



(51) International Patent Classification:

H04L 9/40 (2022.01)

H04L 69/16 (2022.01)

H04L 69/28 (2022.01)

(21) International Application Number:

PCT/EP2023/059477

(22) International Filing Date:

12 April 2023 (12.04.2023)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

17/746,556

17 May 2022 (17.05.2022)

US

22306939.4

19 December 2022 (19.12.2022)

EP

(71) Applicant: AMADEUS S.A.S. [FR/FR]; 485, Route du Pin Montard Sophia Antipolis, 06410 Biot (FR).

(72) Inventors: CHIAPPONI, Elisa; 11 Impasse des Alpes, 06600 Antibes (FR). DACIER, Marc; KAUST, Tulip Lane, G 1369, Thuwal 23955 (SA). THONNARD, Olivier; 38 boulevard Emmanuel Rouquier, Hameau des Aspres, 06130 Grasse (FR). RIGAL, Vincent; 741 chemin des moyennes Breguières, Résidence Aurelia Bat 3, 06600 Antibes (FR). FANGAR, Mohamed; 50 Rue Emmanuel Grout, 06200 Nice (FR).

(74) Agent: SAMSON & PARTNER PATENTANWÄLTE MBB, ASSOCIATION NO. 275; Widenmayerstrasse 6, 80538 München (DE).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

(54) Title: PROXY DETECTION SYSTEMS AND METHODS

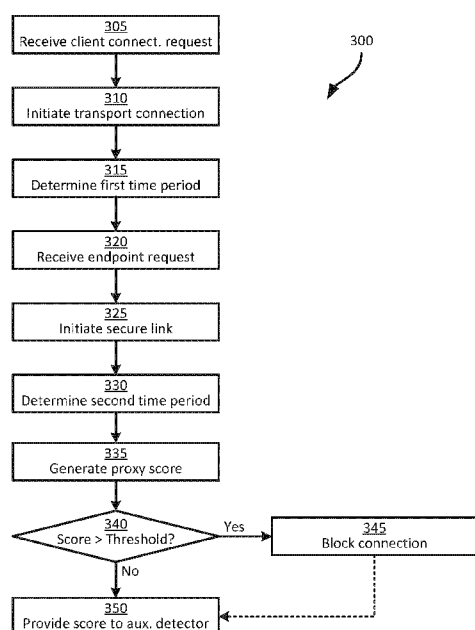


FIG. 3

(57) Abstract: A proxy detection method includes: in response to receiving, from a client device, a first request to establish a transport-layer connection between the client device and the server, transmitting a first message to the client device according to a first handshake sequence, for establishing the transport-layer connection; determining a first time period associated with completion of the first handshake sequence; in response to receiving, from the client device over the transport-layer connection, a second request to establish a secure link between a client endpoint and the server, transmitting a second message to the client endpoint according to a second predefined handshake sequence, for establishing the secure link; determining a second time period associated with completion of the second handshake sequence; and generating, based on the first time period and the second time period, a score indicating a likelihood that the client device is a proxy for the client endpoint.

Published:

- *with international search report (Art. 21(3))*
- *in black and white; the international application as filed contained color or greyscale and is available for download from PATENTSCOPE*

PROXY DETECTION SYSTEMS AND METHODS

BACKGROUND

[0001] Servers receive and respond to requests from client devices, e.g., to deliver
5 data requested by the client devices in connection with web-based services. For certain
services, responding to such requests can be computationally intensive. For example,
servers handling search requests for travel-related services (e.g., flights, hotels, and the
like) may incur significantly higher computational costs to generate responses to such
requests than the costs incurred by other servers responsible for the retrieval of
10 previously generated and indexed data.

[0002] The operators of the above-mentioned servers may derive little or no return for
the cost of servicing fraudulent or abusive client requests. Upon detecting such requests,
discarding or otherwise altering the usual request handling process may therefore be
desirable, to reduce the allocation of computational resources to responding to such
15 requests, with little likelihood of return, e.g., in the form of travel services being purchased
from the server's operator. Fraudulent or abusive client requests, however, may be routed
through proxy devices, which complicates their detection. Detecting such requests may
be particularly challenging when the proxy devices are residential or other consumer-level
devices that may also originate legitimate requests.

20

SUMMARY

[0003] An aspect of the specification provides a proxy detection method in a server,
the method including: in response to receiving, from a client device, a first request to
establish a transport-layer connection between the client device and the server,
25 transmitting a first message to the client device according to a first handshake sequence,
for establishing the transport-layer connection; determining a first time period associated
with completion of the first handshake sequence; in response to receiving, from the client
device over the transport-layer connection, a second request to establish a secure link
between a client endpoint and the server, transmitting a second message to the client

endpoint according to a second predefined handshake sequence, for establishing the secure link; determining a second time period associated with completion of the second handshake sequence; and generating, based on the first time period and the second time period, a score indicating a likelihood that the client device is a proxy for the client endpoint.

[0004] According to embodiments of this aspect, the transport-layer connection is based on the Transport Control Protocol (TCP).

[0005] According to embodiments of this aspect, the first message includes a SYN-ACK message; and wherein the first time period is a time elapsed between transmission of the first message, and receipt of an ACK message from the client device.

[0006] According to embodiments of this aspect, wherein the secure link is based on one of (i) the Transport Layer Security (TLS) protocol, and (ii) the Secure Sockets Layer (SSL) protocol.

[0007] According to embodiments of this aspect, the second time period is a time elapsed between transmission of the second message, and receipt of a next message from the client endpoint according to the second predefined handshake sequence.

[0008] According to embodiments of this aspect, generating the score includes determining a difference between the first and second time periods.

[0009] According to embodiments of this aspect, the method further comprises selecting a handling action for future requests over the transport-layer connection, based on the score.

[0010] According to embodiments of this aspect, the method further comprises a test to determine if the first request to establish a transport-layer connection is coming from a mobile network and selecting a handling action for future requests over the transport-layer connection is based on the result of this test.

[0011] According to embodiments of this aspect, the handling action includes discarding the future requests when the score exceeds a threshold.

[0012] According to embodiments of this aspect, the threshold is a function of network conditions between the client device and the proxy detector, for example if the client device is on a mobile network.

5 **[0013]** According to embodiments of this aspect, the value of threshold increases when a test to determine if the first request to establish a transport-layer connection is coming from a mobile network is positive.

[0014] According to embodiments of this aspect, the method further comprises providing the score to an auxiliary detector.

10 **[0015]** Another aspect of the specification provides a server, including: a communications interface; and a processor configured to: in response to receiving, from a client device, a first request to establish a transport-layer connection between the client device and the server, transmit a first message to the client device according to a first handshake sequence, for establishing the transport-layer connection; determine a first time period associated with completion of the first handshake sequence; in response to
15 receiving, from the client device over the transport-layer connection, a second request to establish a secure link between a client endpoint and the server, transmit a second message to the client endpoint according to a second handshake sequence, for establishing the secure link; determine a second time period associated with completion of the second handshake sequence; and generate, based on the first time period and the
20 second time period, a score indicating a likelihood that the client device is a proxy for the client endpoint.

[0016] According to embodiments of this aspect, the transport-layer connection is based on the Transport Control Protocol (TCP).

25 **[0017]** According to embodiments of this aspect, the first message includes a SYN-ACK message; and wherein the first time period is a time elapsed between transmission of the first message, and receipt of an ACK message from the client device.

[0018] According to embodiments of this aspect, the secure link is based on one of (i) the Transport Layer Security (TLS) protocol, and (ii) the Secure Sockets Layer (SSL) protocol.

5 **[0019]** According to embodiments of this aspect, the second time period is a time elapsed between transmission of the second message, and receipt of a next message from the client endpoint according to the second predefined handshake sequence.

[0020] According to embodiments of this aspect, the processor is configured, to generate the score, to determine a difference between the first and second time periods.

10 **[0021]** According to embodiments of this aspect, the processor is further configured to select a handling action for future requests over the transport-layer connection, based on the score.

[0022] According to embodiments of this aspect, the handling action includes discarding the future requests when the score exceeds a threshold.

15 **[0023]** According to embodiments of this aspect, the processor is further configured to provide the score to an auxiliary detector.

BRIEF DESCRIPTIONS OF THE DRAWINGS

[0024] Embodiments are described with reference to the following figures.

[0025] FIG. 1 is a diagram illustrating a communications system.

20 **[0026]** FIG. 2 is a diagram illustrating certain internal components of the proxy detector of FIG. 1.

[0027] FIG. 3 is a flowchart of a proxy detection method.

[0028] FIG. 4 is a diagram illustrating an example performance of the method of FIG. 3.

25 **[0029]** FIG. 5 is a diagram illustrating another example performance of the method of FIG. 3.

DETAILED DESCRIPTION

[0030] FIG. 1 depicts a communications system 100, including a request handler 104 and a plurality of client devices, referred to collectively as client devices 108 and generically as a client device 108. In the illustrated example, the system 100 includes four
5 client devices 108-1, 108-2, 108-3, and 108-4, although it will be understood that the system 100 can include greater and smaller numbers of client devices 108 in other examples. The client devices 108 are computing devices such as desktop computers, smart phones, laptop computers, or the like. Each client device 108 thus includes suitable
10 hardware elements, such as processing, storage and network communications components, as well as input and output devices (e.g., keyboards, touch panels, displays, and the like), enabling the client device 108 to communicate with the request handler 104 over a network or combination of networks. Communications between the client devices 108 and the request handler 104 include the transmission of requests from client devices 108 to the request handler 104, and the generation and transmission of responses to
15 such requests from the request handler 104 to the relevant client devices 108.

[0031] The request handler 104 can be implemented as a server or set of servers, configured to receive and process requests from the client devices 108. The request handler 104 therefore includes processing and storage hardware components, e.g., executing suitable software to receive and interpret client requests, as well as to generate
20 and return response data to such requests. The requests may include, for example, search requests for travel-related goods or services, such as search requests for flights between specified origin and destination locations (e.g., particular cities or airports), on specified days, or the like. In order to generate response data for a client request, the request handler 104 can be configured to retrieve and process data from various
25 repositories and/or interact with other computing devices (e.g., operated by airlines, or the like), to generate combinations of flights that satisfy search parameters set out in the client request.

[0032] The generation of response data can be computationally complex, as the availability and pricing of flights may be highly variable and dependent on the identity of

an operator of the client device 108, among other factors. The costs (e.g., in terms of financial commitments, staffing, and the like) of the computational resources (e.g., processing time, storage capacity, and the like) allocated to handling search requests from the client devices 108 may be supported in part by purchases of the above-mentioned flights by operators of the client devices 108. Some client requests, however, are highly unlikely to lead to such purchases, and committing computational resources to generate responses to those requests may therefore not be desirable.

[0033] For example, some client requests are originated by scraper bots, and the results generated by the request handler 104 may be used to populate third-party search engines, storefronts, or the like, thus potentially depriving the operator of the request handler 104 of at least some of the financial return associated with those search results, while still having incurred the computational cost of generating the search results. As will be apparent to those skilled in the art, bot-originated requests are not the only type of client request that it may be desirable to detect and handle differently from other client requests. Such requests are simply discussed herein as an illustrative example.

[0034] Bot-originated requests such as those mentioned above, and/or other client requests that the operator of the request handler 104 may seek to detect and handle differently from other requests, may be detected based on the content of the requests, attributes of the requests' senders, or the like. The system 100 may include, for example, an auxiliary detector 110, e.g., in the form of an additional server or set of servers, and/or additional application(s) expected by the request handler 104. The auxiliary detector 110 is configured to process incoming requests 112 to determine whether each request 112 is likely to have originated from a bot or other source for which differential handling is desired (e.g., sources presenting security risks, engaging in fraudulent behavior, or the like). A request may therefore be forwarded to the request handler 104 for further processing, for example, only if the auxiliary detector 110 determines a low likelihood that the request originated from a bot.

[0035] Bot-originated requests, however, may be obfuscated from detection by the auxiliary detector 110 by routing such requests through proxies. For example, the client

devices 108-1 and 108-2 are shown transmitting respective requests 112-1 and 112-2 to the request handler 104 in FIG. 1; the requests 112-1 and 112-2 are generated at the client devices 108-1 and 108-2 themselves. The client device 108-3, on the other hand, is shown transmitting distinct requests 112-3 and 112-4 to the request handler 104. The request 112-3 may be generated at the client device 108-3, e.g., via input data provided by an operator of the client device 108-3. The request 112-4, however, originated not at the client device 108-3, but at a distinct client device 108-4 employing the client device 108-3 as a proxy. In some examples, additional computing devices may handle the request 112-4 between the client device 108-4 and the client device 108-3. For example, a superproxy (not shown) can be configured to receive instructions from the client device 108-4 (e.g., executing a scraper bot application) and generate numerous individual requests for transmission via distinct proxies such as the client device 108-3.

[0036] Various mechanisms are available to detect proxied requests 112 (e.g., filtering requests based on blacklisted Internet Protocol (IP) addresses, or the like). Those mechanisms, however, may only detect a portion of proxied requests. Further, effectiveness of those detection mechanisms may be reduced for certain forms of proxied request. In the illustrated system, for example, the client device 108-3 is referred to as a residential proxy, in that the client device 108-3 is a consumer-level computing device that is unlikely to trigger conventional proxy-detection mechanisms. The client device 108-3, as seen above, can also originate legitimate (e.g., not bot-originated) requests that are preferably processed by the request handler 104 in the same manner as the requests 112-1 and 112-2, in addition to proxied requests for which modified handling may be desirable.

[0037] To detect proxied requests in general, and requests routed via residential proxies in particular, the system 100 therefore also includes a proxy detector 116. The proxy detector 116 can be implemented as a distinct computing device (e.g., one or more servers) from the auxiliary detector 110 and the request handler 104. In other examples, the proxy detector 116 can be implemented as an additional software application executed at the computing device(s) implementing the auxiliary detector 110 or the

request handler 104. As will be discussed below in greater detail, the proxy detector 116 is deployed as the first entity in the request-handling infrastructure (labelled as a request-handling subsystem 120 in FIG. 1) with which the client devices 108 communicate. That is, transport-layer connections and secure links are established between the client devices 108 and the proxy detector 116, rather than between the client devices 108 and the auxiliary detector 110 or request handler 104.

[0038] As will be apparent, the client requests 112 are generally implemented as sequences of messages, e.g., to establish communications between a client device 108 and the proxy detector 116, to serve web content or the like to the client device 108, and to receive the above-mentioned search request from the client device. Establishing communications between a client device 108 and the proxy detector 116 typically involves establishing a transport-layer connection, e.g., based on the Transport Control Protocol (TCP) or another suitable transport-layer protocol. Once the transport-layer connection is established, a secure link is established over the transport-layer connection, e.g., based on the Transport Layer Security (TLS) protocol, Secure Sockets Layer (SSL) protocol, or the like. Web content, search requests, response data and the like, can then be exchanged over the secure link.

[0039] The proxy detector 116 is configured to inspect at least some of the above-mentioned messages to determine whether the client device 108 is likely to be operating as a proxy. In particular, as will be discussed below in greater detail, the transport-layer connection is established between the proxy detector 116 and the client device 108 as the nearest transport-layer device (i.e., ignoring routing hardware implementing link-layer and other lower-level functions). The secure link, however, is established with the ultimate client endpoint, e.g., the device executing the web browser or other application that initiated communication with the subsystem 120 via the proxy.

[0040] In the case of non-proxied requests, the nearest transport-layer device and the client endpoint are one and the same, e.g., the client device 108-1 for the request 112-1. In the case of proxied requests, however, the client endpoint does not reside at the nearest transport-layer device. In the context of FIG. 1, for example, the nearest transport-

layer device involved in transmission of the client request 112-4 is the client device 108-3, but the client endpoint is the client device 108-4. The proxy detector 116 is configured, via the above-mentioned message inspection, to determine round-trip time periods associated with the transport-layer connection and the secure link, and to assess whether the client device 108 is likely to be operating as a proxy based on those time periods. Of particular note, the proxy detector 116 can perform the above-mentioned inspection and assessment without modifying the messages themselves or the processes by which the client devices 108 establish communications with the subsystem 120 (e.g., without delivering executable code or other content to the client devices 108 that would not have been delivered in the absence of the proxy detector 116).

[0041] Before discussing the operation of the system 100, and in particular the functionality of the proxy detector 116, in greater detail, certain internal components of the proxy detector 116 will be described with reference to FIG. 2.

[0042] As noted above, the proxy detector 116 can be implemented as a server in the subsystem 120, distinct from the auxiliary detector 110 and the request handler 104. In the illustrated example, the proxy detector 116 includes at least one processor 200, such as a central processing unit (CPU) or the like. The processor 200 is interconnected with a memory 204, implemented as a suitable non-transitory computer-readable medium (e.g., a suitable combination of non-volatile and volatile memory subsystems including any one or more of Random Access Memory (RAM), read only memory (ROM), Electrically Erasable Programmable Read Only Memory (EEPROM), flash memory, magnetic computer storage, and the like). The processor 200 and the memory 204 are generally comprised of one or more integrated circuits (ICs).

[0043] The processor 200 is also interconnected with a communications interface 208, which enables the proxy detector 116 to communicate with the other computing devices of the system 100. The communications interface 208 therefore includes any necessary components (e.g., network interface controllers (NICs), radio units, and the like) to enable such communication. The proxy detector 116 can also include input and output devices connected to the processor 200, such as keyboards, mice, displays, and the like (not

shown). In other examples, input and output devices can be connected to the proxy detector 116 remotely, via another computing device (not shown).

[0044] The components of the proxy detector 116 mentioned above can be deployed in a single enclosure, or in a distributed format. In some examples, therefore, the proxy detector 116 includes a plurality of processors, either sharing the memory 204 and communications interface 208, or each having distinct associated memories and communications interfaces. Implementing the proxy detector 116 in a distributed format can enable scaling of the computational resources available to the proxy detector 116, geographic distribution of the functionality provided by the proxy detector 116, and the like.

[0045] The memory 204 stores a plurality of computer-readable programming instructions, executable by the processor 200. The instructions stored in the memory 204 include a proxy detection application 212, execution of which by the processor 200 configures the processor 200 to perform various functions related to the above-mentioned inspection and assessment of message exchanged with the client devices 108 to detect client devices 108 operating as proxies. In some examples, the application 212 can be implemented as a set of distinct applications, e.g., a packet sniffer application to collect incoming and outgoing messages, and an analysis application to assess the above-mentioned time periods.

[0046] In other examples, as noted earlier, the proxy detector 116 can be implemented on computing hardware shared with either or both of the auxiliary detector 110 and the request handler 104. For example, the memory 304 can store not only the application 212, but also one or more other applications implementing the functionality of the detector 110 and/or request handler 104. In such embodiments, the application 212 is configured as the endpoint for communications addressed to the illustrated computing platform. That is, the application 212, and not the applications implementing auxiliary detection and/or response handling, is configured to handle the establishment of communications with client devices 108. Configuring the application 212 (or the proxy detector 116 more generally, if the proxy detector 116 is implemented in distinct hardware from the other

components of the subsystem 120) as the endpoint enables the application 212 to inspect messages transmitted by the nearest transport-layer device, as well as the client endpoint.

[0047] Turning to FIG. 3, a method 300 of proxy detection is illustrated. The method 300 will be described below in conjunction with its performance in the system 100, and in particular by the proxy detector 116, e.g., via execution of the application 212 by the processor 200.

[0048] At block 305, the proxy detector 116 is configured to receive a first request from a client device 108. The first request is a request to establish a transport-layer connection between the client device 108 and the proxy detector 116, e.g., a TCP-based connection as noted earlier. The request may include, for example, a TCP 'SYN' message containing a sequence number, an identifier of the client device 108, or the like.

[0049] At block 310, in response to the first request received at block 305, the proxy detector 116 is configured to send a message (or the first in a series of messages, depending on the protocol employed to establish the transport-layer connection) to the client device 108, according to a handshake sequence defined by the relevant protocol. Turning briefly to FIG. 4, the client device 108-1 and the proxy detector 116 are shown in isolation, along with a sequence diagram illustrating various messages exchanged between the client device 108-1 and the proxy detector 116.

[0050] In particular, to establish a transport-layer connection 400, the client device 108-1 sends a first request 400a (e.g., the above-mentioned SYN message) to the proxy detector 116. The proxy detector 116, upon receiving the request 400a at block 305, can store a timestamp representing the time at which the request 400a was received. At block 310, the proxy detector 116 transmits a message 400b, such as a SYN-ACK message (in TCP-based embodiments), containing an acknowledgement of the request 400a, as well as a sequence number and/or other relevant information. The handshake sequence continues with a further message 400c from the client device 108-1, e.g., acknowledging the message 400b. In this example, following receipt of the message 400c at the proxy

detector 116, the transport-layer connection 400 is established, and can be used to exchange other data, e.g., to establish a secure link 404, discussed further below. As will be apparent to those skilled in the art, the handshake sequence used to establish the connection 400 need not be exactly as discussed above, depending on the protocol employed to establish the connection 400.

[0051] The proxy detector 116 is also configured to store timestamps representing the time at which the message 400b was sent, and the time at which the message 400c was received. Returning to FIG. 3, at block 315 the proxy detector 116 is configured to determine a first time period, e.g., associated with the above-mentioned handshake sequence. In this example, at block 315 the proxy detector 116 is configured to determine a time period elapsed between the transmission of the message 400b (i.e., the initiation of block 310), and receipt of the message 400c. The measured time period, as will now be apparent, represents the round trip time (RTT) between the client device 108-1 and the proxy detector 116, and is illustrated as a time period 408 in FIG. 4. Although the RTT measurement at block 315 is discussed in connection with the handshake sequence for establishing the connection 400, in other examples the RTT can be determined after the connection 400 is established, e.g., from any other suitable pair of messages exchanged between the client device 108-1 and the proxy detector 116. A suitable pair of messages is a pair in which the first originates at the proxy detector 116, and the second necessarily follows the first and is expected to be transmitted by the client device 108-1 substantially immediately upon receipt of the first.

[0052] Returning to FIG. 3, at block 320 the proxy detector 116 is configured to receive a second request, to establish a secure link over the transport-layer connection (e.g., the connection 400 shown in FIG. 4). The secure link is between a client endpoint and the proxy detector 116, although it is not yet known whether the client endpoint is co-located with the client device 108 (i.e., with the device 108 from which the request at block 305 was received).

[0053] In response to the second request at block 320, the proxy detector 116 is configured to transmit a message initiating a handshake sequence according to a

selected protocol, to establish a secure link with the client endpoint. In the present example, the protocol employed to establish the secure link is the TLS protocol, although other suitable protocols may be employed. It will be apparent that the handshake sequence involved in establishing the secure link will vary with the protocol employed at
5 block 325.

[0054] Returning to FIG. 4, to establish the secure link 404 over the connection 400, the client device 108-1 can transmit a request 404a, such as a 'Client Hello' message as defined in the TLS protocol. For example, the request 404a can contain an indication of the supported protocol version (e.g., TLS 1.2 or 1.3), indications of cipher suites
10 supported by the client device 108-1, a random number (e.g., for generation of a shared master secret, later used to generate encryption keys) and the like.

[0055] In response to the message 404a, the proxy detector 116 transmits a message 404b, such as an acknowledgment of the message 404a, to the client device 108-1. The proxy detector 116 can then transmit one or more further messages as dictated by the
15 handshake sequence defined by the relevant security protocol. For simplicity of illustration, FIG. 4 shows one additional message 404c sent by the proxy detector 116. The message 404c can include, for example, the 'Server Hello' message as defined in the TLS protocol. The message 404c can contain a protocol version and cipher suites supported by the proxy detector 116, a further random number for later use in key
20 generation. The message 404c can also include a server certificate, or the like.

[0056] In response to the message 404c, the client device 108-1 returns an acknowledgement message 404d, and can then send a final message 404e to complete the handshake sequence, such as a 'Change cipher' message in the TLS 1.3 protocol, or a 'Client key exchange' message in the TLS 1.2 protocol.

[0057] Referring again to FIG. 3, at block 330 the proxy detector 116 is configured to determine a second time period associated with the above-mentioned handshake sequence to establish the secure link 404. As noted earlier in connection with block 315, the proxy detector 116 is configured to maintain timestamps associated with the

transmission of the messages 404b and 404c, as well as with the receipt of the messages 404a, 404d, and 404e. The time period determined at block 330 represents a round-trip time for the establishment of the secure link 404. That is, the RTT measured at block 330 is the time elapsed between the transmission of a message by the proxy detector 116, and the receipt of a following message (e.g., expected to be transmitted substantially immediately by the client endpoint in response to the message from the proxy detector 116) from the client endpoint. In the example shown in FIG. 4, the proxy detector 116 determines a second time period 412 elapsed between transmission of the message 404c, and receipt of the message 404e.

[0058] Of particular note, although the example shown in FIG. 4 involves determining the second time period based on messages exchanged during the handshake sequence, in other examples the time period can be determined based on other messages, after establishment of the secure link 404. The messages employed to determine the RTT at block 330 are selected, however, to ensure that they travel between the proxy detector 116 and the client endpoint, whether or not the client endpoint is behind a proxy. Thus, certain messages, such as acknowledgement messages, may not be suitable for use at block 330 because they cannot be guaranteed to have originated at the client endpoint. In the example of FIG. 4, the messages 404c and 404e are employed because the information contained in those messages is required to establish the secure link 404, and therefore cannot be generated by an intermediate proxy.

[0059] Upon determining the second time period 325, the proxy detector 116 is configured to generate a score indicating a likelihood that the client device 108 (e.g., the client device 108-1, in the example of FIG. 4) is operating as a proxy for the client endpoint. In other examples, the determination at block 315 can be performed substantially simultaneously with the determination at block 330, given that the proxy detector 116 can store timestamps associated with the messages exchanged during the above-mentioned handshake sequences and/or subsequent communications with the client device 108.

[0060] Generation of the score at block 335 is based on the first and second time periods, i.e., on the RTT associated with the transport-layer connection 400, and the RTT associated with the secure link 404. Turning to FIG. 4, solely for illustrative purposes, the first time period is assumed to be forty milliseconds, and the second time period is assumed to be fifty-five milliseconds. These time periods are provided by way of example only, and it will be apparent that varying network conditions between client devices 108 and the proxy detector 116 may lead to a wide variety of other time periods. Notably, it can depend on the type of network used by the client device. It is why the method may comprise a test to determine the type of network of the client device.

[0061] In any case, it is expected that non-proxied client requests exhibit smaller differences between the first and second time periods than proxied client requests.

[0062] The score determined at block 335, therefore, assesses whether a difference between the first and second time periods indicates that the client device 108 with which the transport-layer connection is established is operating as a proxy for the client endpoint with which the secure link is established.

[0063] A wide variety of mechanisms for determining the score at block 335 are contemplated. For example, returning to FIG. 5, the proxy detector can determine a difference between the time periods 408 and 412 (e.g., fifteen milliseconds, in this example), and normalize that difference to a predefined range, based on a configurable maximum difference. For instance, the proxy detector can normalize the difference of fifteen milliseconds to a range between zero and one, with one representing a difference of three hundred milliseconds or more (e.g., the score can be capped at a value of one). In the illustrated example, therefore, the proxy detector 116 generates a score 416 of 0.05 (i.e., 15 ms / 300).

[0064] In other examples, the score can be the difference itself, without normalization. In further examples, the score can be generated by determining the sum of the two time periods, and/or by normalizing the sum according to a predefined range. Various other mechanisms will also occur to those skilled in the art for generating the score. Any

mechanism selected for generating the score at block 335 reflects the fact that when the transport-layer connection is established with a client device 108 that is also the client endpoint for the secure link subsequently established over the transport-layer connection, the separation between first and second time periods is expected to be relatively small.

5 In contrast, when the transport-layer connection is established with a client device 108 that is not the client endpoint, the separation between the first and second time periods is expected to be greater. Thus, the score-generation mechanism is selected to produce higher (or lower) scores for greater differences between time periods, and lower (or higher) scores for smaller differences between time periods.

10 **[0065]** Following generation of the score at block 335, the proxy detector 116 can select a handling action for the client request 112, and/or for subsequent client requests 112 using the same secure link. For example, at block 340, the proxy detector 116 can be configured to compare the score to a threshold. In examples in which higher scores indicate higher likelihoods of proxying, therefore, the proxy detector 116 can determine
15 whether the score exceeds a previously defined threshold. When the determination is affirmative, indicating that the relevant client device 108 is likely operating as a proxy, the proxy detector 116 can discard subsequent requests over the secure link at block 345, block / terminate the secure link previously established, or the like.

[0066] When the determination at block 340 is negative, the proxy detector 116 can
20 forward any client requests received over the secure link to the auxiliary detector 110 and/or request handler 104, along with the score, at block 350. In some examples, blocks 340 and 345 are omitted, and the proxy detector 116 simply forwards the score and request(s) to the auxiliary detector 110. The auxiliary detector 110 can be configured to determine whether the request(s) are likely to have been generated by a bot, based at
25 least in part on the score.

[0067] For example, the threshold to which the score is compared at block 340 can be a function of various network conditions between the client device 108 and the proxy detector 116. But as an alternative it is also possible that another function can be applied to the score depending on such various network conditions.

[0068] If the method comprises a test to determine the type of network of the client device, then for selecting a handling action for future requests over the transport-layer connection based on the result of this test, the threshold is a function of the result of this test. For example, if the test is positive then according to the function, the value of the threshold can be increased by a certain percentage.

[0069] This test can consist in determining whether the client device is on a mobile network. To perform this test, it can be checked if the first request to establish a transport-layer connection between the client device and the server is coming from an IP address which is in a list of IP addresses associated with a mobile network provider.

[0070] Turning to FIG. 5, another example performance of the method 300 is illustrated, to contrast with the performance shown in FIG. 4, in which the client device 108-1 itself is both the nearest transport-layer device and the client endpoint for secure communications. In FIG. 5, on the other hand, the client device 108-3 acts as a proxy for the client device 108-4.

[0071] Prior to receipt of a request at the proxy detector 116 at block 305, the client device 108-4 initiates a transport-layer connection 500 with the client device 108-3, e.g., via a three-way handshake sequence implemented via the messages 500a (e.g., a SYN message), 500b (e.g., a SYN-ACK message), and 500c (e.g., an ACK message). Either after establishment of the connection 500, or (as illustrated) contemporaneously with establishment of the connection 500, the client device 108-1 initiates a transport-layer connection 504 with the proxy detector 116. Specifically, at block 305 the proxy detector receives a message 504a (e.g., a SYN message). At block 310, via the messages 504b and 504c, the proxy detector 116 and the client device 108-3 complete the establishment of the connection 504. At block 315, the proxy detector 116 determines a first time period 512 associated with the transport-layer connection 504, such as the RTT between transmission of the message 504b and receipt of the message 504c.

[0072] Once the connections 500 and 504 are established, the client device 108-4 can request establishment of a secure link 508 over the connections 500 and 504. Of

particular note, the secure link 508 tunnels through the client device 108-3, and therefore cannot be initiated by the client device 108-3 itself. As a proxy, the client device 108-3 is configured only to route encrypted communications between the client device 108-4 and the proxy detector 116, using the connections 500 and 504 (but without accessing the contents of such communications).

[0073] At block 320, therefore, the proxy detector 116 can receive a request 508a (e.g., a Client Hello message) from the client device 108-3. The request 508a was originated at the client device 108-4, although that fact is not visible to the proxy detector 116. The client device 108-3 may acknowledge the message 508a to the client device 108-4 via a message 508b.

[0074] At block 325, the proxy detector 116 is configured to initiate or continue the relevant handshake sequence to establish the secure link 508. For example, as noted earlier, the proxy detector 116 can send an acknowledgement message 508c, which may be relayed to the client device 108-4 in some examples, but is not in the illustrated example. The proxy detector 116 can then send a message 508d, such as the previously mentioned Server Hello message, containing information necessary to establish the secure link 508 (e.g., supported cipher suites, and the like). The message 508d is relayed to the client device 108-4, and acknowledged via the an ACK message 508e by the client device 108-3. The message 508e, however, is not used by the proxy detector 116 to determine a time period 516 associated with the secure link 508, because the message 508e cannot be guaranteed to have originated at the client endpoint. The message 508e, that is, does not contain information that can only be generated or otherwise provided by the client endpoint of the secure link 508, and therefore may not (and in the illustrated example, does not) represent a true RTT between the proxy detector 116 and the client endpoint.

[0075] Once the message 508d is received at the client device 108-4, the client device 108-4 may send an acknowledgement 508f, which is not forwarded to the proxy detector 116 in this example, but can be forwarded in other examples. The client device 108-4 then sends a message 508g to complete the handshake sequence and establish the

secure link 508. The message 508g is analogous to the message 404e shown in FIG. 4. To determine the time period 516 at block 330, the proxy detector 116 determines the time elapsed between transmission of the message 508d, and reception of the message 508g. More generally, as noted earlier, the proxy detector 116 determines the time
5 elapsed between a message transmitted from the proxy detector 116 that necessarily terminates at the client endpoint for the secure connection 508, and a subsequent expected message that necessarily originates at the client endpoint.

[0076] As seen in FIG. 5, the need to relay messages between the client devices 108-3 and 108-4 increases the time elapsed to complete the secure link 508, while the time
10 required to complete the transport-layer connection 504 is unchanged relative to FIG. 4. That is, the presence of a proxy does not affect the connection 504, but lengthens the RTT associated with the secure link 508.

[0077] To determine a score at block 335, the proxy detector 116 can be configured, as in the example of FIG. 4, to determine the difference between the time periods 512
15 and 516, and to normalize that difference, e.g., against a maximum of three hundred milliseconds. The result, as shown in FIG. 5, assuming values of forty milliseconds and two hundred milliseconds for the time periods 512 and 516, is a score 520 of 0.53. That is, the score 520 is significantly higher than the score 416, indicating a greater likelihood that the client device 108-3 is operating as a proxy.

[0078] As will be apparent, therefore, the system 100 and specifically the proxy
20 detector 116 enables the detection of proxied client requests 112 in a manner sufficiently robust to detect residential proxies that may be challenging to detect using previous proxy-detection mechanisms, and in a manner that does not require the deployment of executable code to client devices, or any modification to the message flows between
25 client devices 108 and the proxy detector 116.

[0079] Specific example embodiments have been described above. Those skilled in the art, however, will understand that various modifications can be made to the above-examples, within the scope of above teachings. The scope of the claims below should

therefore not be limited by the specific embodiments set forth in the above examples, but should be given the broadest interpretation consistent with the description as a whole.

[0080] Certain expressions may be employed herein to list combinations of elements. Examples of such expressions include: “at least one of A, B, and C”; “one or more of A, B, and C”; “at least one of A, B, or C”; “one or more of A, B, or C”. Unless expressly indicated otherwise, the above expressions encompass any combination of A and/or B and/or C.

[0081] Those skilled in the art will further understand that in some embodiments, the functionality of the application 212 as described above may be implemented using pre-programmed hardware or firmware elements (e.g., application specific integrated circuits (ASICs), electrically erasable programmable read-only memories (EEPROMs), etc.), or other related components.

CLAIMS

1. A proxy detection method in a server, the method comprising:

in response to receiving, from a client device, a first request to establish a
5 transport-layer connection between the client device and the server, transmitting a first
message to the client device according to a first handshake sequence, for establishing
the transport-layer connection;

determining a first time period associated with completion of the first handshake
sequence;

10 in response to receiving, from the client device over the transport-layer connection,
a second request to establish a secure link between a client endpoint and the server,
transmitting a second message to the client endpoint according to a second handshake
sequence, for establishing the secure link;

determining a second time period associated with completion of the second
15 handshake sequence; and

generating, based on the first time period and the second time period, a score
indicating a likelihood that the client device is a proxy for the client endpoint.

2. The method of claim 1, wherein the transport-layer connection is based on the
20 Transport Control Protocol (TCP).

3. The method of claim 2, wherein the first message includes a SYN-ACK message; and
wherein the first time period is a time elapsed between transmission of the first message,
and receipt of an ACK message from the client device.

25 4. The method according to one of the claims 1 to 3, wherein the secure link is based on
one of (i) the Transport Layer Security (TLS) protocol, and (ii) the Secure Sockets Layer
(SSL) protocol.

5. The method of claim 4, wherein the second time period is a time elapsed between transmission of the second message, and receipt of a next message from the client endpoint according to the second predefined handshake sequence.

5 6. The method according to one of the claims 1 to 5, wherein generating the score includes determining a difference between the first and second time periods.

7. The method according to one of the claims 1 to 6, further comprising selecting a handling action for future requests over the transport-layer connection, based on the
10 score.

8. The method of claim 7, wherein the handling action includes discarding the future requests when the score exceeds a threshold.

15 9. The method according to claim 8, wherein the threshold is a function of network conditions between the client device and the proxy detector.

10. A server, comprising:

a communications interface; and

20 a processor configured to:

in response to receiving, from a client device, a first request to establish a transport-layer connection between the client device and the server, transmit a first message to the client device according to a first handshake sequence, for establishing the transport-layer connection;

25 determine a first time period associated with completion of the first handshake sequence;

in response to receiving, from the client device over the transport-layer connection, a second request to establish a secure link between a client endpoint and the server, transmit a second message to the client endpoint according to a
30 second handshake sequence, for establishing the secure link;

determine a second time period associated with completion of the second handshake sequence; and

generate, based on the first time period and the second time period, a score indicating a likelihood that the client device is a proxy for the client endpoint.

5

11. The server of claim 10, wherein the transport-layer connection is based on the Transport Control Protocol (TCP).

10

12. The server of claim 11, wherein the first message includes a SYN-ACK message; and wherein the first time period is a time elapsed between transmission of the first message, and receipt of an ACK message from the client device.

15

13. The server according to one of the claims 10 to 12, wherein the secure link is based on one of (i) the Transport Layer Security (TLS) protocol, and (ii) the Secure Sockets Layer (SSL) protocol.

20

14. The server of claim 13, wherein the second time period is a time elapsed between transmission of the second message, and receipt of a next message from the client endpoint according to the second predefined handshake sequence.

15. The server according to one of the claims 10 to 14, wherein the processor is configured, to generate the score, to determine a difference between the first and second time periods.

1 / 5

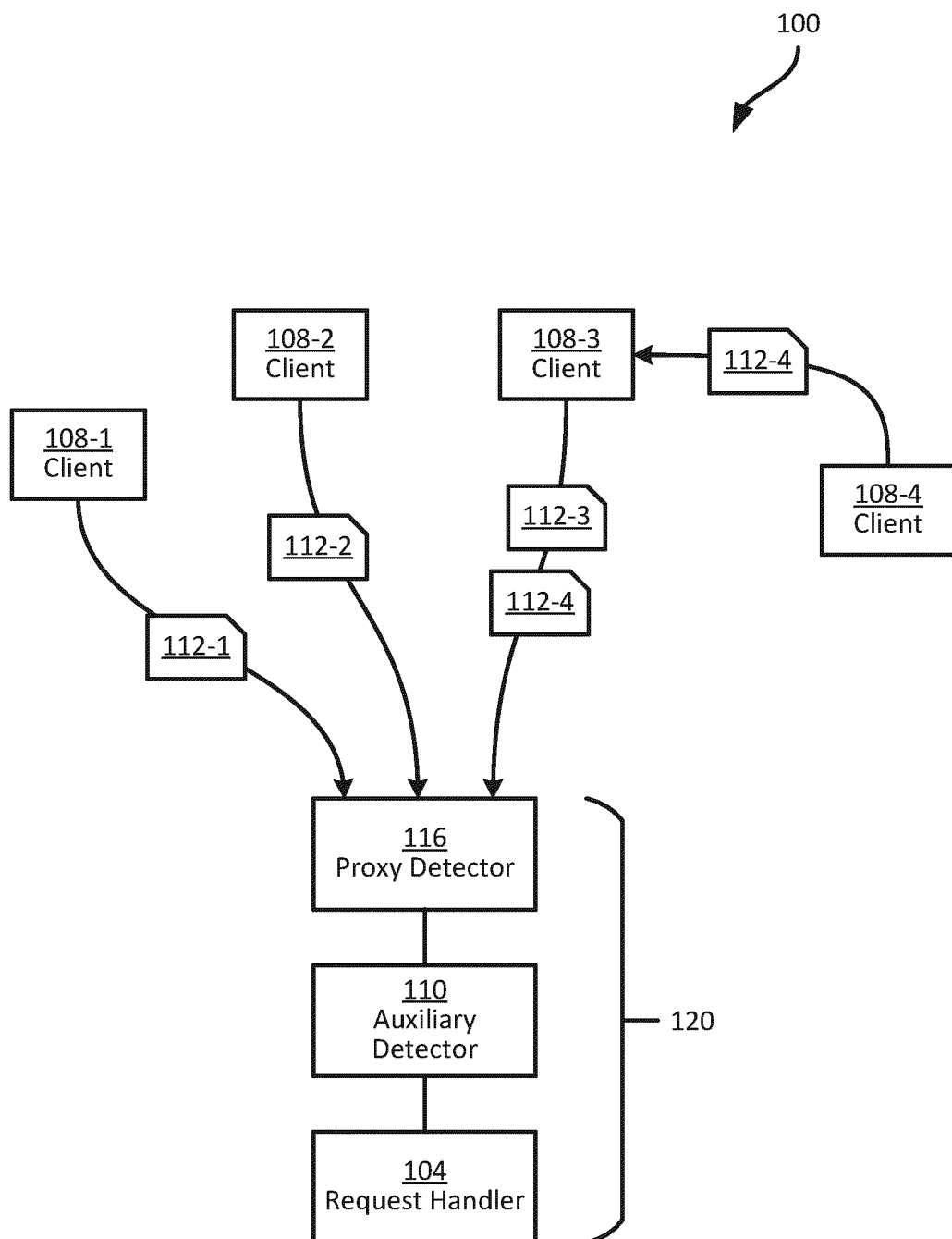


FIG. 1

2 / 5

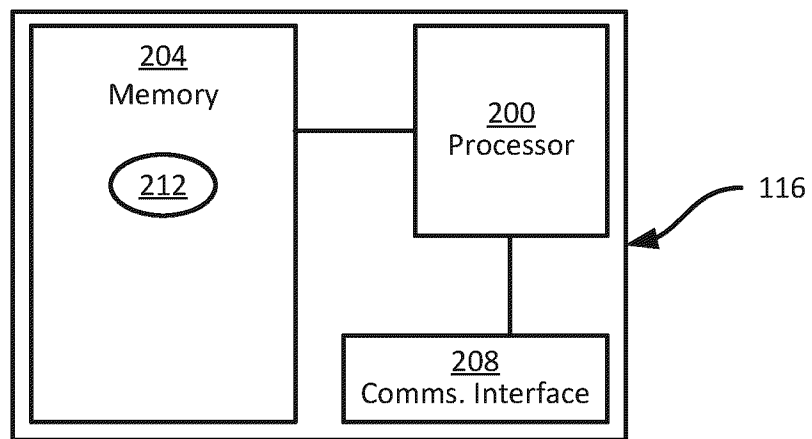


FIG. 2

3 / 5

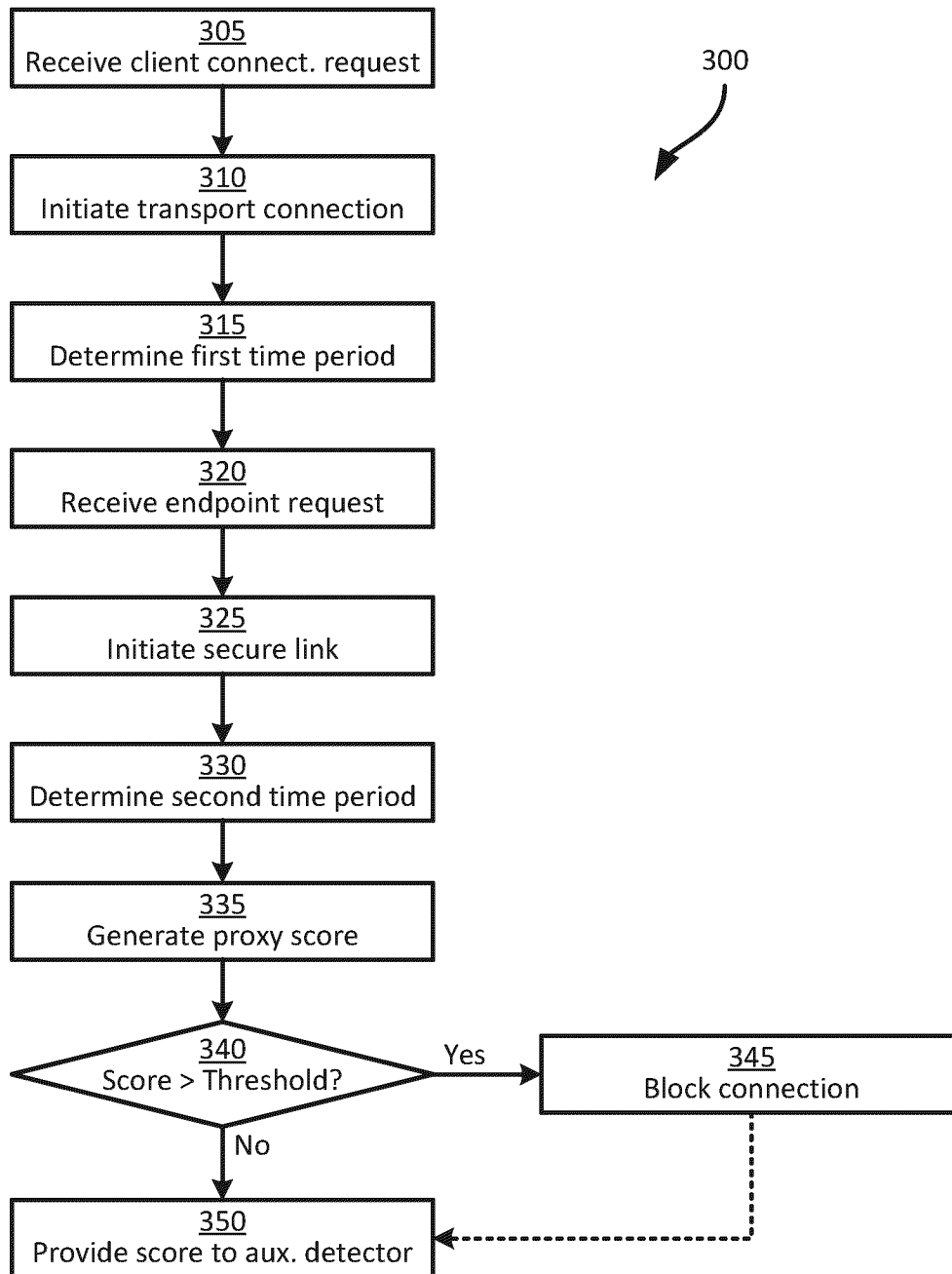


FIG. 3

4 / 5

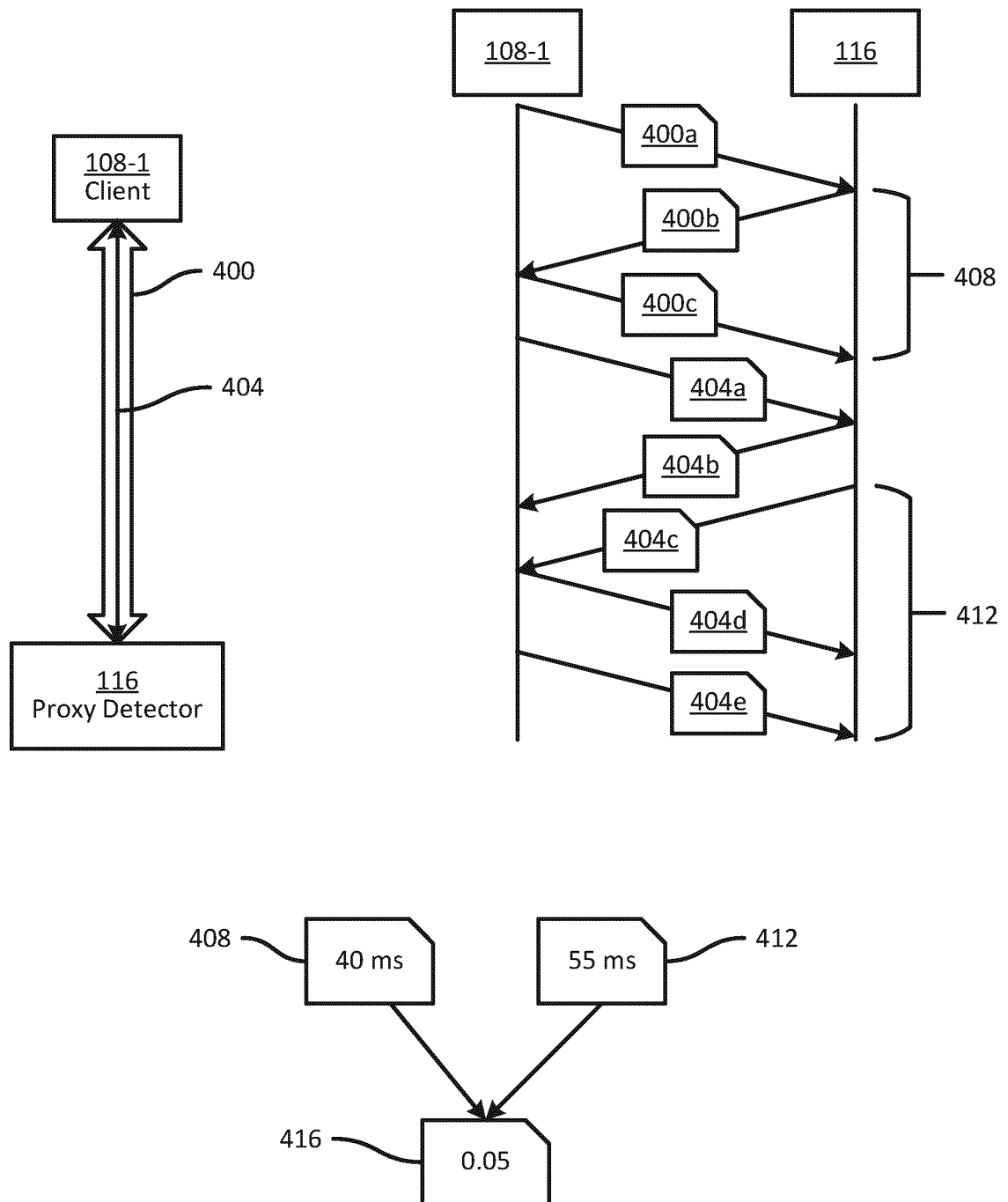


FIG. 4

5 / 5

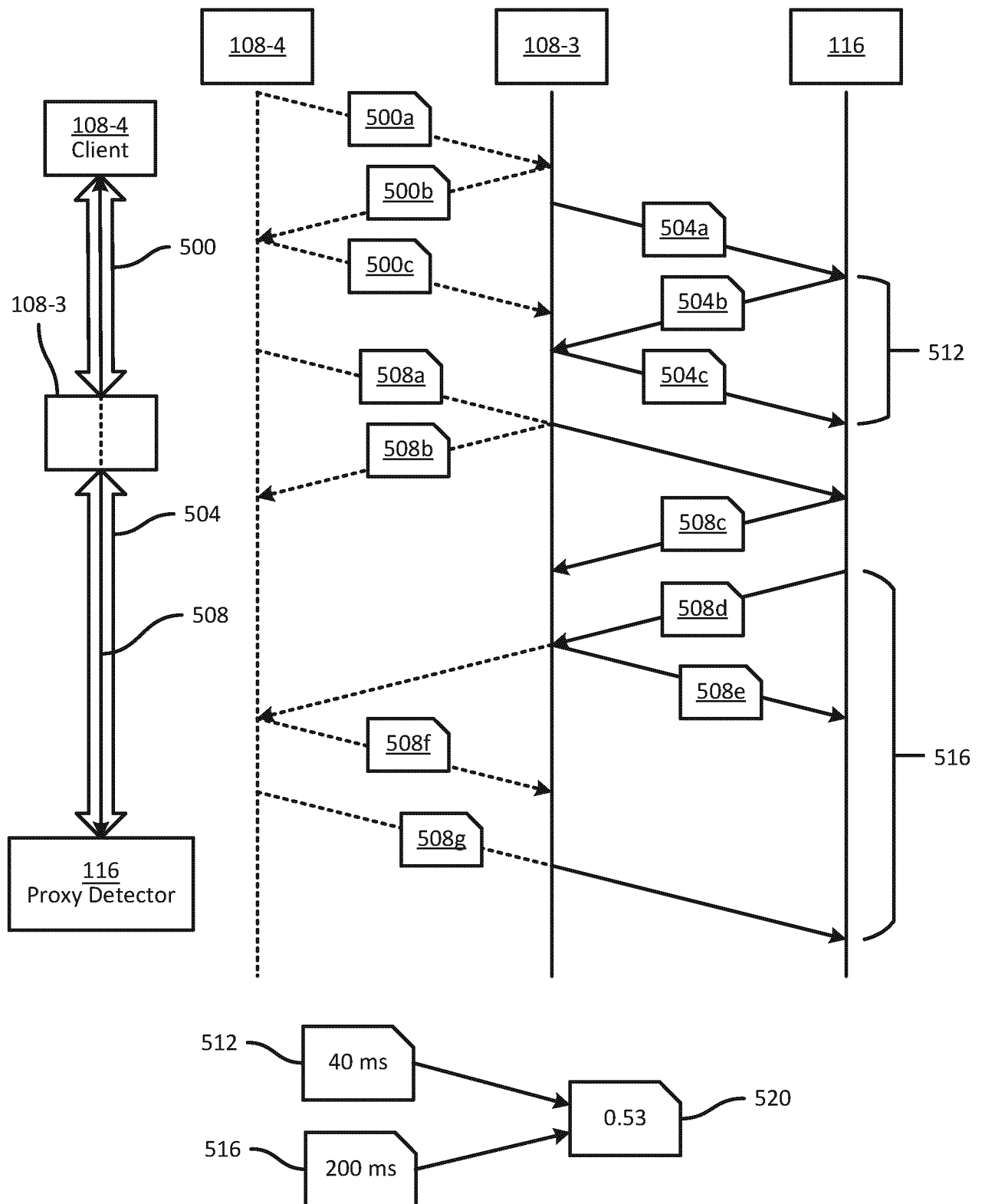


FIG. 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2023/059477

A. CLASSIFICATION OF SUBJECT MATTER

INV. H04L9/40 H04L69/28
ADD. H04L69/16

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, COMPENDEX, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	<p>Cao Zechun: "INTRUSION DETECTION: DETECTING NETWORK INTRUDERS BEHIND ANONYMITY NETWORKS AND IDENTIFYING INTRUDERS ON THE HOSTS BY MODELING USER BEHAVIORS",</p> <p>, 1 December 2020 (2020-12-01), pages 1-109, XP093053753,</p> <p>Retrieved from the Internet:</p> <p>URL:https://uh-ir.tdl.org/bitstream/handle/10657/8018/CAO-DISSERTATION-2020.pdf?sequence=1</p> <p>[retrieved on 2023-06-13]</p> <p>page 24 - page 42</p> <p style="text-align: center;">-----</p> <p style="text-align: center;">-/--</p>	1-15



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

13 June 2023

Date of mailing of the international search report

20/06/2023

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Riegler, Jörg

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2023/059477

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	<p>Hoogstraaten J.M. (hans): "Evaluating server-side internet proxy detection methods",</p> <p>,</p> <p>10 November 2018 (2018-11-10), pages 1-80, XP093053762,</p> <p>Retrieved from the Internet:</p> <p>URL:https://filelist.tudelft.nl/TBM/Over%20faculteit/Afdelingen/Engineering%20Systems%20and%20Services/People/Professors%20emeriti/Jan%20van%20den%20Berg/MasterPhdThesis/Internet-proxy-detection_v1.1-final_Hans%20Hoogstraaten.pdf</p> <p>[retrieved on 2023-06-13]</p> <p>page 33 - page 44</p> <p>-----</p>	1-15
A	<p>US 2017/012988 A1 (TURGEMAN AVI [US] ET AL) 12 January 2017 (2017-01-12)</p> <p>figures 2A, 2B</p> <p>paragraph [0006] - paragraph [0007]</p> <p>paragraph [0034] - paragraph [0041]</p> <p>paragraph [0088] - paragraph [0092]</p> <p>claims 1-5</p> <p>-----</p>	1-15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2023/059477

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2017012988 A1	12-01-2017	US 2017012988 A1	12-01-2017
		US 2018351959 A1	06-12-2018
		US 2020076816 A1	05-03-2020
		US 2021014236 A1	14-01-2021
