(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2010-0133646
(43) 공개일자 2010년12월22일

(51) Int. Cl.
G06F 9/44 (2006.01) *G06F 15/00* (2006.01)
(21) 출원번호 10-2009-0052306
(22) 출원일자 2009년06월12일
심사청구일자 2009년06월12일

(71) 출원인
고려대학교 산학협력단
서울 성북구 안암동5가 1
시큐아이닷컴 주식회사
서울 강남구 대치동 949-11 포르쉐센타 3~6층
(72) 발명자
김명섭
충청남도 연기군 조치원읍 신흥리 푸르지오1차 107-401
박준상
충청남도 연기군 조치원읍 서창리 레미안 C동 304호
(뒷면에 계속)
(74) 대리인
특허법인다나

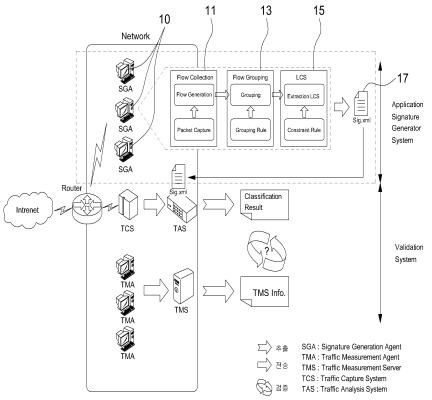
전체 청구항 수 : 총 16 항

(54) 응용프로그램 시그니쳐 생성방법

(57) 요 약

응용프로그램의 시그니쳐 생성방법을 개시한다. 상기 시그니쳐 생성방법은 패킷의 유동적 위치의 바이트 순서를 기반으로 시그니쳐를 생성하는 방법으로서, 컴퓨터에서 발생된 트래픽을 플로우(flow)-플로우는 동일한 소스 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 갖는 패킷의 집합임- 데이터로 저장하고 상기 플로우 데이터를 발생시킨 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 단계, 상기 수집된 플로우 데이터 중 상기 프로세스 정보를 기준으로 하여 응용프로그램과 관련된 프로세스 정보의 상기 플로우 데이터를 그룹핑하는 단계, 및 상기 그룹핑된 플로우 데이터로부터 LCS(Longest Common Subsequence)를 기반으로 상기 시그니쳐를 추출하는 단계를 포함한다.

대 표 도 - 도1



(72) 발명자

윤성호

경상북도 포항시 북구 용흥1동 현대타워맨션 104동 1205호

박진완

경상북도 경주시 동천동 선주아파트 B공구 다동 104호

오영석

서울특별시 노원구 상계9동 보람아파트 106-1208

김경모

서울특별시 강남구 대치3동 949-11 포르쉐타워 5층

특허청구의 범위

청구항 1

패킷의 유동적 위치의 바이트 순서를 기반으로 응용프로그램을 분류할 수 있는 고유한 패턴인 응용프로그램의 시그니쳐 생성방법에 있어서,

컴퓨터에서 발생된 트래픽을 플로우(flow)-플로우는 동일한 소스 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 갖는 패킷의 집합임- 데이터로 저장하고 상기 플로우 데이터를 발생시킨 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 단계;

상기 수집된 플로우 데이터 중 상기 프로세스 정보를 기준으로 하여 응용프로그램과 관련된 프로세스 정보의 상기 플로우 데이터를 그룹핑하는 단계; 및

상기 그룹핑된 플로우 데이터로부터 LCS(Longest Common Subsequence)를 기반으로 상기 시그니쳐를 추출하는 단계를 포함하는 응용프로그램의 시그니쳐 생성방법.

청구항 2

제 1 항에 있어서,

상기 플로우 데이터를 수집하는 단계는

상기 플로우 데이터를 구성하는 패킷의 수가 적어도 10개이거나 FIN 또는 RST 플래그를 포함하는 패킷을 포함하거나 상기 플로우 데이터가 생성된 후 일정 시간 내에 후속 패킷이 발생하는 경우 상기 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 응용프로그램의 시그니쳐 생성방법.

청구항 3

제 2 항에 있어서,

상기 일정 시간은 20초 내지 40초인 응용프로그램의 시그니쳐 생성방법.

청구항 4

제 1 항에 있어서,

상기 플로우 데이터를 그룹핑하는 단계는

상기 수집된 플로우 데이터를 전송 계층 프로토콜, 상기 플로우 데이터의 플로우의 방향, 80포트인지 유무, 고정포트인지 유무, 플로우 데이터의 포트번호 및 패킷 크기의 분포 중 적어도 하나를 기준으로 그룹핑하는 응용프로그램의 시그니쳐 생성방법.

청구항 5

제 4 항에 있어서,

상기 플로우 데이터의 플로우의 방향은 인바운드(inbound) 트래픽과 아웃바운드(outbound) 트래픽인 응용프로그램의 시그니쳐 생성방법.

청구항 6

제 1 항에 있어서,

상기 시그니쳐를 추출하는 단계는

상기 그룹핑된 플로우 데이터로부터 상기 LCS(Longest Common Subsequence)를 기반으로 시그니쳐 후보군을 추출하고 상기 시그니쳐 후보군으로부터 시퀀스 길이가 가장 긴 서브 시퀀스가 있는 플로우 데이터를 시그니쳐로 추출하는 응용프로그램의 시그니쳐 생성방법.

청구항 7

제 6 항에 있어서,

상기 추출된 시그니쳐로부터 패킷의 페이로드 시작 부분을 포함하는 플로우 데이터를 시그니쳐로 추출하는 응용프로그램의 시그니쳐 생성방법.

청구항 8

제 7 항에 있어서,

상기 시그니쳐를 추출하는 단계는

상기 시그니쳐를 구성하는 서브 시퀀스가 1바이트인 서브 시퀀스를 제거하는 단계를 포함하는 응용프로그램의 시그니쳐 생성방법.

청구항 9

제 7 항에 있어서,

상기 시그니쳐를 추출하는 단계는

상기 시그니쳐를 구성하는 서브 시퀀스 중 공통인 서브 시퀀스를 제거하는 단계를 더 포함하는 응용프로그램의 시그니쳐 생성방법.

청구항 10

패킷의 유동적 위치의 바이트 순서를 기반으로 응용프로그램을 분류할 수 있는 고유한 패턴인 응용프로그램의 시그니쳐 생성방법에 있어서,

컴퓨터에서 발생된 트래픽을 플로우(flow)-플로우는 동일한 소스 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 갖는 패킷의 집합임- 데이터로 저장하고 상기 플로우 데이터를 발생시킨 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 단계;및

상기 수집된 플로우 데이터 중 상기 프로세스 정보를 기준으로 하여 응용프로그램과 관련된 프로세스 정보의 상기 플로우 데이터로부터 LCS(Longest Common Subsequence)를 기반으로 상기 시그니쳐를 추출하는 단계를 포함하는 응용프로그램의 시그니쳐 생성방법.

청구항 11

제 10 항에 있어서,

상기 플로우 데이터를 수집하는 단계는

상기 플로우 데이터를 구성하는 패킷의 수가 적어도 10개이거나 FIN 또는 RST 플래그를 포함하는 패킷을 포함하거나 상기 플로우 데이터가 생성된 후 일정 시간 내에 후속 패킷이 발생하는 경우 상기 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 응용프로그램의 시그니쳐 생성방법.

청구항 12

제 10 항에 있어서,

상기 시그니쳐를 추출하는 단계는

상기 수집된 플로우 데이터로부터 상기 LCS(Longest Common Subsequence)를 기반으로 시그니쳐 후보군을 추출하고 상기 시그니쳐 후보군으로부터 시퀀스 길이가 가장 긴 서브 시퀀스가 있는 플로우 데이터를 시그니쳐로 추출하는 응용프로그램의 시그니쳐 생성방법.

청구항 13

제 12 항에 있어서,

상기 추출된 시그니쳐로부터 패킷의 페이로드 시작 부분을 포함하는 플로우 데이터를 시그니쳐로 추출하는 응용프로그램의 시그니쳐 생성방법.

청구항 14

제 13 항에 있어서,

상기 시그니쳐를 추출하는 단계는

상기 시그니쳐를 구성하는 서브 시퀀스가 1바이트인 서브 시퀀스를 제거하는 단계를 포함하는 응용프로그램의 시그니쳐 생성방법.

청구항 15

제 14 항에 있어서,

상기 시그니쳐를 추출하는 단계는

상기 시그니쳐를 구성하는 서브 시퀀스 중 공통인 서브 시퀀스를 제거하는 단계를 더 포함하는 응용프로그램의 시그니쳐 생성방법.

청구항 16

제1항 내지 제15항 중 어느 한 항에 기재된 응용프로그램의 시그니쳐 생성방법을 수행하기 위해 디지털 처리 장치에 의해 실행될 수 있는 명령어들의 프로그램이 구현되어 있으며, 상기 디지털 처리 장치에 의해 판독될 수 있는 프로그램이 기록된 기록매체.

명 세 서

발명의 상세한 설명

기 술 분 야

[0001] 본 발명은 응용 레벨 트래픽 분류방법에 관한 것으로, 특히 응용프로그램의 시그니쳐 생성방법에 관한 것이다.

배 경 기 술

[0002] 응용프로그램의 시그니쳐는 응용프로그램 별로 그들만이 사용하는, 다른 응용프로그램과는 구분되는 전체 응용프로그램의 트래픽으로부터 해당 응용프로그램을 분류할 수 있는 고유한 패턴을 말한다.

[0003] 종래 응용프로그램 시그니쳐 생성방법의 데이터 수집 방법은 시그니쳐 생성을 위한 응용프로그램의 특정 기능을 독립적으로 반복 수행하여 데이터를 수집하였다. 패킷 페이로드(packet payload)의 고정된 위치에서 특정 바이트의 발생 빈도에 기초하여 시그니쳐를 추출한다. 이와 같은 방법은 목적에 맞는 정확한 데이터 수집이 어렵기 때문에 시그니쳐의 신뢰도를 보장할 수 없다.

발명의 내용

해결 하고자하는 과제

[0004] 본 발명이 이루고자 하는 기술적 과제는 높은 정확성을 보장하는 응용프로그램별 시그니쳐를 생성하고, 시그니쳐의 생성 효율을 높여 응용프로그램의 지속적인 업그레이드에 대처하기 위한 시그니쳐 생성방법을 제공하는 데 있다.

과제 해결수단

[0005] 본 발명의 일 양태에 따르면 패킷의 유동적 위치의 바이트 순서를 기반으로 응용프로그램을 분류할 수 있는 고유한 패턴인 응용프로그램의 시그니쳐 생성방법에 있어서, 컴퓨터에서 발생된 트래픽을 플로우(flow)-플로우는 동일한 소스 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 갖는 패킷의 집합임- 데이터로 저장하고 상기 플로우 데이터를 발생시킨 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 단계, 상기 수집된 플로우 데이터 중 상기 프로세스 정보를 기준으로 하여 응용프로그램과 관련된 프로세스 정보의 상기 플로우 데이터를 그룹핑하는 단계, 및

상기 그룹핑된 플로우 데이터로부터 LCS(Longest Common Subsequence)를 기반으로 상기 시그니쳐를 추출하는 단계를 포함하는 응용프로그램의 시그니쳐 생성방법을 제공한다.

[0006] 본 발명의 다른 양태에 따르면 패킷의 유동적 위치의 바이트 순서를 기반으로 응용프로그램을 분류할 수 있는 고유한 패턴인 응용프로그램의 시그니쳐 생성방법에 있어서, 컴퓨터에서 발생된 트래픽을 플로우(flow)-플로우는 동일한 소스 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 갖는 패킷의 집합임- 데이터로 저장하고 상기 플로우 데이터를 발생시킨 프로세스 정보를 기준으로 상기 플로우 데이터를 수집하는 단계, 및 상기 수집된 플로우 데이터 중 상기 프로세스 정보를 기준으로 하여 응용프로그램과 관련된 프로세스 정보의 상기 플로우 데이터로부터 LCS(Longest Common Subsequence)를 기반으로 상기 시그니쳐를 추출하는 단계를 포함하는 응용프로그램의 시그니쳐 생성방법을 제공한다.

효 과

[0007] 본 실시예에 의한 시그니쳐 생성방법은 시그니쳐 생성을 위한 데이터 수집과 시그니쳐 추출과정에서 관리자의 개입을 최소화한다. 패킷을 기준으로 시그니쳐를 생성하므로, 플로우의 스트림 형태로 재조합하는 과정이 불필요하여 분류 시스템의 부가적인 프로세싱 과정과 저장공간을 줄일 수 있다. 또한, 패킷의 손실, 비대칭 라우팅으로 인해 플로우의 스트림이 완전하게 구성되지 못하는 경우에도 유연한 대처가 가능하다. 본 실시예에 의한 시그니쳐 생성방법으로부터 추출된 시그니쳐는 QoS, IPS 장비들에서 효과적으로 사용되며 장비의 정확성을 보장하고 유지한다. 높은 정확성을 보장하면서 시그니쳐의 생성 효율을 높여 응용프로그램의 지속적인 업그레이드에 대처할 수 있다.

발명의 실시를 위한 구체적인 내용

[0008] 본 발명은 다양한 변경을 가할 수 있고 여러 가지 실시예를 가질 수 있는 바, 특정 실시예들을 도면에 예시하고 설명하고자 한다. 그러나, 이는 본 발명을 특정한 실시 형태에 대해 한정하려는 것이 아니며, 본 발명의 사상 및 기술 범위에 포함되는 모든 변경, 균등물 내지 대체물을 포함하는 것으로 이해되어야 한다.

[0009] 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.

[0010] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가지고 있다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥 상 가지는 의미와 일치하는 의미를 가지는 것으로 해석되어야 하며, 본 출원에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.

[0011] 이하, 첨부된 도면을 참조하여 실시예를 상세히 설명하되, 도면 부호에 관계없이 동일하거나 대응하는 구성 요소는 동일한 참조 번호를 부여하고 이에 대한 중복되는 설명은 생략하기로 한다.

[0012] 본 실시예에서 응용프로그램의 시그니쳐는 패킷의 페이로드로부터 추출하고 해당 응용프로그램의 플로우내에서 변하지 않은 바이트의 시퀀스로 정의한다. 여기서, 플로우는 동일한 소스 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 갖는 패킷의 집합을 나타낸다.

[0013] 본 실시예에 의한 시그니쳐 생성방법은 개별 컴퓨터에서 발생하는 모든 트래픽을 프로세스 기준으로 수집하고 패킷의 페이로드를 포함한 플로우 단위로 수집한다. 유동적 위치의 바이트 순서를 기반으로 시그니쳐를 추출한다. 본 실시예에 의한 시그니쳐 생성방법은 응용 프로그램에서 발생하는 모든 플로우를 대상으로 하며, 패킷을 기준으로 응용 프로그램 시그니쳐를 생성한다.

[0014] 도 1은 본 실시예에 의한 시그니쳐 생성장치의 구조를 나타내는 도면이다.

[0015] 도 1을 참조하면, 시그니쳐 생성장치(10)는 플로우 데이터 수집부(11), 플로우 데이터 그룹핑부(13) 및 시그니쳐 추출부(15)를 포함한다. 플로우 데이터 수집부(11)는 시그니쳐 생성장치(10)의 입력 데이터로서 플로우 데이

터를 수집하고, 플로우 데이터 그룹핑부(13)는 이러한 플로우 데이터를 그룹핑하며, 시그니쳐 추출부(15)는 그룹핑된 플로우 데이터로부터 최적의 시그니쳐를 추출한다. 시그니쳐 생성장치(10)를 통해 추출된 시그니쳐(17)는 XML파일로 저장된다.

[0016] 이하, 본 실시예에 의한 플로우 데이터 수집부(11)의 플로우 데이터 수집방법을 후술한다.

[0017] 컴퓨터가 실행되면서 프로세스를 발생시키는 모든 트래픽을 캡쳐하여 프로세스를 기준으로 트래픽을 저장한다. 트래픽은 IP(source IP), 목적지 IP(destination IP), 소스 포트(source port), 목적지 포트(destination port) 및 전송 계층 프로토콜(protocol)을 기준으로 플로우 형태로 저장되며, 플로우데이터는 캡쳐 시간을 기준으로 10번째까지의 패킷을 포함할 수 있다.

[0018] 플로우 데이터 수집부(11)의 입력 데이터는 시그니쳐를 생성하고자하는 응용프로그램의 트래픽을 포함한 컴퓨터에서 발생한 모든 트래픽이다. 이러한 트래픽은 플로우 형태로 저장되며 해당 플로우를 발생시킨 프로세스 정보를 맵핑하여 프로세스를 기준으로 플로우 데이터가 저장된다.

[0019] 플로우 데이터 수집부(11)의 출력 데이터는 해당 프로세스의 디렉토리에 저장된다. 하나의 응용프로그램이 발생시키는 트래픽은 응용프로그램의 동작에 따라 다른 형태를 갖는다. 예를 들어 하나의 응용프로그램은 검색, 업데이트, 파일다운 등의 다양한 형태의 트래픽을 발생시킨다. 서로 다른 패턴을 갖는 트래픽을 시그니쳐 생성장치의 입력 데이터로 사용한다면 시그니쳐의 추출 가능성이 감소하고, 정확성이 떨어진다. 따라서 파일 이름을 결정할 때 응용프로그램의 각 기능을 이루는 최소 단위로 생성할 수 있다.

[0020] 플로우 데이터의 데이터 포맷은 전송 계층 프로토콜, 플로우의 전송 방향, 프로세스명, 서버 IP, 서버 Port를 기준으로 결정된다. 이때 서버/클라이언트는 SYN 패킷에 의해서 결정된다. 예를 들어 A와 B 사이의 통신에서 A에서 B로 SYN 연결 요청이 발생하는 경우 도착지의 IP가 서버로 결정된다. 즉 B가 두 연결의 서버가 되며, A가 클라이언트가 된다. 서버/클라이언트가 결정되면 플로우의 전송 방향을 서버를 기준으로 결정한다. 서버에서 수신하는 플로우는 인바운드(Inbound) 트래픽이 되며, 서버에서 송신하는 플로우는 아웃바운드(Outbound) 트래픽으로 결정된다. 이와 같은 데이터 포맷은 본 실시예에 의한 시그니쳐 추출 장치의 입력데이터를 결정하는 조건으로 사용될 수 있다.

[0021] 도 2는 본 실시예에 의한 플로우 데이터 수집 방법을 나타내는 순서도이다.

[0022] 도 2를 참조하면, 우선 네트워크 카드를 선택한다(S110). 그런 다음, 선택된 네트워크 카드에 송수신되는 패킷을 수집한다(S120). 패킷이 캡쳐되면(S130) 캡쳐된 패킷의 IP 및 TCP/UDP 헤더를 추출하고(S140) 패킷이 캡쳐되지 않으면 다시 패킷을 수집한다(S120). 그런 다음, 파싱(parsing) 결과 캡쳐된 패킷이 TCP 패킷인지 판단한다(S150).

[0023] S150에서 TCP 패킷이라면, SYN 플래그(flag)이 설정되어 있는 패킷인지 판단한다(S160). SYN 플래그(flag)이 설정되어 있는 패킷이라면 실행중인 프로세스들이 사용하는 소켓정보 및 캡쳐된 패킷을 비교하여 해당 패킷을 발생한 프로세스 이름을 저장한다(S170). 그런 다음, 플로우 데이터를 생성하고 프로세스 이름을 함께 저장한다(S180).

[0024] S160에서 SYN 플래그(flag)이 설정되어 있는 패킷이 아니라면 플로우 데이터의 마지막인 FIN 또는 RST 플래그가 설정된 패킷인지 판단한다(S190). FIN 또는 RST 플래그가 설정된 패킷이라면 데이터의 프로세스명을 기준으로 디렉토리를 생성하고 플로우 데이터를 저장한다(S200). FIN 또는 RST 플래그가 설정된 패킷이 아니라면 패킷이 캡쳐된 시간을 기준으로 순차적 플로우 데이터에 저장되는 패킷이 10개 이상이거나 플로우 데이터가 생성된 후 30초(time expired) 내외 이내에 후속 패킷이 발생했는지 판단한다(S210). 플로우 데이터가 생성된 후 20초 내지 40초(time expired) 이내에 후속 패킷이 발생했는지 판단한다. S210에서 패킷이 캡쳐된 시간을 기준으로 순차적 플로우 데이터에 저장되는 패킷이 10개 이상이거나 플로우 데이터가 생성된 후 30초(time expired)이내에 후속 패킷이 발생했다면 데이터의 프로세스명을 기준으로 디렉토리를 생성하고 플로우 데이터를 저장한다(S200) S210에서 그렇지 않다면 플로우 데이터에 해당하는 패킷을 추가한다(S220).

[0025] 다시 정리하면, S180을 통해 플로우 데이터를 생성하고 메모리에 저장하고 S220를 통해 해당 플로우 데이터의 다음 패킷이 캡쳐되면 해당 플로우 데이터에 패킷을 추가한다. S180 및 S220를 통해서 메모리에 저장되어 있는 플로우 데이터가 패킷저장조건인 S190 및 S210의 조건을 만족하면 데이터의 프로세스명을 기준으로 디렉토리를 생성하고 플로우 데이터를 저장한다(S200).

[0026] 다시 S150에서 TCP 패킷이 아니고 UDP 패킷이라면(S230) 플로우 데이터의 시작을 알 수 없기 때문에 플로우 데

이터의 초기 패킷을 설정하는 단계(S160)과 끝을 결정하는 단계(S190)을 제외하고는 전술한 나머지와 동일하다.

[0027] 이하, 본 실시예에 의한 플로우 데이터 그룹핑부(13)의 플로우 데이터 그룹핑방법을 후술한다.

[0028] 플로우 데이터 수집부(11)를 통해 응용프로그램별로 플로우 데이터를 생성하고 결정된 데이터 포맷에 따라 플로우 데이터를 저장한다. 본 실시예에 의한 플로우 데이터 그룹핑부(13)는 이로부터 시그니쳐의 추출 가능한 그룹을 생성하여 시그니쳐 추출부(15)에 정확한 입력 데이터를 제공한다.

[0029] 플로우 데이터 그룹핑부(13)의 입력 데이터는 플로우 데이터 수집부(11)의 결과로 저장되어진 각 응용프로그램의 모든 플로우 데이터가 된다. 플로우 데이터 그룹핑부(13)에 입력되는 모든 플로우 데이터에 대해서 전송 계층 프로토콜을 기준으로 그룹핑할 수 있다. UDP 프로토콜을 사용하는 플로우는 응용프로그램 내에서 하나의 독립된 기능을 수행하기 때문에 전송 계층 프로토콜 단계에서 그룹핑할 수 있다. TCP 프로토콜을 사용하는 플로우는 응용프로그램의 다양한 기능을 수행하기 때문에 세분화하여 그룹핑할 수 있다. 따라서, 전송 계층 프로토콜로 그룹핑된 TCP 플로우는 전송 방향에 따라 그룹핑될 수 있다. 전송 방향은 서버를 기준으로 인/아웃바운드(In/Outbound) 트래픽으로 분류된다. HTTP의 요청 메시지와 응답 메시지는 다른 형태의 페이로드를 갖는다. 요청 메시지는 페이로드의 시작 부분에 "GET" 또는 “POST”를 포함하며, 응답 메시지의 페이로드의 시작 부분은 "HTTP1."을 포함한다. 전송 방향에 따라 분류된 플로우는 서버의 포트를 기준으로 그룹핑된다. 예를 들어, 웹(web) 포트(80, 80xx), 고정 포트, 유동 포트를 고려하여 그룹핑 될 수 있다. 여기서, 고정 포트는 응용프로그램에서 특정 서비스를 제공하기 위해 사용하는 고유 포트번호를 가진 포트를 말하고, 유동 포트는 응용프로그램에서 서비스를 제공하기 위해 동적으로 생성하는 포트를 말한다.

[0030] 웹 포트의 경우, 플로우의 페이로드가 존재하는 첫 번째, 두 번째의 패킷의 크기가 유동적이기 때문에 패킷 크기에 따른 분류가 불가능하다. 웹 포트의 시그니쳐의 발생 형태를 보면, 대부분 URL에서 추출되기 때문에 웹 포트 레벨에서 시그니쳐가 생성 가능하다. 따라서 80포트 또는 80xx포트에 대해서는 포트 레벨에서 그룹핑한다. 80, 80xx 포트를 제외한 나머지 포트에 대해서는 고정 포트와 유동 포트를 구분하여 그룹핑할 수 있다. 이하에서, 80포트는 80포트 또는 80xx포트를 나타낸다.

[0031] 도 3은 본 실시예에 의한 플로우 데이터 그룹핑 방법을 나타내는 순서도이다.

[0032] 도 3을 참조하면, 프로세스명을 기준으로 생성된 디렉토리내에 저장된 플로우 데이터로부터 응용프로그램과 관련된 프로세스명의 디렉토리내의 플로우 데이터가 입력된다(S410). 그런 다음, 입력된 플로우 데이터를 프로토콜 즉, TCP 및 UDP을 기준으로 그룹핑한다. 따라서, 입력된 플로우 데이터가 TCP 프로토콜인지 판단한다(S420). 입력된 플로우 데이터가 TCP 프로토콜이라면 플로우의 방향에 따라 그룹핑한다(S430) 여기서, 플로우의 방향은 인바운드(inbound) 트래픽과 아웃바운드(outbound) 트래픽으로 분류될 수 있다. 그런 다음, 입력된 플로우 데이터가 사용하는 포트가 80 포트 또는 80xx 포트인지를 판단한다(S440). 웹 포트 즉, 80 포트 또는 80xx 포트라면, 다시 포트를 기준으로 그룹핑한다(S450). 예를 들어, 플로우의 포트번호를 기준으로 그룹핑한다.

[0033] S440에서 입력된 플로우 데이터가 사용하는 포트가 80 포트 또는 80xx 포트가 아니라면, 데이터 포맷에 따라 하나의 파일에 하나의 플로우만 저장되는지 판단한다. 즉, 입력된 플로우 데이터가 사용하는 포트가 유동포트인지 판단한다(S460). 입력된 플로우 데이터가 사용하는 포트가 유동포트라면, 패킷 크기의 분포를 통해 그룹핑한다(S480).

[0034] 도 4는 2개의 ID로 메신저를 이용해 채팅 초기의 패킷 크기 분포를 나타내는 그래프이고, 도 5는 2개의 ID로 메신저를 이용해 로그인 초기의 패킷 크기 분포를 나타내는 그래프이며, 도 6은 2개의 ID로 메신저를 이용해 파일 전송 초기의 패킷 크기 분포를 나타내는 그래프이고, 도 7은 2개의 ID로 메신저를 이용해 원격접속 초기의 패킷 크기 분포를 나타내는 그래프이다.

[0035] 도 4 및 도 5를 참조하면, 2개의 ID 응용프로그램의 동작별로 동일한 패킷의 크기를 보인다. 응용프로그램은 응용 레벨의 프로토콜을 통해 구현되고 프로토콜은 일정한 규칙을 갖고, 플로우의 초기에는 콘텐츠의 전송이 아닌 컨트롤을 위한 메시지를 송수신하기 때문이다. 도 6 및 도 7을 참조하면, 2개의 ID의 패킷의 크기가 서로 다른 것을 확인할 수 있다.

[0036] 다시 도 3을 참조하면, 입력된 플로우 데이터가 사용하는 포트가 유동포트가 아니라면, 플로우 데이터의 포트번호를 기준으로 그룹핑한다(S470). 그런 다음, 패킷 크기의 분포를 통해 그룹핑한다(S480).

[0037] S420에서 입력된 플로우 데이터가 TCP 프로토콜이 아니라면 별다른 그룹핑을 하지 않고 TCP 프로토콜이 아닌 입력된 플로우 데이터를 그룹핑한다(S490).

[0038] 그런 다음, 응용프로그램과 관련된 프로세스명의 디렉토리내에 플로우 데이터의 존재유무를 확인한다(S500). 모든 플로우 데이터의 처리가 끝나면 종료한다.

[0039] 이하, 본 실시예에 의한 시그니쳐 추출부(15)의 시그니쳐 추출방법을 후술한다.

[0040] 본 실시예에 의한 시그니쳐 추출방법는 LCS(Longest Common Subsequence)를 기반으로 한다. LCS는 2개의 입력 스트링으로부터 가장 긴 공통 서브시퀀스(subsequence)를 찾는 문제이다. 예를 들어, 2개의 문자열 A와 B가 주어지고, 문자열 Z가 문자열 A와 B에 공통되는 서브시퀀스일 경우, 문자열 Z를 공통 서브시퀀스(Common Subsequence)라 한다. 두 문자열 사이에는 문자열 Z와 같은 여러 개의 공통 서브시퀀스가 존재하는 것이 가능하며, LCS는 이러한 여러 개의 공통 서브시퀀스 중 가장 긴 서브시퀀스를 말한다.

[0041] 예를 들어, 입력스트링 X와 Y가 X = <A, B, C, B, D, A, B> 및 Y = <B, D, C, A, B, A>와 같은 스트링을 가진다고 가정한다. 입력스트링 X와 Y의 LCS 결과는 <B,C,B,A>로 나타나며, 입력스트링 Y와 X의 LCS 결과는 <B,A,D,B>로 나타난다. 이는 서로 다른 크기의 LCS 테이블을 생성하고 다른 경로를 통해 LCS의 결과가 얻어지기 때문이다. 또한, 입력스트링 X와 Y로부터 추출 가능한 LCS는 <B,C,B,A>, <B,C,A,B>, <B,D,A,B>으로 다양한 경우의 수로 나타날 수 있다. 이러한 LCS를 찾는 알고리즘은 브루트 알고리즘, 재귀적 알고리즘 및 동적 프로그램법 등이 있다.

[0042] 도 8은 본 실시예에 의한 시그니쳐 추출방법을 나타내는 순서도이다.

[0043] 도 8을 참조하면, 플로우 데이터 그룹핑부(13) 결과의 각 디렉토리를 입력받는다(S710). 그런 다음, 입력받은 디렉토리내의 플로우 데이터의 페이로드 데이터를 읽는다(S720). 플로우 데이터의 페이로드 데이터에 LCS를 적용하기 위해 페이로드 데이터를 x축 및 y축에 나열하고 매트릭스를 생성하고 LCS 알고리즘으로 추출가능한 모든 경로를 매트릭스에 표현한다(S730). 그런 다음, 표현된 추출가능한 모든 경로를 바탕으로 모든 가능한 시그니쳐 후보군을 추출한다(S740). 시그니쳐 후보군은 길이가 가장 긴 서브 시퀀스가 존재하는 시그니쳐를 추출한다. 이러한 시그니쳐는 하나일 수 있지만, 그 이상이 존재할 수 있다. 길이가 가장 긴 서브 시퀀스가 존재하는 시그니쳐가 두 개 이상인 경우에는 패킷의 페이로드 시작 부분을 포함하는 시그니쳐를 추출한다.

[0044] 그런 다음, 모든 가능한 시그니쳐 후보군 중에서 최적의 시그니쳐를 추출한다(S750). 최적의 시그니쳐는 모든 가능한 시그니쳐 후보군 중에 시그니쳐를 구성하는 서브 시퀀스의 최소 길이를 2바이트로 제한하여 추출하거나 시그니쳐를 구성하는 서브 시퀀스의 최소 길이가 1바이트인 서브 시퀀스는 제거할 수 있다. 그런 다음, 시그니쳐를 구성하는 서브 시퀀스 중 공통인 서브 시퀀스를 제거하여 획득할 수 있다.

[0045] 그런 다음, 플로우 데이터의 최적의 추출된 시그니쳐를 저장한다(S760). 디렉토리 내에 플로우 데이터의 존재유무를 확인한다(S770). S770에서 디렉토리 내에 플로우 데이터가 있다면 다시 디렉토리내의 플로우 데이터의 페이로드 데이터를 읽는다(S720). 디렉토리 내에 플로우 데이터가 없다면 해당 디렉토리내의 플로우 데이터에 대한 시그니쳐를 저장한다(S780).

[0046] 본 실시예에서 사용되는 '~부'라는 용어는 소프트웨어 또는 FPGA(field-programmable gate array) 또는 ASIC과 같은 하드웨어 구성요소를 의미하며, '~부'는 어떤 역할들을 수행한다. 그렇지만 '~부'는 소프트웨어 또는 하드웨어에 한정되는 의미는 아니다. '~부'는 어드레싱할 수 있는 저장 매체에 있도록 구성될 수도 있고 하나 또는 그 이상의 프로세서들을 재생시키도록 구성될 수도 있다. 따라서, 일 예로서 '~부'는 소프트웨어 구성요소들, 객체지향 소프트웨어 구성요소들, 클래스 구성요소들 및 태스크 구성요소들과 같은 구성요소들과, 프로세스들, 함수들, 속성들, 프로시저들, 서브루틴들, 프로그램 코드의 세그먼트들, 드라이버들, 펌웨어, 마이크로코드, 회로, 데이터,데이터베이스, 데이터 구조들, 테이블들, 어레이들, 및 변수들을 포함한다. 구성요소들과 '~부'들 안에서 제공되는 기능은 더 작은 수의 구성요소들 및 '~부'들로 결합되거나 추가적인 구성요소들과 '~부'들로 더 분리될 수 있다. 뿐만 아니라, 구성요소들 및 '~부'들은 디바이스 또는 보안 멀티미디어카드 내의 하나 또는 그 이상의 CPU들을 재생시키도록 구현될 수도 있다.

[0047] 이상 본 발명에 대하여 실시예를 참조하여 설명하였지만, 해당 기술 분야의 통상의 지식을 가진 자는 본 발명의 기술적 사상 및 영역으로부터 벗어나지 않는 범위 내에서 본 발명을 다양하게 수정 및 변경시켜 실시할 수 있음을 이해할 수 있을 것이다. 따라서 상술한 실시예에 한정되지 않고, 본 발명은 이하의 특허청구범위의 범위 내의 모든 실시예들을 포함한다고 할 것이다.

도면의 간단한 설명

[0048] 도 1은 본 실시예에 의한 시그니쳐 생성장치의 구조를 나타내는 도면이다.

[0049] 도 2는 본 실시예에 의한 플로우 데이터 수집 방법을 나타내는 순서도이다.

[0050] 도 3은 본 실시예에 의한 플로우 데이터 그룹핑 방법을 나타내는 순서도이다.

[0051] 도 4는 2개의 ID로 메신저를 이용해 채팅 초기의 패킷 크기 분포를 나타내는 그래프이다.

[0052] 도 5는 2개의 ID로 메신저를 이용해 로그인 초기의 패킷 크기 분포를 나타내는 그래프이다.

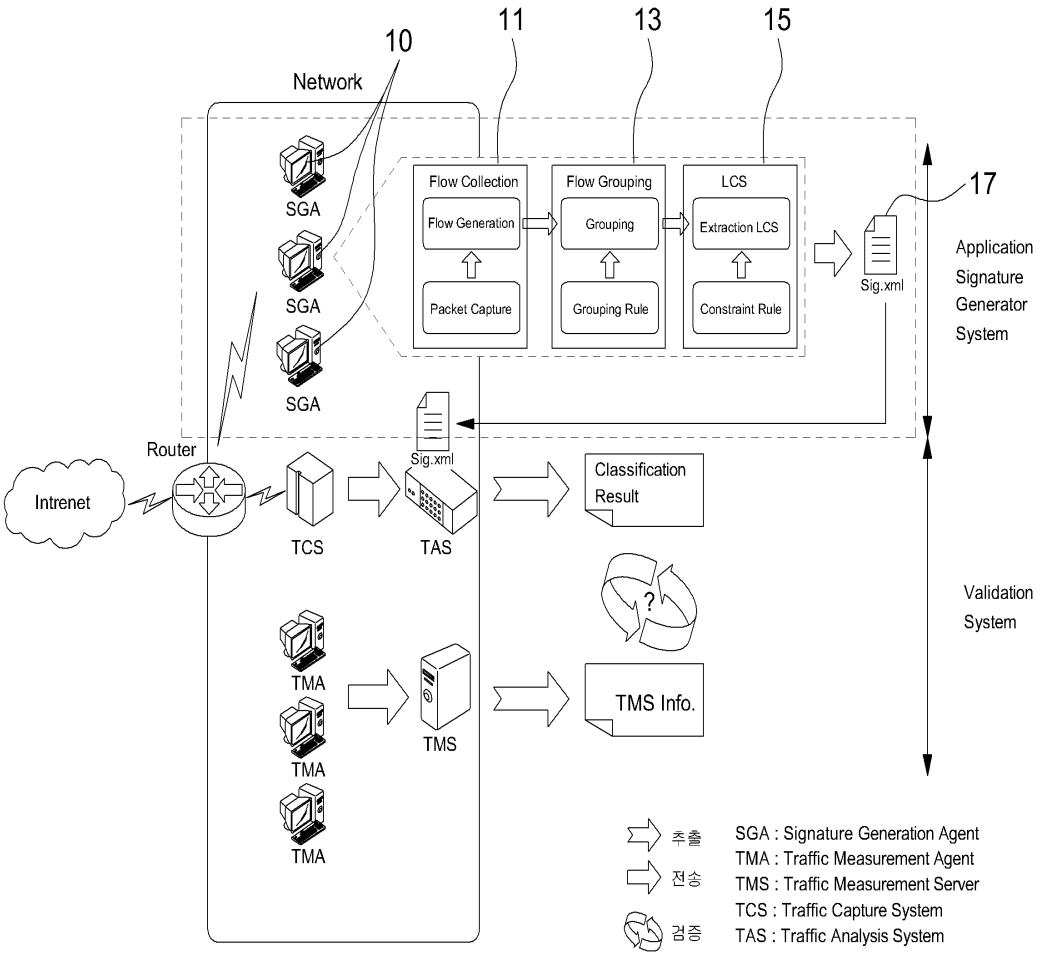
[0053] 도 6은 2개의 ID로 메신저를 이용해 파일전송 초기의 패킷 크기 분포를 나타내는 그래프이다.

[0054] 도 7은 2개의 ID로 메신저를 이용해 원격접속 초기의 패킷 크기 분포를 나타내는 그래프이다.

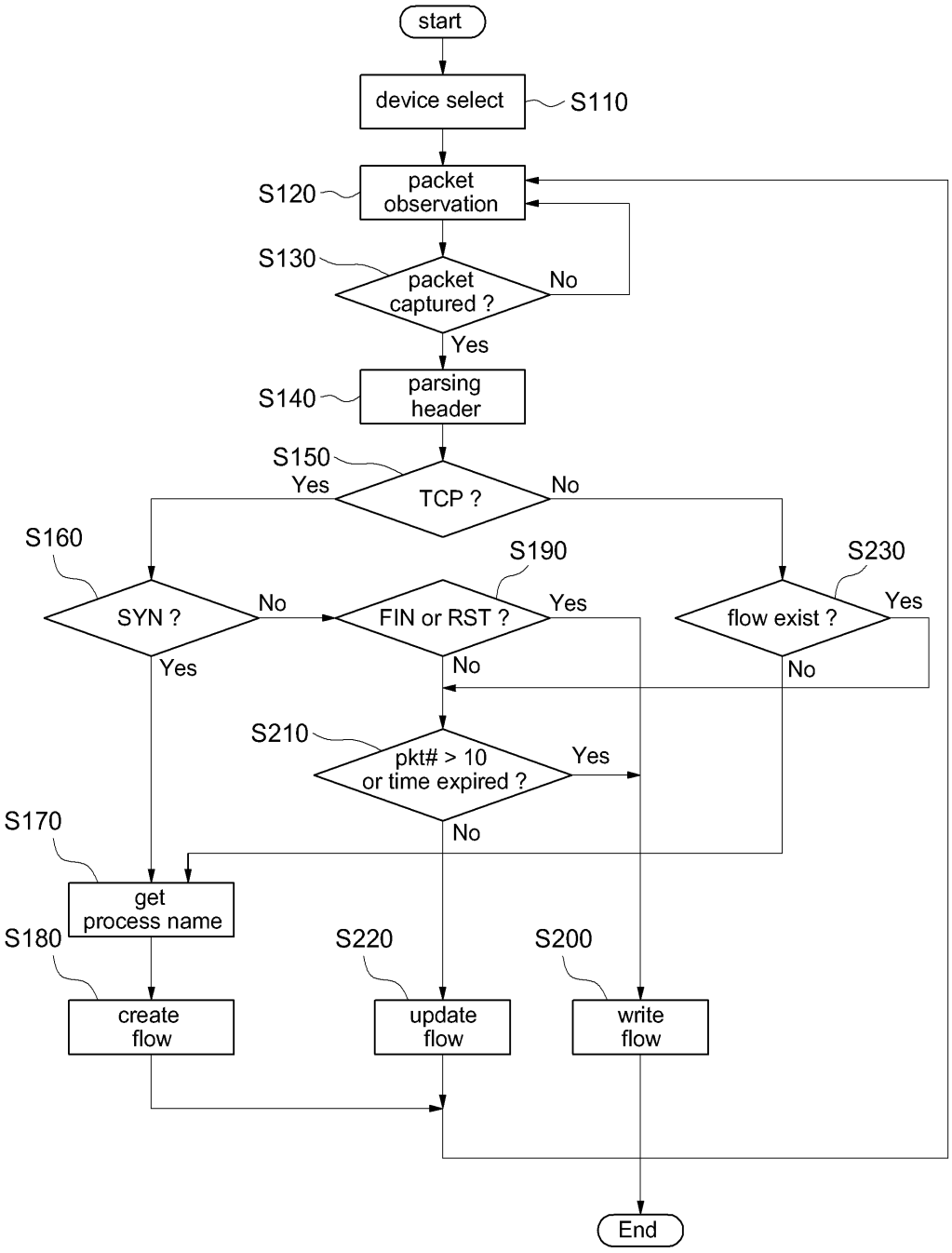
[0055] 도 8은 본 실시예에 의한 시그니쳐 추출방법을 나타내는 순서도이다.

도면

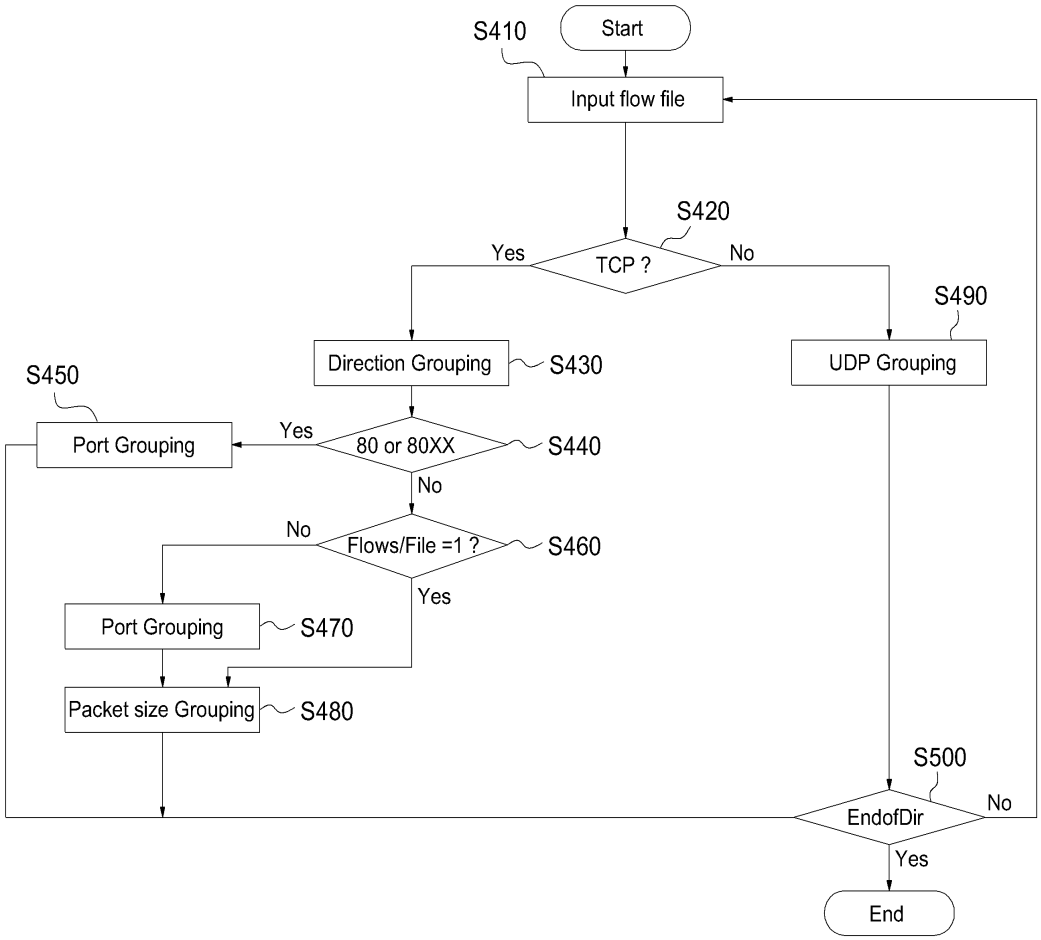
도면1



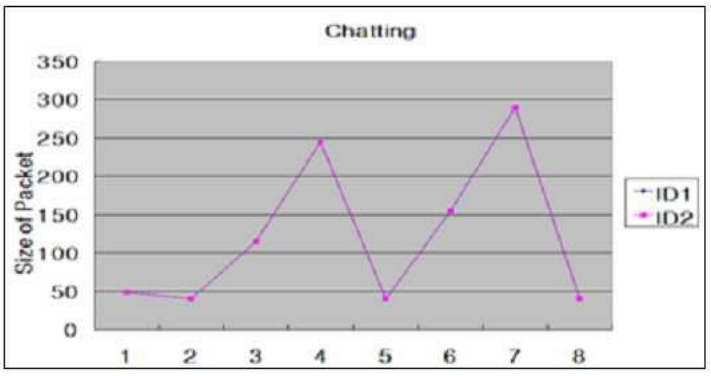
도면2


start
device select
S110
S120
packet observation
S130
packet captured ?
No
Yes
S140
parsing header
S150
Yes
TCP ?
No
S160
SYN ?
No
Yes
S190
FIN or RST ?
Yes
No
S230
flow exist ?
Yes
No
S210
pkt# > 10 or time expired ?
Yes
No
S170
get process name
S180
create flow
S220
update flow
S200
write flow
End

도면3



도면4



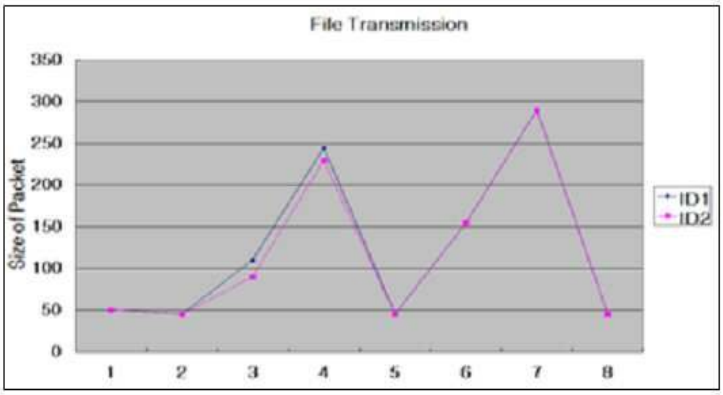
채팅 초기 패킷 크기 분포

도면5



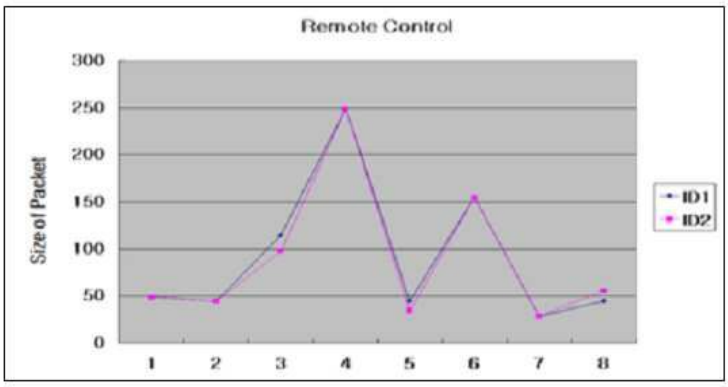
로그인 초기 패킷 크기 분포

도면6



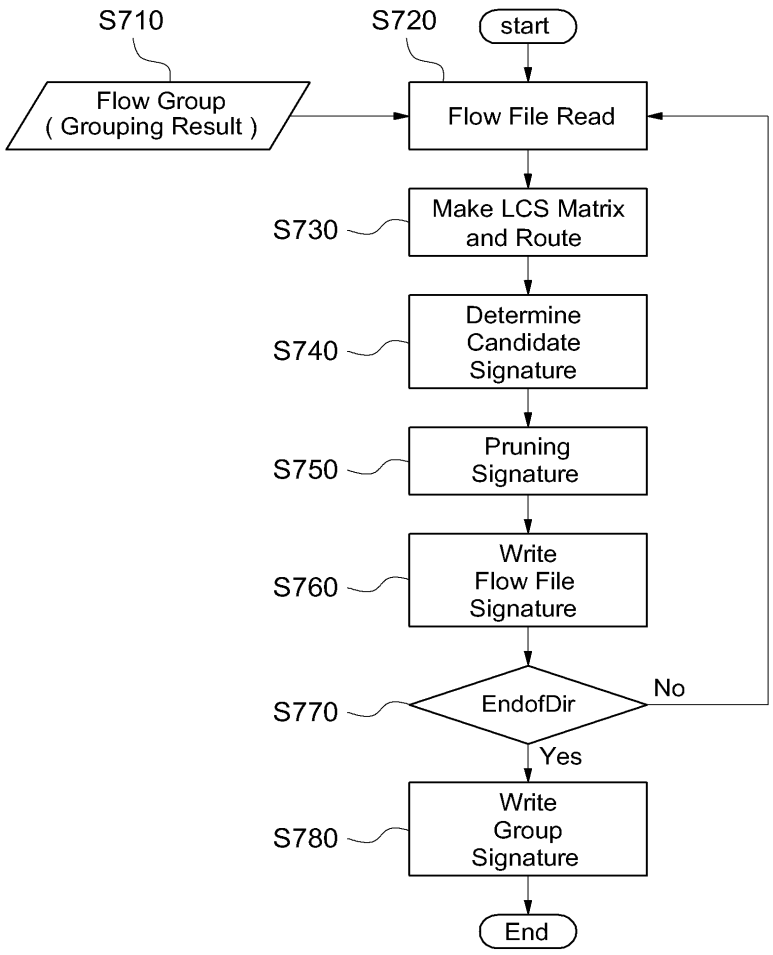
파일전송 초기 패킷 크기 분포

도면7



원격 접속 초기 패킷 크기 분포

도면8


S710
S720
start
Flow Group
(Grouping Result)
Flow File Read
S730
Make LCS Matrix
and Route
S740
Determine
Candidate
Signature
S750
Pruning
Signature
S760
Write
Flow File
Signature
S770
EndofDir
No
Yes
S780
Write
Group
Signature
End