Research and Design of Next Generation Internet (IPV9) Datagram

Wang Zhongsheng
[1.]State and Provincial Joint Engineering Lab. of Advanced Network, Monitoring and Control, China
[2.]School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
e-mail: wzhsh1681@163.com

Lin Zhao
[1.]Chinese Decimal Network Working Group
Shanghai, China
[2.]Shanghai Decimal System Network Information Technology Ltd.
e-mail: chinalinzhao@126.com

Xie Jianping
[1.]Chinese Decimal Network Working Group
Shanghai, China
[2.]Shanghai Decimal System Network Information Technology Ltd.
e-mail: 13386036170@189.cn

Zhong Wei
[1.]Chinese Decimal Network Working Group
Shanghai, China
[2.]Shanghai Decimal System Network Information Technology Ltd.
e-mail: 13331860961@189.cn

***Abstract*—The current global Internet USES the TCP/IP protocol cluster. IP is the network layer protocol and the core protocol in this protocol family. The current version is IPv4 with 32-bit addresses. With the popularity of Internet applications, the limited address space defined by IPv4 has been exhausted. To expand the address space, the Internet Engineering Task Force (IETF) has designed the next generation IP protocol, IPv6, to replace IPv4. IPv6 redefines the address space, using a 128-bit address length that provides almost unlimited addresses. However, with the development and application of the Internet of things, big data and cloud storage, IPv6 has some shortcomings in its address structure design, security and network sovereignty, so it is urgent to develop a new generation or future internet with security and reliability, autonomy and controllable , and it becomes a research hotspot in the world.**

This paper developed a new generation Internet (IPV9) datagram by researching the existed IPv4 and IPv6, and it is based on the method of assigning addresses to computers connected to the Internet by full decimal character code which designed by the Chinese Decimal Network Working Group with the leader of Xie Jianping. It is a subsequent version with RFC1606, RFC1607, a new generation of network data structure, it is not the updating of IPv4 [RFC - 791] and IPv6 [RFC - 1883], [RFC - 2464], it is a new vision to demonstration and testing.

Keywords-Future Network; IPv4; IPv6; IPV9; Datagram

A datagram is the basic unit of data transmitted on the network. It contains a header and the data itself, in which the header describes the destination of the data and its relationship to other data. Datagram is a complete and independent data entity, which carries the information to be transferred from the source computer to the destination computer and does not rely on the previous exchange between the source and the destination and the transmission network [1]. TCP/IP protocol defines a packet that is transmitted on the Internet; called IP Datagram, which are the network layer protocol and the core protocol of the TCP/IP protocol family.

DOI: 10.21307/ijanmc-2019-030

IP is a virtual package consisting of a header and data. The IPv4 header is a fixed length of 20 bytes, which is required for all IP datagram. After the fixed portion of the header are optional fields whose length is variable. Both the source and destination addresses in the header are IP protocol addresses. The current global Internet USES the TCP/IP protocol cluster, the current version is IPv4 with 32-bit addresses. With the popularity of Internet applications, the limited address space defined by IPv4 has been exhausted. To expand the address space, the Internet Engineering Task Force (IETF) has designed the next generation IP protocol, IPv6, to replace IPv4. IPv6 redefines the address space, using a 128-bit address length that provides almost unlimited addresses. However, with the development and application of the Internet of things, big data and cloud storage, IPv6 has some shortcomings in its address structure design, security and network sovereignty. The Chinese researchers developed a new generation Internet (IPV9) datagram by researching the existed IPv4 and IPv6, and it is based on the method of assigning addresses to computers connected to the Internet by full decimal character code. It is a subsequent version with RFC1606, RFC1607, a new generation of network data structure, it is not the updating of IPv4 and IPv6, it is a new vision to demonstration and testing.

I. OVERALL DESIGN OBJECTIVE

In order to be compatible with the existing Internet system, on the basis of studying the existing standard IPv4 [RFC -791] and IPv6 [RFC -1883] and [RFC -2464], the overall goal of IPV9 datagram design is formulated.

1) Extended address capacity

IPV9 increases the length of IP addresses from 32 (IPv4) and 128 (IPv6) bits to 2048 bits to support more address hierarchies, more addressable nodes, and simpler automatic address configurations. It also increases the IPv4 32 bit address length reduced to 16 bits, in order to solve mobile communication.

2) Variable length and uncertain number digits

In order to reduce the network overhead, this datagram designing adopts the method of indefinite length and uncertain number of digits. By adding a "range" field to the multicast address, the scalability of multicast routing is improved. It defines a new address type, "any broadcast address," for sending datagrams to any one of a set of nodes.

3) Simplify and improve header format

Some IPv4 header fields have been eliminated or made optional to reduce the overhead of common processing on packet control and to limit IPV9 header bandwidth overhead. The encoding of header options has been changed to allow more efficient forwarding, reduce restrictions on the length of options, and gain more flexibility to introduce new options in the future.

4) Label data streams

To attach labels to data streams belonging to a particular data communication, the sender may require special processing of these data streams, such as non-default quality of service or real-time service, such as using virtual circuits to achieve the purpose of circuit communication.

5) High safety and reliability

To get security and reliability, the new datagram added expansionary support for IP address encryption and authentication, data integrity, and data security (optional) in IPV9 designing. It extension headers and options take into account the length of packets, the semantics of flow control labels and categories, and the impact on high-level protocols.

6) Direct routing addressing

The ICMP (Internet Control Message Protocol) version of IPV9 contains all the requirements for implementing IPV9. The function of route character arrangement authentication, recognition and addressing is added, which reduces the routing cost and improves the efficiency.

7) Compatible with IPv4 and IPv6

In order to ensure a smooth transition from IPv4 to IPV9, considering the protection of the original investment and not changing user habits, this datagram defines IPV9 header and transitional header. The last segment address is used for the header of IPv4 or IPv6 is used, but the version number is changed to 9, the connection protocols used are IPv4 or IPv6.

8) The virtual and real circuit design

In order to smoothly transmit voice, image and video and other big data real-time applications, it is necessary to adopt long-stream code, absolute value, return code and other technologies, and adopt three-layer and four-layer network hybrid architecture. Three-layer and four-layer network hybrid architecture design should be implemented in virtual and real circuits.

II. GENERAL DESIGN OF SYSTEM

A. Some basic terminologies

The basic concepts in system designing are defined as follows.

1) Node: a device installed with IPV9 or IPV9 device compatible with IPv4 and IPv6.

2) Router: the device that is responsible for forwarding and explicitly not sending IPV9 data to itself.

3) Host: any node that does not belong to the router.

4) Upper agreement: Protocols located in the layer above IPV9. For example, transport layer protocols TCP, UDP, a communication facility or medium in the link layer, on which nodes can carry out link-layer communication, that is, the layer just below IPV9. Examples of links include Ethernet (or bridged), PPP links, X.25, frame relay, or ATM networks.

5) Neighbor: each node connected to the same link.

6) Interface: node to link connection.

7) Address: IPV9 layer identifier of an interface or a group of interfaces.

8) Packet: An IPV9 header plus load.

9) Link MUT (Maximum Transmission Unit): the Maximum Transmission Unit that can be transmitted on a section of link, namely the Maximum data packet length in eight-bit group.

10) Path MUT: the smallest MUT of all links in the path between the source node and the destination node.

Note: for a device with multiple interfaces, it can be configured to forward non-self-directed packets from one of its interfaces and drop from other interfaces. When such a device receives data from a previous interface or interacts with a neighbor, the protocol requirements of the host must be followed.

B. IPV9 header format

The format design of IPV9 datagram header is shown in table 1.

TABLE I. IPV9 HEADER FORMAT

<table>
<tr><td rowspan="2">Version Number</td><td colspan="4">Category Type</td><td rowspan="2">Flow Label</td></tr>
<tr><td>Address Length</td><td>Priority class Communication</td><td>Authentication Address</td><td>Absolute Communication</td></tr>
<tr><td>Payload Length</td><td colspan="4">Next Header</td><td>Hop Limit</td></tr>
<tr><td colspan="6">Source Address (16-2048 bit)</td></tr>
<tr><td colspan="6">Destination Address (16-2048bit)</td></tr>
<tr><td colspan="6">Time</td></tr>
<tr><td colspan="6">Authorization Code</td></tr>
</table>

The design of table 1 is explained below.

1) Version Number: The length is 4 bits, representing the Internet protocol version number. For IPV9, this field must be 9.

2) Category Type: The length is 8 bits, the high 3 bits are used to specify the address length, its volume is 0 to 7, and it's 2 to the power. Contains address length of 1Byte ~ 128Byte; the default value is 256 bits, where 0 is 16 bits, 1 is 32 bits, 2 is 64 bits, 3 is 128 bits, 4 is 256 bits, 5 is 512 bits, 6 is 1024 bits, and 7 is 2048 bits. The last five bits specify the communication class and authentication for the source and destination addresses. 0 to 15 is used to specify the priority class of the communication, 6 to 7 is used to specify the communication method after the first authentication, which is used by the packet sending place for control and whether the source address and destination address authentication is needed. 8 to 14 are used to specify absolute communication that will not fall back when congestion is encountered, and 15 are used for virtual circuits. 16 and 17 are used to assign audio and video, called absolute value, to ensure the uninterrupted transmission of audio and video. The other values are reserved for future use.

3) Flow Label: with a length of 20 bits, it is used to identify packages belonging to the same business flow.

4) Payload Length (Net Load Length): the length is 16 bits, including the net load byte length, that is, the number of bytes contained in the packet after IPV9 header.

5) Next Header: the length is 8 bits. This field indicates the protocol type in the field following the IPV9 header.

6) Hop Limit: the length is 8 bits, and this field is subtracted one each time a node forwards a packet.

7) Source Address: the length is 8bit ~ 2048bit, and the sender address of IPV9 packet is specified. Adopt the method of Variable length and uncertain number digits.

8) Destination Address: the length is 8 bit ~ 2048 bit, and the destination address of IPV9 packet is specified. Adopt the method of Variable length and uncertain number digits.

9) Time: it is used to control the lifetime of the address in the header.

10) Authorization Code: it is used to identify the authenticity of the address in the header.

C. Extended headers of IPV9

In IPV9 datagrams, Internet optional information is placed in specialized headers which between the packet IPV9 header and the high-level protocol header. The number of such extended headers is not too much, and each identified by a different value for the next header. An IPV9 packet can have zero to more than one extension header, each of which is defined in the next header S field in the previous header, as shown in table 2.

TABLE II. EXTENDED HEADER FORMAT

IPV9 Header Label NEXT HEADER =TCP	TCP HEADER +DATA		
IPV9 Header Label NEXT HEADER = ROUTE	ROUTING HEADER NEXT HEADER =TCP	TCP HEADER +DATA	
IPV9 Header Label NEXT HEADER = ROUTE	ROUTING HEADER NEXT HEADER = DATA SEGMENT	DATA SEGMENT HEADER NEXT HEADER =TCP	TCP DATA SEGMENT HEADER +DATA

On the path of the packet passing, no node to checks or processes the extended header until the packet reaches the node specified by the destination address field in the IPV9 header (or, if it is a multicast address, it would be a group of nodes). For the normal multiplexing of the next header field of IPV9 header, the processing module is called to process the first extended header. If an extended header does not exist, the high level header is processed. Therefore, the extension headers must be processed exactly in the order in which they appear in the packet, and the receiver cannot scan the packet to find a particular extension header and process it before processing other preceding headers.

If a node, after processing the header wants to process the next header, but it does not recognize the value of the "next header" , it will lost the packet, and sends the source a "ICMP parameter problem" message, the message ICMP code has a value of 2 (can't identify the "next header" type), "ICMP indicator" contained in the field could not identify the "next header values" offset location in the original packets. The same is done if a node finds the "next header" field is 0 in any non-IPV9 header.

Each extended header is an integer multiple of the 8-bit array so that subsequent headers can be aligned along 8-bit array boundaries. In the extended header, the fields made up of multiple 8-bit groups are internally aligned with their internal natural boundaries, that is, the position in the fields of width n 8-bit groups are placed at the beginning of the header, an integer times n 8-bit groups, where n=1,2,4 or 8.

The fully installed IPV9 includes the following extension headers:

- Hop-by-Hop Option(segment options)
- Routing（type 0）
- Fragment;
- Destination Options;
- Authentication;
- Encapsulating Security Payload.

This paper defines the first four extension headers, and the next two additional definitions.

1) The sequence of extension headers

When multiple extension headers are used in the same packet, they appear in the following order:

a IPV9 header;
b segment options header;
c Destination options header(annotation 1);
d Routing header;
e Data segment header;
f Authentication header;
g Encapsulate security load header;
h Destination option header(annotation 2);
i The upper header;

Annotation 1: Options for the first destination node to appear in the IPV9 destination address field and for subsequent destination nodes listed in the routing header.

Annotation 2: The option to be processed only by the destination node of the packet.

Each extended header can occur at most once, but destination option headers can occur at most twice, once before routing headers and once before high-level headers.

If the high-level header is another IPV9 header (that is, when IPV9 is encapsulated in another IPV9 tunnel), it can be followed by its own extended headers, which, as a whole, follow the same sequence.

When defining other extended headers, it must specify a sequential relationship between them and the headers listed above

2) Options

When the extension header is defined the segment-by-segment option and the destination option header have an unequal number of options encoded in the form of Type length value (TLV). The format is shown in table 3.

TABLE III. OPTION FORMAT

Option Type	Option data length	Option data

Where,

Option type: 8-bit identifier.

Option data length: 8-bit unsigned integer. It is the length of the data field for this option, in 8-bit groups.

Option data: Variable length field, the data is related to the option type.

The order of the options in the header must be handled in strict accordance with the order in which they appear in the header. The receiver cannot scan the header for a particular type of option and cannot process it before processing all previous options.

The option type identifier is internally defined, and its highest two bits specify what must be done if the node handling IPV9 cannot identify the option type.

00: Skip this option and continue with the header.

01: Abandon the packet.

10: Abandon the packet and send a "ICMP parameter problem, code 2," message to the source address, regardless of whether the destination address is a multicast address, indicating that the option type is not recognized.

11: Abandon the packet. Only when the destination address of the packet is not a multicast address, send a message of "ICMP parameter problem, code 2" to the source address, indicating the type of option it cannot recognize.

The high third bit of the option type specifies whether the data for this option can change on the way to the destination of the packet.

0: Option data cannot be changed during transport.

1: Option data can be changed during transport.

When the authentication header appears in the packet, when calculating the authentication value of the packet, the whole field in which any data can change in the way is treated as an 8-bit group of all zeros.

Each option can have its own alignment requirements to ensure that the values of multiple 8-bit groups in the option data field fall to the natural boundaries. The alignment of the choices is required in terms of Xn+y, the option type must be an integer multiple of x 8-bit groups plus y 8-bit groups from the beginning of the header. Such as:

2n means the offset is any multiple of two 8-bit groups from the beginning of the header.

8n+2 means the offset is any multiple of eight 8-bit groups from the beginning of the header plus two 8-bit groups.

3) Pad option

a) Pad1 option(Alignment requirement: none)

0

Notice: the format of the Pad1 option is a special case where there is no length field or value field. The Pad1 option is used to insert an 8-bit group of fill bits in the header option field.

b) PadN option

If need to fill out multiple 8-bit groups, it should be used the PadN option instead of multiple Pad1 options. The PadN option format (alignment requirement: none) is shown in table 4.

TABLE IV. PADN OPTION FORMAT

1	Option data length	Option data

The PadN option inserts two or more 8-bit groups of fill bits into the header option field. To fill in N 8-bit groups, the value of the option data length field should be N-2, and the option data field contains N-2 all-0 8-bit groups.

4) Segment options header

Segment options header is used to carry the option information that must be checked and processed by all nodes on the path the packet travels through. In IPV9, the header of each network segment option is represented by the value of the next header is 0, as shown in table 5.

TABLE V. SEGMENT OPTIONS HEADER FORMAT

Next header	The extension length of the header	
Options		

Next header：It's an 8-bit selector. It is used to identify the header type just following the each segment header option. It is the same value as the IPv4 protocol field [rfc-1700].

The extension length of the header: It is an 8-bit unsigned integer. The length of the header of each segment option, in 8-bit groups, does not include the first 8-bit group.

Option: variable length field whose length makes the length of each segment header an 8-bit integer multiple. It contains one or more TLV encoding options.

In addition to the Pad1 and PadN options specified above, the large load options (alignment requirement: 4n+2) are defined, as shown in table 6.

TABLE VI. LARGE LOAD OPTION FORMAT

	194	Option data length
Heavy load length		

The large load option is used to send IPV9 packets with a load length of more than 65,535 8-bit groups, the large load length is the length of the packet, in 8-bit groups, excluding IPV9 headers but including segment option headers, it has to be greater than 65,535. If a packet with a large load option is received and the large load length is less than or equal to 65535, a message of "ICMP parameter problem, code 0" is sent to the source node, pointing to the high bit of the invalid large load length field.

If the packet has a large load option, the load length in the IPV9 header must be set to 0. If received a packet that contains both a payload length option and an IPV9 payload length field that is not 0, it need to send a message to the source node with "ICMP parameter has a problem, code 0" and pointing to the large payload option type field.

The large load option cannot be used in packets with segment headers. If the data segment header is encountered in the packet containing the high-load option, a message "ICMP parameter problem, code 0" will be sent to the source node, pointing to the first 8-bit group of the data segment header.

If the installed IPV9 does not support the large load option, it does not have an interface to a link with MTU greater than 65535 (IPV9 header with 72 8-bit groups, plus 4G 8-bit group loads).

5) Routing header

IPV9 USES the routing header to list one or more intermediate nodes that needs to be accessed on the path of the packet to the destination node. This function is very similar to the source routing options of IPv4. The routing header is identified by the next header field whose previous header median value is 43; the format is shown in table 7.

TABLE VII. ROUTING HEADER FORMAT

Next header	Length of the extension header	Routing type	Remaining data segment
Data related to type			

Next header: It's an 8-bit selector. It Use the same value as the IPv4 protocol field [RFC-1700].

Length of the extension header: It is an 8-bit unsigned integer. The routing header length is in 8-bit groups, does not contain the first 8-bit group.

Routing type 0: the route header variable's 8-bit identifier

Remaining data segment: It is an 8-bit unsigned integer. The number of remaining routing data segments, that is, the number of intermediate nodes explicitly listed, which are the nodes to be accessed before reaching the final destination node.

Data related to type: It is a variable length field, whose format is determined by the route type, its length makes the entire route header an integer, multiple of the 8-bit group.

If a node encounters an unrecognized routing type value while processing the received packet, the action that the node will take depends on the value of the remaining data segment field. It includes the following two cases:

a) If the value of the remaining data segment field is 0, the node must ignore the routing header and continue to process the next header of the packet. The header type is given in the header field next to the routing header.

b) If the value of the remaining data segment field is not 0, the node must abandon the packet and send a message of "ICMP parameter exists problem, code 0" to the source address of the packet, pointing to the unrecognized routing type

The format of the route header for type 0 is shown in table 8.

TABLE VIII. TYPE 0 ROUTING HEADER FORMAT

Next header	Length of extension header	Routing Type =0	Remaining data segment
Reserve	Strict/loose bit innuendo		
Address [1]			
Address [2]			
……			
Address [n]			

Next header: It is an 8-bit selector. Its identity follows the routing header type. it USES the same value as the IPv4 protocol field [RFC-1700].

Length of extension header: It is an 8-bit unsigned integer. The routing header length is in 8-bit groups that does not contain the first 8-bit group. For routing headers of type 0, the header extension length is equal to twice the number of addresses in the header and is an even number less than or equal to 46.

Routing type is 0.

Remaining data segment: It is an 8-bit unsigned integer. The number of remaining routing data segments, that is, the number of intermediate nodes explicitly listed, which are the nodes to be accessed before reaching the final destination node. The maximum effective value is 23.

Reserve: It is an 8-bit reserved field. The sender initializes it to 0, and the receiver ignores the field.

Strict/loose bit innuendo: 1 to 23 from left to right. For each routing data segment, indicate whether the next destination address must be the neighbor of the previous node: 1 indicates strict (must be neighbor), 0 indicates loose (need not be neighbor).

Address [1…..n]: It's a 256-bit address vector, value from 1 to n.

Multicast addresses cannot appear in routing header packets of type 0 or in routing header packets of type 0 in the destination address field of IPV9.

If the value of the zero bit of the Strict/loose bit innuendo is 1, the destination address field in the IPV9 header of the original packet must indicate a neighbor of the source node. If the zero bit is 1, the sender can

use any legal, non-multicast address as the initial destination address.

6) Data segment header

IPV9 source hosts use segment headers to send MTU packets that are longer than the packet delivery path. Different from IPv4, in IPV9, only the source host completes the segmentation, rather than the router on the path of the packet. The data segment header is identified by setting the next header to 44 in its previous header, as shown in table 9.

TABLE IX. DATA SEGMENT HEADER FORMAT

Next header	Reserve	Data segment offset	Reserve	M
Identifier				

Next header: It is an 8 bit selector. The initial header type (defined below) that identifies the data segment portion of the original packet. Use the same values as the IPv4 protocol fields [RFC-1700].

Reserve: It is an 8-bit field. It is initialized to zero at the sender and ignored at the receiver.

Data segment offset: It is an 8-bit field. The number of bytes moved forward or backward from the specified position.

M: Flag 1 indicates that there are more data segments, and 0 indicates the last data segment.

Identifier: It has 32-bit field. In order to send MTU packets whose length is longer than the transmission path, the source node can split the packet into several data segments and send each data segment as a separate data packet, and then reassemble the data packet at the receiver.

For each packet to be segmented, the source node generates an identifier value for it. Any piece of data in any recently delivered packet with the same source and destination addresses must have a different identifier. If a routing header is present, the destination address being considered is the final destination address.

7) Destination options header

The destination option header is used to carry the option that only needs to be checked by the packet's final node. The destination option header is identified by the header before it, with the next header field value of 60; the format is shown in table 10.

TABLE X. HEADER FORMAT OF DESTINATION ADDRESS OPTIONS

Next header	Length of extension header	
Option		

Next header: Bit selector. This is an identifier type of header that follows the destination option header. Use the same value as the IPv4 [RFC-1700].

Length of extension header: It is an 8-bit unsigned integer. The destination option header length is in 8-bit groups, and does not contain the first 8-bit group.

Option: It’s a Variable length field, whose length makes the destination option header length an integer multiple of 8-bit group. It contains one or more TLV encoding options.

Optional destination information in IPV9 packets is encoded in two ways: defined in the destination options header, or as a separate extended header. Data segment headers and authentication headers are two examples of the latter. Which one to take is depends on the action if the destination node could not recognize the option information.

a) If the destination node operation is want to abandon packets, and only in the destination node address of the packet is not the multicast address, then send the packet source address a "ICMP Unrecognized Type" message, and then these messages can be encapsulated into a separate header, or destination option in a header option, and the highest two digits of the option type are 11. This choice depends on a number of factors, such as fewer bytes, better alignment, or easy to parse.

b) If both operations are required, the messages must be encoded as an Option at the head of the destination Option, whose Option type has a highest two digits is 00, 01, or 10, specifying which actions will be take.

Note: When the next header field of an IPV9 header or any extended header is 59, which means there's nothing behind the header. If the IPV9 header payload length field indicates that there are 8-bit groups after the next header field of 59, these 8-bit groups must be ignored, and the content is passed as is when the packet must be forwarded.

III. PACKET LENGTH DESIGN

IPV9 requires a minimum of 576 MTU per link on the Internet. On any link, if it cannot pass 576 8-bit groups in one packet, then the data segment and reassembly associated with the link must be supported by the hierarchy below IPV9.

For each link directly connected to the node, the node must be able to accept packets as large as MTU. Links with configurable MTU (such as PPP links [RFC-1661]) must be configured with at least 576 8-bit groups, and larger MTU is recommended to accommodate possible encapsulations (such as tunnels) without fragmentation.

IPV9 nodes are recommended to implement Path MTU Discovery [RFCc-1191] in order to discover and take advantage of MTU links larger than 576. However, a minimal IPV9 implementation (such as in a BootROM) can simply restrict itself from sending packets larger than 576 and omit the Path MTU Discovery implementation.

In order to send MTU packets with a length greater than the link, the node can segment the packet at the source node and assemble it at the destination node by using the data segment header of IPV9. However, this fragmentation is not recommended in any application unless it can resize packets to fit the MTU of the link being measured.

A node must be guaranteed to accept segmented packets that exceed 1500 bytes after reassembly, including IPV9 headers. However, a node must ensure that it does not send segmented packets larger than 1500 bytes after reassembly unless it is explicitly told that the destination node can assemble such a large packet.

When sending an IPV9 Packet to an IPv4 node (that is packets go through the transition from IPV9 to IPv4)), the IPV9 source node may receive an "ICMP Packet TOO Big" (ICMP Packet is TOO Big) message reporting that next-hop MTU must be less than 576. In this case, IPV9 does not need to reduce the size of subsequent packets to less than 576, but must include a segment header in those packets so that the IPV9-IPv4 conversion router can obtain an appropriate identifier value for the constructed IPv4. This means that the load can be reduced to 496 8-bit groups (576 minus 72 bytes for IPV9 headers and 8 bytes for data segment headers) or even smaller if additional extended headers are used.

In order to send MTU data packets whose length is longer than the link, such as audio, image and video, long-stream code and absolute return code can be selected. The node can use the data segment header of IPV9 to identify the data packets in the source node without segmentation and assemble them in the destination node. However, when the sender and the receiver receive the signal disconnected by the return code, they will return to normal working condition.

Note: unlike IPv4, IPV9 does not require a "Don't Fragment" flag in the packet header to perform Path MTU Discovery, which is an implicit feature of IPV9. And the process associated with using MTU in RFC-1191 is not applicable to IPV9, because the message of IPV9 "Datagram Too Big" is always identifies the exact MTU being used.

Also, the procedures associated with the use of MTU tables in rfc-1191 are not applicable to IPV9 because the IPV9 version of the "Datagram Too Big" message always identifies the exact MTU being used.

Unlike IPv4 and IPv6, IPV9 can transmit the practical applications such as audio or video, it need to use the ever-flowing code and absolute return code, thus formed in the reserved the actual circuit actually has become a three layer structure, so there is no try to transfer the concept of content as guaranteed delivery channels and reliable safety, guarantee the transmission content don't interrupt. This results in the co-existence of the three - and four-tier architectures within the IPV9 network.

IV. FLOW LABEL

A data flow is a sequence of packets sent from one source to another destination address (point-to-point or multicast), and the source nodes require the intermediate router to have special control over these packets. These special processes can be transferred to routers through control protocols, such as resource reservation protocols, or through the information carried by the packets themselves in the data stream, such as segment options.

There may be multiple active streams between a pair of source and destination nodes, as well as many communications independent of any flow. A flow is uniquely identified by a combination of a source address and a non-zero flow label. The flow label field for packets that do not belong to any flow is set to 0.

The flow label is assigned by the source node of the data flow. New flow tags must be randomly selected (pseudo random), ranging from 1 to 16777215 (decimal). The purpose of the random assignment is to make the bits in the flow label suitable for use as hash keys in routers to find the relevant state of the flow.

All packets belonging to the same flow must be sent with the same source node address, destination node address, priority, and flow label. If any of these packets contain a segment option header, they must all have the same segment option header content. If any of their packets contain a routing header, all of their extended headers preceding the routing header must have the same content, including the routing header (except for the header field next to the routing header). Allows, but does not require, the router and destination node to check whether the above requirements are met. If a collision is detected, it should sending "ICMP parameter has a problem, code 0" message and then pointing to the high bit of the flow label (i.e., within the IPV9 packet with an offset of 1).

Routers are free to set the flow control state of based on "timing" even when there is no explicit flow control protocol, segment options, or other methods provide them with flow creation information. For example, when a flow label with an unknown, non-zero label is received from a particular source node, the router can process its IPV9 header and any other necessary extension headers as if the flow control label were 0.

The flow control state described above, after being set and cached according to "timing" must be discarded within 6 seconds, whether or not packets of the same class continue to arrive. If another packet with the same source address and flow control label arrives after the cache state has been discarded, then the packet must undergo full normal processing (as if the flow control label is 0), this process may cause the flow control state of the flow to be re-established and cached.

The lifetime of explicitly established flow control states, such as flow control states created by control protocols or segment options, must be specified as part

of the explicit establishment mechanism and can exceed 6 seconds.

During the lifetime of any flow control state that was previously created, the source node must not use this control label for new flows. Since any flow control state created depending on "timing" has a lifetime of six seconds, the minimum time between the last packet of a flow and the first packet of a new flow to use the same flow label is six seconds. The flow label has a longer lifetime and cannot be reused for new flows during the lifetime.

When a node is off and restarted (for example due to a system crash), care must be taken to avoid using flow label that it might have used for previously created flow that have not yet expired.

This can be achieved by record flow label in the memory, so that it can recall flow label previously used after a system crash, or until the previously created, there may be one of the biggest lifetime timeout before does not use flow label (at least for 6 seconds, if it use an explicit flow and establish a mechanism, and specifies the longer life span, even longer time). If the reboot time of a node is known (usually more than 6 seconds), the amount of time to wait before starting to allocate flow tags can be deducted accordingly.

V. CATEGORY TYPE DESIGN

The 8-bit Category Type field in the IPV9 header enables the source node to identify the desired level of packet delivery determination, certainly relative to other packets sent from the same node. Category Type bits contain two parts: 3 bits high is used to specify the address length, the value is 0 ~ 7, is 2 to the power, the address length is 1Byte ~ 128Byte; the default value is 256 bits, where 0 is 16 bits, 1 is 32 bits, 2 is 64 bits, 3 is 128 bits, 4 is 256 bits, 5 is 512 bits, 6 is 1024 bits, and 7 is 2048 bits. The last five bits specify two ranges of communication categories, with values 0 ~ 7 used to specify the information priority provided by the source node for congestion control, that is, the priority of information sent with a delay in the face of congestion, such as TCP information. 8 ~ 15 are used to specify the priority of messages that are sent without delay in the face of congestion, that is, the priority of "real time" packets that are sent at a fixed rate.

For crowd-constrained information, the following priority values can be used for specific application classes.

0: Non-character information,

1: Fill in the information (such as: net news),

2: Unattended information (such as: Email),

3: Reserve,

4: Large quantities of supervised information (such as: FTP、NFS),

5: Reserve,

6: Interactive information (such as: Telnet, X),

7: Internet control information (Such as: Routing protocol, SNMP),

8: For audio,

9: For video,

10: For video or audio compression will not be error due to alignment error,

11: Broadcast with audio and video,

12: Emergency use.

For messages that are not congested, the lowest rating value of 8 should be used for packets that the sender most wants to discard in crowded conditions (such as high-fidelity video messages), and the highest rating value of 15 should be used for packets that the sender least wants to discard (such as low-fidelity audio messages). There is no corresponding sequential relation between the rank of un-crowded and the priority of crowded.

VI. UPPER PROTOCOL DESIGN

A. Upper protocol check

If any transport protocol or other upper layer protocol includes the address in the IP header when calculating its checksum, then in order to be able to run on IPV9, the algorithm that calculates the checksum must be modified to include addresses with a length of 16-2048 bits rather than 32-bit IPv4 addresses. TCP and UDP headers for IPV9 are shown in table 11.

TABLE XI. TCP AND UDP HEADERS FOR IPV9

Source address	
Destination address	
Time	
Identify code	
Payload Length	
0	Next header

1) If the packet contains the routing header, the destination address in the pseudo-header is the final address. In the source node, this address is the last address in the routing header; at the receiver, this address will be in the IPV9 header address field.

2) The value of the next header in the pseudo-header identifies the upper protocol (e.g., TCP is 6, UDP is 17). If there is an extended header between the IPV9 header and the upper protocol header, the value of the next header in the pseudo-header is different from the value of the next header in the IPV9 header.

3) The value of the load length field in the pseudo-header is the length of the upper protocol packet, including the upper protocol header. If there is an extended Payload between the IPV9 Payload and the upper protocol Payload it will take less Payload Length than the IPV9 Payload Length (or in the large Payload option).

4) Different from IPv4, when a UDP packet is sent from an IPV9 node, UDP checksum is not optional. That is, whenever an IPV9 node sends a UDP packet, it must calculate the UDP checksum. The checksum is generated by the packet and pseudo-header, and if the result is 0, it must be converted to hex FFFF and placed in the UDP header. The IPV9 receiver must abandon the UDP packet containing the checksum 0 and record the error.

The checksum of ICMP version of IPV9 includes the above pseudo-header in its verification and calculation. This is a modification of the IPv4 version of ICMP, which does not include a pseudo-header in its verification and calculation. This change is made to ensure that ICMP is not passed incorrectly or corrupted by the IPV9 header fields on which it depends, which, unlike IPv4; these fields are not checked and protected by the Internet layer. The next header field in the pseudo-header of ICMP contains the value 58, which identifies the IPV9 version of ICMP.

B. Maximum packet lifetime

Unlike IPv4, IPV9 nodes like IPv6 do not require a mandatory maximum packet lifetime. This is why the "lifetime" field of IPv4 has been renamed IPV9's "segment limit".

In practice, very fewer IPv4 complies with the current packet lifetime, so this is not a practical change. Any protocol that relies on the Internet layer (whether IPV9 or IPv4) to limit the lifetime of packets should be upgraded to rely on its own mechanism to detect and discard stale packets.

C. Maximum upper layer load

When calculating the maximum load available for upper level protocol, it must be taking into account that the IPV9 header is larger than the IPv4 header.

For example, in IPv4, TCP's MSS option is calculated by the maximum datagram length (the default value obtained through Path MTU Discovery) minus 40 8-bit groups (20 8-bit groups for the minimum length of IPv4 headers and 20 for the minimum length of TCP headers) from.

When using TCP over IPV9, the MSS must be calculated by the maximum length minus 60 8-bit groups, because the minimum IPV9 header (when

IPV9 without an extended header) is 20 8-bit larger than the minimum IPv4 header.

VII. FORMAT OF OPTIONS

It is required to design the fields first when designing new options for segment option headers or destination option headers; these are based on the following assumptions.

1) Any field in the option data of an option that consists of multiple 8-bit groups should be aligned with their natural boundaries, that is, fields with n 8-bit groups of width should be placed at integer multiples of n 8-bit groups from the beginning of the segment header or destination option header, where n= 1,2,3,4 or 8.

2) Each segment header or destination option header takes up as little space as possible and must meet the requirement that the header length is an integer multiple of an 8-bit group.

3) It can be assumed that when any header with options appears, they only have a fewer options, usually only one.

These assumptions mean that it needs planning the individual fields of an option, arrange the fields from smallest to largest, with no padding in the middle, and then derive the alignment requirements for the entire field based on the alignment requirements for the largest field.

Examples are given below.

Case1. If option X requires two data fields, one with a length of 8 8-bit groups and one with a length of 4 8-bit groups, it should be arranged according to table 12.

TABLE XII. TWO FIELD DESIGN TABLE

	Option Type =X	Option data length =12
Four 8-bit group fields		
Eight 8-bit group fields		

Its alignment requirement is 8n+2 to ensure that the eight 8-bit fields start with an 8-fold offset from the header. The full segment header or destination header with the above options is shown in table 13.

TABLE XIII. FULL HEADER OR DESTINATION HEADER FORMAT

Next header	length of extension header =1	Option Type =X	Option data length =12
Four 8-bit group fields			
Eight 8-bit group fields			

Case2. If option Y requires three fields, one with a length of four 8-bit groups, one with a length of two 8-bit groups, and one 8-bit group, the format is shown in table 14.

TABLE XIV. THREE FIELD DESIGN FORMAT

		Option Length=Y
Option data length =7	One 8-bit group fields	Two 8-bit group fields
Four 8-bit group fields		

Its alignment requirement is 4n+3 to ensure that the four 8-bit leader fields start at a 4 times offset from the header. The full hop-by-hop or destination option header with the above options is shown in table 15.

TABLE XV. A THREE-FIELD FULL DATA FORMAT

Next header	length of extension header =1	Pad1 option =0	Option Type =Y
Option data length =7	One 8-bit group field	Two 8-bit group fields	
Four 8-bit group fields			
PadN option=1	Option data length =2	0	0

Case3. The segment header or destination header for each option X and option Y in both case 1 and case 2 should be one of the following two formats, depending on which option appears first, as shown in tables 16 and 17.

TABLE XVI. ONE FORMAT OF CONTAIN BOTH TWO - AND THREE-FIELD ADDRESS

Next header	length of extension header =3	Option Type=X	Option data length =12
Four 8-bit group fields			
Eight 8-bit group fields			
PadN Option =1	Option data length =1	0	Option Type=Y
Option data length =7	One 8-bit group fields	Two 8-bit group fields	
One 8-bit group fields			
PadN option=1	Option data length =2	0	0

TABLE XVII. ANOTHER FORMAT OF CONTAIN BOTH TWO - AND THREE-FIELD ADDRESS

Next header	length of extension header =3	PadN option=1	Option Type=Y
Option data length =7	One 8-bit group fields	Two 8-bit group fields	
Four 8-bit group fields			
PadN option=1	Option data length =4	0	0
0	0	Option Type =X	Option data length =12
Four 8-bit group fields			
Eight 8-bit group fields			

VIII. ENCAPSULATE SECURITY PAYLOAD HEADER DESIGN

A. Format of encapsulate security payload header

ESP (Encapsulating Security Payload) Header is designed to provide mixed security services in IPv4. The ESP mechanism can be applied with the authentication header or in a nested manner in tunnel mode alone. Security services may be provided between a pair of communicating hosts, or between a pair of communicating security gateways, or between a security gateway and a host.

The primary difference between the authentication header and the ESP mechanism is the effective area service. The encapsulation security payload mechanism does not protect any IP header fields unless they are encapsulated by the ESP, such as in tunnel mode where the IP header is encapsulated underneath.

The encapsulation security header is inserted after the IP header. In transport mode, the encapsulated security header is in front of the upper layer protocol header, and in tunnel mode, the encapsulated security header is in front of the encapsulated IP header.

ESP mechanisms provide services such as confidentiality, data origin authentication, connectionless integrity, anti-replay services (a form of partial sequence integrity), and limited communication confidentiality. The business provided by this mechanism depends on the options and the location of the application when the security association is established.

Confidentiality can be independent of other business options. However, the use of confidentiality alone without integrity authentication can lead to attacks that compromise the confidentiality of communication. Data origin authentication and connectionless integrity are federated services that can be provided as an option along with confidentiality services. The anti-replay service can only be selected if the data origin authentication service is selected, which is entirely up to the receiver.

The confidential service requires selection in tunnel mode, and this service is most effective when it used in the security gateway, because the clustering of communication on the gateway may mask the true source and host address modes. Note that while both confidentiality and authentication services are optional, at least one of them should be chosen.

A protocol header (IPv4 header, IPV9 base header, or extended header) that precedes the ESP header will have a value of 50 in its protocol field (if IPv4 header) or in its next header field (if it is the IPV9 extended header). The format of ESP groups and headers is shown in table 18.

TABLE XVIII. FORMAT OF ESP GROUPS AND HEADERS

0 8 16 24 31		
Security Parameters Index（SPI）		
Sequence Number		
Payload Data（variable length）		
	Fill field（0~255B））	
	Pad Length	Next Header
Authentication Data（variable length）		

Note: the scope of the certification business is the part before the certification data (authentication data is not included); the scope of the encryption service provided is the portion of the data that follows the serial number and precedes the authentication data (Serial Numbers and authentication data are not included).

B. Description of safe load format

The fields in the header format are explained below. The "optional" of the text indicates that if the field is not selected, the field is ignored and is not used when calculating the overall check value. If "required", this field must appear in the ESP group.

1) Security Parameters Index（SPI)

It is a required field of 32 bits. This field is associated with the security of the datagram that uniquely identifies the address of the address IP and the security protocol. The value of the SPI can be any, currently from 1 to 255 is reserved by IANA (). The value 0 of SPI is reserved for local, specific application use.

2) Sequence Number

It is a 32bit monotonically increasing counter (serial number). This field is required even if the receiver does not select to enable the playback service for a particular security association. The processing of this sequence number field is entirely done by the receiver, that is, the sender must transmit this field, and the receiver may or may not comply with the field.

When a security association is established, both sender and receiver counters are set to 0. If the anti-replay is started (default is enabled), the serial number

transferred does not allow loops. Therefore, after a secure association group, the sender and receiver counters must be reset.

3) Payload Data

It is a variable-length field that contains the data described by the next header field. The payload field is required and is an integer multiple of bytes in length.

4) Fill field

This field is used for encryption. The purposes of use fill fields in the ESP header are as follows.

a) If an encryption algorithm requires the body to be an integer multiple of bytes, the padding bytes are used to the body. (In addition to the filling fields themselves, the payload data, the filling length, and the next header fields are also included) to meet the data length requirements of the encryption algorithm.

b) Even *without considering the requirements of the encryption algorithm, fields need to be filled in to ensure the length of the* encrypted *data terminates at the boundary of 4B.* In particular, the length of the fill length field and the next header field must be aligned to 4B.

c) Apart from above algorithmic and alignment requirements, padding fields may also be used to hide the true length of the payload and partially encrypt the communication. However, this additional padding obviously consumes bandwidth resources and should be used with caution.

The sender can add 0 to the 255B to the fill field. Padding is optional in an ESP group, but all applications must support the generation and use of padding fields to satisfy the encryption algorithm for the length of the encrypted data while ensuring that the authentication data is aligned to the 4B boundary.

d) Pad Length

This field is required, and the valid fill length value should be from 0 to 255, with 0 indicating no fill bytes.

e) Next header

This field is required and is an 8-bit field indicating the data type in the payload data field.

f) Authentication Data

It is a variable-length field that contains the group's Integrity Check Value (ICV), which is calculated from the ESP group except the authentication data. This field is optional and only occurs if the authentication business is included in the security association. The authentication algorithm must account for the full length of the verification value and the relative rules of validation and processing steps.

C. Processing of security payload header

Encapsulated security payloads (ESP) are used in two ways: transport mode or tunnel mode.

1) Transmission mode

Transmission mode applies only to host applications. In this mode, the ESP header only protects the upper layer protocol and not the IP header field. In this mode, the ESP header is inserted after other IP headers and before the upper layer protocols of TCP, UDP, and ICMP. In IPV9, the ESP header is treated as an end-to-end payload, so the header must appear after the hop-to-hop, route, and segment headers.

The host option header may appear before or after the ESP header, depending on the semantics required. The locations of the ESP headers in a typical IPV9 packet in transport mode are shown in tables 19 and 20.

TABLE XIX. DATAGRAMS BEFORE THE APPLICATION OF ESP HEADER

Basic header	Extension header(if any)	TCP	Data

TABLE XX. DATAGRAM AFTER THE APPLICATION OF ESP HEADER

Basic header	Hop-to-hop, Destination header Route header Segment header	ESP	Destination Options header	TCP	Data	ESP Trailer	ESP authentication

The encrypted portion of the above packet can be a basic header encryption or a host option header, TCP, data, and ESP tail. The authenticated part in addition to the above part is encrypted, but also the package security load.

2) Tunnel mode

The ESP header in tunnel mode can be used for host or security gateway. Tunnel mode must be used when the ESP header is applied to the security gateway to protect the user's transmission communication.

In tunnel mode, the "lower" IP header carries the final source and destination address, while the "upper" IP headers contain the other addresses, such as the address of a security gateway.

In tunnel mode, the ESP header is positioned relative to the "upper" IP header as it is in transport mode. The position of the ESP header in a typical IPV9 packet in tunnel mode is shown in table 21.

TABLE XXI. ESP HEADERS IN TYPICAL IPV9 PACKETS IN TUNNEL MODE

upper Basic header	Upper Extension header (if any)	ESP	lower Basic header	Upper Extension header (if any)	TCP	Data	ESP Trailer	ESP authentication

In the above group, the encrypted part can be the upper basic header, the lower basic header, the lower extended header, TCP, data and ESP. The authenticated part in addition to the above part is encrypted, but also the ESP.

IX. CONCLUSION

This paper is a specific research and design scheme of RFC1606 and RFC1607 for the future network. The 42-layer routing address space is described according to the document in RFC1606. IPV9 has a routing hierarchy of up to 42 layers, and this routing hierarchy is a key feature in its wide application.

In order to protect previous investments, IPv4 and IPv6 compatible addresses have been set inside, with layers 1-41 designed for IPv4 and IPv6 compatibility and layer 42 described in the RFC1606 document. The large number of address Spaces in IPV9 also makes it possible to allocate addresses in a direct way

In order to the application of IP mobile, IPTV, IP phone, Internet of things and other network applications that need to use Arabic numerals to represent and need to use characters that do not have to be analyzed again, this design also designed a character router.

IPV9 address length is designed according to the document of RFC1607 that the network address length is 1024 bits in the future network, and the address space length is designed as 2048 bits according to the actual demand, thus solves the address space capacity problem in the next 750 years.

In order to meet the technical demand of RFC1606 and RFC1607, the definitions of routing hierarchy, address length, address working mode, address space

resource, address text representation method, compression definition and separator were redefined, Please refer to other related articles of this design team.

REFERENCES

[1] Tang Xiaodan etc. Computer Operating System (third edition) [M]. Xi'an: Xidian university press, 2010.

[2] Xie Jianping etc. A method of assigning addresses to network computers using the full decimal algorithm [P]. CN: ZL00135182.6, 2004.2.6.

[3] Xie Jianping etc. Method of using whole digital code to assign address for computer [P]. US: 8082365, 2011.12.

[4] RFC - Internet Standard. Internet Protocol, DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION, RFC 791, 1981.09.

[5] S. Deering, R. Hinden, Network Working Group. Internet Protocol, Version 6 (IPv6)-Specification, RFC-1883, 1995.12.

[6] M. Crawford. Network Working Group. Transmission of IPv6 Packets over Ethernet Networks. RFC-2464, 1998.12.

[7] J. Onions, Network Working Group. A Historical Perspective on the usage of IP version 9. RFC1606. 1994.04.

[8] V. Cerf, Network Working Group. A VIEW FROM THE 21ST CENTURY, RFC1607. 1994.04.

[9] Xie Jianping, Xu Dongmei, etc. Digital domain name specification. SJ/T11271-2002, 2002.07.

[10] Information technology-Future Network- Problem statement and requirement-Part 2: Naming and addressing, ISO/IEC DTR 29181-2, 2014, 12.

[11] Wang Wenfeng, Xie Jianping, etc. Product and service digital identification format for information procession. SJ/T11603-2016, 2016. 06.

[12] Radio frequency identification tag information query service network architecture technical specification. SJ/T11606-2016, 2016. 06.