(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 845 468 A1

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
17.10.2007 Bulletin 2007/42

(51) Int Cl.:
G06F 21/00 *(2006.01)* ***H04L 29/06*** *(2006.01)*

(21) Application number: **06007693.2**

(22) Date of filing: **12.04.2006**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IS IT LI LT LU LV MC NL PL PT RO SE SI SK TR
Designated Extension States:
AL BA HR MK YU

(71) Applicants:
- **Deutsche Telekom AG**
  53113 Bonn (DE)
- **Technische Universität Berlin**
  10623 Berlin (DE)

(72) Inventors:
- **Karrer, Roger, Dr.**
  10555 Berlin (DE)
- **Kühn, Ulrich, Dr.**
  10110 Berlin (DE)

(74) Representative: **Blumbach - Zinngrebe**
Patentanwälte
Alexandrastrasse 5
65187 Wiesbaden (DE)

(54) **Method, computer network system and gate for identifying and controlling unsolicited traffic**

(57) It is a major aspect of the present invention to identify and control unsolicited traffic, in particular by limiting or entirely blocking unsolicited traffic in an access network (107) or in a core network (104), e. g. the Internet.

This aim is achieved by at least one gate (109), wherein each gate is located between an access network (105, 107) and the core network (109). The gate (109) operates to identify and control unsolicited traffic received from malicious sources (102-1 to 102-N) and to forward permissions or capabilities to sources (101-1 to 101-N; 102-1 to 102-N) wishing to send data by using an authentication authority (113). These functions performed by the gate (109) and the authentication authority (113) are also called edge-based capability (EC) functions. Each source, e. g. a user, a web client or a host computer wanting to send data to a destination (103-1) must receive a permission to send prior to sending. This permission is used to compute a mark for every packet so that the gate (109) is able to distinguishing authenticated (111) from unsolicited (112) traffic and thus to allow only authenticated traffic to pass the gate (109).

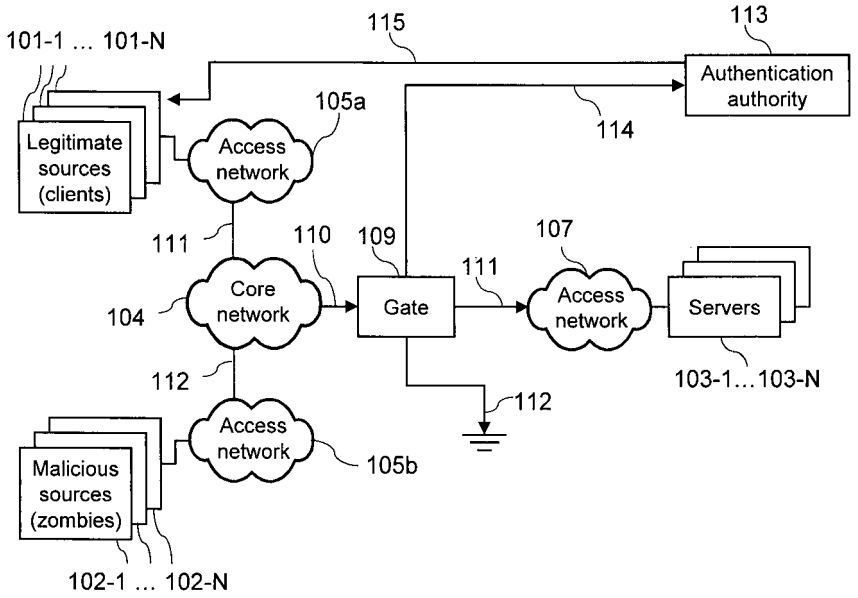

Fig. 2

EP 1 845 468 A1

Description

Technical Field

[0001] The present invention relates to a method for identifying and controlling unsolicited traffic in a computer network system which comprises access networks for connecting sources and destinations, respectively, to a core network, like the Internet. The present invention further relates to a computer network system and a gate adapted to perform a method for identifying and controlling unsolicited traffic.

Backround of the Invention

[0002] The open design of the Internet has been a key factor for its growth and success. However, the ability to send anything to anybody at any time has shown to have severe drawbacks. The lack of authentication and security is increasingly exploited to inject unsolicited traffic into the network, leading to uncontrolled virus and worm spread and devastating Distributed Denial-of-Service (DDoS) attacks.

[0003] Viruses and worms have been shown to clog entire parts of the Internet by causing extensive traffic. Moreover, they can carry code that modifies or destroys infected end systems up to a complete loss of data or steal private data.

[0004] DDoS attacks aim at overloading a resource such that legitimate users are denied access to the resource. Well-known DDoS targets include, but are not limited to, Web servers and Internet Data Centers.

[0005] The damage caused by unsolicited traffic has rapidly increased in the past years and its damage exceeds today billions of dollars.

[0006] There already exist approaches which aim to protect end systems or end devices against unsolicited traffic such as Denial of Service Attacks, worms or virus.

[0007] The present invention aims at protecting network resources. In particular, access networks that have a significantly lower capacity than the core must be protected against unsolicited traffic. Protection includes the prevention of flooding attacks with unsolicited traffic (DDoS), such that legitimate traffic is no longer able to reach its destination due to exceeded network capacity, as well as the early prevention of unsolicited (malicious or infectious) traffic spreading into the core to infect other systems.

[0008] One known approach is based on statistical methods as, for example, disclosed by Y. Kim et al. in "Packetscore: statistics-based overload control against distributed denial-of-service attacks", Proceedings of IEEE INFOCOM'04, Hong Kong, March 2004.

[0009] Statistical methods require no user interaction that might annoy the user. However, the increasing sophistication of attacks, such as mimicking attacks as flash crowds or attack with specific traffic patterns, makes it significantly more difficult to address unsolicited traffic by statistical methods.

[0010] Further approaches are disclosed by S. Savage et al. in "Network support for IP-traceback", IEEE/ACM Transactions on Networking, 9(3):226-237, June 2001 and A. Yaar et al. "A stateless internet flow filter to mitigate DDoS flooding attacks; Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, May 2004. In these approaches all or at least a large number of routers along a communication path within a network participate in the defending end systems against DDoS attacks. In practice, such solutions are slow to be deployed as they require changes by multiple providers or even support from routers in the network.

[0011] A still further solution is disclosed by X. Yang et al. in "A DoS-limiting network architecture", ACM SIGCOMM'05, Philadelphia, PA, August 2005. This solution requires a collaboration of multiple entities at the edge of an access network and a core network. Again, a deployment is slow and difficult to achieve in practice as it is difficult to imagine that multiple Internet Service providers (ISPs) exchange information, in particular regarding security-related issues.

[0012] A further approach to protect end devices against Denial-of-Service attacks is known wherein a source wanting to send data to a destination must first receive a permission to communicate prior to sending data. This permission to communicate is also termed capability. See for example the publication of T. Anderson et al. "Preventing Internet Denial-of-Service with capabilities", Proceedings of Hot Topics in Networking (HotNetsII), Boston, October 2003. The capabilities are used to authenticate sources on an end-to-end basis. This means, that the capabilities are issued by a destination after checking the legitimacy of a source, and thus traffic is checked for capabilities by the destination to protect end system resources. However, such an end-to-end scheme does not protect the access networks and the core network from unsolicited traffic. Therefore, DDoS attacks on an access network or malware spread into the core network are not addressed.

[0013] Furthermore, EP 1 284 558 B1 discloses a method and apparatus for limiting the effect upon certain clients that access an electronic merchant's server from a denial of service attack against the electronic merchant's server. Therefore, the e-merchant designates some clients as VIP clients, wherein a packet addressed to the e-merchant's server is marked for forwarding in a privileged class of services, if the source of the packet is a VIP client.

Intuition Machines Exhibit 1030, Page 2 of 22

Summary of the Invention

[0014] It is an object of the present invention to provide a method, a computer network system and a gate which are capable of preventing unsolicited traffic from spreading into the core network or preventing unsolicited traffic from overloading a destination's access network.

[0015] It is a major aspect of the invention to identify and control unsolicited traffic, in particular by limiting or entirely blocking unsolicited traffic in an access network or the core network, e. g. the Internet. In particular, the present invention relates to a computer network system and a method using authentication or permissions, also termed capabilities, to

a) thwart Distributed Denial of Service (DDoS) attacks against a destination's access network at the edge of a core network and thus by implication on servers, and
b) stop worm and virus spread prior to reaching the core network.

[0016] This aim is achieved by at least one gate, wherein each gate is located between an access network and the core network. The gate operates to identify and control unsolicited traffic received from malicious sources and to forward permissions or capabilities to sources wishing to send data by using an authentication authority. These functions performed by the gate and the authentication authority are also called edge-based capability (EC) functions due to the fact that the gate is located at an edge of the core network. Every source, e. g. a user, a web client or a host computer wanting to send data to a destination must receive a permission to send (a capability) prior to sending. This permission is used to generate and include a tag in every packet so that the EC gate is able to distinguish authenticated from unsolicited traffic and thus to allow only authenticated traffic to pass the gate.

[0017] Therefore, the present invention based on edge-based capabilities protects the core network and/or a destinations' access network, whereas the above mentioned known schemes protect end systems or destination resources.

[0018] According to one aspect of the invention, the authentication authority uses state-of-the-art authentication techniques, such as CAPTCHAS as described by L. v. Ahn et al. in "CAPTCHA: Using Hard AI Problems for Security"; Advances in Cryptology-EUROCRYPT 2003, volume 2656 of Lecture Notes in Computer Science, pages 294-311, Springer Verlag and/or IP Puzzles as described by D. Dean et al. in "Using client puzzles to protect TLS", Proceedings of the 10th USENIX Security Symposium, Washington, D.C., August 2001 to transmit permission information to the sources. The sources, i.e. users and/or computers try to obtain the valid tagging key from the permission information, used for tagging packets to be sent to a destination.

[0019] It is to be noted, that in a preferred embodiment the authentication information is shared among three distributed components, i.e. sources, the gates and the authentication authority in a secure, efficient and scalable way.

[0020] It is a further aspect of the invention that the computer network system can be implemented readily, e.g. by a single Internet Service Provider (ISP). Owners of end systems connected to the ISP's network have the incentive to install the respective components of the present invention to profit from the protection against unsolicited traffic provided by the gate.

[0021] The above mentioned technical problem is solved by a a method for identifying and controlling unsolicited traffic in a computer network system. The computer network system comprises at least a first access network to which at least one source is connected, at least a second access network to which at least one destination is connected and a core network connecting the access networks. The method comprises the following steps:

- directing traffic units addressed to a destination from the at least one source to a gate which is interconnected between the core network and the second access network;
- verifying each traffic unit at the gate to determine whether or not the traffic unit has been marked validly;
- identifying each traffic unit as unsolicited if it has been marked invalidly or if it has not been marked at all;
- controlling the flow of each traffic unit having been identified as unsolicited in response to at least one predetermined system parameter.

[0022] This method helps to thwart Distributed Denial of Service (DDoS) attacks against the second access network at the edge of a core network and thus by implication on servers.

[0023] The technical problem is further solved by a method which comprises the steps:

- directing traffic units addressed to a destination from the at least one source to a gate which is interconnected between the core network and the first access network;
- verifying each traffic unit at the gate to determine whether or not the traffic unit has been marked validly;
- identifying each traffic unit as unsolicited if it has been marked invalidly or if it has not been marked at all;
- controlling flow of each traffic unit having been identified as unsolicited in response to at least one predetermined system parameter.

Intuition Machines Exhibit 1030, Page 3 of 22

[0024] This method aims to stop worm and virus spread prior to reaching the core network.

[0025] Preferred embodiments are set forth in the dependent claims.

[0026] If the Internet is used as the core network traffic units are IP packets.

[0027] The step of controlling the flow of each traffic unit having been identified as unsolicited may include blocking, delaying and/or limiting the flow of each traffic unit having been identified as unsolicited to a destination for example in dependence of the congestion condition of the core network and/or the second access network.

[0028] In order to improve the network based flow control of traffic units having been identified as unsolicited a permission message is generated and transmitted to the at least one source. Traffic units to be sent are then marked at the source using information from the permission message. It is to be noted, that the permission message comprises as an information a tagging key, in the description also called session keys, to mark traffic units. To allow the gate to verify traffic units information relating to the permission message is stored at the gate. This information might be the tagging key or information to derive a tagging key.

[0029] In a preferred embodiment, the permission message is transmitted in form of a challenge to be solved by the at least one source to obtain the tagging key. Using challenges only computers working properly are able to obtain a valid tagging key from the permission message.

[0030] The challenge is to be designed such that it can distinguish legitimate users or end systems from other sources. In a preferred embodiment that allows a distinction between human users and remotely controlled machines (zombies, bot-nets) a challenge is constructed such that it can be solved only with an interaction of a human user.

[0031] A CAPTCHA algorithm and/or an IP puzzle algorithm as mentioned above can be used to generate a challenge to transmit the permission message to the at least one source. In this case cryptographic keys are used for tagging traffic units.

[0032] In a preferred embodiment the tagging key is used in a message authentication code (MAC) to compute a tag to mark traffic units. The tagging key or information to derive the tagging key is stored at the gate to verify traffic units received from sources.

[0033] In a preferred embodiment, the tagging key is a cryptographic key that is used by the source to generate tags for marking traffic units and that is used by the gate to verify tags included in traffic units.

[0034] Preferably, the permission message is generated by the gate and forwarded to an authentication means which in turn transmits the permission message to the at least one source.

[0035] Alternatively, the permission message is generated by an authentication means which transmits the permission message to the at least one source and the gate.

[0036] The above mentioned problem is also solved by a computer network system for identifying and controlling unsolicited traffic. The system comprises a core network, at least a first access network, at least one source connected to said first access network, at least a second access network, at least one destination connected to said second access network, and at least one gate located between the first access network and the core network and/or between the second access network and the core network. Furthermore, means for generating and transmitting a permission message to the at least one source are provided, wherein the at least one source comprises means for obtaining a tagging key from that permission message used to mark traffic units addressed to a destination. The gate comprises storage means for storing information related to a permission message, means for verifying received traffic units on the basis of the information stored to determine whether or not the traffic units are marked validly, means for identifying each traffic unit being marked invalidly as unsolicited, and means for controlling the flow of each traffic unit being marked invalidly in response to at least one predetermined system parameter.

[0037] It is to be noted that the feature "traffic unit being marked invalidly" also covers traffic units which are not marked at all.

[0038] In a preferred embodiment, each source comprises a client computer, each destination comprises a server and the core network is the Internet.

[0039] In order to facilitate authentication of sources, the means for generating and transmitting a permission message to the at least one source is implemented in an authentication unit which can be located in a separate device connected to the gate or in at least one destination.

[0040] In a preferred embodiment the authentication unit is adapted to generated a permission message in form of a challenge to be solved by the at least one source, wherein only sources having solved the challenge are able to mark traffic units validly.

[0041] The above mentioned technical problem is further solved by a gate used in a computer network system according to any one of the claims 14 to 17. The gate comprises storage means for storing information related to a permission message, means for verifying traffic units on the basis of the information stored to determine whether or not the traffic units are marked validly, means for identifying each traffic unit being marked invalidly as unsolicited, and means for controlling the flow of each traffic unit being marked invalidly in response to at least one predetermined system parameter. It is to be noted that the feature "traffic unit being marked invalidly" also covers traffic units which are not marked at all.

Intuition Machines Exhibit 1030, Page 4 of 22

Brief Description of the Drawing:

[0042]

Fig. 1 is a principle block diagram of a communication system showing the basic concept of the present invention;
Fig. 2 is a block diagram of a communication system adapted to thwart DoS attacks against a destination's access network;
Fig. 3 is a block diagram of a communication system adapted to stop worm and virus spread prior to reach the core network;
Fig. 4 is a block diagram showing the function modules associated to the respective components of the communication systems as depicted in Fig. 2 and 3;
Fig. 5 is a flowchart showing the steps performed at the gate.

Detailed Description

[0043] Fig. 1 shows a block diagram of a preferred communication system 10 based on a commonly used distributed networked system including an aspect of the present invention. The communication system 10 comprises end systems or end devices and a network. End systems are separated into sources 101 and destinations 103, wherein only one source and one destination are shown. The network is divided into a core network 104, e. g. the Internet, and access networks. In order to facilitate the illustration only a single source-side access network 105 and a single destination-side network 107 are shown. The conjunction of an access network and the core network is called edge. Thus, we distinguish between a source-side edge 106 and a destination-side edge 108. Each edge usually comprises one or multiple hardware components, such as routers. A common characteristic of the edges is that the network capacity is significantly different between an access network and the core network, i.e., the network capacity is one or several orders of magnitude smaller in the access networks 105, 108 than in the core network 104. Therefore, an access network is significantly more vulnerable to traffic flooding than the core network 104. Therefore, unsolicited traffic must be identified and controlled, e.g. limited or completely blocked, at an edge. For this purpose, a specific network element termed gate 109 is added to the network infrastructure at least at one edge, for example at the source side and/or the destination side that ensures that the network resources are available for legitimate traffic.

[0044] Fig. 2 shows a first preferred embodiment of a communication system as generally depicted in Fig. 1. This communication system comprises a gate 109 at a destination side edge of a core network 104, i. e. between an destination-side access network 107 and the core network 104. The destination-side access network 107 serves to connect destinations, e. g. Web servers 103-1 to 103-N to the core network 104. It is to be understood, that basically a plurality of destination-side access networks are connected to the core network 104. The gate 109 operates to protect the destination-side access network 107 against DDoS flooding attacks.

[0045] In this embodiment, legitimate sources 101-1 to 101-N, such as Web clients are connected to the core network 104 via a source-side access network 105a, whereas malicious sources (zombies) 102-1 to 102-N are connected to the core network 104 via a separate access network 105b. In Fig. 2, wanted traffic generated by the legitimate sources 101-1 to 101-N is designated by reference number 111, whereas unsolicited traffic generated by the malicious sources 102-1 to 102-N is designated by reference numbers 112. It will be described later how clients are identified and distinguished as legitimate sources and malicious sources, respectively.

[0046] The attack or unsolicited traffic 112 from the malicious sources 102 to 102-N sent to one or multiple destinations 103-1 to 103-N converges at the edge joining the core network 104 with the destination-side access network 107. Without gate 109, the access network 107 with its lower capacity can be flooded with unsolicited traffic 112.

[0047] In the communication system, whenever a source wants to communicate with a destination, it must first receive a permission to communicate. This permission to communicate is also termed capability in literature. The permission 115 is generated and forwarded to one or more sources by a specific software module called authentification authority 113. The authentification authority 113 might be implemented in one or more destinations 103 to 103-N or in a separate device associated to the gate 109. By generating and forwarding permissions to sources wishing to communicate with destinations the sources become implicitly authenticated, i. e. legitimate sources, only if they are able to obtain a valid tagging key from the received permission. This procedure will be explained below in detail. With other words, capabilities or permissions are used to authenticate sources implicitly. A legitimate source uses the tagging key included in the permission to tag packets to be sent to a destination.

[0048] With gate 109, the attack traffic 112 is identified and can be filtered out such that only a controllable number of legitimate packets 111 are forwarded from the core network 104 to the access network 107. The unsolicited traffic 112 filtered out by the gate 109 is symbolically depicted by a ground symbol.

[0049] Additionally or alternatively, the amount of attack traffic 112 flowing through the access network 107 can be controlled by the gate 109 in dependence of the condition of the access network 107. The gate 109 identifies traffic as

Intuition Machines Exhibit 1030, Page 5 of 22

unsolicited traffic if the received packets do not contain a valid tag generated on the basis of the permission.

[0050] Furthermore, as will be described later, the gate 109 uses a key generation algorithm to generate tagging keys. The tagging keys are transported to the authentication authority 113 which forwards the permission and the tagging key or portion of the key to the sources. Only sources which work correctly are able to calculate the tagging key from the permission which is then used to tag or mark packets to be sent to respective destinations.

[0051] Fig. 3 shows a second embodiment of a communication system. This communication system differs from that shown in Fig. 2 in that a gate 109 is arranged at a source side edge of a core network 104, i.e. between a source-side access network 105 and the core network 104. Assume that malware, e. g. a virus or a worm has infected a set of sources 102-1 to 102-N, for example computers and replicates itself with horrendous speed to infect other source computers. These source computers 102-1 to 102-N will fail to obtain a correct permission generated by and received from an authentication authority 115. Assume further, that the malware 112 spreads via email. Gate 109 is adapted to automatically determine whether an email 111 has been sent by a legitimate source 101-1 to 101-N or by a malicious computer 102-1 to 102-N. Only emails 111 sent by a legitimate source 101 to 101-N have been marked correctly and will pass the gate 109. Therefore, the malware which are not marked or carrying an invalid tag can be identified as unsolicited traffic 112 and prevented from spreading past the gate 109 into the Internet 104.

[0052] The internal operation of the gate 109 as shown in Fig. 2 and 3 is depicted in Fig. 5. The gate 109 has the ability to identify and control, e.g. filter out or limit unsolicited traffic. When a packet 110 arrives from sources 101 or 102 at the gate 109, the gate 109 first checks in step 123 whether the packet contains a tag. If the packet contains a tag, then the gate 109 retrieves in step 120 a tagging key previously generated for a given session. If a suitable tagging key is available, the gate 109 uses the tagging key to determine in step 124 whether or not the packet includes a valid tag. If the tag is valid, the packet 111 can be forwarded immediately to an output queue of the gate 109.

[0053] If, however, a tag is not included in a packet 118 or not valid 122 or a tagging key is not available 121 in the gate 109, the packet is considered as an unsolicited or a malicious packet 112 and thus will be filtered or controlled 125 by the gate 109. The term "filtered" or "controlled" 125 can be interpreted and implemented in various ways. Examples include, but are not limited, to simply drop unsolicited packets 112 or to forward them to a low-priority queue 125. An efficient solution is required such that the gate 109 does not slow down the traffic in the destination's access network 107 or in the core network 104. Furthermore, the solution needs to scale to a large number of flows.

[0054] As a brief overview, the capability exchange for legitimating sources to send wanted traffic 111 involves three independent components: sources which must acquire a capability, the authentication authority 113 to issue and forward capabilities and the gate 109 to check capabilities in order to identify and control unsolicited traffic. The general concept is shown in Fig. 4.

[0055] For this purpose, the gate 109 uses a tagging scheme for verifying packets and a keying scheme for generating tagging keys. The keying scheme can be implemented in key generation means 114 and the tagging scheme can be located in a tag verification means 117, both located within the gate 109.

[0056] Furthermore, a key transport scheme is used by the authentication authority 113. The key transport scheme can be implemented in a key transport means 118. In a preferred embodiment both the authentication authority 113 as well as the key transport scheme are implemented in one or more of the destinations 103-1 to 103-N. In addition, a tagging scheme is used by the sources to generate tags and embed them into packets to be sent. Therefore, tag generating means 116 can be implemented in each source to perform the tagging scheme. A detailed description of these schemes will be given below.

[0057] In a first step, a tagging key is generated by the key generation means 114 of the gate 109. Derivations of this key are passed to the authentication authority 113. The authentication authority 113 is a logical unit or software module that may be mapped onto an independent device, or may be collocated with other devices, such as the server 103-1. The authentication authority 113 is adapted to implicitly authenticate clients 101. This is done by generating a permission or a capability, for example in form of a challenge which includes a tagging key or a portion of a tagging key that is valid for a single session. The challenge is forwarded to the sources 101 and 102. Each source tries to solve the received challenge for obtaining a valid tag which is used to mark packets to be sent to destinations. In the gate 109 packets marked with a valid tag are considered as a legitimate traffic 111 whereas traffic without a tag or with an invalid tag are identified as unsolicited traffic 112 and thus filtered respectively, as shown in Fig. 2, 3 and 5.

[0058] An illustrative operation of the computer network system as shown in Fig. 2 will be described in more detail.

[0059] This computer network system is adapted to prevent the access network 107 from DDoS attacks, but as mentioned above, it can be used for other scenarios as well.

[0060] It is assumed, that the destinations 103-1 to 103-N are servers, that the sources 101-1 to 102-N are client computers and that the authentication authority 113 is located in at least one of the servers 103.

[0061] In a preferred embodiment, the Keying Scheme built into the key generation means 114 is responsible for generating a key at the gate 109 and communicating it to one or more of the servers 103-1 to 103-N or, in general, the authentication authority 113. It is important to realize that the gate 109 defines and manages keys and exchanges them with the servers, but not vice versa. The key sent to the server 103-1 is server-specific. The server 103-1 stores its key

Intuition Machines Exhibit 1030, Page 6 of 22

locally. Once the server 103-1 receives a request from a client, e. g. client 101-1, it derives a session-specific key as a tagging key to be sent to the sources.

[0062] The Key Transport Scheme associated to the authentication authority and implemented for example in the server 103-1 is responsible for communicating the session or tagging key to clients, e.g. in form of a challenge generated by the authentication authority 113. When the user of the client 101-1 and/or the client itself solves this challenge, the client 101-1, the user uses, is considered as authenticated.

[0063] In the present embodiment, the clients 101-1 to 101-N are assumed to be able to solve the received challenge correctly and thus are legitimated or authenticated. It is assumed that the clients 102-1 to 102-N are not able to solve the received challenge correctly and thus are considered as zombies.

[0064] The Tagging Scheme implemented in the clients allow the legitimate clients to embed a tag in each packet it send, after being authenticated and after obtaining a tagging key by solving the challenge correctly.

[0065] At gate 109, the Tagging Scheme defines how the gate 109 verifies the tags included in the packets with information stored at the gate 109.

[0066] In the following, we describe the three schemes in detail. For the description we introduce the following notation: *Notation 1:* For an IP packet *P* we use $IP_s(P)$, $IP_d(P)$, Ident(P), Flow(P), Len(P) to denote the respective header fields source address, destination address, identification (in IPv4), flow label (in IPv6), and payload length of the packet. Likewise, we use $Port_s(P)$, $Port_d(P)$, Seq(*P*), Ack(*P*) to denote the source port, destination port, sequence and acknowledge number in case that P is a TCP packet.

[0067] *Notation 2:* For a bitstring *X* let $[X]_r$ denote the leftmost r bits of X; if X has less than r bits, it is filled up by zero bits. Let $[X]_{r,s}$ with $r \leq s$ denote the sub-bitstring consisting of the bits *r,...,s,* filled up by zeroes if necessary.

1. Keying Scheme

[0068] In the following two schemes are described for tagging key generation and management for the gate, servers, and clients. The first scheme, which is called *stateless keying,* derives all tagging keys from a central master key held at the gate 109. In the second scheme the gate 109 needs to keep some state for each flow routed through it.

a) Stateless Keying

[0069] In this keying scheme, the gate 109 has a central master or gate key k_G. As all other keys are derived from this key it is kept secret at all times. The scheme further uses two key derivation functions KDF_1, KDF_2 to derive server keys and session or client keys. Session keys are also termed tagging keys. Given a server S with address IP(*S*), its key is

$$k_S \leftarrow KDF_1(k_G, IP(S))$$

[0070] Furthermore, for a client C with address IP(C), the tagging key used to tag any packet sent to server S is

$$k_{C,S} \leftarrow KDF_2(k_S, IP(C), IP(S))$$

[0071] The interface to the tag verification means 117 in the gate 109 can be easily implemented using these functions. Note that the client does only receive $k_{c,s}$, so it does not need to know about KDF_1, KDF_2.

[0072] Adding a cache (not shown) in the gate 109 for the results of key derivation does not affect the property that the scheme is stateless. However, it does improve computational efficiency of the scheme. We use two LRU (least recently used) tables in the gate 109 holding entries (IP(*S*),k_s) and (IP(S), IP(C), $k_{c,s}$). That way we can amortize the computational complexity of the tagging key derivation over a significant number of tag verifications while the tables are easy to maintain. The achievable improvement in computation time depends on available memory at the gate 109 and the number of different flows passing the gate at any single time.

b) Keying with State at the Gate

[0073] Rather than deriving all the keys from a single master key kept at the gate 109, the gate 109 could issue a new tagging key for each single connection it encounters. In the following we describe how such an alternative keying scheme could work.

[0074] In this scheme the gate 109 generates a session or tagging key k whenever it sees a connection initiation

Intuition Machines Exhibit 1030, Page 7 of 22

involving a client C and a server 5 pass by, i.e. in form of a SYN packet. It then stores k along with (C, S) and other information identifying the flow, e.g. port numbers, for future use.

[0075] Then the gate 109 passes on the tagging key to the respective server in which the authentication authority 113 is implemented such that the server can transport it to the client. This requires that the session or tagging key *k* is encrypted with some key k_s shared between the gate 109 and the server 103-1. The communication between the gate 109 and the server 103-1 can be done in-band, i.e. by adding the encrypted key to the original SYN packet of the client 101-1 before passing it on to the server 103-1. To expire session keys the gate 109 has to watch the connection and know about protocol timeouts. This requires a similar state-keeping as in a stateful-inspection firewall.

c) Selection of Algorithms and Parameters

[0076] For the stateless keying scheme the key sizes as well as the key derivation functions need to be specified. Given that we select CBC-MAC with AES and a key size of 128 bits for the tagging scheme, it is reasonable also use these parameters for the keying scheme. Furthermore, the key derivation functions KDF_1, KDF_2 can be implemented using CBC-MAC, given that it is a good pseudorandom function. This function is described, for example, by M. Bellare et al. "The security of the cipher block chaining message authentication code.", Journal of Computer and Systems Sciences, 61(3):362-399, December 2000. This way no additional code needs to be added. Concretely,

$$KDF_1(k_G, IP(S) := CBC - MAC[AES]_{k_G}(c_1 \| IP(S)),$$

$$KDF_2(k_S, IP(C), IP(S) := CBC - MAC[AES]_{k_S}(c_2 \| IP(C) \| IP(S))$$

using two different constants c_1, c_2 for separating the input to the encryption function. We use constants of 12 or 8 octets each. The performance of these functions is then as follows: For IPv4 the IP adress(es) and constants result in inputs of 16 octets for KDF_1 resp. KDF_2, i.e. a single block cipher operation. For IPv6 the IP address(es) and constants result in inputs of 28 or 40 octets, resulting in KDF_1 resp. KDF_2 using 2 or 3 block cipher operations. Furthermore, a key scheduling operation is needed when putting the resulting key to use.

2. Key Transport Scheme

[0077] In this section we propose methods to transport the cryptographic key generated by the gate 109 from the server 103-1 or authentication authority 113 to the client 101-1 at the start of the respective connection. The client uses this key to tag each packet to be sent to a server.

[0078] We employ the general idea of a CAPTCHA test (Completely Automated Public Turing Test to tell Computers and Humans Apart) to distinguish between a human user and a machine or client. The CAPTCHA test is for example described by L.v.Ahn et al. in "CHAPTCHA: Using Hard AI Problems for Security", Advances in Cryptology-EUROCRYPT 2003, vol. 2656 of Lecture Notes in Computer Science, pages 294-311, Springer Verlag.

[0079] This way our scheme ensures that human users can still access a server even under attack. On the other hand, a malicious client, e.g. 102-1, that is potentially part of a DDoS attack network, and that can therefore not rely on help of a human user, is not able to get a valid tagging key.

[0080] The CAPTCHAs were originally designed as tests. In this embodiment, we use a subclass of those problems that allow the user to input some information derived from a given artificial intelligence (AI) problem. In particular, such an AI problem is called a challenge. The challenge is generated by the authentication authority 113 using a tagging key received by the gate 109. The challenge including the tagging key is then transmitted to the clients 101 and 102.

[0081] Then we use this input for a computation that will give a cryptographic key to the client computer.

[0082] One problem we face here is that probably only a minor fraction of potential user is willing to input the equivalent of a 128-bit tagging key, which is a tedious and error-prone task, e.g. reading the key from a distorted image. Therefore we must restrict ourselves to using only a small number of bits, resulting in a low-entropy input.

[0083] In the following we propose two solutions that combine the CAPTCHA with a variant of IP puzzles in a way that either the user helps the client machine in solving the puzzle faster, or allows the client machine to use the solution at all. IP puzzles are known and described for example by D. Dean et al. in "Using client puzzles to protect TLS", Proceedings of the 10th USENIX Security Symposium, Washington, D.C., August 2001.

[0084] In fact, in the first scheme the puzzle-solving is helped by the human user, while the second scheme requires a correct solution of the CAPTHCA by the user.

Intuition Machines Exhibit 1030, Page 8 of 22

[0085] In both schemes the client machine tries to obtain the key that is used by the server 103-1 or the authentication authority 113 to encrypt the tagging key.

[0086] Our proposals are scalable in the sense that a busy server 103-1 can select which effort, e. g. in computing cycles or waiting time a client computer has to invest in order to access the server.

[0087] The schemes make use of a preimage-resistant cryptographic hash function h (such as SHA-256) with

$$h: \{0,1\}^* \rightarrow \{0,1\}^n.$$

[0088] The hash function SHA-256 is described, for example, in National Institute of Standards and Technology (NIST), Computer Systems Laboratory. Secure Hash Standard. Federal Information Processing Standards Publication (FIPS PUB 180-2), February 2004.

[0089] We assume that h is either shared somehow between the client and the server or is otherwise public.

[0090] Furthermore it should be noted that due to the way the key transport schemes are integrated into the EC communication system, the schemes need to be realized as a single round, i.e. a single back-and-forth exchange of messages. The idea is to piggyback a client-to-server message, if existing, into an initial SYN packet, while the server-to-client message is integrated into the SYN-ACK packet. Preferably, we are using in-band communication.

[0091] With the two key transport schemes the system's designer or operator has a choice whether client machines shall be able to connect even though no human user stands essentially behind it. This choice might also be dependent on the current system and network load.

a) Scheme Helped by CAPTCHA Solution

[0092] In the first scheme, we are effectively realizing a differential work load for a client computer, i.e. a client computer and a human user working together have a vastly faster access to the tagging key than the machine alone.

[0093] Our IP puzzle works as follows. The server 103-1 or authentication authority 113 selects an $m<n$, i. e. the work factor, and a random string r* of length n. The value r^* itself will not be transmitted to the client computer. Additionally the server computes

$$k_e \leftarrow h(c_p \| IP(S) \| IP(C) \| r^*)$$

and uses k_e to encrypt the tagging key. Let $n_p=n-m$ denote the number of cleartext bits, m_u the number of bits to be entered by the user, and m_c the number of bits determined by the machine. We require $m_u+m_c=m$. Let $r\hat{}=[r^*]_{np}$, $r_c=[r^*]_{n-m+1,n-mu}$, and $r_u=[r^*]_{n-mu+1,n}$. Then $r^*=r\hat{}\|r_c\|r_u$.

[0094] The client is given the tuple

$$(r\hat{}, s^*, C)$$

with

$$s^* = h(r^* \| IP(C) \| IP(S)),$$

$$C = Enc_{k_e}(k_t),$$

where k_t is the tagging key to be transmitted.

[0095] In order to obtain the cleartext k_t, the client needs to find r_c and r_u in order to retrieve r*. The client can check that it found a solution r by testing if

Intuition Machines Exhibit 1030, Page 9 of 22

$$h(r \| IP(C) \| IP(S)) = s^{\bullet}$$

[0096] Given that h is a cryptographic hash function that is providing some partial-input preimage resistance, this needs to be done by brute force. This allows us to roughly predict the average amount of computing time the client needs to invest. After finding r^* the client can compute k_e and decrypt k_t.

[0097] The CAPTCHA now plays into this by giving an additional hint on r^* by providing r_u in the form of a CAPTCHA, so that the user can type in r_u into the client computer and thus effectively help the client computer to solve the puzzle and retrieve the tagging key.

b) Scheme Requiring a CAPTCHA Solution

[0098] A slight variation of the above scheme allows making sure that the client computer can make use of the solution to the client puzzle only if a human user provides the correct solution to the CAPTCHA. The client puzzle is very similar to the one given above.

[0099] Using the notation r^*, $r\hat{}$, r_c, r_u established above, the server 103-1 selects, instead of r^*, a random $r\hat{}$, r_c, r_u, computes $r\tilde{}_u = [h(c_u \| r_u)]_{mu}$. Then we use $r^* = r\hat{} \| r_c \| r\tilde{}_u$ and

$$k_e \leftarrow h(c_p \| IP(S) \| IP(C) \| r^*) \oplus h(c'_p \| IP(S) \| IP(C) \| h(c'_u \| r_u))$$

as key to en- or decrypt k_t. Here c_u and c'_u are again two publicly known constants (differing also from the other constants used above). Like in the first key transport scheme, the user will be given r_u in the form of a CAPTCHA. The crucial observation here is that the client is not provided a means to integrity check either k_e or k_t and therefore has no means to directly verify that the correct key has been found. This means that essentially the correct key is obtained only if both the correct solution to the IP puzzle and the CAPTCHA are known to the client computer.

c) Selection of Algorithms and Parameters

[0100] We propose a possible instantiation of our first scheme for key transport, the one realizing a differential work factor. We instantiate the 2nd preimage resistant hash function h with SHA-256. For the encryption function $Enc_k(.)$ we employ the block cipher AES with a 128-bit key; we obtain the key k_e from the output of h by truncating the output to the required size.

[0101] The selection of m_c and m_u (and thus m) is critical for the security as well as the usability of the system. We assume that a graphical CAPTCHA consisting of characters and digits (and resolving some ambiguities such as '0' vs. 'O', '1' vs. '1') is acceptable. We use only the lowercase characters and convert uppercase letters if necessary. This will give us about 5 bit per character typed in by the user. In our opinion 4 characters or 20 bits in total should be acceptable. Furthermore, on a Pentium M (1.5GHz) system the implementation of SHA-256 runs at about 2^{18} operations per second.

[0102] This choice of $m_u=20$ results in a differential work factor of about 2^{20}, so that a choice of m_c=18 and thus m=38 should be sufficient at the current time. Solving the puzzle with human help should need about 0.5 seconds on average on the machine given as reference above. The value of m_c would need to be gradually increased in the future when vastly faster machines are commonplace.

3. Tagging Scheme

[0103] First, we recall the requirements that the tagging scheme performed by a client must fulfill in order to efficiently and securely work. Then we describe the concrete scheme, and provide some performance figures.

[0104] To explain the tagging scheme we assume that a client computer C wishing to communicate with a server S behind the gate 109 has a cryptographic key $k_{c,s}$ that is specific to the client-server pair and possibly time-dependent or even session-dependent. Furthermore, we assume that the gate 109 itself does also know $k_{c,s}$, possibly by deriving it via the keying scheme from other information known to the gate 109.

Intuition Machines Exhibit 1030, Page 10 of 22

a) Requirements

[0105] The purpose of the tagging scheme is to provide proof to the gate 109 about the authenticity of a packet sent from the client computer *C* to the server *S*, as the gate sets -in the event of an attack or overload - the priority with which the packet is forwarded via the access network 107 to the server.

* Security. The tagging mechanism shall keep an adversary from learning the cryptographic key $k_{C,S}$ from the packets of a flow. Furthermore, an adversary shall be unable to reuse the tag from one packet to forge a valid tag for another packet, even if the adversary spoofs the client address as origin.
* Efficiency of verification. The gate 109 has to check every packet that is tagged, so the tagging scheme shall not impose too much computational overhead in order to keep latencies small.

[0106] It should be noted that the server *S* or the authentication authority does not have to cope with generation or verification of tags.

b) The Concrete Scheme

[0107] For tagging a packet the tag generation means 116 of a client uses a special tag field in the IP header of a packet to be sent. The gate 109 will then look for available tags in received packets, check the tags, and for example base a decision about the packet's priority on this.

[0108] Given the requirements discussed above we employ a secure Message Authentication Code (MAC) scheme, i.e. a pair of possibly randomized algorithms *(Mac,* Ver) for tagging packets and verification, making use of parameters k and t for tagging key and tag lengths measured in bits:

$$Mac : \{0,1\}^k \times \{0,1\}^* \rightarrow \{0,1\}^t$$

$$Ver : \{0,1\}^k \times \{0,1\}^* \times \{0,1\}^t \rightarrow \{0,1\}$$

with *Ver(k, M, Mac(k, M))* ⇒ 1.

[0109] In the following we describe the tagging and verification scheme based on these functions.

[0110] First we describe which data from a packet P is included in the tag computation and verification. Here we distinguish between IPv4 and IPv6: For a given packet P, we use the tuples

$$M_{hdr}(P) \leftarrow \begin{cases} IP_s(P), IP_d(P), Len(P), Ident(P)) & IPv4 \\ IP_s(P), IP_d(P), Len(P), Flow(P) & IPv6 \end{cases}$$

and for a TCP packet

$$M_{tcp}(P) \leftarrow (Port_s(P), Port_d(P), Seq(P), Ack(P))$$

(or M_{tcp} *(P)* ← *()* for non-TCP packets) to set up the input to the tagging and verification operations as

$$M(P) \leftarrow (M_{hdr}(P), M_{tcp}(P))$$

[0111] For IPv4 the maximal total size of *M(P)* is 12+12 octets or 96+96=192 bits, for IPv6 these numbers are 37+12=49

Intuition Machines Exhibit 1030, Page 11 of 22

octets resp. 296+96=392 bits (where we count *Flow(P)* as 3 octets or 24 bits).

[0112] Now the procedures for computing a tag at the client 101-1 and for checking a tag at gate 109 are as follows:

* The client tries to retrieve from its internal key storage the tagging key for each packet P it sends to the server:

$$k \leftarrow \text{client_get_session_key}(\text{IP}_d(P)).$$

If no tagging key is available, the packet is sent untagged. Otherwise the client sets up M(P), computes

$$\tau(P) \leftarrow \text{Mac}(k, M(P)),$$

and includes $\tau(P)$ in P using the tag option.

* The gate 109 maintains a label (cfg. Fig.5)

$$\lambda(P) \in \{\text{Untagged, Invalid, Valid}\}$$

for each packet for further queuing decisions, determined as follows. If a packet P is not tagged

$$\lambda(P) \leftarrow \text{Untagged}.$$

If a packet P carries a tag option, the gate obtains

$$k \leftarrow \text{gate_get_session_key}(\text{IP}_s(P), \text{IP}_d(P)).$$

If no tagging key is available $\lambda(P) \leftarrow$ Invalid. Otherwise the gate extracts the alleged tag r'(P), obtains *M(P)* from the other packet information as described above, and computes

$$b \leftarrow Ver(k, M(P), \tau'(P)),$$

Then

$$\lambda(P) \leftarrow \begin{cases} Valid & b = 1 \\ Invalid & b = 0 \end{cases}$$

Based on the label the gate 109 decides what further to do with the packet, e.g. forward it with high or low priority, or drop it entirely. Note, however, that dropping of SYN packets is not a good idea, as it would essentially prevent the server to be reachable.

d) Selection of Algorithms and Parameters

[0113] For implementing the MAC scheme several straight-forward choices are available, such as CBC-MAC with various block ciphers or the HMAC construction, e.g. HMAC-SHA1. This choice directly affects the natural key size and maximal tag length of our scheme. For IPv4 *M(P)* fits into two AES blocks, so using CBC-MAC with AES would require just two encryption operations, while for IPv6 four encryption operations would be needed. In both cases HMAC-SHA1 would need, depending on implementation, 2 or 4 compression function operations, so we chose CBC-MAC with AES

Intuition Machines Exhibit 1030, Page 12 of 22

and a 128 bit key.

[0114] We shorten the output of 128 bits to the first 80 bits of tag data, so that the option for IPv4, including option type and length, has a total length of 96 bits or 12 octets. This still offers a level of security against an adversary that is appropriate for our purpose.

Claims

1. A method for identifying and controlling unsolicited traffic in a computer network system (10), which comprises at least a first access network (105, 105a, 105b) connecting at least one source (101-1 to 101-N; 102-1 to 102-N) to a core network (104) and at least a second access network (107) connecting at least one destination (103-1 to 103-N) to said core network (104), wherein said method comprises the steps:

   - directing traffic units addressed to at least one destination (103-1) from the at least one source (101-1) to the gate (109) which is interconnected between the core network (104) and the second access network (107);
   - verifying each traffic unit at the gate (109) to determine whether or not the traffic unit has been marked validly;
   - identifying each traffic unit (112) as unsolicited if it has been marked invalidly or if it has not been marked at all;
   - controlling the flow of each traffic unit (112) having been idenitified as unsolicited in response to at least one predetermined system parameter.

2. A method for identifying and controlling unsolicited traffic in a computer network system, which comprises at least a first access network (105) connecting at least one source ((101-1 to 101-N; 102-1 to 102-N) to a core network (104) and at least a second access network (107) connecting at least one destination (103-1 to 103-N) to said core network (104), wherein said method comprises the steps:

   - directing traffic units addressed to at least one destination (103-1) from the at least one source (103-1) to the gate (109) which is interconnected between the core network (104) and the first access network (105);
   - verifying each traffic unit at the gate (109) to determine whether or not the traffic unit has been marked validly;
   - identifying each traffic unit (112) as unsolicited if it has been marked invalidly or if it has not been marked at all;
   - controlling flow of each traffic unit having been identified as unsolicited in response to at least one predetermined system parameter.

3. The method according to claim 1 or 2, wherein the traffic units are IP packets or TCP packets and wherein the core network (104) is the Internet.

4. The method according to any one of the claims 1 to 3, wherein the step of controlling comprises the following steps:

   blocking, delaying and/or limiting flow of each traffic unit having been identified as unsolicited to a destination (103-1 to 103-N) in dependence of the congestion condition of the core network (104) and/or second access network (107).

5. The method according to any one of the preceding claims, comprising the following steps:

   - Generating and transmitting a permission message to the at least one source (101-1);
   - Marking traffic units to be sent at the source using information from the permission message; and
   - Storing information relating to the permission message at the gate (104) to allow the gate to verify traffic units.

6. The method according to claim 5, wherein the permission message includes a tagging key used to mark traffic units to be sent at the source.

7. The method according to claim 5 or 6, wherein the permission message is transmitted in form of a challenge to be solved by the at least one source to obtain the tagging key.

8. The method of claim 7, wherein the challenge is solved at the source only with an interaction of a human user.

9. The method of claim 7 or 8, wherein a CAPTCHA algorithm and/or an IP puzzle algorithm are used to generate a challenge to transmit the permission message to the at least one source.

Intuition Machines Exhibit 1030, Page 13 of 22

10. The method of claim 6, wherein the tagging key is used in a message authentication code (MAC) to compute a tag to mark traffic units, wherein the tagging key or information to derive the tagging key is stored at the gate.

11. The method of claim 10, wherein the tagging key is a cryptographic key that is used by the source to generate tags for marking traffic units and that is used by the gate to verify tags included in traffic units.

12. The method according to any one of the claims 5 to 11, wherein the permission message is generated by the gate (104) and forwarded to an authentication means (113) which in turn transmits the permission message to the at least one source.

13. The method according to any one of the claims 5 to 11, wherein the permission message is generated by an authentication means (113) which transmits the permission message to the at least one source and the gate.

14. A computer network system for identifying and controlling unsolicited traffic, comprising
a core network (104),
at least a first access network (105; 105a, 105b),
at least one source (101-1 to 101-N; 102-1 to 102-N) connected to said first access network (105; 105a, 105b),
at least a second access network (107),
at least one destination (103-1 to 103-N) connected to said second access network (107),
at least one gate (109) located between the first access network (105; 105a, 105b) and the core network (104) and/or between the second access network (107) and the core network (104),
means (113; 114)) for generating and transmitting a permission message to the at least one source (101-1 to 101-N; 102-1 to 102-N), wherein
the at least one source (101-1 to 101-N; 102-1 to 102-N) comprises means (116) for obtaining a tagging key from that permission message used to mark traffic units addressed to a destination,
the gate (109) comprises:

- storage means for storing information related to a permission message,
- means (117) for verifying received traffic units on the basis of the information stored to determine whether or not the traffic units are marked validly,
- means (117) for identifying each traffic unit being marked invalidly as unsolicited, and
- means (117) for controlling the flow of each traffic unit being marked invalidly in response to at least one predetermined system parameter.

15. The computer network system according to claim 14, wherein each source (101-1 to 101-N; 102-1 to 102-N)comprises a client computer, each destination (103-1 to 103-N) comprises a server and wherein the core network (104) is the Internet.

16. The computer network system according to claim 14, wherein the means for generating and transmitting a permission message to the at least one source is implemented in an authentication unit (113) which is located in a device connected to the gate (104) or in at least one destination.

17. The computer network system according to claim 16, wherein the authentication unit (113) is adapted to generate a permission message in form of a challenge to be solved by the at least one source, wherein only sources (101-1 to 101-N) having solved the challenge are able to mark traffic units validly.

18. A gate (109) used in a computer network system according to any one of the claims 14 to 17, comprising

- storage means for storing information related to a permission message,
- means (117) for verifying traffic units on the basis of the information stored to determine whether or not the traffic units are marked validly,
- means (117) for identifying each traffic unit being marked invalidly as unsolicited, and
- means (117) for controlling the flow of each traffic unit being marked invalidly in response to at least one predetermined system parameter.

Intuition Machines Exhibit 1030, Page 14 of 22

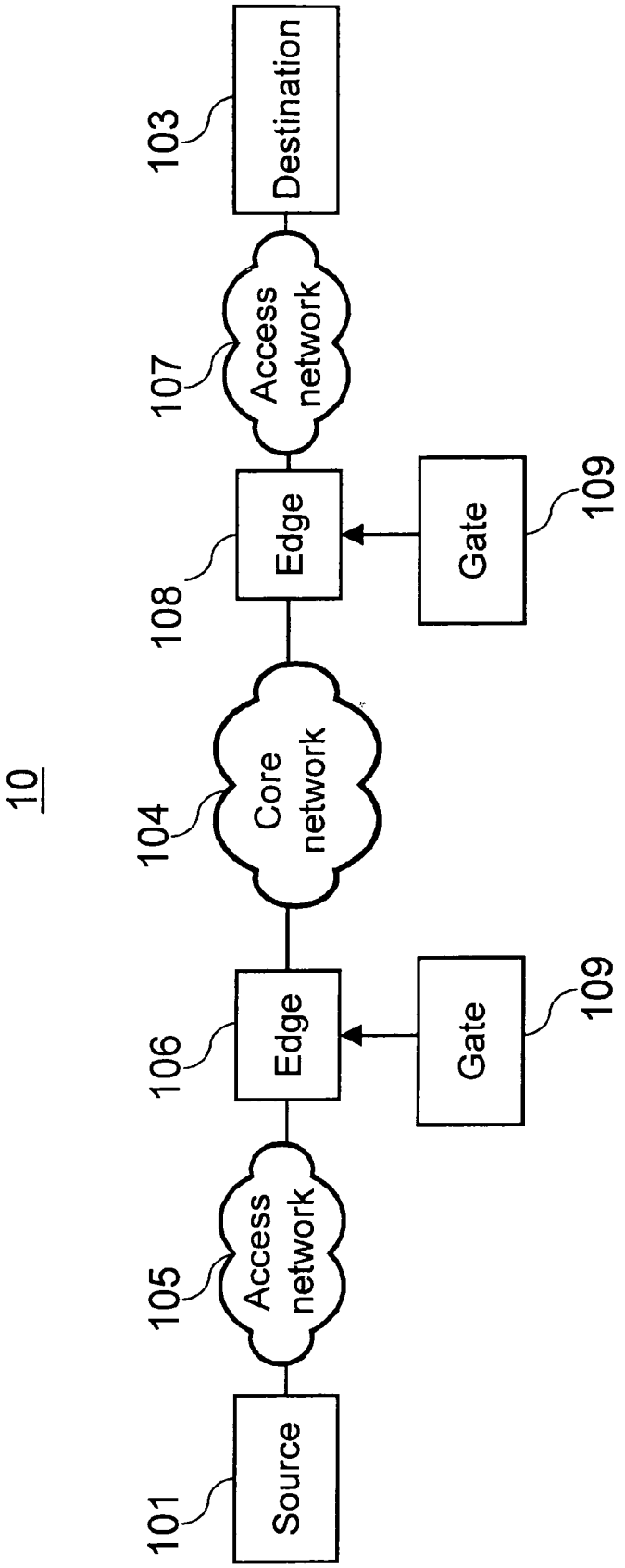

Fig. 1

Intuition Machines Exhibit 1030, Page 15 of 22

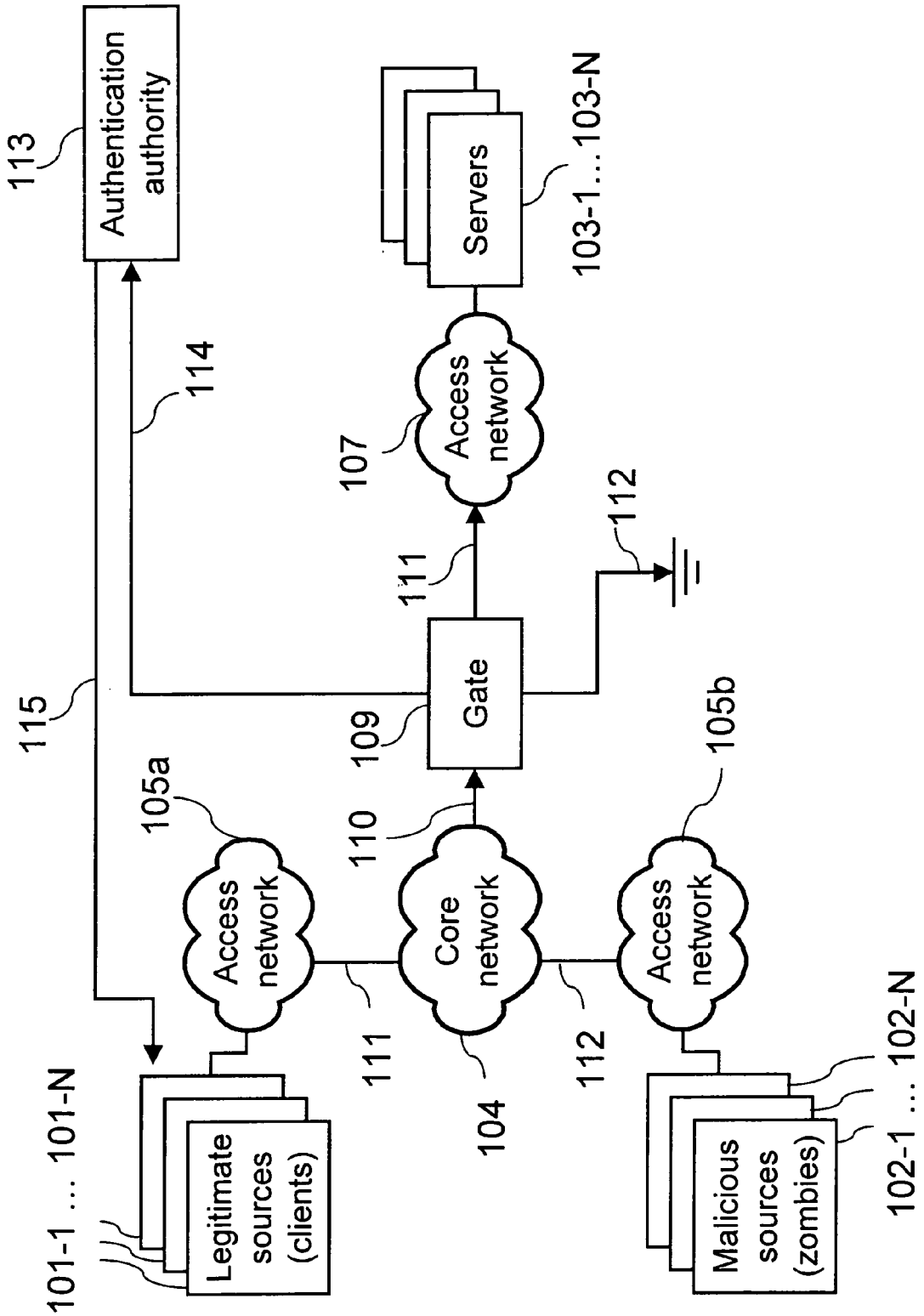

Fig. 2

Intuition Machines Exhibit 1030, Page 16 of 22

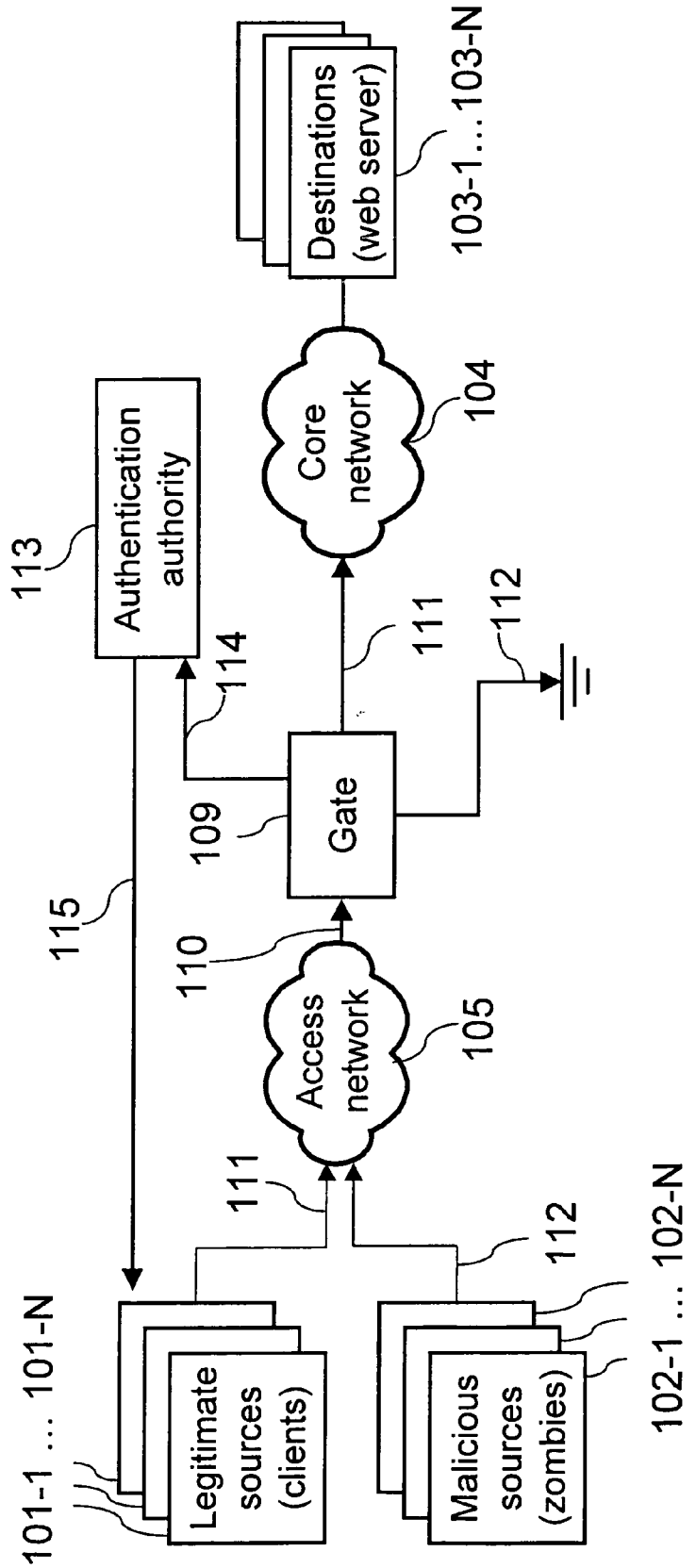

Fig. 3

Intuition Machines Exhibit 1030, Page 17 of 22

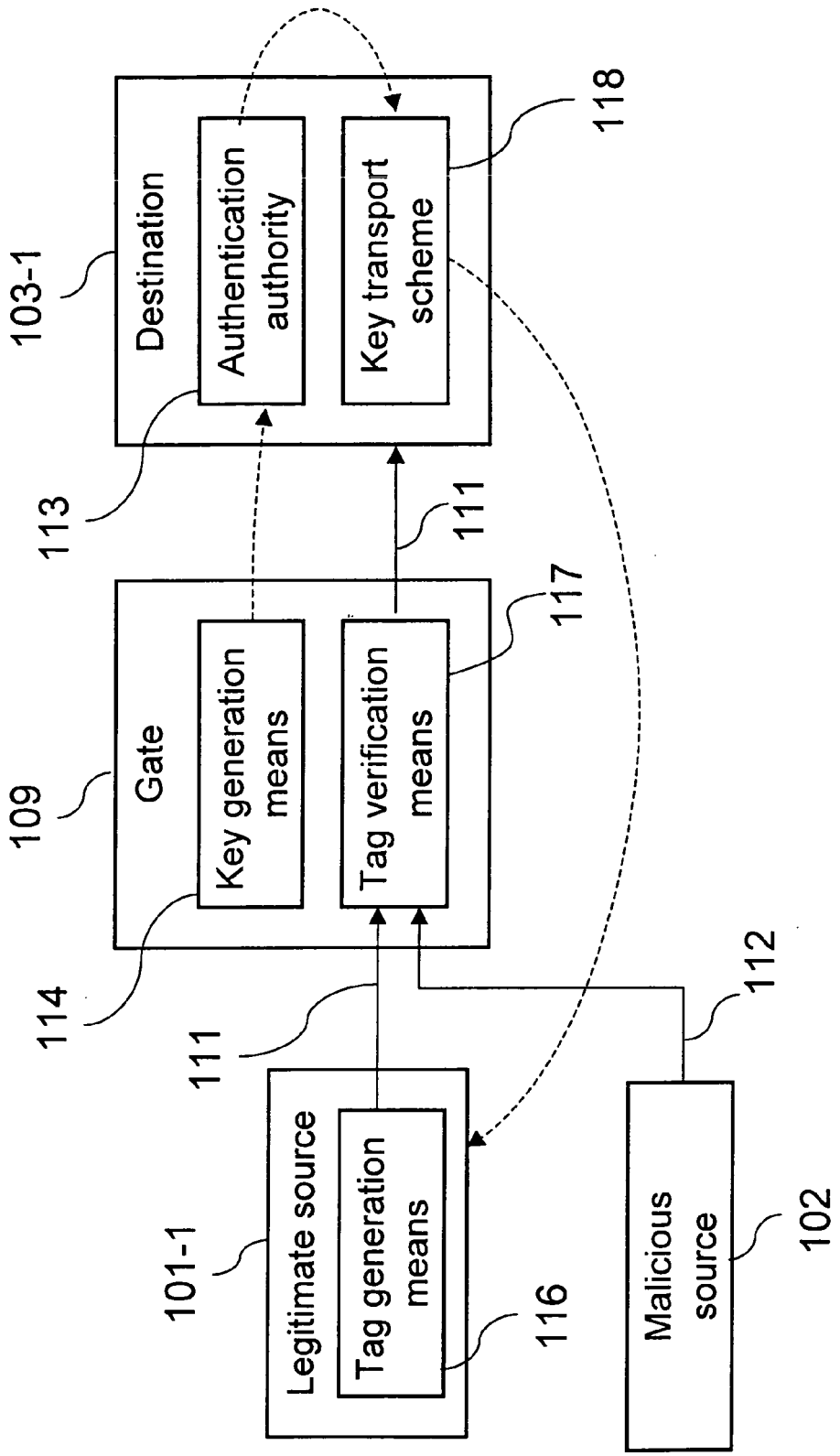

Fig. 4

Intuition Machines Exhibit 1030, Page 18 of 22

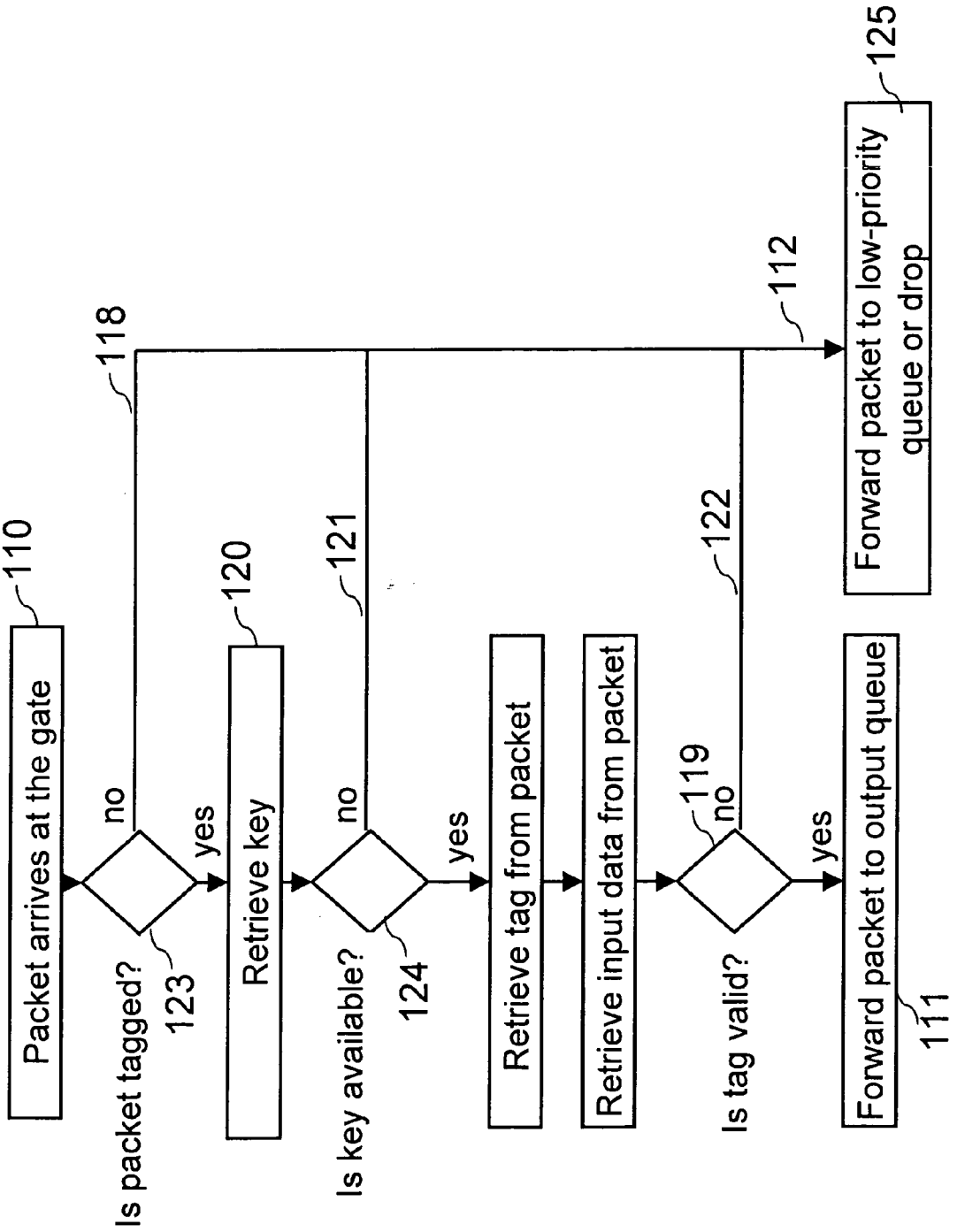

Fig. 5

Intuition Machines Exhibit 1030, Page 19 of 22

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 06 00 7693

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	US 2004/230797 A1 (OFEK YORAM [US] ET AL) 18 November 2004 (2004-11-18) * paragraphs [0085] - [0090] * * paragraphs [0098], [0099] * * paragraphs [0104] - [0110] * * paragraphs [0123] - [0128] * * paragraphs [0137], [0161] * * paragraphs [0225] - [0230] * * figures 1,2,4,5,8,9,18,19 * -----	1-18	INV. G06F21/00 H04L29/06
X	ANDERSON T ET AL: "PREVENTING INTERNET DENIAL-OF-SERVICE WITH CAPABILITIES" COMPUTER COMMUNICATION REVIEW, ACM, NEW YORK, NY, US, vol. 34, no. 1, January 2004 (2004-01), pages 39-44, XP001223359 ISSN: 0146-4833 * page 41, right-hand column, paragraph 2 * * page 42, left-hand column, line 14 - last line * * page 42, right-hand column, paragraph 4 - page 43, left-hand column, paragraph 1 * -----	1-18	TECHNICAL FIELDS SEARCHED (IPC) G06F H04L
D,A	EP 1 284 558 A1 (LUCENT TECHNOLOGIES INC [US]) 19 February 2003 (2003-02-19) * abstract * * paragraphs [0021], [0028] * * paragraphs [0034], [0035] * * figures 1-4 * -----	1-18	

The present search report has been drawn up for all claims

1

Place of search	Date of completion of the search	Examiner
Munich	15 January 2007	BENGI-AKYUEREK, K

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons
.....
& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

Intuition Machines Exhibit 1030, Page 20 of 22

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 06 00 7693

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

15-01-2007

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2004230797 A1	18-11-2004	NONE	
EP 1284558 A1	19-02-2003	DE 60201716 D1	02-12-2004
		DE 60201716 T2	22-12-2005
		US 2003036970 A1	20-02-2003

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- EP 1284558 B1 **[0013]**

Non-patent literature cited in the description

- **Y. KIM et al.** Packetscore: statistics-based overload control against distributed denial-of-service attacks. *Proceedings of IEEE INFOCOM'04,* March 2004 **[0008]**
- **S. SAVAGE et al.** Network support for IP-traceback. *IEEE/ACM Transactions on Networking,* June 2001, vol. 9 (3), 226-237 **[0010]**
- **A. YAAR et al.** A stateless internet flow filter to mitigate DDoS flooding attacks. *Proceedings of the IEEE Symposium on Security and Privacy,* May 2004 **[0010]**
- **X. YANG et al.** A DoS-limiting network architecture. *ACM SIGCOMM'05,* August 2005 **[0011]**
- **T. ANDERSON et al.** Preventing Internet Denial-of-Service with capabilities. *Proceedings of Hot Topics in Networking (HotNetsII,* October 2003 **[0012]**
- CAPTCHA: Using Hard AI Problems for Security. **L. V. AHN et al.** Advances in Cryptology-EUROCRYPT. Springer Verlag, 2003, vol. 2656, 294-311 **[0018]**
- Using client puzzles to protect TLS. **D. DEAN et al.** Proceedings of the 10th USENIX Security Symposium. August 2001 **[0018] [0083]**
- **M. BELLARE et al.** The security of the cipher block chaining message authentication code. *Journal of Computer and Systems Sciences,* December 2000, vol. 61 (3), 362-399 **[0076]**
- CHAPTCHA: Using Hard AI Problems for Security. **L.V.AHN et al.** Advances in Cryptology-EUROCRYPT. Springer Verlag, 2003, vol. 2656, 294-311 **[0078]**
- Federal Information Processing Standards Publication (FIPS PUB 180-2. February 2004 **[0088]**

Intuition Machines Exhibit 1030, Page 22 of 22