(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
04 July 2024 (04.07.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/145367 A1

(51) International Patent Classification:
H04L 9/08 (2006.01) *H04L 9/40* (2022.01)
H04L 12/46 (2006.01) *H04L 45/24* (2022.01)

(21) International Application Number:
PCT/US2023/086050

(22) International Filing Date:
27 December 2023 (27.12.2023)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
18/147,369 28 December 2022 (28.12.2022) US

(71) Applicant: **CISCO TECHNOLOGY, INC.** [US/US]; 170 West Tasman Drive, San Jose, CA 95134-1706 (US).

(72) Inventors: **SHARMA, Govind, Prasad**; 35975 Nickel Street, Union City, CA 94587 (US). **KAILAS, Sivakumar**; 7882 Kennard Lane, San Ramon, CA 94582 (US). **BALAKANNAN, Prabhu**; 1257 Burdett Way, Milpitas, CA 95035 (US).

(74) Agent: **SHU, Haining** et al.; Lee & Hayes, P.C., 601 W. Riverside Ave, Suite 1400, Spokane, WA 99201 (US).

(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,

(54) Title: HIGH BANDWIDTH ENCRYPTION ENGINES IN A MULTIPATHING IP NETWORK

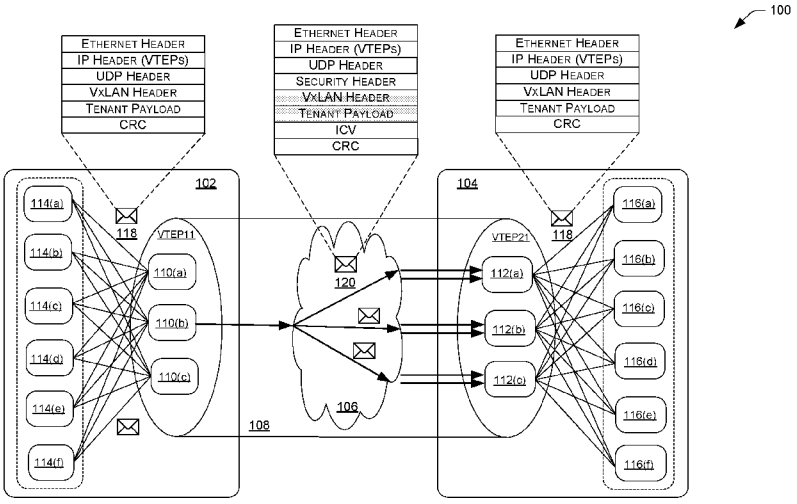

FIG. 1

(57) Abstract: Techniques for generating a per-packet initialization vector for high bandwidth encryption engines in a multipathing IP network are described herein. In examples, a network switch of a first datacenter site may receive a data packet to be sent to a second datacenter site over a network. The data packet may be encrypted according to a virtual extensible LAN (VxLAN) protocol and to be transmitted in a VxLAN tunnel created for the first datacenter site and the second datacenter site. An encryption engine implemented at the network switch may generate an initialization vector (IV) for the data packet based on a packet number (PN) associated with the data packet. The encryption engine may use the IV and information associated with a security association (SA) assigned to the packet to encrypt the data packet. In some examples, a full 64-bit PN may be used to compute the IV for the data packet.

WO 2024/145367 A1

[Continued on next page]

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).

Published:

— *with international search report (Art. 21(3))*

HIGH BANDWIDTH ENCRYPTION ENGINES IN A MULTIPATHING IP NETWORK

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Patent Application No. 18/147,369, filed on December 28, 2022, the entire contents of which are incorporated herein by reference.

TECHNICAL FIELD

[0002] The present disclosure relates generally to encryption and decryption of the data packets transmitted in an inter-site tunnel in a cloud deployed datacenter network.

BACKGROUND

[0003] Traditional datacenter networks have evolved towards CLOS based network designs, which allow the datacenter networks to scale and grow incrementally on demand. A virtualized datacenter site nowadays may comprise one or more performance optimized data centers (PODs). Each POD comprises a large number of switches (e.g., approximate 500-1000 leaf switches and 8-16 spine switches) that serve the virtual networks of a tenant on a shared physical network infrastructure. To address the IP mobility, disaster recovery, scale and redundancy concerns in the datacenter network, the datacenter sites may use a virtual extensible LAN (VxLAN) tunnel to provide a high bandwidth for traffic between two sites. The traffic that traverses in the VxLAN tunnel may be encrypted for data security and integrity.

[0004] Existing encrypting solutions that use GCM-AES-XPN-128 and GCM-AES-XPN-256 may carry 32 lowest significant bits of a packet number (PN) in the encrypted data packet, even though the encryption and decryption engines, according to the standards, use a 64-bit PN number on the transmitter site and the receiver site, respectively. As the uplink bandwidth of the VxLAN tunnel is very high, the 32-bit PN used for generating the per-packet unique initialization vector (IV) for encryption may exhaust quickly, requiring a security association key (SAK) rekeying, redistribution, and re-deployment across multiple datacenter sites in very short time intervals. To overcome the fast PN exhaustion in the data plane for the high speed uplinks, the control plane may exchange the metadata in very short intervals across the multiple datacenter sites. However, it is challenging for a software based control plane to perform the very short interval rekey and key distribution. In addition, existing 64-bit PN recovery mechanisms that carry the 32 lowest significant bits of the PN in the security header is not applicable

to multipathing IP networks. In a multipathing IP network, there are multiple upstream encrypting devices transmitting packets to multiple downstream decrypting devices for decryption in the VxLAN tunnel between the sites. The encrypted packets transmitted from the upstream site from one of the encrypting devices of the tunnel may land on any of the receiving decrypting devices on the downstream site, which may cause the 32 most significant bits of the PN maintained locally on the receivers become invalid. As a result, the data packet recovery may fail due to multipathing packet sprays inside the service provider IP networks connecting the VxLAN tunnel across sites.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The detailed description is set forth below with reference to the accompanying figures. In the figures, the left-most digit(s) of a reference number identifies the figure in which the reference number first appears. The use of the same reference numbers in different figures indicates similar or identical items. The systems depicted in the accompanying figures are not to scale and components within the figures may be depicted not to scale with each other.

[0006] FIG. 1 illustrates a system-architecture diagram of an example multipathing IP network data-plane that includes a tunnel, such as a VxLAN tunnel, created for two datacenter sites.

[0007] FIG. 2 illustrates a block diagram of an example encryption engine configured to encrypt the data packet transmitted in the example VxLAN tunnel.

[0008] FIG. 3 illustrates a block diagram of an example decryption engine configured to decrypt the data packet transmitted in the example VxLAN tunnel.

[0009] FIGS. 4A and 4B illustrate a flow diagram of an example method for encrypting and decrypting the data packet transmitted in the example VxLAN tunnel.

[0010] FIG. 5 illustrates a computer architecture diagram showing an illustrative computer hardware architecture for implementing a network device that can be utilized to implement aspects of the various technologies presented herein.

DESCRIPTION OF EXAMPLE EMBODIMENTS

OVERVIEW

[0011] Aspects of the invention are set out in the independent claims and preferred features are set out in the dependent claims. Features of one aspect may be applied to each aspect alone or in combination with other features.

[0012] This disclosure describes techniques for generating a per-packet unique initialization vector for high bandwidth encryption engines in a multipathing IP network. In examples, a tunnel, such as a VxLAN tunnel, may be created for a pair of datacenter sites for traversing the inter-site traffic. The endpoints of the VxLAN tunnel may include one or more spine switches of the datacenter sites. A transmitter endpoint of the VxLAN tunnel may receive an Ethernet frame from one of a plurality of leaf switches of the corresponding datacenter site and encapsulate the Ethernet frame to a VxLAN encapsulated packet. In examples, a security association (SA) may be generated for the VxLAN encapsulated packet to be transmitted in the VxLAN tunnel. The SA may indicate information used for encryption and decryption, such as a next packet number (PN), a security association key (SAK), a secure channel identifier (SCI), an associate number (AN), etc.

[0013] In examples, an encryption engine may be implemented by a spine switch of a datacenter site. Alternatively, or additionally, the encryption engine may be implemented by a spine switch corresponding to an endpoint of the VxLAN tunnel. The encryption engine may use a full 64-bit next PN to generate a per-packet unique initialization vector (IV) to encrypt the VxLAN encapsulated packet. The IV together with the SAK and SCI may be further used to encrypt the user data in the VxLAN encapsulated packet to obtain encrypted user data. The user data of the VxLAN encapsulated packet may include a tenant payload and a VxLAN header. In examples, the IV may be used to compute an integrity checksum value (ICV) for packet integrity check. The encryption engine may further construct a security header to include information related to encryption and transmission of the VxLAN encapsulated packet in the VxLAN tunnel, such as the next PN, the SCI and the AN. In examples, the encryption engine may further replace the user data of the VxLAN encapsulated packet with the encrypted user data and insert the ICV and the security header into the VxLAN encapsulated packet to generate an encrypted VxLAN encapsulated packet to be transmitted in the VxLAN tunnel.

[0014] Additionally, the techniques described herein may be performed as a method and/or by a system having non-transitory computer-readable media storing computer-executable instructions that, when executed by one or more processors, performs the techniques described above.

EXAMPLE EMBODIMENTS

[0015] As described above, in a multi-site datacenter network, a data packet is encrypted for security and integrity before it is transmitted to a VxLAN tunnel between two datacenter sites. Due to the high speed uplinks of the VxLAN tunnel, existing encryption schemes may exhaust the PN bits when 32 bits of the next PN are used for generating the per-packet unique initialization vector (IV) for encryption. It is also challenging for the software based control plane to offset the shortage of using 32 bits of the next PN for the per-packet unique IV. Accordingly, this disclosure describes a data plane approach that uses a full 64-bit PN, as an example, to generate the per-packet IV for encrypting the data packet to be transmitted in the VxLAN tunnel.

[0016] In examples, a VxLAN tunnel may be created for a pair of datacenter sites for traversing the inter-site traffic. The network switches in the datacenter site may be organized into two or more stages, including at least the lower level stage switches (also known as leaf switches) and the higher level stage switches (also known as spine switches). Each endpoint of the VxLAN tunnel may include one or more spine switches of the datacenter site. In examples, a spine switch of a transmitter endpoint of the VxLAN tunnel may receive an Ethernet frame from a leaf switch of the corresponding datacenter site and encapsulate the Ethernet frame into a VxLAN encapsulated packet. In examples, the VxLAN encapsulated packet may include at least an Ethernet header, an IP header (VTEPs) indicating IP addresses of the VxLAN tunnel endpoints, a user datagram protocol (UDP) header, a VxLAN Header, a tenant payload, and a cyclic redundancy check (CRC).

[0017] In examples, a security association (SA) may be generated for the VxLAN encapsulated packet with a set lifetime. The SA may include the information used for encrypting and decrypting the data packet, including but not limited to, a next packet number (PN), a security association key (SAK), a secure channel identifier (SCI), an associate number (AN). The SA may be allocated and distributed to both the transmitter site and the receiver site. Information of the SA may be stored locally at the transmitter site and the receiver site for the encryption and decryption.

[0018] In examples, when an encryption engine receives a VxLAN encapsulated packet, the encryption engine may retrieve a destination IP (DIP) address from the IP header (VTEPs) of the VxLAN encapsulated packet. The encryption engine may use the DIP to search the local database at the transmitter site to obtain information of the SA assigned for the VxLAN encapsulated packet. In examples, the encryption engine may

retrieve the next PN from the SA and compute the IV for the VxLAN encapsulated packet based at least in part the next PN.

[0019] The encryption engine may further use the IV together with the security association key (SAK) and the secure channel identifier (SCI) to encrypt the user data of the VxLAN encapsulated packet and generate encrypted user data. In examples, the user data of the VxLAN encapsulated packet may include the VxLAN header and the tenant payload.

[0020] In examples, the encryption engine may also generate an integrity checksum value (ICV) using the IV for integrity check purpose. In a further example, the encryption engine may further construct a security header that includes the information for encryption and decryption, such as the next PN associate with the VxLAN encapsulated packet, the security channel identifier (SCI) and the association number (AN). The encryption engine may then replace the user data of the VxLAN encapsulated packet with the encrypted user data and insert the ICV and the security header into the VxLAN encapsulated packet to generate an encrypted VxLAN encapsulated packet. The encrypted VxLAN encapsulated packet may be further transmitted to another datacenter site in the VxLAN tunnel.

[0021] In a further example, once the encrypted VxLAN encapsulated packet is sent, the next PN may be incremented by one in the transmitter site SA for further use.

[0022] In examples, the encryption engine may be implemented at a spine switch of the datacenter site. Alternatively, or additionally, the encryption engine may be implemented at a spine switch corresponding to an endpoint of the VxLAN tunnel. In examples, the encryption engine may use a full 64-bit next PN to compute the unique initialization vector (IV) for the VxLAN encapsulated packet and to encrypt the user data in the VxLAN encapsulated packet.

[0023] Comparing to the existing encryption techniques, in which 32 lowest significant bits of the next PN are used for the per-packet unique IV, this disclosure uses a 64-bit based per-packet unique IV during the lifetime of the SA for confidentiality and integrity protection and does not suffer from the extra-short interval rekey requirements on the PN exhaustion.

[0024] In examples, when a decryption engine receives an encrypted VxLAN encapsulated packet from the VxLAN tunnel, the decryption engine may retrieve a source IP (SIP) address from the IP header (VTEPs). The decryption engine may further obtain the information used for encryption from the security header, such as the SCI and the AN.

The decryption engine may then use the SCI and the AN together with the SIP address to search the local database to obtain the information of SA locally stored at the receiver site. In examples, the decryption engine may retrieve the copy of a security association key (SAK) from the SA stored at the receiver site.

[0025] As described herein, the security header of the encrypted VxLAN encapsulated packet carries the 64-bit next PN as-is, and the decrypt engine may retrieve the next PN directly from the security header of the encrypted VxLAN encapsulated packet and use it as is for the per-packet unique IV. The decrypt engine may regenerate the integrity checksum value (ICV) using the next PN carried by the security header, i.e., the per-packet unique IV, the SCI, the SAK, and the encrypted user data.

[0026] In examples, the decrypt engine may validate the integrity of the encrypted VxLAN encapsulated packet based on the regenerated ICV. Once the ICV validation passes, the decrypt engine may decrypt the encrypted user data in the encrypted VxLAN encapsulated packet using the next PN, the security association key (SAK) and the SCI. The decrypt engine may further remove the security header and the ICV from the encrypted VxLAN encapsulated packet to recover the original VxLAN encapsulated packet. The original VxLAN encapsulated packet may then be transmitted to the downstream device, such as a leaf switch in the datacenter site.

[0027] In a further example, once the original VxLAN encapsulated packet is successfully recovered and sent to the downstream device, the next PN in the receiver site SA may be set according to the next PN carried by the security header for future use.

[0028] Certain implementations and embodiments of the disclosure will now be described more fully below with reference to the accompanying figures, in which various aspects are shown. However, the various aspects may be implemented in many different forms and should not be construed as limited to the implementations set forth herein. The disclosure encompasses variations of the embodiments, as described herein. Like numbers refer to like elements throughout.

[0029] FIG. 1 illustrates a system-architecture diagram of an example multipathing IP network data-plane 100, which includes a tunnel to transmit the TCP or UPD packets that are encrypted using the cryptographic machinery of IEEE MACsec. In examples, the tunnel may include a VxLAN tunnel, iVxLAN tunnel, or VxLAN-GPE tunnel, etc. As shown in FIG. 1, a VxLAN tunnel 108 is created to transmit traffic between a datacenter site 102 and a datacenter site 104 over a network 106.

[0030] In examples, the datacenter site 102 and the datacenter site 104 may refer to any virtualized datacenter site in a datacenter network. The datacenter network may generally implement CLOS based network designs in which the switches in a datacenter site may be organized into two or more stages. In a two-stage design, the lower level stage switches (i.e., leaf switches) may provide the network connectivity to the end hosts (e.g., endpoints or virtual machines) and implement the layer-2 bridging and the layer-3 routing functions. The higher level stage switches (i.e., spine switches) may provide the redundant paths and the network connectivity to a previous lower level stage switch in the network fabric.

[0031] As shown, the datacenter site 102 may comprise a plurality of leaf switches 114(a)-114(f) that connect to the endpoint devices (not shown in FIG. 1) and provide the layer-2 bridging and the layer-3 routing to the endpoint devices. The datacenter site 102 may further comprise a plurality of spine switches 110(a)-110(c) that connect to the leaf switches 114(a)-114(f) in the datacenter site 102 and provide the multipath communications to another site (e.g., the datacenter site 104) in the datacenter network. Similarly, the datacenter site 104 may comprise a plurality of leaf switches 116(a)-116(f) and a plurality of spine switches 112(a)-112(c).

[0032] In examples, the datacenter site 102 and the datacenter site 104 may be connected using data center interconnect (DCI) strategies, such as a virtual extensible LAN (VxLAN) that uses a site-to-site VxLAN overlay IP tunnel over the Internet or WAN to send the data traffic between the two sites. The endpoints of the VxLAN tunnel may comprise one or more spine switches that share an IP address in the uplinks. For instance, one endpoint of the VxLAN tunnel 108 comprises the spine switches 110(a)-110(c) that share the IP address VTEP11 in the uplinks while another endpoint of the VxLAN tunnel 108 comprises the spine switches 112(a)-112(c) that share the IP address VTEP 21 in the uplinks. The IP traffic may flow inside the VxLAN tunnel 108, egressing from an uplink of one of the spine switches 110(a)-110(c) on the datacenter site 102, and ingressing to any uplinks of the spine switches 112(a)-112(c) on the datacenter site 104.

[0033] In examples, a layer-2 frame from a leaf switch may be encapsulated according to the VxLAN protocol at a spine switch to generate a layer-3 packet, e.g., the VxLAN encapsulated packet 118. As shown, the VxLAN encapsulated packet 118 may include a tenant payload (e.g., the payload of the layer-2 frame) encapsulated in a VxLAN header. The tenant payload together with the VxLAN header may form a user datagram

protocol (UDP) payload. A UDP header may be added to the UDP payload to form a UDP-IP packet (i.e., the layer-3 packet). The UDP-IP packet may be further attached to an IP header (VTEPs) and an Ethernet header. In examples, the IP header (VTEPs) may indicate the information related to the VxLAN tunnel end points. In a further example, the VxLAN encapsulated packet 118 may further include a cyclic redundancy check (CRC) originally carried by the layer-2 frame.

[0034] In examples, to mitigate the security threats on data confidentiality and integrity for the inter-site traffic flows, various encryption schemes may be used to secure the traffic that traverses inside the VxLAN tunnel 108. The encryption schemes may generally utilize the allocation and distribution of the cryptography symmetric keys and set up of two unidirectional security associations (SAs) between the datacenter site 102 and the datacenter site 104 to secure the inter-site VxLAN overlay traffic. For instance, at the datacenter site 102 (Tx site), an encryption engine may encrypt the VxLAN encapsulated packet 118 to generate an encrypted VxLAN encapsulated packet 120 to be sent to the datacenter site 104 (Rx site) in the VxLAN tunnel 108. In examples, the encryption engine may be implemented at one of the spine switches 110(a)-110(c) at the Tx site of the VxLAN tunnel 108.

[0035] In examples, the encryption engine may encrypt the tenant payload together with the VxLAN header of the VxLAN encapsulated packet 108 using at least a security association key (SA) and a per-packet unique initialization vector (IV) to generate encrypted user data. In examples, the encrypt engine may further construct a security header to include the information for encryption and decryption. The security header may be included in the encrypted VxLAN encapsulated packet 120. In examples, the encrypted VxLAN encapsulated packet 120 may further include an integrity checksum value (ICV) to validate the integrity of the encrypted VxLAN encapsulated packet 120 received at the Rx site of the VxLAN tunnel 108. In examples, the encryption of the tenant payload and the VxLAN header, and the computing of the IV may use a full 64-bit next packet number (PN) associated with the VxLAN encapsulated packet 118.

[0036] In examples, at the Rx site of the VxLAN tunnel 108, a decryption engine may validate the integrity of the encrypted VxLAN encapsulated packet 120 based on the ICV. Upon determining that the integrity of the encrypted VxLAN encapsulated packet 120 is valid, the decryption engine may decrypt the encrypted VxLAN encapsulated packet

120 to recover the VxLAN encapsulated packet 108. In examples, the decryption engine may be implemented at the Rx site of the VxLAN tunnel 108.

[0037] In examples, the network 106 may facilitate communications of information or data between any datacenter cites, such as the datacenter site 102 and the datacenter site 104. In examples, the network 106 may include one or more networks implemented by any viable communication technology, such as wired and/or wireless modalities and/or technologies. The network(s) 106 may include any combination of Personal Area Networks (PANs), Local Area Networks (LANs), Campus Area Networks (CANs), Metropolitan Area Networks (MANs), extranets, intranets, the Internet, short-range wireless communication networks (e.g., ZigBee, Bluetooth, etc.) Wide Area Networks (WANs)—both centralized and/or distributed—and/or any combination, permutation, and/or aggregation thereof.

[0038] FIG. 2 illustrates a block diagram of an example encryption engine configured to encrypt the data packet transmitted in the example VxLAN tunnel.

[0039] In examples, the encryption engine 200 may include encrypt logic 202 communicatively coupled to a TxSA database 204. As described herein, when the VxLAN tunnel 108 is established between the pair of datacenter sites (e.g., datacenter site 102 and datacenter site 104), a security association (SA) between the pair of the datacenter sites may be set up for a lifetime period. Information of the SA may be stored locally on the pair of the datacenter sites. For instance, the TxSA database 204, as a Tx site database, may store a copy of the SA for encrypting the data packets to be transmitted in the VxLAN tunnel 108.

[0040] In examples, the SA may define one or more parameters for encrypting and decrypting the data packets transmitted between the datacenter site 102 and the datacenter site 104 in the VxLAN tunnel 108. By way of example and without limitation, the one or more parameters of the SA may include a security association key (SAK), a next packet number (PN), a secure channel identifier (SCI), an association number (AN), etc. The SAK may include the attributes, such as the cryptographic algorithm or the traffic encryption key that is used to secure the traffic in the VxLAN tunnel 108. The next PN may be a sequence number assigned by the transmission site (e.g., the datacenter site 102) to each data packet to be sent to the receiver site (e.g., the datacenter site 104) in the VxLAN tunnel 108. The SCI may be a unidirectional secure identifier that is used to transmit the secure data between the datacenter site 102 and the datacenter site 104. The

AN may be an identifier of the SA within the VxLAN tunnel 108. In examples, a combination of the SCI and the AN may identify the SAK used between the datacenter site 102 and the datacenter site 104.

[0041] In examples, when the VxLAN encapsulated packet 118 is received at an endpoint of the VxLAN tunnel 108, e.g., one of the spine switches 112(a)-112(c), the encrypt logic 202 may retrieve a destination IP (DIP) address 206 from the IP header (VTEPs) of the VxLAN encapsulated packet 118. The encrypt logic 202 may search the TxSA database 204 using the DIP 206 and obtain the information of the SA assigned to the VxLAN encapsulated packet 118, for instance, the SAK 208, the next packet number (PN) 210, and the SCI 212. The encrypt logic 202 may determine the per-packet unique initialization vector (IV) for the VxLAN encapsulated packet 108 based on the PN 210. The encrypt logic 202 may further encrypt the user data 214 (e.g., the tenant payload and the VxLAN header) based at least in part on the IV, the SAK 208, and the SCI 212 to generate the encrypted user data 216.

[0042] In examples, the encrypt logic 202 may generate an integrity checksum value (ICV) 218 based at least in part on the IV for checking the integrity of the packet after the packet is transmitted through the VxLAN tunnel. In examples, the ICV 218 may be generated using any cryptographic algorithms, including but not limited to, the message digital algorithm 5 (MD5), the secure hash algorithm (SHA), such as SHA-1, SHA-256, SHA-512, etc.

[0043] In examples, the encrypt logic 202 may further construct a security header 220 to include the information used for encryption and decryption, such as the next PN 210, the SCI 212 and the AN 222. As shown in FIG. 2, the security header 220 may include a tag control information (TCI) and AN field in 1 octet, a reserved (RSVD) field in 1 octet, a PN field in 8 octets, and an SCI field in 8 octets.

[0044] In examples, the encrypt logic 202 may replace the user data 214 in the VxLAN encapsulated packet 118 (e.g., the tenant payload and the VxLAN header) with the encrypted user data 216 and insert the security header 220 and the ICV 218 to the VxLAN encapsulated packet 118 to generate the encrypted VxLAN encapsulated packet 120. In examples, the security header 220 may be inserted between the UDP header and the VxLAN header of the VxLAN encapsulated packet 118 and the ICV 218 may be inserted between the tenant payload and the CRC of the VxLAN encapsulated packet 118.

[0045] FIG. 3 illustrates a block diagram of an example decryption engine configured to decrypt the data packet transmitted in the example VxLAN tunnel.

[0046] In examples, the decryption engine 300 may include decrypt logic 302 communicatively coupled to a RxSA database 304. As described herein, the information of the security association (SA) created for the communication between the datacenter sites (i.e., the datacenter site 102 and datacenter site 104) in the VxLAN tunnel 108 is not only stored in the transmission site TxSA database 204 but also in the receiver site RxSA database 304.

[0047] In examples, when the encrypted VxLAN encapsulated packet 120 is received at the receiver site (e.g., the datacenter site 104), the decrypt logic 302 may retrieve the next packet number (PN) 210 from the security header 220 of the encrypted VxLAN encapsulated packet 120 and use the next PN 210 as is for the per-packet unique IV. The decrypt logic 302 may retrieve a source IP (SIP) address 308 from the IP header (VTEPs) of the encrypted VxLAN encapsulated packet 120. The decrypt logic 302 may further retrieve the SCI 212 and AN 222 from the security header 220 of the encrypted VxLAN encapsulated packet 120. The decrypt logic 302 may then search the RxSA database 304 using the SIP 308, the SCI 212, and the AN 222 and obtain the security association key (SAK) 208 used for regenerating the ICV and the decryption. The decrypt logic 302 may then regenerate the ICV based at least in part on the IV (i.e., the next PN 210), the SAK 208, the SCI 212, and the encrypted user data 216. The decrypt logic 302 may compare the regenerated ICV with the ICV 218 carried by the encrypted VxLAN encapsulated packet 120. If the regenerated ICV does not match the ICV 218 carried by the encrypted VxLAN encapsulated packet 120, the decrypt logic 302 may determine that the integrity check of the encrypted VxLAN encapsulated packet 120 fails and drop the encrypted VxLAN encapsulated packet 120.

[0048] In examples, if the regenerated ICV matches the ICV 218 carried by the encrypted VxLAN encapsulated packet 120, the decrypt logic 302 may determine that the encrypted VxLAN encapsulated packet 120 passes the integrity check. The decrypt logic 302 may decrypt the encrypted user data 216 in the encrypted VxLAN encapsulated packet 120 using the IV (i.e., the next PN 210), the SAK 208, and the SCI 212 to recover the user data 214 (e.g., the tenant payload and the VxLAN header originally carried by the VxLAN encapsulated packet 118). In examples, the decrypt logic 320 may further remove the security header 220 and the ICV 218 from the encrypted VxLAN encapsulated packet 120

to recover the original VxLAN encapsulated data packet 118 and transmit the VxLAN encapsulated data packet 118 to the destination device.

[0049] FIGS. 4A and 4B illustrate a flow diagram of an example method for encrypting and decrypting the data packet transmitted in the example VxLAN tunnel.

[0050] Although FIGS. 4A and 4B illustrate these processes in a step-by-step order, and although these figures will be explained in a step-by-step order as well, any of the steps of these example processes may be performed independently of other steps, in any order, and/or in parallel with other steps. Additionally, steps may be omitted from the example processes and/or replaced with other steps described herein.

[0051] At block 402, the process begins by receiving, at a first device, a packet to be sent to a second device in an encrypted tunnel over a network. In examples, the first device and the second device may include one or more datacenter sites or virtualized datacenter sites in a cloud deployed datacenter network. The encrypted tunnel may implement a VxLAN protocol that facilitates the datacenter connectivity using tunneling to stretch the layer-2 connections over an underlying layer-3 network. Each of the first device and the second device may include a plurality of leaf switches and spine switches. In examples, the endpoints of the encrypted tunnel may include a plurality of spine switches that connect to the downstream leaf switches in their respective datacenter sites. In examples, a layer-2 frame may be encapsulated according to a VxLAN protocol to be transmitted in the VxLAN tunnel.

[0052] At block 404, the process includes obtaining, from a first database, a first security association (SA), information of the first SA including a next packet number (PN), a security association key (SAK), a security channel identifier (SCI), and an association number (AN). In examples, a security association (SA) may be created for the first device and the second device for transmitting traffic in the encrypted tunnel with a lifetime period. The information of the SA may be symmetrically stored at the local databases associated with the first device and the second device, respectively. For instance, the first SA may be a copy of the SA stored in the transmission site database. In examples, the first device may retrieve a destination IP (DIP) address from an IP header (VTEPs) of the VxLAN encapsulated packet and search the first database using the DIP address to obtain the first SA.

[0053] At block 406, the process includes generating, at the first device, a per-packet initialization vector (IV) based on the next packet number (PN). In examples, the per-

packet IV may be a unique random or pseudorandom number computed using the next PN. In a further example, the first device may use the full 64-bit next PN as is to generate the per-packet IV for the VxLAN encapsulated packet. As discussed herein, the next PN may indicate a next packet number maintained in the hardware of the first device.

[0054] At block 408, the process includes encrypting, at the first device, user data of the packet based at least in part on the IV, the SAK, and the SCI to generate the encrypted user data. In examples, the user data of the VxLAN encapsulated packet may generally include a tenant payload and a VxLAN header. The first device may encrypt the tenant payload together with the VxLAN header using the IV, the SAK, and the SCI to generate an encrypted user data.

[0055] At block 410, the process includes constructing, at the first device, a security header to include the next packet number (PN), the SCI and the AN. In examples, the security header may be constructed to leverage the cryptographic machinery of IEEE MACsec for the VxLAN encapsulated packet transmitted in the VxLAN tunnel. In a further example, the security header may include a complete sequence of the 64-bit PN instead of the 32 lowest significant bits of the next PN.

[0056] At block 412, the process includes generating, at the first device, a first integrity checksum value (ICV) based at least in part on the IV. In examples, the first ICV may be generated using any cryptographic algorithms, including but not limited to, the message digital algorithm 5 (MD5), the secure hash algorithm (SHA), such as SHA-1, SHA-256, SHA-512, etc.

[0057] At block 414, the process includes inserting, at the first device, the security header and the first ICV to the packet. In examples, the security header may be inserted between the UDP header and the VxLAN header of the VxLAN encapsulated packet. In a further example, the first ICV may be inserted between the CRC and the tenant payload of the VxLAN encapsulated packet.

[0058] At block 416, the process includes replacing the user data of the packet with the encrypted user data. As discussed herein, the tenant payload together with the VxLAN header are replaced with the encrypted tenant payload and the encrypted VxLAN header.

[0059] At block 418, the process includes transmitting, from the first device, the packet to the second device in the encrypted tunnel. As discussed herein, the packet may include the Ethernet header, the IP header (VTEPs), the UDP header, the security header, the encrypted VxLAN header, the encrypted tenant payload, the first ICV, and the CRC.

[0060] In further examples, the first device may increment the next PN value by one and store the next PN in the first SA for future use.

[0061] At block 420, the process includes receiving, at the second device, the packet from the first device sent in the encrypted tunnel. As discussed herein, the packet may be encapsulated according to the VxLAN protocol and further encrypted based at least in part on the per-packet unique IV and the SAK.

[0062] At block 422, the process includes retrieving, at the second device, the next packet number from the security header of the packet, the next packet number being used as is for IV. As described herein, the security header may carry the full 64-bit next PN as is. Thus, a full 64-bit next packet number may be directly retrieved from the security header of the encrypted VxLAN encapsulated packet and used as the per-packet unique IV.

[0063] At block 424, the process includes obtaining, from a second database, a second security association (SA), the second SA including the SAK. In examples, the encryption engine at the first device and the decryption engine at the second device may use a symmetric SAK for packet encryption and decryption. The second SA may include the symmetric SAK to decrypt the encrypted VxLAN encapsulated packet.

[0064] At block 426, the process includes generating, at the second device, a second ICV based on the IV, the SAK, the SCI, and the user data of the packet. In examples, the decryption engine at the second device may retrieve the SCI from the security header of the packet. In examples, the decryption engine at the second device may generate the second ICV based at least in part on the IV, e.g., the full 64-bit next packet number carried in the security header of the encrypted VxLAN encapsulated packet, the SCI, the SAK, and the user data of the encrypted VxLAN encapsulated packet.

[0065] At block 428, the process includes determining whether the second ICV corresponds to the first ICV carried in the received packet.

[0066] When the second ICV corresponds to the first ICV carried in the packet, at block 430, the process includes decrypting, at the second device, the user data of the packet based on the IV, the SAK, and the SCI. In examples, the decryption engine at the second device may retrieve the SAK from a second database (i.e., a local database) and retrieve the information of the next PN and the SCI from the security header and use the SAK, the next PN, and the SCI to decrypt the user data, e.g., the tenant payload and the VxLAN header.

[0067] At block 432, the process includes removing, at the second device, the security header and first ICV from the received packet. Once the integrity check passes and the user data is successfully decrypted, the security header and the first ICV may be removed from the encrypted VxLAN encapsulated packet, thus recovering the originally VxLAN encapsulated packet received at the first device.

[0068] At block 434, the process includes transmitting, from the second device, the packet to a destination device. In examples, the packet may be routed to the destination device based at least in part on the destination IP address and information included in the Ethernet header of the VxLAN encapsulated packet.

[0069] When the second ICV does not correspond to the first ICV carried in the packet, at block 436, the process includes discarding, at the second device, the packet. In examples, the process may further include performing the CRC check for the received packet. When the ICV and/or the CRC check fail, the received packet may be dropped by the second device.

[0070] In examples, the decryption engine may set the next PN at the second device according to the next PN in the security header of the encrypted VxLAN encapsulated packet and store the next PN in the second database for future use.

[0071] FIG. 5 is a computer architecture diagram showing an illustrative computer hardware architecture for implementing a network device that can be utilized to implement aspects of the various technologies presented herein. The computer architecture shown in FIG. 5 illustrates a conventional server computer, network device, router, switch, gateway, workstation, desktop computer, laptop, tablet, network appliance, e-reader, smartphone, or other computing device, and can be utilized to execute any of the software components presented herein. Additionally, or alternatively, the computer 500 may, in some examples, correspond to a switch of a virtualized datacenter site, such as the spine switch 110(a)-110(c) and 112(a)-112(c) and the leaf switches 114(a)-114(f) and 116(a)-116(f), as described herein.

[0072] The computer 500 includes a baseboard 502, or "motherboard," which is a printed circuit board to which a multitude of components or devices can be connected by way of a system bus or other electrical communication paths. In one illustrative configuration, one or more central processing units ("CPUs") 504 operate in conjunction with a chipset 506. The CPUs 504 can be standard programmable processors that perform arithmetic and logical operations necessary for the operation of the computer 500.

[0073] The CPUs 504 perform operations by transitioning from one discrete, physical state to the next through the manipulation of switching elements that differentiate between and change these states. Switching elements generally include electronic circuits that maintain one of two binary states, such as flip-flops, and electronic circuits that provide an output state based on the logical combination of the states of one or more other switching elements, such as logic gates. These basic switching elements can be combined to create more complex logic circuits, including registers, adders-subtractors, arithmetic logic units, floating-point units, and the like.

[0074] The chipset 506 provides an interface between the CPUs 504 and the remainder of the components and devices on the baseboard 502. The chipset 506 can provide an interface to a RAM 508, used as the main memory in the computer 500. The chipset 506 can further provide an interface to a computer-readable storage medium such as a read-only memory ("ROM") 510 or non-volatile RAM ("NVRAM") for storing basic routines that help to startup the computer 500 and to transfer information between the various components and devices. The ROM 510 or NVRAM can also store other software components necessary for the operation of the computer 500 in accordance with the configurations described herein.

[0075] The computer 500 can operate in a networked environment using logical connections to remote computing devices and computer systems through a network, such as the network 524. The chipset 506 can include functionality for providing network connectivity through a NIC 512, such as a gigabit Ethernet adapter. The NIC 512 is capable of connecting the computer 500 to other computing devices over the network 524, such as the destination device 526. It should be appreciated that multiple NICs 512 can be present in the computer 500, connecting the computer to other types of networks and remote computer systems.

[0076] The computer 500 can be connected to a storage device 518 that provides non-volatile storage for the computer. The storage device 518 can store an operating system 520, programs 522, and data, which have been described in greater detail herein. The storage device 518 can be connected to the computer 500 through a storage controller 514 connected to the chipset 506. The storage device 518 can consist of one or more physical storage units. The storage controller 514 can interface with the physical storage units through a serial attached SCSI ("SAS") interface, a serial advanced technology attachment ("SATA") interface, a fiber channel ("FC") interface, or other type

of interface for physically connecting and transferring data between computers and physical storage units.

[0077] The computer 500 can store data on the storage device 518 by transforming the physical state of the physical storage units to reflect the information being stored. The specific transformation of physical state can depend on various factors, in different embodiments of this description. Examples of such factors can include, but are not limited to, the technology used to implement the physical storage units, whether the storage device 518 is characterized as primary or secondary storage, and the like.

[0078] For example, the computer 500 can store information to the storage device 518 by issuing instructions through the storage controller 514 to alter the magnetic characteristics of a particular location within a magnetic disk drive unit, the reflective or refractive characteristics of a particular location in an optical storage unit, or the electrical characteristics of a particular capacitor, transistor, or other discrete component in a solid-state storage unit. Other transformations of physical media are possible without departing from the scope and spirit of the present description, with the foregoing examples provided only to facilitate this description. The computer 500 can further read information from the storage device 518 by detecting the physical states or characteristics of one or more particular locations within the physical storage units.

[0079] In addition to the mass storage device 518 described above, the computer 500 can have access to other computer-readable storage media to store and retrieve information, such as program modules, data structures, or other data. It should be appreciated by those skilled in the art that computer-readable storage media is any available media that provides for the non-transitory storage of data and that can be accessed by the computer 500. In some examples, the operations performed by the network 524 and or any components included therein, may be supported by one or more devices similar to computer 500. Stated otherwise, some or all of the operations performed by the network 524, and or any components included therein, may be performed by one or more computer devices 500 operating in a cloud-based arrangement.

[0080] By way of example, and not limitation, computer-readable storage media can include volatile and non-volatile, removable and non-removable media implemented in any method or technology. Computer-readable storage media includes, but is not limited to, RAM, ROM, erasable programmable ROM ("EPROM"), electrically-erasable programmable ROM ("EEPROM"), flash memory or other solid-state memory

technology, compact disc ROM ("CD-ROM"), digital versatile disk ("DVD"), high definition DVD ("HD-DVD"), BLU-RAY, or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium that can be used to store the desired information in a non-transitory fashion.

[0081] As mentioned briefly above, the storage device 518 can store an operating system 520 utilized to control the operation of the computer 500. According to one embodiment, the operating system comprises the LINUX operating system. According to another embodiment, the operating system comprises the WINDOWS® SERVER operating system from MICROSOFT Corporation of Redmond, Washington. According to further embodiments, the operating system can comprise the UNIX operating system or one of its variants. It should be appreciated that other operating systems can also be utilized. The storage device 518 can store other system or application programs and data utilized by the computer 500.

[0082] In one embodiment, the storage device 518 or other computer-readable storage media is encoded with computer-executable instructions which, when loaded into the computer 500, transform the computer from a general-purpose computing system into a special-purpose computer capable of implementing the embodiments described herein. These computer-executable instructions transform the computer 500 by specifying how the CPUs 504 transition between states, as described above. According to one embodiment, the computer 500 has access to computer-readable storage media storing computer-executable instructions which, when executed by the computer 500, perform the various processes described above with regard to FIGS. 1-4. The computer 500 can also include computer-readable storage media having instructions stored thereupon for performing any of the other computer-implemented operations described herein.

[0083] The computer 500 can also include one or more input/output controllers 516 for receiving and processing input from a number of input devices, such as a keyboard, a mouse, a touchpad, a touch screen, an electronic stylus, or other type of input device. Similarly, an input/output controller 516 can provide output to a display, such as a computer monitor, a flat-panel display, a digital projector, a printer, or other type of output device. It will be appreciated that the computer 500 might not include all of the components shown in FIG. 5, can include other components that are not explicitly shown in FIG. 5, or might utilize an architecture completely different than that shown in FIG. 5.

[0084] The programs 522 may comprise any type of programs or processes to perform the techniques described in this disclosure for a data plane approach for per-packet unique initialization vector (IV) for high bandwidth encryption engines in multipathing IP network. The programs 522 may enable the computer 500 to perform various operations described above with regard to FIGS. 3-4.

[0085] In summary, techniques for generating a per-packet initialization vector for high bandwidth encryption engines in a multipathing IP network are described herein. In examples, a network switch of a first datacenter site may receive a data packet to be sent to a second datacenter site over a network. The data packet may be encrypted according to a virtual extensible LAN (VxLAN) protocol and to be transmitted in a VxLAN tunnel created for the first datacenter site and the second datacenter site. An encryption engine implemented at the network switch may generate an initialization vector (IV) for the data packet based on a packet number (PN) associated with the data packet. The encryption engine may use the IV and information associated with a security association (SA) assigned to the packet to encrypt the data packet. In some examples, a full 64-bit PN may be used to compute the IV for the data packet.

[0086] While the invention is described with respect to the specific examples, it is to be understood that the scope of the invention is not limited to these specific examples. Since other modifications and changes varied to fit particular operating requirements and environments will be apparent to those skilled in the art, the invention is not considered limited to the example chosen for purposes of disclosure and covers all changes and modifications which do not constitute departures from the true spirit and scope of this invention.

[0087] Although the application describes embodiments having specific structural features and/or methodological acts, it is to be understood that the claims are not necessarily limited to the specific features or acts described. Rather, the specific features and acts are merely illustrative some embodiments that fall within the scope of the claims of the application.

CLAIMS

WHAT IS CLAIMED IS:

1. A method comprising:

receiving, at a first device, a packet to be sent, over a network and in an encrypted tunnel, to a second device;

generating, at the first device, an initialization vector (IV) for the packet based at least in part on a packet number (PN) associated with the packet;

encrypting, at the first device, the packet based at least in part on the IV and information associated with a security association (SA) assigned to the packet to generate an encrypted packet; and

transmitting, from the first device, the encrypted packet to the second device in the encrypted tunnel over the network.

2. The method of claim 1, wherein the PN is represented in a length of 64 bits, and the method further comprises:

generating, at the first device, the IV for the packet using the 64 bits.

3. The method of claim 1 or 2, further comprising:

obtaining, from a database, the SA assigned to the packet, wherein the information associated with the SA includes the PN, a security association key (SAK), a security channel identifier (SCI), and an association number (AN).

4. The method of claim 3, further comprising:

constructing, at the first device, a security header to include at least the PN, the SCI and the AN.

5. The method of claim 4, further comprising:

generating, at the first device, an integrity checksum value (ICV) based on the IV.

6. The method of claim 5, wherein the encrypting comprises:

encrypting, at the first device, user data of the packet based at least in part on the IV, the SAK, and the SCI to generate encrypted user data;

replacing, at the first device, the user data of the packet with the encrypted user data; and

inserting, at the first device, the ICV and the security header into the packet.

7. The method of any of claims 1 to 6, further comprising:

incrementing, at the first device, the PN by one to generate a second PN; and

updating, at the first device, the SA to include the second PN.

8. The method of any of claims 1 to 7, wherein the packet is encapsulated using a virtual extensible local area network (VxLAN) overlay protocol.

9. A first device comprising:

a processor; and

a non-transitory computer-readable media storing instructions that, when executed by the processor, cause the processor to perform operations including:

receiving a packet to be sent, over a network and in an encrypted tunnel, to a second device;

generating an initialization vector (IV) for the packet based at least in part on a packet number (PN) associated with the packet;

constructing a security header to include at least the PN;

encrypting the packet based at least in part on the IV and information associated with a security association (SA) assigned to the packet to generate an encrypted packet; and

transmitting, over the network and in the encrypted tunnel, the encrypted packet to the second device.

10. The first device of claim 9, wherein the PN is represented in a length of 64 bits, and the operations further comprise:

generating the IV for the packet using the 64 bits.

11. The first device of claim 9 or 10, wherein the operations further comprise:

obtaining, from a database, the SA assigned to the packet, wherein the information associated with the SA includes the PN, a security association key (SAK), a security channel identifier (SCI), and an association number (AN).

12. The first device of claim 11, wherein the operations further comprise:

constructing a security header to include at least the PN, the SCI and the AN.

13. The first device of claim 12, wherein the operations further comprise:

generating an integrity checksum value (ICV) based at least in part on the IV.

14. The first device of claim 13, wherein the encrypting comprises:

encrypting, at the first device, user data of the packet based at least in part on the IV, the SAK, and the SCI to generate encrypted user data;

replacing, at the first device, the user data of the packet with the encrypted user data; and

inserting, at the first device, the ICV and the security header into the packet.

15. The first device of any of claims 9 to 14, wherein the operations further comprise:

incrementing the PN by one to generate a second PN; and

updating the SA to include the second PN.

16. The first device of any of claims 9 to 15, the packet is encapsulated using a virtual extensible local area network (VxLAN) overlay protocol.

17. A second device comprising:

a processor; and

a non-transitory computer-readable media storing instructions that, when executed by the processor, cause the processor to perform operations including:

receiving, over a network and via an encrypted tunnel, a packet from a first device;

retrieving a packet number (PN) associated with the packet from a security header of the packet;

generating a first integrity checksum value (ICV) based at least in part on the packet number;

performing an integrity check for the packet based at least in part on the first ICV;

in response to the packet passing the integrity check, decrypting the packet based at least in part on information associated with a security association (SA) assigned to the packet to generate a decrypted packet; and

transmitting, over the network, the decrypted packet to a destination device.

18. The second device of claim 17, wherein the PN is in a length of 64 bits, and the operations further comprise:

using the PN as is for an initialization vector (IV) for the packet;

obtaining, from the database, the SA assigned to the packet, wherein the information associated with the SA includes at least a security association key (SAK);

retrieving a security channel identifier (SCI) from the security header; and

generating the first ICV based at least in part on the IV, the SAK, the SCI, and user data of the packet,

wherein the integrity check includes comparing the first ICV with a second ICV in the packet.

19. The second device of claim 18, wherein the decrypting comprises:

decrypting the user data of the packet based at least in part on the IV, the SAK, and the SCI; and

removing the security header and the second ICV from the packet.

20. The second device of any of claims 17 to 19, wherein the encrypted tunnel is a virtual extensible local area network (VxLAN) tunnel.

21. Apparatus comprising:

means for receiving, at a first device, a packet to be sent, over a network and in an encrypted tunnel, to a second device;

means for generating, at the first device, an initialization vector (IV) for the packet based at least in part on a packet number (PN) associated with the packet;

means for encrypting, at the first device, the packet based at least in part on the IV and information associated with a security association (SA) assigned to the packet to generate an encrypted packet; and

means for transmitting, from the first device, the encrypted packet to the second device in the encrypted tunnel over the network.

22. The apparatus according to claim 21 further comprising means for implementing the method according to any of claims 2 to 8.

23. A method comprising:

receiving, over a network and via an encrypted tunnel, a packet from a first device;

retrieving a packet number (PN) associated with the packet from a security header of the packet;

generating a first integrity checksum value (ICV) based at least in part on the packet number;

performing an integrity check for the packet based at least in part on the first ICV;

in response to the packet passing the integrity check, decrypting the packet based at least in part on information associated with a security association (SA) assigned to the packet to generate a decrypted packet; and

transmitting, over the network, the decrypted packet to a destination device;

the method optionally further comprising the further operations of any of claims 18 to 20.

24. Apparatus comprising:

means for receiving, over a network and via an encrypted tunnel, a packet from a first device;

means for retrieving a packet number (PN) associated with the packet from a security header of the packet;

means for generating a first integrity checksum value (ICV) based at least in part on the packet number;

means for performing an integrity check for the packet based at least in part on the first ICV;

means for decrypting the packet, in response to the packet passing the integrity check, based at least in part on information associated with a security association (SA) assigned to the packet to generate a decrypted packet; and

means for transmitting, over the network, the decrypted packet to a destination device;

the apparatus optionally further comprising means for implementing the further operations of any of 18 to 20.

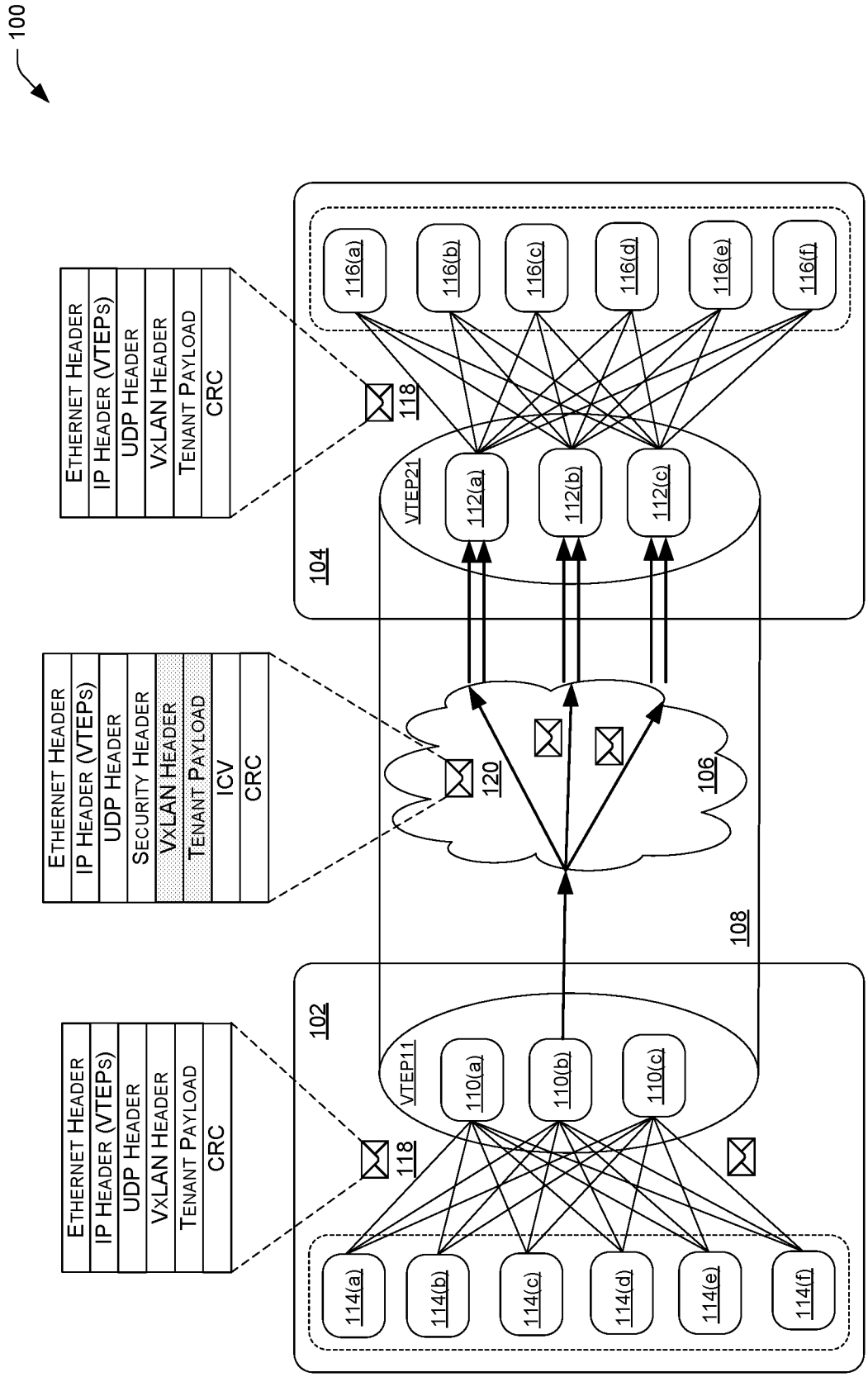
100
ETHERNET HEADER
IP HEADER (VTEPS)
UDP HEADER
VXLAN HEADER
TENANT PAYLOAD
CRC
118
ETHERNET HEADER
IP HEADER (VTEPS)
UDP HEADER
SECURITY HEADER
VXLAN HEADER
TENANT PAYLOAD
ICV
CRC
120
ETHERNET HEADER
IP HEADER (VTEPS)
UDP HEADER
VXLAN HEADER
TENANT PAYLOAD
CRC
118
102
VTEP11
110(a)
110(b)
110(c)
114(a)
114(b)
114(c)
114(d)
114(e)
114(f)
104
VTEP21
112(a)
112(b)
112(c)
116(a)
116(b)
116(c)
116(d)
116(e)
116(f)
106
108

FIG. 1

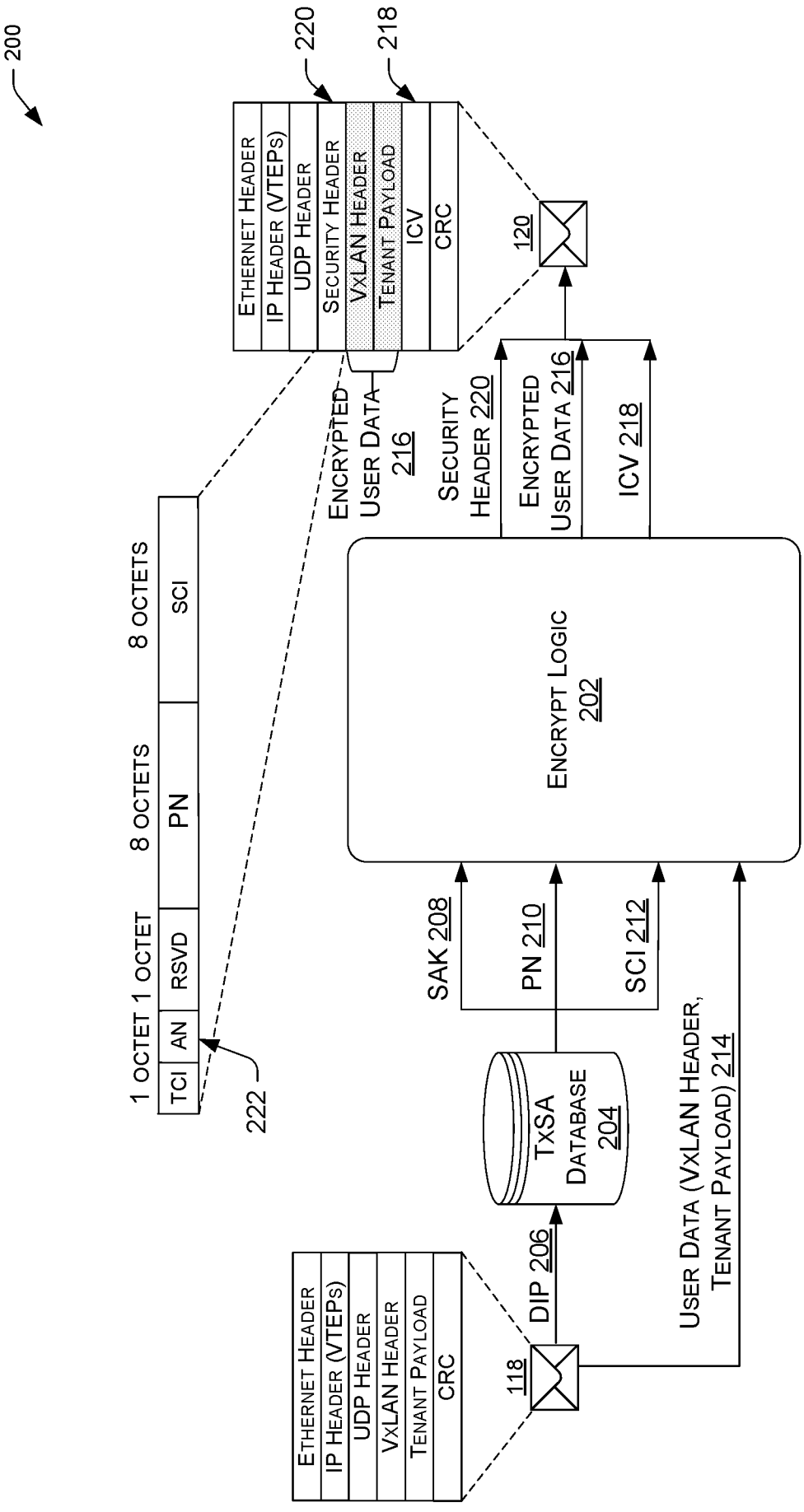
200
ETHERNET HEADER
IP HEADER (VTEPs)
UDP HEADER
SECURITY HEADER
VXLAN HEADER
TENANT PAYLOAD
ICV
CRC
220
218
120
ENCRYPTED USER DATA 216
SECURITY HEADER 220
ENCRYPTED USER DATA 216
ICV 218
ENCRYPT LOGIC 202
1 OCTET
1 OCTET
8 OCTETS
8 OCTETS
TCI
AN
RSVD
PN
SCI
222
SAK 208
PN 210
SCI 212
TXSA DATABASE 204
DIP 206
118
USER DATA (VXLAN HEADER, TENANT PAYLOAD) 214

FIG. 2

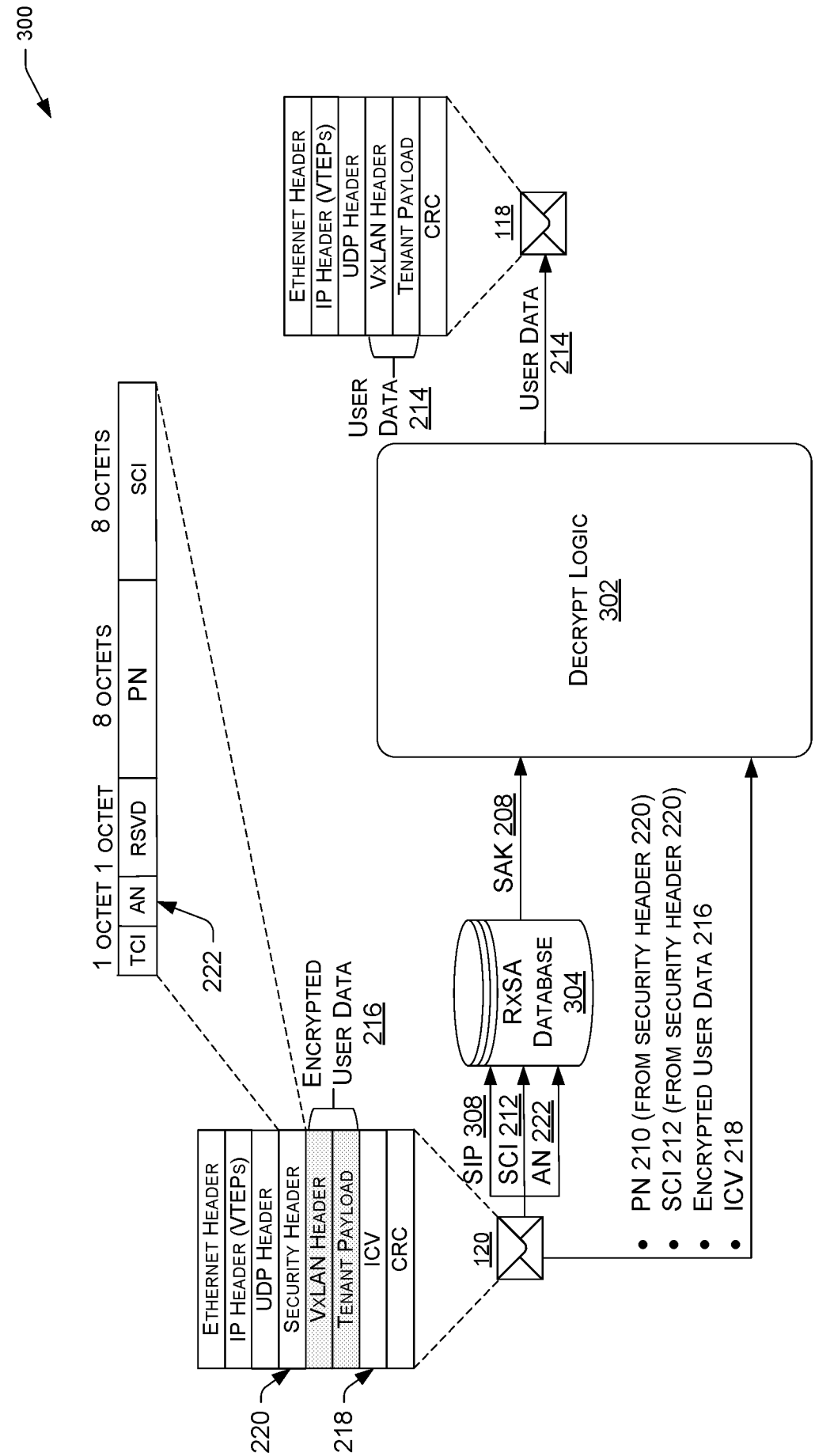
300
ETHERNET HEADER
IP HEADER (VTEPS)
UDP HEADER
SECURITY HEADER
VXLAN HEADER
TENANT PAYLOAD
ICV
CRC
220
218
ENCRYPTED USER DATA 216
1 OCTET 1 OCTET 8 OCTETS 8 OCTETS
TCI AN RSVD PN SCI
222
120
SIP 308
SCI 212
AN 222
RXSA DATABASE 304
SAK 208
DECRYPT LOGIC 302
• PN 210 (FROM SECURITY HEADER 220)
• SCI 212 (FROM SECURITY HEADER 220)
• ENCRYPTED USER DATA 216
• ICV 218
USER DATA 214
USER DATA 214
118
ETHERNET HEADER
IP HEADER (VTEPS)
UDP HEADER
VXLAN HEADER
TENANT PAYLOAD
CRC

FIG. 3

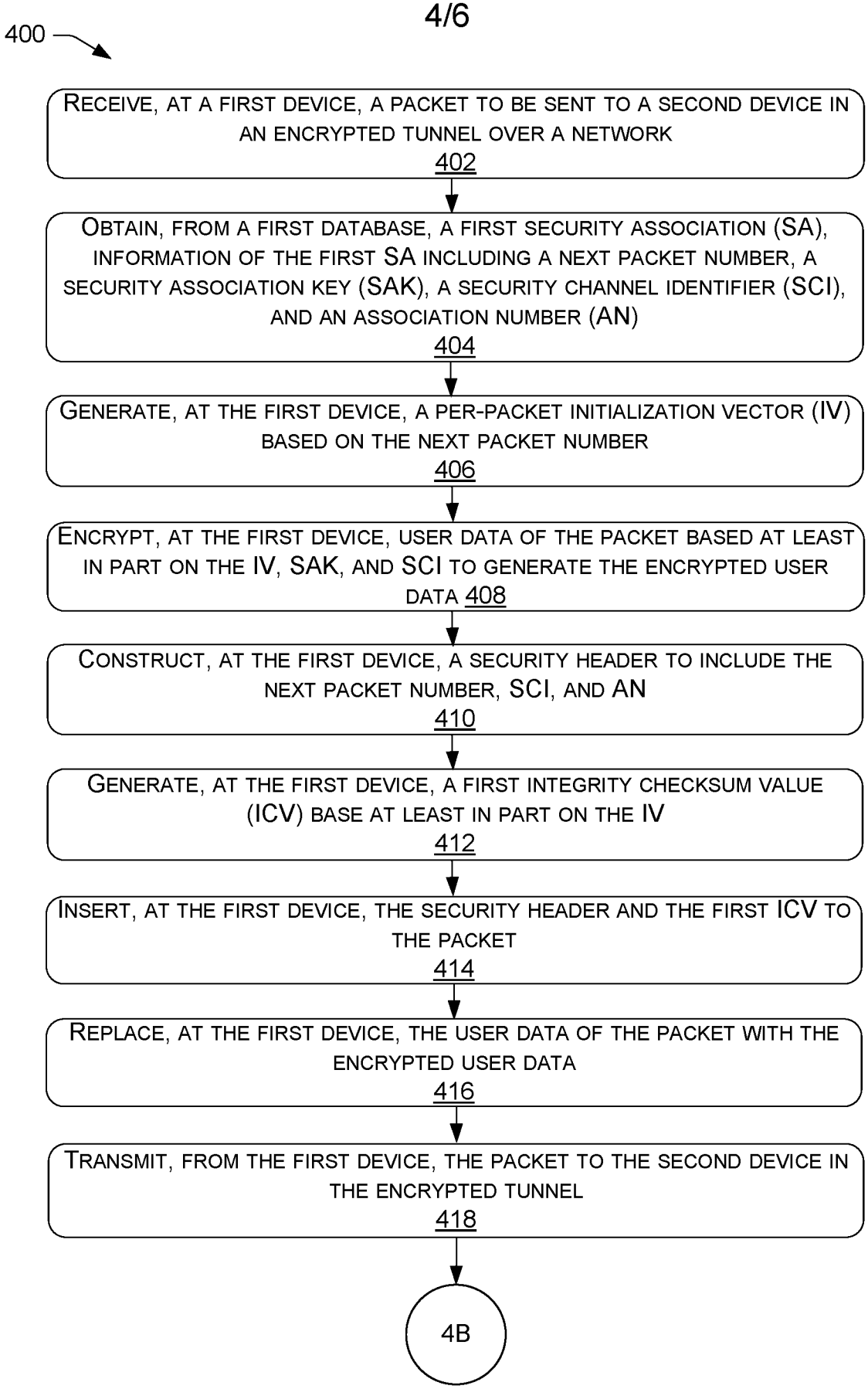
400
RECEIVE, AT A FIRST DEVICE, A PACKET TO BE SENT TO A SECOND DEVICE IN AN ENCRYPTED TUNNEL OVER A NETWORK
402
OBTAIN, FROM A FIRST DATABASE, A FIRST SECURITY ASSOCIATION (SA), INFORMATION OF THE FIRST SA INCLUDING A NEXT PACKET NUMBER, A SECURITY ASSOCIATION KEY (SAK), A SECURITY CHANNEL IDENTIFIER (SCI), AND AN ASSOCIATION NUMBER (AN)
404
GENERATE, AT THE FIRST DEVICE, A PER-PACKET INITIALIZATION VECTOR (IV) BASED ON THE NEXT PACKET NUMBER
406
ENCRYPT, AT THE FIRST DEVICE, USER DATA OF THE PACKET BASED AT LEAST IN PART ON THE IV, SAK, AND SCI TO GENERATE THE ENCRYPTED USER DATA 408
CONSTRUCT, AT THE FIRST DEVICE, A SECURITY HEADER TO INCLUDE THE NEXT PACKET NUMBER, SCI, AND AN
410
GENERATE, AT THE FIRST DEVICE, A FIRST INTEGRITY CHECKSUM VALUE (ICV) BASE AT LEAST IN PART ON THE IV
412
INSERT, AT THE FIRST DEVICE, THE SECURITY HEADER AND THE FIRST ICV TO THE PACKET
414
REPLACE, AT THE FIRST DEVICE, THE USER DATA OF THE PACKET WITH THE ENCRYPTED USER DATA
416
TRANSMIT, FROM THE FIRST DEVICE, THE PACKET TO THE SECOND DEVICE IN THE ENCRYPTED TUNNEL
418
4B

FIG. 4A

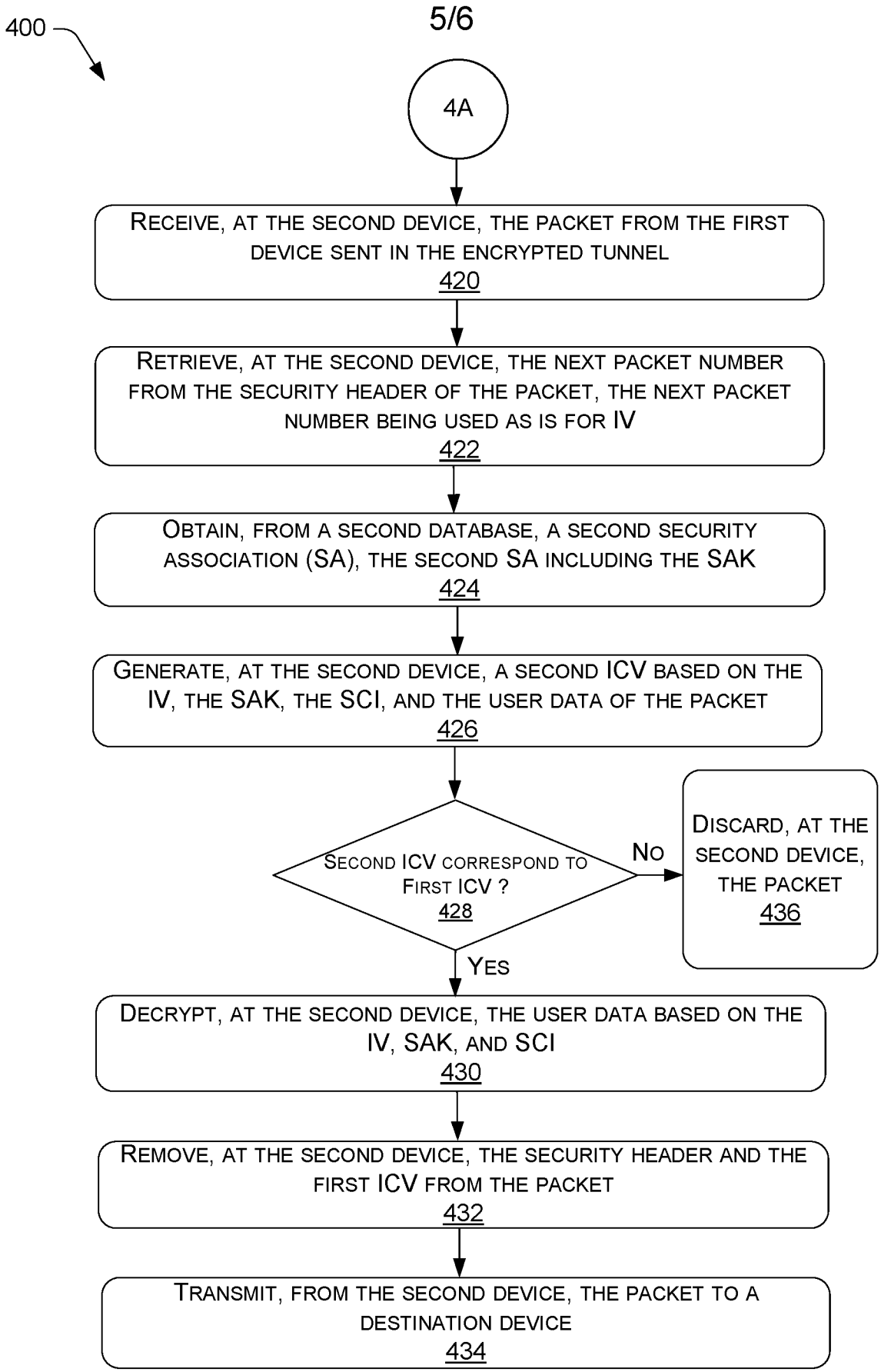
400
4A
RECEIVE, AT THE SECOND DEVICE, THE PACKET FROM THE FIRST DEVICE SENT IN THE ENCRYPTED TUNNEL
420
RETRIEVE, AT THE SECOND DEVICE, THE NEXT PACKET NUMBER FROM THE SECURITY HEADER OF THE PACKET, THE NEXT PACKET NUMBER BEING USED AS IS FOR IV
422
OBTAIN, FROM A SECOND DATABASE, A SECOND SECURITY ASSOCIATION (SA), THE SECOND SA INCLUDING THE SAK
424
GENERATE, AT THE SECOND DEVICE, A SECOND ICV BASED ON THE IV, THE SAK, THE SCI, AND THE USER DATA OF THE PACKET
426
SECOND ICV CORRESPOND TO FIRST ICV ?
428
NO
DISCARD, AT THE SECOND DEVICE, THE PACKET
436
YES
DECRYPT, AT THE SECOND DEVICE, THE USER DATA BASED ON THE IV, SAK, AND SCI
430
REMOVE, AT THE SECOND DEVICE, THE SECURITY HEADER AND THE FIRST ICV FROM THE PACKET
432
TRANSMIT, FROM THE SECOND DEVICE, THE PACKET TO A DESTINATION DEVICE
434

FIG. 4B

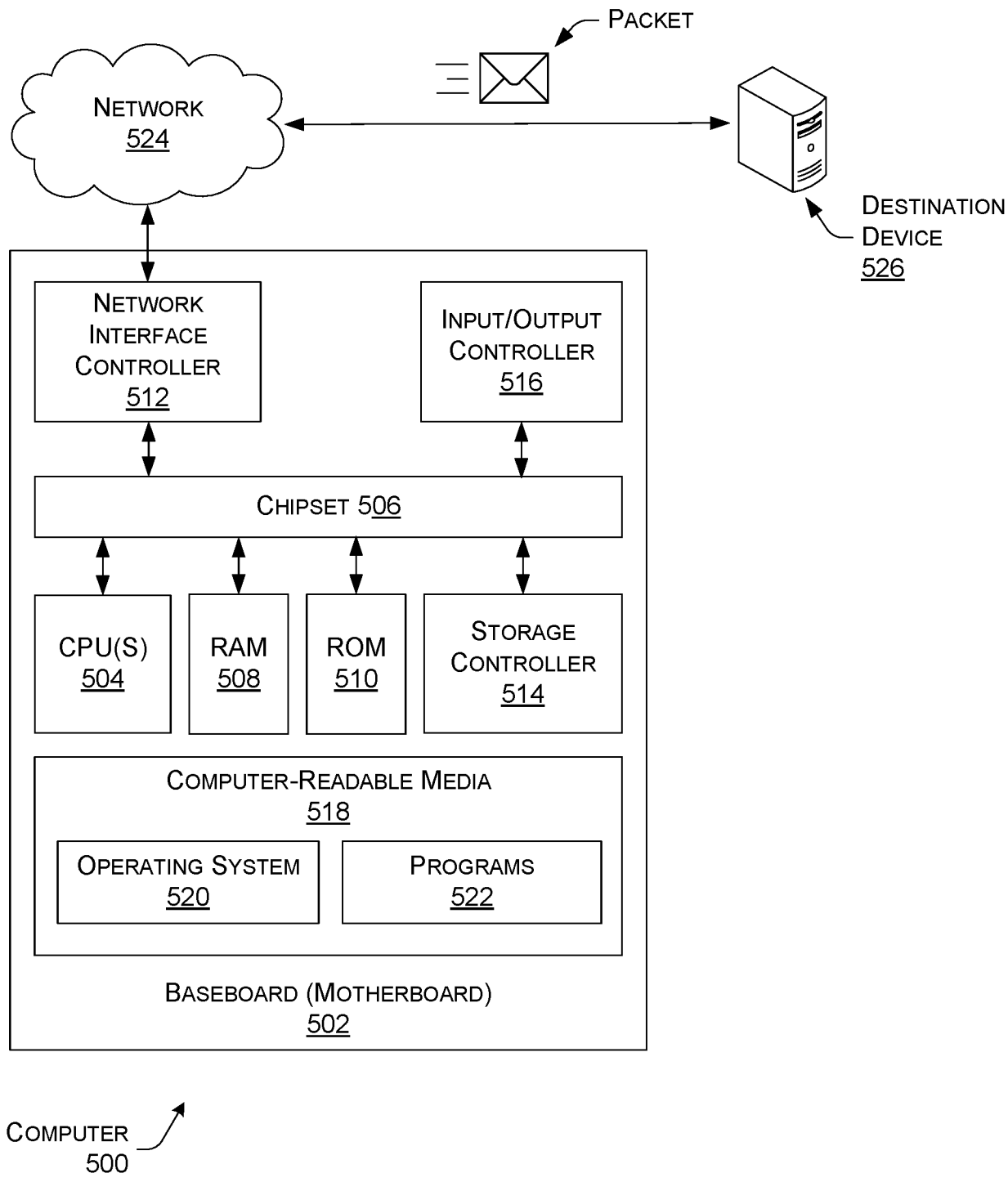
PACKET
NETWORK
524
DESTINATION
DEVICE
526
NETWORK
INTERFACE
CONTROLLER
512
INPUT/OUTPUT
CONTROLLER
516
CHIPSET 506
CPU(S)
504
RAM
508
ROM
510
STORAGE
CONTROLLER
514
COMPUTER-READABLE MEDIA
518
OPERATING SYSTEM
520
PROGRAMS
522
BASEBOARD (MOTHERBOARD)
502
COMPUTER
500

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2023/086050

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L9/08 H04L12/46 H04L9/40
ADD. H04L45/24

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	KHAN FERDOUS WAHID ET AL: "Secure bridging in large scale deployment of Ethernet", SECURITY AND CRYPTOGRAPHY (SECRYPT), PROCEEDINGS OF THE 2010 INTERNATIONAL CONFERENCE ON, IEEE, 26 July 2010 (2010-07-26), pages 1-11, XP031936460, figures 1, 4 page 2 - page 6	1-24

-/--

[x] Further documents are listed in the continuation of Box C.

[x] 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	Date of mailing of the international search report
20 March 2024	28/03/2024

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Dely, Peter

2

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	"IEEE Standard for Local and metropolitan area networksMedia Access Control (MAC) Security Amendment 2: Extended Packet Numbering", IEEE STANDARD, IEEE, PISCATAWAY, NJ, USA, 12 February 2013 (2013-02-12), pages 1-67, XP068045832, ISBN: 978-0-7381-8148-6 page 20 - page 23	1-24
A	US 10 778 662 B2 (CISCO TECH INC [US]) 15 September 2020 (2020-09-15) the whole document	1-24

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2023/086050

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 10778662 B2	15-09-2020	CA 3115517 A1	30-04-2020
		CN 112889253 A	01-06-2021
		EP 3871390 A1	01-09-2021
		US 2020127987 A1	23-04-2020
		US 2020358750 A1	12-11-2020
		US 2024080309 A1	07-03-2024
		WO 2020086252 A1	30-04-2020