서 사용하는 프로세스에 대한 정보를 포함하는 구조체로서, 프로세스구조체의 스레드리스트헤드(ThreadListHead) 멤버에는 프로세스 내에서 동작중인 모든 스레드에 대한 정보가 포함된다.
스레드리스트헤드(ThreadListHead) 멤버는 연결리스트 형태이며, 여기에 연결된 각각의 엔트리를 통해서 스레드구조체(EThread)를 구할 수 있다.
스레드구조체(EThread)는 커널 레벨(Kernel Level)에서 사용하는 스레드에 대한 정보를 포함하는 구조체이다. 따라서, 스레드 정보 수집부(313a)는 스레드리스트헤드(ThreadListHead) 멤버를 통해 스레드구조체를 수집하고 검사하여, 해당 프로세스에 존재하는 모든 스레드들에 대한 정보를 수집한다.

[52] 다음, 스레드 정보 수집부(313a)가 ZwQuerySystemInformation 함수를 사용하여 스레드 정보를 수집하는 방법을 설명한다. ZwQuerySystemInformation 함수는 시스템 정보를 구할 때 사용하는 함수로서, SystemInformationClass에 따라서 얻어지는 정보 타입이 달라진다. 이 발명에서는 SystemInformationClass를 SystemProcessesAndThreadsInformation으로 설정하여 ZwQuerySystemInformation 함수를 호출함으로써, 프로세스별 스레드 정보를 얻는다.

[53] ZwQuerySystemInformation 함수 호출이 성공하면 SystemInformation으로 해당 정보에 접근할 수 있다. SystemInformation는 모든 프로세스와 각 프로세스에 대한 스레드 정보를 담고 있는 메모리 공간으로, SystemInformation를 통해서 SYSTEM_PROCESSES 구조체로 접근할 수 있고 결국 배열로 존재하는 SYSTEM_THREADS를 얻을 수 있다. 또한, SYSTEM_PROCESSES 구조체의 NextEntryDelta 멤버를 사용하여 다음 프로세스에 대한 SYSTEM_PROCESSES 구조체 영역을 찾아갈 수 있다.

[54] 다음, 주소 비교부(313b)는 프로세스에서 실행되는 임의의 스레드의 시작 주소값(Win32StartAddress)과, 상기 프로세스의 코드 인젝션된 가상주소공간의 가상 주소 범위를 비교하여 상기 수집된 스레드의 시작 주소값이 상기 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하면 상기 코드 인젝션 기반 스레드가 존재하는 것으로 판단하고, 그 코드 인젝션 기반 스레드의 프로세스 아이디 및 스레드 아이디를 차단스레드목록에 추가하여 네트워크

차단부(322)에게 전달한다.

[55] 네트워크 필터부(321)는 통상적인 네트워크 필터링을 수행하여 네트워크 패킷 트래픽을 감시한다. 네트워크 필터부(321)의 네트워크 필터링 방법으로서, 커널레벨에서는 네트워크 필터 드라이버를 사용할 수 있으며, 네트워크 필터 드라이버를 사용하는 방법에는 NDIS_OPEN_BLOCK 후킹 방법, Ndis Intermediate Driver 구현 방법, TDI 후킹 방법, WFP(Windows Filtering Platform)를 사용하는 방법 등 다양한 방법들이 포함된다. 한편, 네트워크 필터부(321)의 네트워크 필터링 방법으로서, 유저레벨에서는 특정 프로세스를 타깃으로 해당 특정 프로세스가 사용하는 소켓함수를 후킹하여 네트워크 패킷 트래픽을 감시할 수 있다.

[56] 네트워크 차단부(322)는 네트워크 패킷의 송신 또는 수신 주체의 프로세스 아이디와 스레드 아이디를 이용하여 해당 네트워크 패킷의 트래픽을 허용 또는 차단한다. 즉, 네트워크 패킷의 스레드 아이디가 차단스레드목록에 포함되면 상기 네트워크 패킷의 트래픽을 차단하고, 네트워크 패킷의 스레드 아이디가 차단스레드목록에 포함되지 않으면서 네트워크 패킷의 프로세스 아이디가 허용프로세스목록에 포함되면 상기 네트워크 패킷의 트래픽을 허용한다. 한편, 상기 네트워크 패킷의 프로세스 아이디가 허용프로세스목록에 포함되지 않으면 사용자 검증을 받고 상기 사용자 검증 결과를 기반으로 해당 트래픽을 허용 또는 차단한다.

[57] 상기와 같이 구성된 네트워크 접근 제어시스템의 동작을 설명한다.

[58] 도 6은 이 발명에 따른 네트워크 접근 제어방법을 도시한 동작 흐름도이다.

[59] 이 발명에 따른 네트워크 접근 제어시스템은, 컴퓨터에서 실행되는 프로세스 정보를 수집한다(S61). 다음, 수집된 각 프로세스에 대해 코드 인젝션된 가상주소공간 영역을 탐지한다(S62). 다음, 그 코드 인젝션된 가상주소공간에 코드 인젝션 기반 스레드가 존재하는지를 탐지한다(S63). 다음, 네트워크 패킷 트래픽이 탐지되면(S64), 상기 코드 인젝션 기반 스레드를 기반으로 상기 네트워크 패킷의 트래픽을 허용 또는 차단한다(S65).

[60] 도 7은 상기 도 6에서 임의의 한 프로세스의 코드 인젝션된 가상주소공간 영역을 탐지하는 단계(S62)를 상세하게 도시한 동작 흐름도이다.

[61] 코드 인젝션된 가상주소공간 영역을 탐지하는 단계(S62)는 다음과 같이 이루어진다. 네트워크 접근 제어시스템은 프로세스를 이루는 가상주소공간들에 대한 정보를 수집한다(S71). 하나의 가상주소공간을 검사대상 가상주소공간으로 선택하고(S72), 상기 검사대상 가상주소공간의 메모리타입이 프라이빗 타입인지를 검사한다(S73). 단계 S73에서 검사대상 가상주소공간의 메모리타입이 프라이빗 타입이면 상기 검사대상 가상주소공간의 프로텍트 플래그값이 실행가능(EXECUTABLE)인지를 검사한다(S74). 단계 S74에서 검사대상 가상주소공간의 프로텍트 플래그값이 실행가능이면, 상기 검사대상 가상주소공간을 코드 인젝션 영역으로 판단하고(S75), 코드 인젝션된

가상주소공간의 가상 주소 범위 정보를 코드 인젝션 기반 스레드 탐지부에게 전달한다(S76).

[62] 이후, 상기 검사대상 가상주소공간이 마지막 가상주소공간인지를 검사하여(S77), 마지막 가상주소공간이 아니면 단계 S72부터 반복 수행한다. 한편, 단계 S73에서 검사대상 가상주소공간의 메모리타입이 프라이빗 타입이 아니거나, 단계 S74에서 검사대상 가상주소공간의 프로텍트 플래그값이 실행가능이 아니면, 단계 S77로 바로 진행한다.

[63] 도 8은 도 6의 단계 S62에서 탐지된 코드 인젝션된 가상주소공간을 기반으로 한 스레드 즉, 코드 인젝션 기반 스레드를 탐지하는 단계(S63)를 상세하게 도시한 동작 흐름도이다.

[64] 코드 인젝션 기반 스레드를 탐지하는 단계(S63)는 다음과 같이 이루어진다. 네트워크 접근 제어시스템은 컴퓨터에서 실행되는 스레드 정보를 수집한다(S81). 수집된 스레드 정보 중 하나의 스레드를 검사대상 스레드로 선택하고(S82), 검사대상 스레드의 시작 주소값이 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하는지를 비교한다(S83). 단계 S83의 비교 결과, 검사대상 스레드의 시작 주소값이 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하면(S84), 상기 코드 인젝션된 가상주소공간에 코드 인젝션 기반 스레드가 존재하는 것으로 판단하고(S85), 코드 인젝션 기반 스레드의 프로세스 아이디 및 스레드 아이디를 차단스레드목록에 포함한 후 네트워크 차단부로 전달한다(S86). 단계 S86에서 상기 코드 인젝션 기반 스레드의 프로세스 아이디 및 스레드 아이디에 대해 사용자 검증 후 차단스레드목록에 포함시킬 수도 있다.

[65] 이후, 상기 검사대상 스레드가 마지막 스레드인지를 검사하여(S87), 마지막 스레드가 아니면 단계 S82부터 반복 수행한다. 한편, 단계 S84에서 검사대상 스레드의 시작 주소값이 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하지 않으면, 단계 S87로 바로 진행한다.

[66] 도 9는 도 6의 단계 S63에서 탐지된 코드 인젝션 기반 스레드 정보를 이용하여 네트워크 패킷을 제어하는 단계(S65)를 상세하게 도시한 동작 흐름도이다.

[67] 네트워크 패킷 제어단계(S65)는 다음과 같이 이루어진다. 네트워크 접근 제어시스템은 네트워크 필터링을 수행하여 컴퓨터에서 송수신되는 네트워크 패킷을 수집한다(S91). 그리고, 수집된 네트워크 패킷의 프로세스 아이디와 스레드 아이디 및 전체 경로 정보를 수집한다(S92). 상기 프로세스 아이디 및 스레드 아이디가 차단 스레드 목록에 포함되면(S93), 상기 수집된 네트워크 패킷의 트래픽을 차단한다(S94). 단계 S93에서 상기 수집된 네트워크 패킷의 프로세스 아이디 및 스레드 아이디가 차단 스레드 목록에 포함되지 않으면, 상기 프로세스 아이디가 허용 프로세스 목록에 포함되는지를 검사한다(S95). 단계 S95에서 프로세스 아이디가 허용 프로세스 목록에 포함되면 상기 수집된 네트워크 패킷의 트래픽을 허용한다(S96).

[68] 한편, 단계 S95에서 프로세스 아이디가 허용 프로세스 목록에 포함되지 않으면

상기 프로세스 아이디에 대해 사용자 검증을 요청하고(S97), 사용자로부터 허용되면(S98), 상기 수집된 네트워크 패킷의 프로세스 아이디를 허용 프로세스 목록에 추가한(S99) 후 상기 수집된 네트워크 패킷의 트래픽을 허용한다(S96). 한편, 단계 S98에서 사용자로부터 허용되지 않으면, 상기 수집된 네트워크 패킷의 트래픽을 차단한다(S94).

[69] 이상에서 본 발명에 대한 기술사상을 첨부도면과 함께 서술하였지만, 이는 본 발명의 가장 양호한 실시예를 예시적으로 설명한 것이지 본 발명을 한정하는 것은 아니다. 또한, 이 기술분야의 통상의 지식을 가진 자라면 누구나 본 발명의 기술사상의 범주를 이탈하지 않는 범위 내에서 다양한 변형 및 모방이 가능함은 명백한 사실이다.

[70]

청구범위

[청구항 1] 프로세스에 포함된 코드 인젝션 기반 스레드를 탐지하는 프로세스 검사부와,
네트워크 필터링을 수행하여 네트워크에 접근하는 네트워크 패킷을 탐지하고 상기 탐지된 네트워크 패킷의 통신 주체가 상기 코드 인젝션 기반 스레드이면 상기 탐지된 네트워크 패킷의 트래픽이 차단되도록 하는 네트워크 감시부를 포함한 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 2] 제 1 항에 있어서, 상기 프로세스 검사부는 컴퓨터에서 실행중인 프로세스 정보를 수집하는 프로세스 정보 수집부와, 상기 프로세스 정보 수집부에서 수집된 프로세스에서 코드 인젝션된 영역을 탐지하는 코드 인젝션 탐지부와, 상기 코드 인젝션된 영역을 기반으로 생성된 스레드를 탐지하는 코드 인젝션 기반 스레드 탐지부를 포함한 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 3] 제 2 항에 있어서, 상기 프로세스 정보 수집부는 상기 컴퓨터에서 실행중인 프로세스별로 프로세스구조체를 수집한 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 4] 제 2 항에 있어서, 상기 코드 인젝션 탐지부는 상기 수집된 프로세스에 할당된 가상주소공간의 가상 주소 범위와 속성 정보를 수집하는 가상주소공간 정보 수집부와, 상기 가상주소공간 정보 수집부에서 수집된 가상주소공간에 코드 인젝션 여부를 조사하는 가상주소공간 조사부를 포함한 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 5] 제 4 항에 있어서, 상기 가상주소공간 정보 수집부는 커널레벨(Kernel Level)에서 프로세스구조체(EProcess)의 배드루트(VadRoot) 멤버를 검사하여 상기 수집된 프로세스에 할당된 가상주소공간의 가상 주소 범위와 속성 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 6] 제 4 항에 있어서, 상기 가상주소공간 정보 수집부는 유저레벨(User Level)에서 VirtualQueryEx 함수 사용하여 상기 수집된 프로세스에 할당된 가상주소공간의 가상 주소 범위와 속성 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 7] 제 4 항에 있어서, 상기 가상주소공간 조사부는 상기 가상주소공간 정보 수집부에서 수집된 상기 가상주소공간의 속성 정보 중 메모리타입이 프라이빗 타입(Private type)이고, 프로텍트 플래그값이 실행가능(executable)이면, 상기 가상주소공간을 코드

인젝션 영역으로 판단하는 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 8] 제 2 항에 있어서, 상기 코드 인젝션 기반 스레드 탐지부는 컴퓨터에서 실행중인 스레드 정보를 수집하는 스레드 정보 수집부와, 상기 스레드 정보 수집부에서 수집된 스레드의 시작 주소값과 상기 가상주소공간 조사부에서 제공된 코드 인젝션된 가상주소공간의 가상 주소 범위를 비교하여 상기 수집된 스레드의 시작 주소값이 상기 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하면 상기 코드 인젝션된 가상주소공간에 코드 인젝션 기반 스레드가 존재하는 것으로 판단하는 주소 비교부를 포함한 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 9] 제 8 항에 있어서, 상기 스레드 정보 수집부는 프로세스구조체(EProcess)의 스레드리스트헤드(ThreadListHead)를 검사하여 상기 컴퓨터에서 실행중인 스레드 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 10] 제 8 항에 있어서, 상기 스레드 정보 수집부는 ZwQuerySystemInformation 함수를 사용하여 상기 컴퓨터에서 실행중인 스레드 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 11] 제 1 항에 있어서, 상기 네트워크 감시부는 상기 네트워크 패킷을 필터링하는 네트워크 필터부와, 상기 네트워크 패킷의 통신 주체의 프로세스 아이디 및 스레드 아이디를 기반으로 상기 네트워크 패킷의 트래픽을 허용 또는 차단하는 네트워크 차단부를 포함한 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 12] 제 11 항에 있어서, 상기 네트워크 필터부는 커널레벨에서 네트워크 필터 드라이버를 사용하여 네트워크 필터링하는 것을 특징으로 네트워크 접근 제어시스템.

[청구항 13] 제 11 항에 있어서, 상기 네트워크 필터부는 유저레벨에서 프로세스가 사용하는 소켓함수를 후킹하여 네트워크 필터링하는 것을 특징으로 하는 네트워크 접근 제어시스템.

[청구항 14] 네트워크 접근 제어시스템이 컴퓨터에서 실행되는 프로세스 정보를 수집하는 제1단계와,
상기 네트워크 접근 제어시스템이 상기 수집된 프로세스에 포함된 코드 인젝션된 가상주소공간 영역을 탐지하는 제2단계와,
상기 네트워크 접근 제어시스템이 상기 코드 인젝션된 가상주소공간에 코드 인젝션 기반 스레드가 존재하는지를 탐지하는 제3단계와,
상기 네트워크 접근 제어시스템이 네트워크 필터링하여 네트워크

패킷 트래픽을 수집하는 제4단계와,
상기 네트워크 접근 제어시스템이 상기 코드 인젝션 기반 스레드 정보를 이용하여 상기 네트워크 패킷의 트래픽을 허용 또는 차단하는 제5단계를 포함하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 15] 제 14 항에 있어서, 상기 제1단계는 상기 컴퓨터에서 실행중인 프로세스별로 프로세스구조체를 수집한 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 16] 제 14 항에 있어서, 상기 제2단계는, 상기 네트워크 접근 제어시스템이 상기 수집된 프로세스에 할당된 상기 가상주소공간들에 대한 정보를 수집하는 제21단계와,
상기 네트워크 접근 제어시스템이 검사대상 가상주소공간을 선택하고, 상기 검사대상 가상주소공간의 메모리타입과 프로텍트 플래그값을 검사하는 제22단계와,
상기 제22단계의 검사 결과, 상기 검사대상 가상주소공간의 메모리타입이 프라이빗 타입이고 프로텍트 플래그값이 실행가능(EXECUTABLE)이면 상기 검사대상 가상주소공간을 코드 인젝션 영역으로 판단하는 제23단계를 포함하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 17] 제 16 항에 있어서, 상기 제21단계는 커널레벨(Kernel Level)에서 프로세스구조체(EProcess)의 배드루트(VadRoot) 멤버를 검사하여 상기 수집된 프로세스에 할당된 가상주소공간의 가상 주소 범위와 속성 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 18] 제 16 항에 있어서, 상기 제21단계는 유저레벨(User Level)에서 VirtualQueryEx 함수 사용하여 상기 수집된 프로세스에 할당된 가상주소공간의 가상 주소 범위와 속성 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 19] 제 14 항에 있어서, 상기 제3단계는 상기 네트워크 접근 제어시스템이 상기 컴퓨터에서 실행되는 스레드 정보를 수집하는 제31단계와,
상기 네트워크 접근 제어시스템이 검사대상 스레드를 선택하고 상기 검사대상 스레드의 시작 주소값이 상기 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하는지를 비교하는 제32단계와,
상기 제32단계의 비교 결과, 상기 검사대상 스레드의 시작 주소값이 상기 코드 인젝션된 가상주소공간의 가상 주소 범위에 속하면 상기 코드 인젝션된 가상주소공간에 코드 인젝션 기반

스레드가 존재하는 것으로 판단하는 제33단계를 포함하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 20] 제 19 항에 있어서, 상기 제33단계 후 상기 코드 인젝션 기반 스레드의 프로세스 아이디 및 스레드 아이디를 차단스레드목록에 포함하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 21] 제 19 항에 있어서, 상기 제31단계는 프로세스구조체(EProcess)의 스레드리스트헤드(ThreadListHead)를 검사하여 상기 컴퓨터에서 실행중인 스레드 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 22] 제 19 항에 있어서, 상기 제31단계는 ZwQuerySystemInformation 함수를 사용하여 상기 컴퓨터에서 실행중인 스레드 정보를 수집하는 것을 특징으로 하는 네트워크 접근 제어방법.

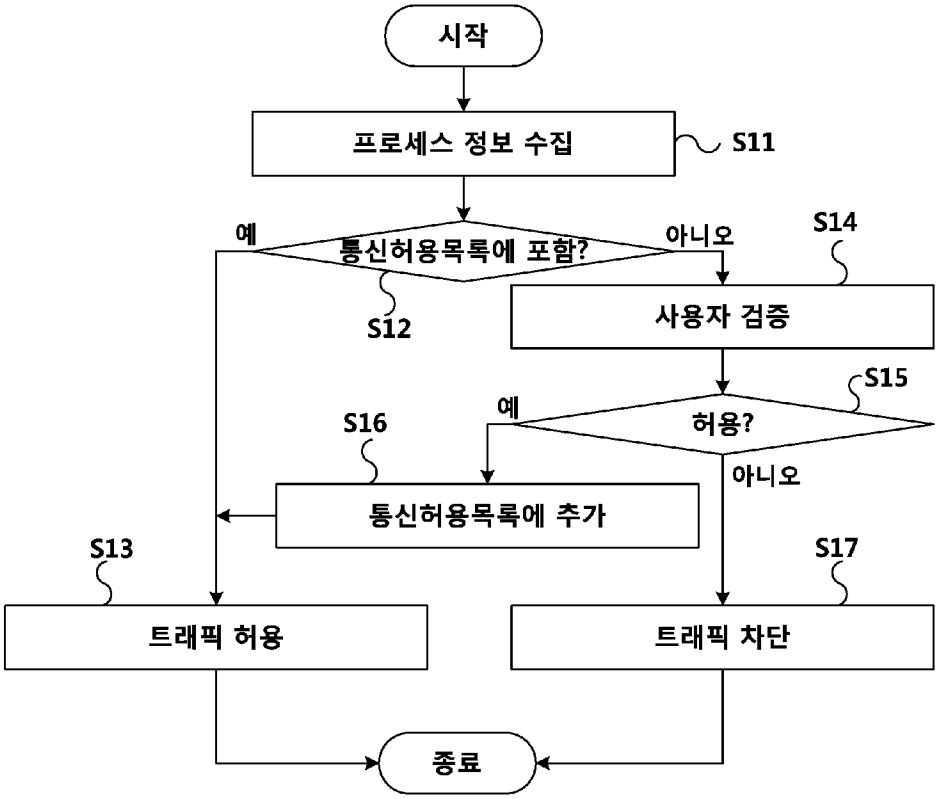
[청구항 23] 제 14 항에 있어서, 상기 제4단계는 커널레벨에서 네트워크 필터 드라이버를 사용하여 네트워크 필터링하는 것을 특징으로 네트워크 접근 제어방법.

[청구항 24] 제 14 항에 있어서, 상기 제4단계는 유저레벨에서 프로세스가 사용하는 소켓함수를 후킹하여 네트워크 필터링하는 것을 특징으로 하는 네트워크 접근 제어방법.

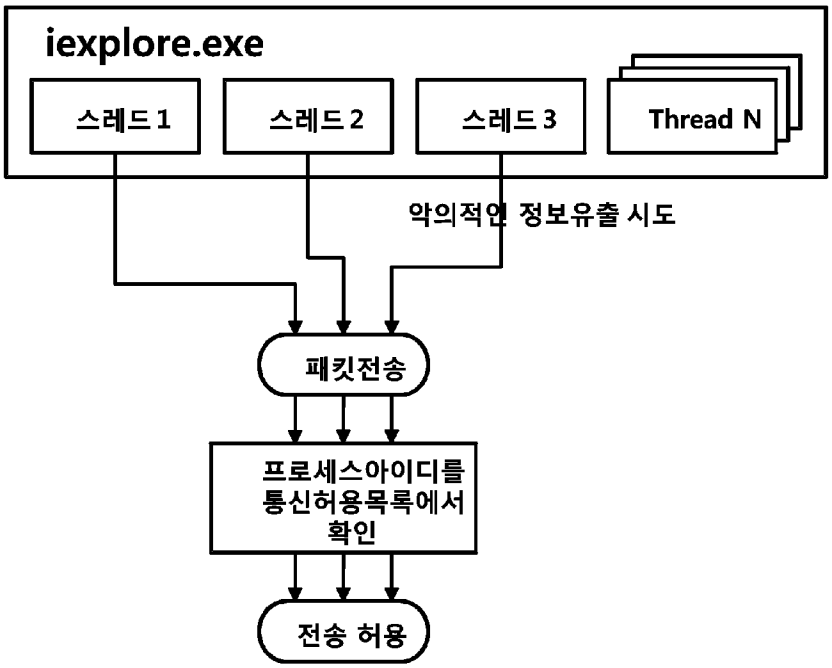
[청구항 25] 제 14 항에 있어서, 상기 제5단계는 상기 네트워크 접근 제어시스템이 상기 코드 인젝션 기반 스레드의 프로세스 아이디 및 스레드 아이디를 차단스레드목록에 저장하는 제51단계와,
상기 네트워크 접근 제어시스템이 상기 수집된 네트워크 패킷의 프로세스 아이디와 스레드 아이디 정보를 수집하는 제52단계와,
상기 네트워크 접근 제어시스템이 상기 제52단계에서 수집된 상기 프로세스 아이디 및 스레드 아이디가 상기 차단 스레드 목록에 포함되면, 상기 수집된 네트워크 패킷의 트래픽을 차단하는 제53단계와,
상기 네트워크 접근 제어시스템이 상기 수집된 네트워크 패킷의 프로세스 아이디 및 스레드 아이디가 차단 스레드 목록에 포함되지 않고 상기 프로세스 아이디가 허용 프로세스 목록에 포함되면, 상기 수집된 네트워크 패킷의 트래픽을 허용하는 제54단계를 포함하는 것을 특징으로 하는 네트워크 접근 제어방법.

[청구항 26] 제 25 항에 있어서, 상기 제51단계는 상기 네트워크 접근 제어시스템이 사용자 검증 후 상기 코드 인젝션 기반 스레드의 프로세스 아이디 및 스레드 아이디를 차단스레드목록에 저장하는 것을 특징으로 하는 네트워크 접근 제어방법.

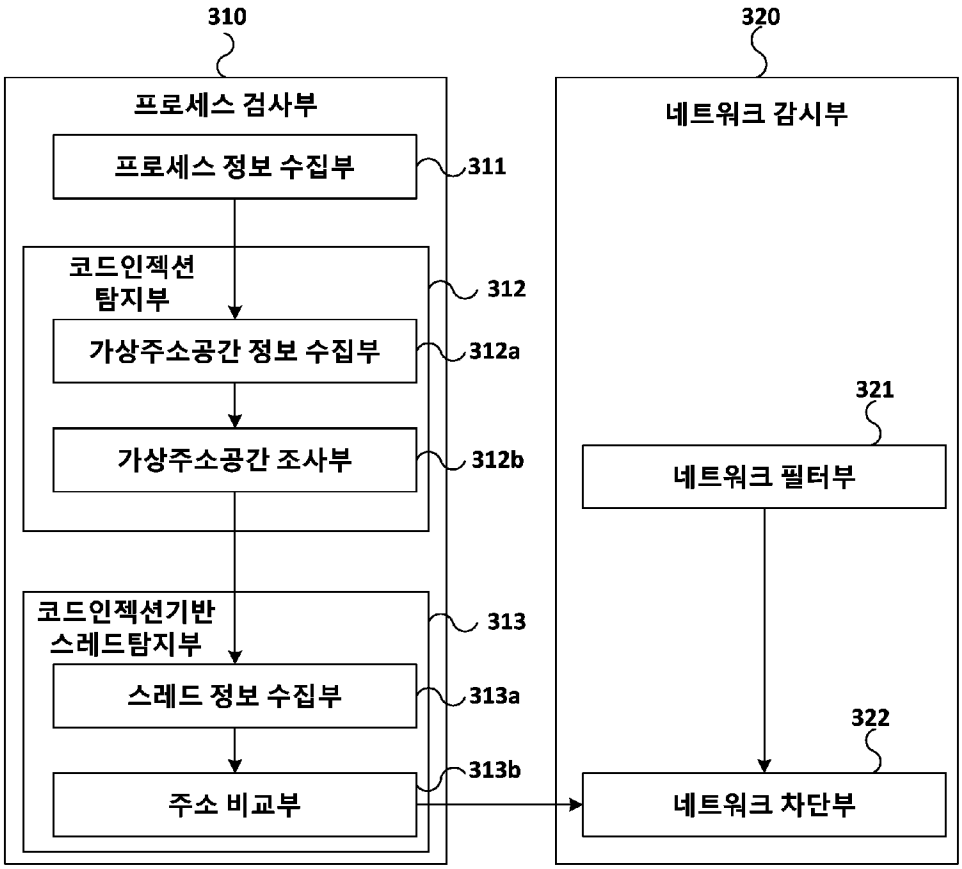
[Fig. 1]


시작
프로세스 정보 수집
S11
예
통신허용목록에 포함?
아니오
S14
S12
사용자 검증
S15
예
허용?
S16
아니오
통신허용목록에 추가
S13
S17
트래픽 허용
트래픽 차단
종료

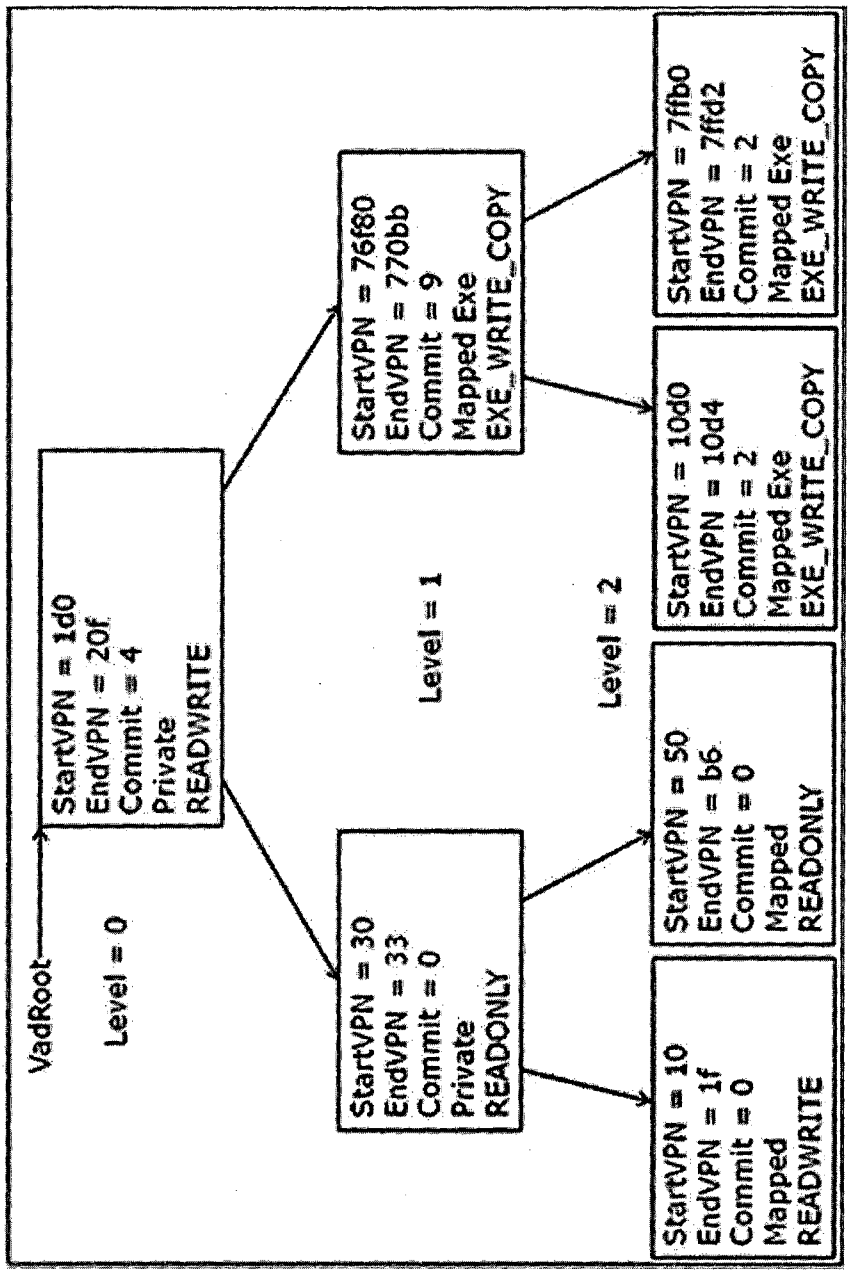
[Fig. 2]


iexplore.exe
스레드 1
스레드 2
스레드 3
Thread N
악의적인 정보유출 시도
패킷전송
프로세스아이디를
통신허용목록에서
확인
전송 허용

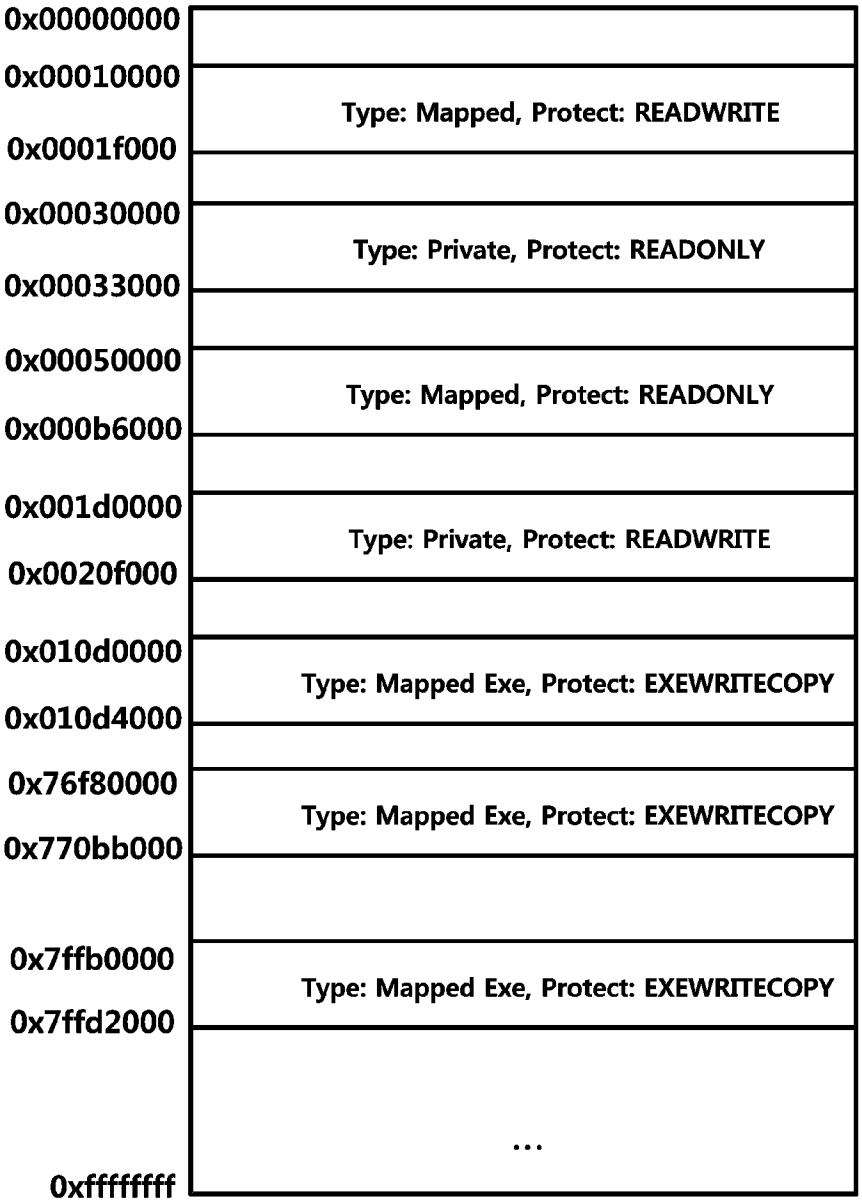
[Fig. 3]


310
320
프로세스 검사부
프로세스 정보 수집부
311
코드인젝션
탐지부
312
가상주소공간 정보 수집부
312a
가상주소공간 조사부
312b
코드인젝션기반
스레드탐지부
313
스레드 정보 수집부
313a
주소 비교부
313b
네트워크 감시부
321
네트워크 필터부
322
네트워크 차단부

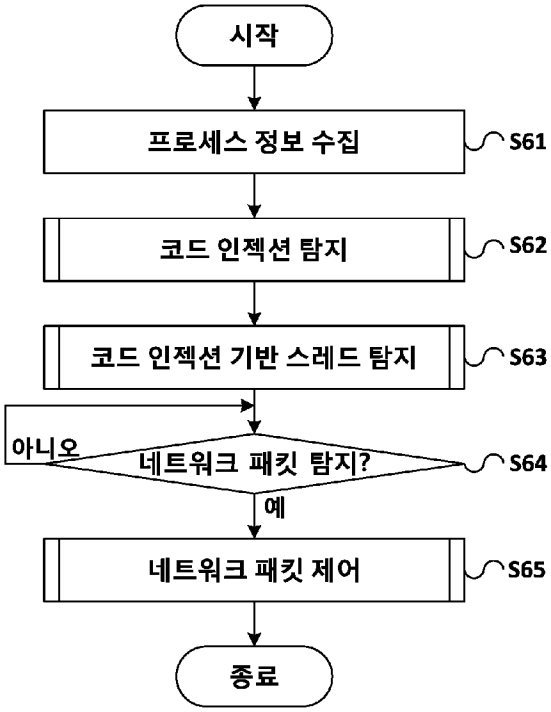
[Fig. 4]


VadRoot
Level = 0
StartVPN = 1d0
EndVPN = 20f
Commit = 4
Private
READWRITE
StartVPN = 30
EndVPN = 33
Commit = 0
Private
READONLY
Level = 1
StartVPN = 76f80
EndVPN = 770bb
Commit = 9
Mapped Exe
EXE_WRITE_COPY
StartVPN = 10
EndVPN = 1f
Commit = 0
Mapped
READWRITE
StartVPN = 50
EndVPN = b6
Commit = 0
Mapped
READONLY
Level = 2
StartVPN = 10d0
EndVPN = 10d4
Commit = 2
Mapped Exe
EXE_WRITE_COPY
StartVPN = 7ffb0
EndVPN = 7ffd2
Commit = 2
Mapped Exe
EXE_WRITE_COPY

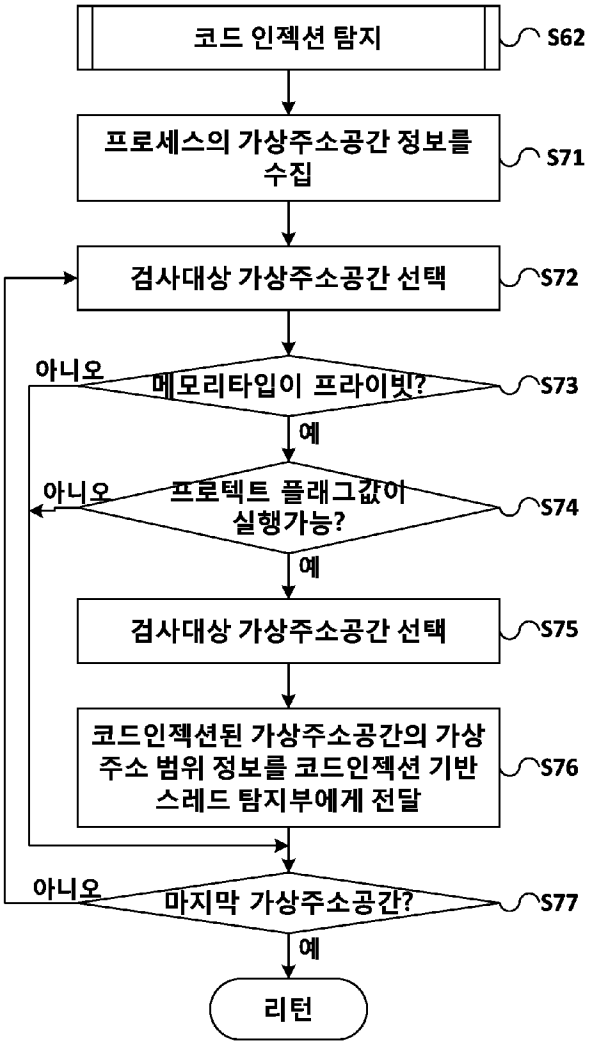
[Fig. 5]


0x00000000
0x00010000
Type: Mapped, Protect: READWRITE
0x0001f000
0x00030000
Type: Private, Protect: READONLY
0x00033000
0x00050000
Type: Mapped, Protect: READONLY
0x000b6000
0x001d0000
Type: Private, Protect: READWRITE
0x0020f000
0x010d0000
Type: Mapped Exe, Protect: EXEWRITECOPY
0x010d4000
0x76f80000
Type: Mapped Exe, Protect: EXEWRITECOPY
0x770bb000
0x7ffb0000
Type: Mapped Exe, Protect: EXEWRITECOPY
0x7ffd2000
...
0xffffffff

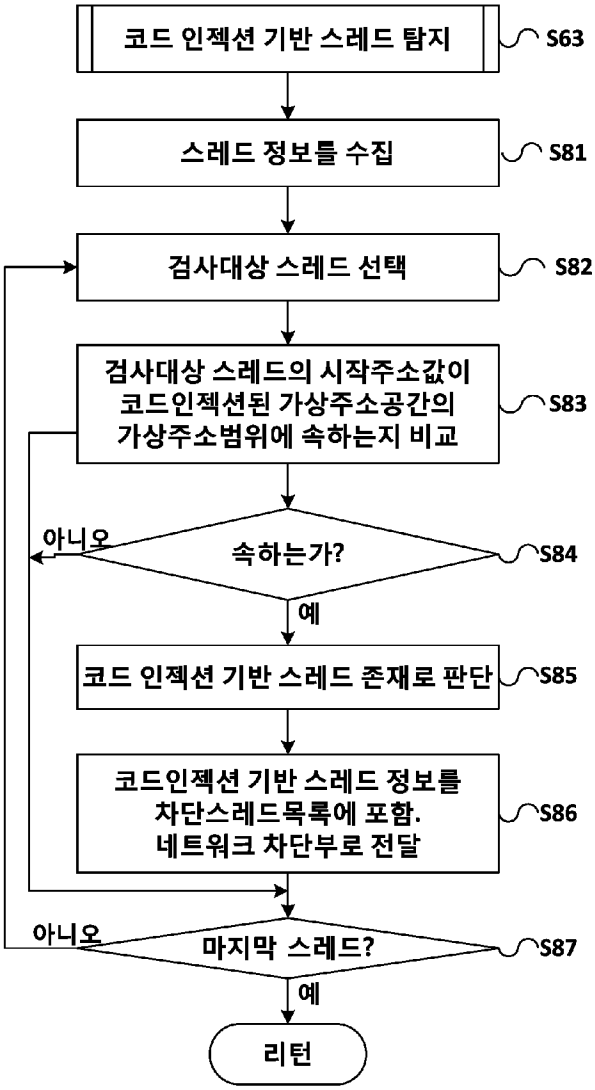
[Fig. 6]


시작
프로세스 정보 수집
S61
코드 인젝션 탐지
S62
코드 인젝션 기반 스레드 탐지
S63
아니오
네트워크 패킷 탐지?
S64
예
네트워크 패킷 제어
S65
종료

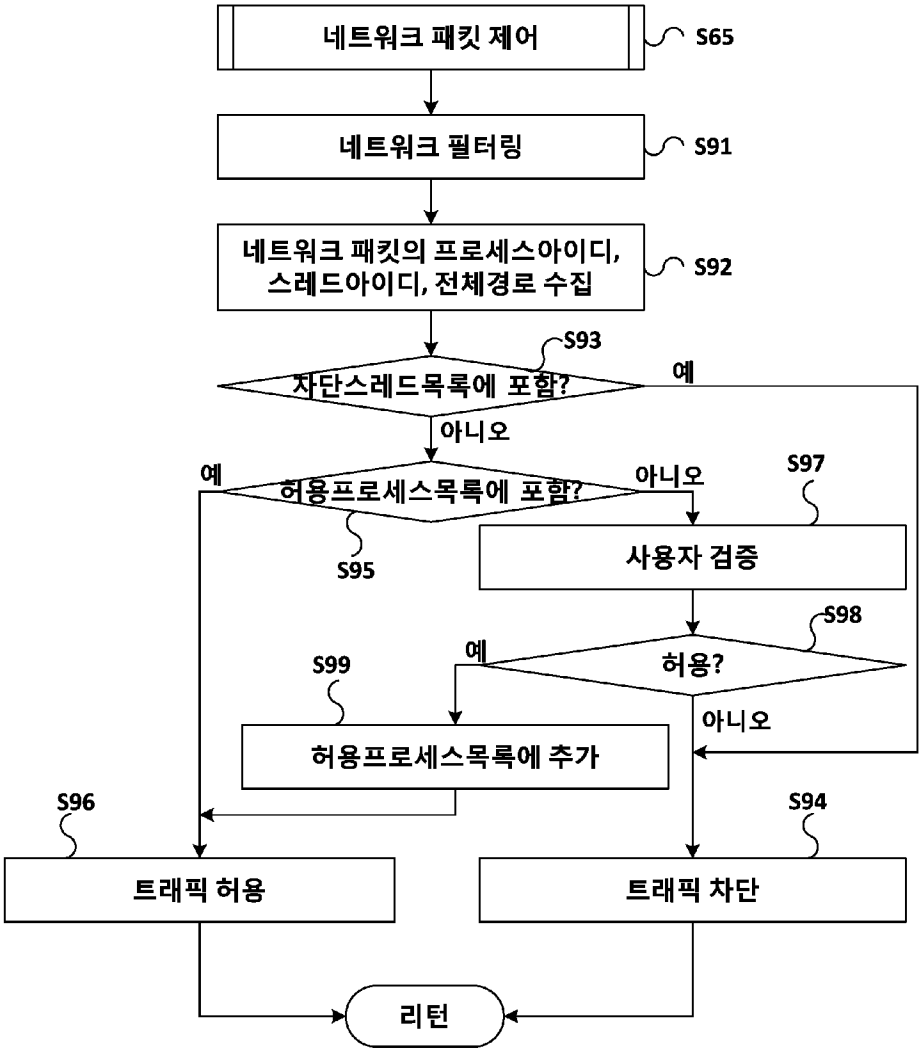
[Fig. 7]


코드 인젝션 탐지
S62
프로세스의 가상주소공간 정보를 수집
S71
검사대상 가상주소공간 선택
S72
아니오
메모리타입이 프라이빗?
S73
예
아니오
프로텍트 플래그값이 실행가능?
S74
예
검사대상 가상주소공간 선택
S75
코드인젝션된 가상주소공간의 가상주소 범위 정보를 코드인젝션 기반 스레드 탐지부에게 전달
S76
아니오
마지막 가상주소공간?
S77
예
리턴

[Fig. 8]


코드 인젝션 기반 스레드 탐지
S63
스레드 정보를 수집
S81
검사대상 스레드 선택
S82
검사대상 스레드의 시작주소값이
코드인젝션된 가상주소공간의
가상주소범위에 속하는지 비교
S83
아니오
속하는가?
S84
예
코드 인젝션 기반 스레드 존재로 판단
S85
코드인젝션 기반 스레드 정보를
차단스레드목록에 포함.
네트워크 차단부로 전달
S86
아니오
마지막 스레드?
S87
예
리턴

[Fig. 9]


네트워크 패킷 제어
S65
네트워크 필터링
S91
네트워크 패킷의 프로세스아이디, 스레드아이디, 전체경로 수집
S92
S93
차단스레드목록에 포함?
예
아니오
예
허용프로세스목록에 포함?
아니오
S95
S97
사용자 검증
S98
예
허용?
S99
아니오
허용프로세스목록에 추가
S96
S94
트래픽 허용
트래픽 차단
리턴