electronics

MDPI

Article

MultiSec: A Multi-Protocol Security Forwarding Mechanism Based on Programmable Data Plane

Zeying Liu [1], Pengshuai Cui [1,*], Yongji Dong [1], Lei Xue [2] and Yuxiang Hu [1]

[1] Institute of Information Technology, PLA Strategic Support Force Information Engineering University, Zhengzhou 450002, China; zeyingl698@gmail.com (Z.L.); yongjid@gmail.com (Y.D.); chxachxa@126.com (Y.H.)
[2] Zhejiang Lab, Research Institute of Intelligence Network, Hangzhou 310000, China; xuel@zhejianglab.com
* Correspondence: cuipengshuai@nudt.edu.cn

Abstract: With the development of network technology, various network protocols different from TCP/IP have emerged. The heterogeneous integrated network has been proposed to realize the interconnection between heterogeneous networks running different protocols. However, current protocol conversion mechanisms often can only handle a few pre-defined protocols and do not support the flexible expansion of new protocols, which cannot meet the needs of the efficient convergence of different heterogeneous networks. Addirionally, due to the lack of security mechanisms, data in the core network is confronted with the risk of stealing and tampering. Our aim is to provide a protocol-extensible protocol conversion and secure transmission integration mechanism, MultiSec, for heterogeneous converged networks. First, based on the programmable data plane, the parser is reconfigured to realize multi-protocol parsing. Furthermore, the encryption mechanism implemented in the P4 extern is proposed and unified to the data plane together with the protocol conversion mechanism. Finally, the MultiSec prototype is implemented on a programmable software switch and accelerated by a dedicated encryption card. Experiments show that MultiSec successfully realizes multi-protocol conversion and data encryption, and the system performance is significantly improved with the help of an encryption card.

Keywords: MultiSec; protocol conversion; VPN technology; programmable data plane; heterogeneous integrated network.

Citation: Lastname, F.; Lastname, F.; Lastname, F. MultiSec: A Multi-Protocol Security Forwarding Mechanism Based on Programmable Data Plane. *Electronics* **2022**, *11*, 2389. https://doi.org/10.3390/electronics11152389

Academic Editor: Irene Moser

Received: 19 June 2022
Accepted: 26 July 2022
Published: 30 July 2022

1. Introduction

With the development of various network applications and the extensive access of various heterogeneous terminals, the current TCP/IP protocol struggles to meet the requirements of various applications on bandwidth, delay, power consumption, mobility, and so on. New protocols such as Named Data Networking(NDN) [1], MobilityFirst(MF) [2], and GeoNetWorking(Geo) [3] have emerged to meet users' application needs. The NDN protocol uses content as the addressing condition and caches the data through a switch, making data transfer faster and improving the efficiency of content retrieval. The MF protocol uses name-based routing to support not only endpoints but also network mobility, i.e., migration of entire networks. The Geo protocol uses geographical positions for packet transport to support the communication among individual Intelligent Transport Systems (ITS) [4] stations as well as the distribution of packets in geographical areas.

Heterogeneous protocols [5] meet the needs of some network applications. However, they also encounter problems when trying to connect with the Internet due to incompatible protocols. Thus, protocol conversion mechanisms are proposed to interconnect heterogeneous networks with each other. A heterogeneous integrated network usually consists of a core network and multiple edge heterogeneous sub-networks. The sub-networks with new protocols are connected to the core network through gateways. Take, as an example, heterogeneous subnet A and heterogeneous subnet B to show the network diagram of

the heterogeneous integrated network, as shown in Figure 1. Specifically, the data in heterogeneous subnet A are converted into the IP protocol after passing through gateway A and then to the core network. Then, the data is restored to the original protocol through gateway B and sent to the receiving terminal in heterogeneous subnet B.

There are many protocols in the heterogeneous network. Supposing that different protocol conversion devices are deployed to support communication between heterogeneous subnets and the core network, there should have been a variety of gateways, and it will become highly bloated as new protocols emerge. Nick McKeown et al. propose the Programmable Protocol-independent Packet Processor (P4) [6] and the corresponding forwarding model [7,8]. With the data plane programming capabilities that P4 brings, network devices can be more flexible in handling multi-protocol packets [9]. However, most research on protocol conversion mechanisms primarily targets reducing design cost and improving conversion speed rather than implementing multi-protocol conversion [10].

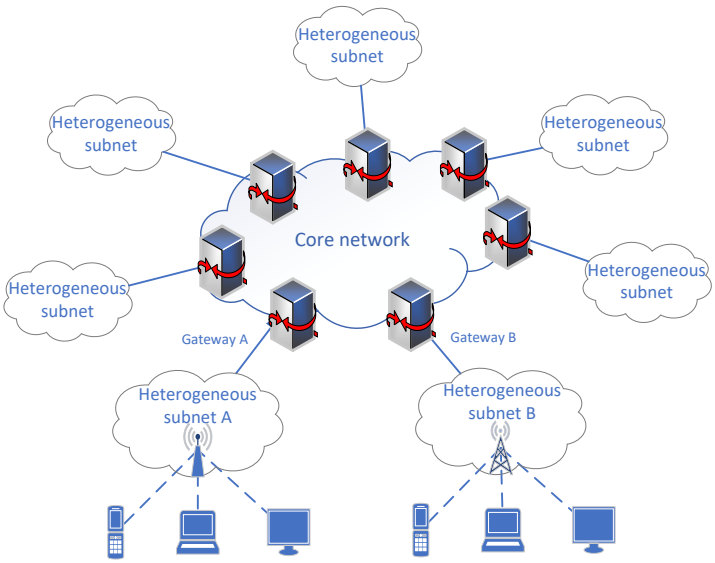

Figure 1. The heterogeneous integrated network.

Many new methods are introduced into the protocol conversion to provide more convenience for network convergence. However, the lack of consideration of security issues makes IP-based communication vulnerable to tapping and forgery of external information, resulting in an insecure and unreliable network [11]. Encrypted communication technologies [12] such as Multi-Protocol Label Switching (MPLS), Secure Sockets Layer (SSL), and Internet Protocol Security (IPsec) solve the data security issues at different layers in the network. However, they are protocol-dependent and cannot provide a general-purpose encryption mechanism for packets of other protocols.

To address the issues mentioned above, we propose MultiSec, a multi-protocol security forwarding mechanism based on programmable data plane, which successfully implements a prototype for the Polymorphic Processing Kit (PPK) P4 software switch (see Section 4 for details) and conducts a performance evaluation. MultiSec flexibly modifies the protocol parsing process according to the input protocol type and can process any protocol. MultiSec combines protocol conversion and encryption mechanisms, realizing the integration of protocol conversion and secure forwarding. We refer to devices running MultiSec as secure forwarding gateways that support multi-protocol access or security gateways. Every security gateway implements the same MultiSec functionality, i.e., encryption, decryption, and conversion between private and IP protocols, which facilitates flexible deployments. The implementation of MultiSec has made essential contributions to developing heterogeneous converged networks and improving reliable data transmission in the core network.

The major contributions of this paper are as follows:

1. We analyze the problem that existing protocol conversion mechanisms can only process specific protocols and propose a programmable multi-protocol conversion mechanism. By reconfiguring the programmable parser and designing the packet header-caching method, this mechanism realizes the interconnection between heterogeneous networks with multiple addressing methods and the Internet.
2. Encryption, decryption, and authentication functions are added to the P4-based programmable data plane. We integrate protocol conversion and secure forwarding functions by designing a recirculation mechanism. Security gateway configuration files are leveraged in the Software Defined Network (SDN) controller to distribute and update Security Association (SA) dynamically.
3. We build a heterogeneous network test environment with the help of a virtual heterogeneous network. We evaluate the implementation of the PPK P4 software switch. The experiments show that the goodput increases by 20 times, and transmission latency is reduced by two orders of magnitude through hardware acceleration.

The rest of the paper is structured as follows. We review the relevant work in Section 2. Section 3 gives the architecture and details of MultiSec. The experiments are analyzed in Section 4. Finally, Section 5 gives the conclusion and prospects of this work.

2. Relevant Work and Background

2.1. Protocol Conversion Mechanism

In recent years, there has been an increasing amount of literature on hardware-based protocol conversion mechanisms. In terms of performance improvement, the ACR128 protocol conversion motherboard is used to develop a protocol converter based on Modbus and NMEA0183, which effectively reduce the time required for industrial monitoring devices to analyze communication protocols [13]. Based on this, Ananda, M.P. et al. [14] implement a low-cost protocol converter by using a generic socket modulator–demodulator. A protocol conversion system is constructed through the BF548 and universal media interface devices, which can provide connectivity between analog public-switched telephone network (PSTN) cellular, WIFI, and Bluetooth. In terms of applicability, a multi-layer system-on-chip protocol conversion algorithm [15] is proposed. For different layers of the Open System Interconnection (OSI) model, it is allowed to generate realizable converters for IP protocols by using any existing converter generation algorithm. Similarly, Nie et al. [16] present a Field Programmable Gate Array (FPGA)-based Ethernet and G. SHDSL protocol converter, which adopts MAC soft core to process Ethernet data frame and enables Ethernet users to access the local area network through a single twisted pair over long distances. However, the above research studies on protocol conversion mainly focus on performance optimization, such as reducing latency and costs, and do not realize multi-protocol conversion. In the study of multi-protocol conversion, Luo [17] investigates the bridge between Fibre Channel (FC) protocol and other network protocols based on network virtualization, where an overall network architecture with FC as the primary network is designed to apply the advantages of FC to large embedded systems. This scheme needs to design mapping tables for two virtualization nodes in different networks, which is not conducive to the flexible deployment of the new protocol. Wang [18] implements multi-protocol conversion between various industrial bus protocols based on FPGA. A new architecture is proposed, which is dedicated to meeting the needs of multi-protocol conversion. However, it can only process a few specific protocols and cannot process complex private protocols. Currently, most hardware-based multi-protocol conversion mechanisms can only deal with limited protocols. The extraction order of data fields is also fixed in the hardware and cannot be modified, which cannot meet the needs of multi-protocol parsing in heterogeneous integrated networks.

The main methods to realize protocol conversion include the direct mapping method [19,20] (DMM for short), the mapping method based on intermediate descriptors [21,22] (MMID for short), and the protocol conversion matrix [23](PCM for short). DMM is generally implemented using hardware. It first searches for the target protocol

field information corresponding to the input field in the storage module according to the field mapping relationship between the original protocol and the target protocol. Then, the corresponding field information is generated according to the matching result. Finally, together with the payload, a new target protocol packet is formed. However, when the number of protocols in the system exceeds three, the processing efficiency of this method decreases significantly. MMID adds intermediate descriptors to the direct mapping method to solve this problem. Specifically, the original protocol is first converted into intermediate descriptor information and then mapped to the destination protocol. The method can improve the overall efficiency of the system during multi-protocol conversion. However, the conversion efficiency of each pair of protocols is reduced by half due to the need for secondary mapping. PCM designs a parameter matrix for protocol conversion by analyzing the characteristics of the protocols between the two sides of the communication. The method first abstracts each component of the protocol into a set of parameters then calculates the product of the set of parameters and the protocol conversion matrix to obtain the parameters corresponding to the target protocol, and finally generates the target protocol. Since both protocols need to meet the requirement of being able to abstract the parameter matrix, this method does not have broad applicability. Aiming at the above problems, we propose a protocol conversion method based on the programmable data plane (PCPDP for short). This method realizes the parsing of any protocol by reconfiguring the programmable parser. It implements protocol conversion by encapsulating the entire heterogeneous protocol packet as part of the payload into an IP packet in the network layer, simplifying the conversion complexity. We compare the pros and cons of the above methods in Table 1.

Table 1. Comparison of four protocol conversion methods.

Method	Mode	Protocol	Flexibility	Complexity
DMM	Mapping	Fixed	Low	High
MMID	Mapping	Fixed	Low	High
PCM	Parameter Matrix	Fixed	Low	High
PCPDP	Encapsulation	Any	High	Low

The DMM, MMID, and PCM methods convert all the key fields of the original protocol to the target protocol by means of mapping and parameter matrix, which is a complicated conversion process. In addition, the above methods are usually done in hardware and can only handle specific protocols with low flexibility. PCPDP is designed to enable the transmission of multiple heterogeneous protocol packets across the core network and does not involve complex mapping algorithms. Therefore, compared with other methods, PCPDP can achieve flexible expansion of new protocols with lower engineering overhead.

2.2. Encrypted Communication Technology and Mechanism

The rapid development of Virtual Private Network (VPN) technology further ensures data security. MPLS is a technology that uses tags to guide the efficient transmission of data. Xia [24] ensures the end-to-end service quality of MPLS-VPN users by mapping the IP Precedence and Differentiated Services Code Point (DSCP) priorities in data traffic to the EXP field in MPLS. On the other hand, scholars do a lot of research on the performance optimization and security improvement of security protocols. A high-performance SSL VPN system scheme based on Data Plane Development Kit(DPDK) technology and user mode network protocol stack is proposed [25], which significantly improves the transmission performance of the SSL VPN system. Similarly, Ren [26] optimizes both authentication and session key transmission in the traditional SSL protocol, enhancing the attack resistance and confidentiality of SSL. However, the above research does not expand new application scenarios, and the innovation is slightly insufficient. Besides, Xiao [27] proposes a MACsec security association scheme under the trusted computing environment, which adds plat-

form authentication to the terminal equipment, thereby effectively enhancing the security of the local area network. However, the above research mainly focuses on the optimization of the traditional security protocols and does not expand new application scenarios, and the innovation is slightly insufficient.

Some research provides lightweight mechanisms to support encryption and authentication for real-world scenarios. In order to better realize the integration and development of continuous authentication (CA) [28] in mobile devices, M. Frank et al. [29] implement and compare different ML agent models for CA in mobile environments. Experiments show that among the mentioned ML classifiers, ensemble methods (RFC, ETC, and GBC) perform the best, while SVM performs the worst among all classifiers. However, the main problem in their scheme is that decisions were based on a single key event resulting in relatively low performance. Aiming at the existing healthcare applications that are vulnerable to weak security against some known attacks, M. Al-Zubaidie et al. [30] propose a new robust authentication protocol (RAMHU) to prevent internal, external, passive, and active attacks. This scheme uses multiple pseudonyms for both users and medical centers to prevent the transmission of real information in the authentication request and a MAC address to prevent counterfeit devices from connecting to the network. Experiments show that RAMHU is resistant to a variety of known attacks, but the study did not verify its performance. Significant performance degradation will affect the actual deployment of the mechanism. The studies of [31,32] also use lightweight encryption and authentication methods. However, they have problems such as low encryption efficiency and unclear encryption mode.

AES and SM4 are two typical symmetric encryption algorithms [33]. AES is widely used and has four modes for cryptographic operations in different scenarios [34]. The SM4 algorithm is a Chinese secret grouping algorithm, which is safe, efficient, and simple in design, and meets the requirements of a lightweight security communication mechanism. Ge, G. et al. [35] propose a video surveillance key management scheme based on SM3 [36] and SM4 algorithms for identity identification. The SM4 symmetric encryption algorithm is used to encrypt the video data collected by the video front-end equipment, and the SM3 hash algorithm is used to implement the integrity verification of the video signaling, which enhances the overall efficiency of video surveillance data storage and sharing. However, given security, asymmetric encryption algorithms may be more suitable for encryption keys. To improve the high security of power grid data transmission, Liu et al. [37] propose a security encryption method for power Internet of Things terminals based on the SM3 algorithm. However this mechanism can only protect a single-terminal device but cannot protect the entire local area network.

2.3. Application of Programmable Data Plane

In recent years, the new type of network represented by SDN has been developing rapidly. Moreover, the research on the application of P4 programmable language has gradually come into the vision of scholars. The authors in the literature [38] implement cryptographic hash functions in the data plane to mitigate potential attacks against hash conflicts. Furthermore, Chen et al. [39] implement the Advanced Encryption Standard (AES) protocol at the data layer using scrambled lookup tables. AES is one of the most widely used symmetric encryption algorithms, which applies several rounds of encryption to 128-bit input data blocks to improve security. In P4-MACsec [40], the authors implement IEEE 802.1AE (MACsec) at P4 and introduce an automatic deployment mechanism to provide MACsec for links detected between P4 switches. Link monitoring relies on encrypted payloads and sequence numbers to prevent Link Layer Discovery Protocol (LLDP) packet manipulation and replay attacks. Hauser et al. [41] attempt to implement host-to-site IPsec in P4 switches. Another study focuses on protecting the identity of Internet users. Moghaddam et al. [42] propose Practical Anonymity at Network Level (PANEL), a lightweight and low-overhead in-network solution to provide anonymity for the Internet forwarding infrastructure. Similarly, Datta et al. [43] propose surveillance protection in network ele-

ments (SPINE), a system for anonymous communication to hostile autonomous systems (as) in the data plane by hiding IP addresses and associated TCP fields (e.g., sequence numbers). However, the above studies still have limitations. These studies only strengthen the security of data transmission, but cannot meet the conversion requirements between other protocols to IP protocols.

2.4. Programmable Data Plane

The programmable data plane extends network programmability to data plane functions. The packet processing flow can be defined using a dedicated programming language, keeping the packet processing behavior separate from the underlying hardware. P4 is the most widely used data plane programming language today. Figure 2 depicts the packet processing pipeline and the core concepts of P4.

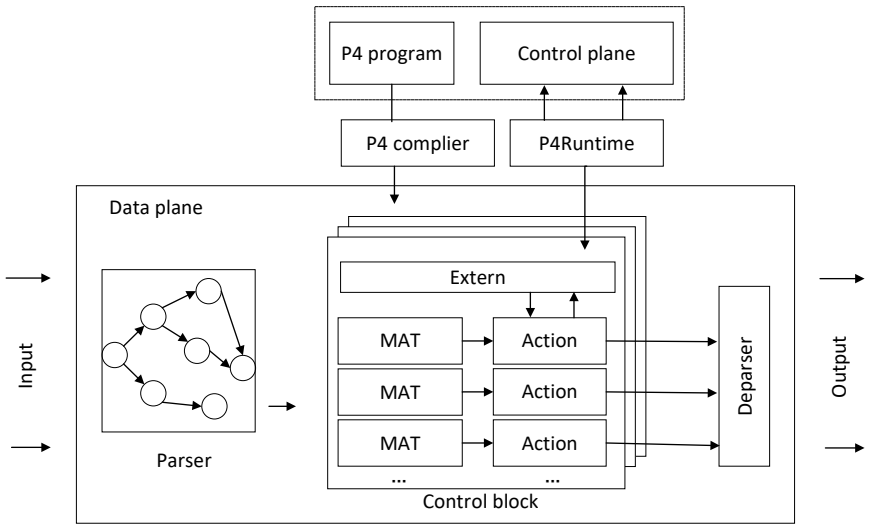

Figure 2. The packet processing pipeline and core concepts of P4.

The P4 parser extracts header fields of packets by applying predefined procedures, where the extraction behavior of the parser is expressed as a finite state machine. The primary function of the control block is to modify packet headers and perform forwarding operations according to the match-action table (MAT) issued by the control plane. The MAT gives the matching relationship between keywords and actions. Moreover, the control blocks also provide additional functionality for packet processing by developing a component called extern. The deparser reassembles the individual packet headers and loads them into well-formed network packets, which are sent out through the out port of the packet forwarding device.

3. MultiSec Design

3.1. Overview

Figure 3 outlines an application scenario for a security gateway running MultiSec. A secure connection is established between security gateway A and security gateway B to provide encryption protection for data transmitted in the core network.

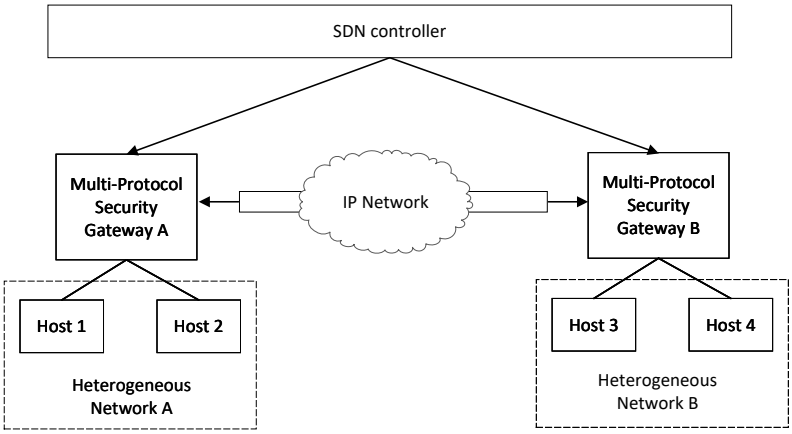

Figure 3. The application scenario of MultiSec.

Terminals in different heterogeneous networks establish secure connections through two security gateways supporting multi-protocol access, with the SDN controller unifying the configuration of the security gateways. As a core principle of MultiSec, each security gateway implements the same protocol conversion and security functions, with the specific actions performed determined by the type of packets being processed and the table entries configured by the controller. Suppose a packet is sent by host 1 inside heterogeneous network A. After receiving it, security gateway A encrypts the packet. It converts the private protocol to IP protocol, where the source address is the IP address of gateway A, and the destination address is the IP address of gateway B. The packet is forwarded through the core network to security gateway B, which decrypts the encrypted data and converts the IP protocol to the original protocol. The functional unification of the security gateway facilitates flexible deployment in the network.

3.2. Framework

The processing flow of MultiSec in the data plane is shown in Figure 4. It is mainly comprised of the programmable parser, Security Policy Database (SPD) module, (Conversion and Encryption) C&E module, (Restoration and Decryption) R&D module, and other modules. The main function blocks are as follows:

Parser: used to extract the input packet header.
SPD module: Matches given packets with SPD rules and determines the corresponding security policy. The packets that need to be encrypted are sent to the C&E module.
C&E module: integrates encryption and conversion of the private protocol to the IP protocol and is implemented in an extern.
R&D module: integrates the decryption and conversion of IP protocol to the original protocol (protocol restoration) and is implemented in another extern.

During the protocol conversion, the original packet is encrypted using the Encapsulate Security Payload(ESP) protocol [44] according to the SA information configured by the control plane. SA is a part of the Security Association Database (SAD) which has all the data required for the encryption and decryption process, such as the Security Parameter Index (SPI), encryption algorithm, encryption key, authentication key, and SA survival period.

When a packet arrives via the ingress, the programmable parser extracts the packet header according to the configuration. In case of a header other than ESP and the recirculate_flag is 0, the packet will be sent to the protocol-related basic function module, which implements user-defined operations for the specific protocol. The SPD module then processes the packet, and the matched entry in the SPD MAT (similar to the packet forwarding MAT) determines the action to be executed on the packet. In the case of DROP, the packet is dropped. In the case of BYPASS, the packet is forwarded directly to the protocol conversion module, which performs protocol conversion of the packet and then sends it to the Layer 3 forwarding module. In the case of PROTECT, the packet is passed to the C&E module

to implement the protocol conversion and encryption. Let us go back to the parser: if the packet contains an ESP header or the recirculate_flag is 1, it is forwarded to the R&D module. This module verifies the authenticity of the packet, decrypts the ESP message, and restores the IP protocol to the original protocol. Then it forwards the packet to the protocol-related forwarding module according to the flow table. The packets parsed for the second time will be sent directly to this module to apply the protocol-related forwarding actions (See Section 3.4). If a matching entry is missing in the MAT of the SPD, C&E, or R&D module, the packet will be dropped. The deparser reassembles all headers and re-calculates the IP checksum, such as time-to-live (TTL) and header checksum (checksum) fields. The matching and forwarding behavior of the data plane is managed uniformly through the SDN controller manipulating the MAT. The C&E module and the R&D module are described in detail below.

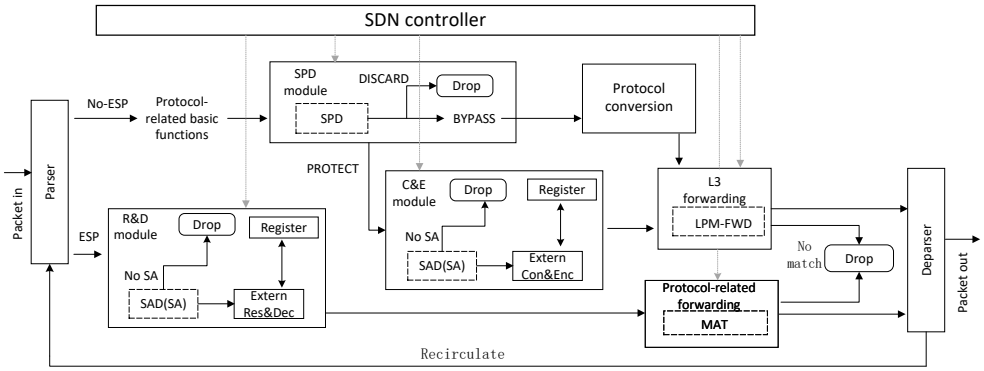

Figure 4. The processing flow of MultiSec in the data plane.

A. C&E module

Figure 5 shows the working principle of the C&E module. We introduce a C&E MAT similar to the packet forwarding MAT. Packets are first matched against keywords in the C&E MAT. If there is no corresponding matching entry, the packet will be discarded; if the match is successful, the Con&Enc action will be performed. Among them, the specific encryption and protocol conversion is implemented in the extern; the register saves the number of received packets to realize the timely update of SA; the packet descriptor records the information of the packet before encryption to avoid errors during decryption. The parameters in the MAT are generated by the controller as SA information and updated periodically.

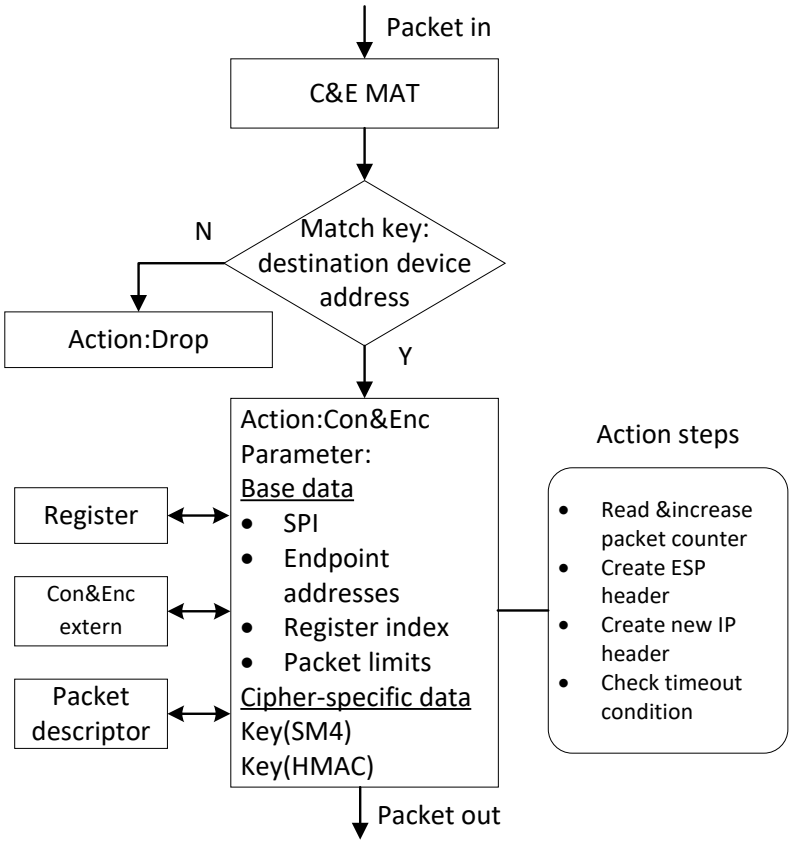

Figure 5. The C&E module.

We implement protocol conversion and encryption as actions that depend on externs, registers, and packet descriptors. Taking SM4-SM3 [45] as an example, the Con&Enc action receives two types of parameters: base data required for all cipher suites and cipher-specific data. We first describe the base data. SPI is part of the ESP header and is used to identify the SA. The endpoint addresses (IPv4 source/destination addresses) identify the source and destination of the secure connection, both of which are part of the new outer IPv4 header that encapsulates the ESP frame. Register index points to a particular index that holds the packet counter for a particular SA used by the cipher suite extern. Packet limits define timeout conditions for packets based on SA counter thresholds. Packet limits include soft and hard limits. If the soft limit is reached, rekeying will be triggered. If the hard limit is reached, packets encrypted with this SA information will be dropped. In addition to the base data, cipher suites that implement specific encryption and authentication mechanisms require cipher-specific parameters such as keys, initialization vectors (IVs), and even additional structures to hold cipher states such as registers. As an example of such a cipher suite, SM4-SM3 requires a key for SM4 and a key for the keyed-hash message authentication code. The packet descriptors hold essential information about the packet header and determine where the new IP header should be in the packet.

The processing flow of this module is as follows: the header information of the packet is matched with the matching key. If there is no matching entry, the packet will be discarded. If there is, the Con&Enc action will be executed. First, the packet counter for a particular SA is read from the register and incremented. Second, the SA information generated by the controller is passed to the corresponding extern, and the ESP packet header is created. We use the SM4 encryption algorithm to protect the original packet while the SM3 hash algorithm is applied to the complete ESP packet. If the packet length is less than 128 bits, it will be padded at the ESP trailer. The checksum value of the ESP packet is saved in the ESP authentication. Third, a new IP packet header is created using the endpoint address to encapsulate the ESP packet and complete the protocol conversion. Finally, check for

timeout conditions. If the soft limit is reached, rekeying will be triggered. If the hard limit is reached, packets will be dropped. The packet encapsulation mode of this module is shown in Figure 6. The original packet is encrypted after passing through the C&E module and encapsulated in a new IP packet as part of the payload.

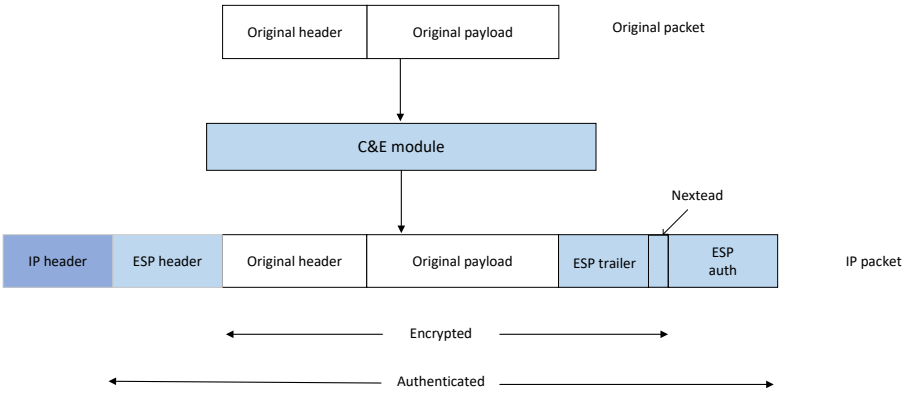

Figure 6. The packet encapsulation mode of C&E module.

B. R&D module

The R&D module also introduces an R&D MAT to implement the protocol restoration and decryption. The parameters used by the R&D MAT are the same as those of the C&E MAT, while the operation process is opposite to that of the C&E function. The processing flow of this module is as follows: first, delete the new IP header and extract the ESP packet. Second, pass the ESP packet and the key required for the cipher suite to the corresponding extern, which decrypts the ESP packet while restoring the IP protocol to the original protocol. Finally, parse the original packet, add the Ethernet header, and perform a forwarding operation based on the parsed content.

3.3. Programmable Protocol Conversion Mechanism

We reconfigure the processing flow of the programmable parser according to different private protocols to allow any protocol type network to access the security gateway with the help of the P4 language. The extraction behavior of the parser is abstracted as a finite state machine, which extracts the packet and outputs it in the form of the headers. Figure 7 shows the analysis and processing flow of packets by taking MF as an example. We simplify the program to show only the key parts.

The process of the parser is shown in Figure 7a. The parser is initiated in the state **start**. It parses the Ethernet packet header, checks the packet protocol according to the Ethertype, and selects the next state with an exact match. The possible states in this example are **parse_mf** or **parse_ipv4**, representing the MobilityFirst header and IPv4 header, respectively. The default state of the parser is **accept**. After parsing the last header, the corresponding field information and names are saved (not shown in the figure). The **INGRESS** (shown in Figure 7b,c) defines some **action**s and the corresponding match-action tables (**tables**). The action **Con&Enc** performs protocol conversion and encryption, and the action **Res&Dec** performs protocol restoration and decryption. The **tables** declare the matching keys of the above actions. The **apply** action declares the processing logic of the packet: if the ESP header is valid, apply the relevant actions of the dec table to decrypt the packet and forward it out; if the MF header is valid, the spd flag is added. If the flag bit is 0, the data packet will be bypassed without encryption; if the flag bit is 1, the enc table is applied and sent to the forwarding table. Finally, the **DEPARSER** (shown in Figure 7c) regroups the packet and sends it to the output ports.

```
/******PARSER******/
state start
   transition parse_ethernet;

state parse_ethernet
   packet.extract(hdr.ethernet);
   transition select(hdr.ethernet.etherType)
      0X27C0: parse_mf;
      0X0800:parse_ipv4;
      default: accept;

state parse_mf
   packet.extract(hdr.mf);
   transition select(hdr.mf.protocol)
     TYPE_ESP:parser_esp;
     default: accept;

state parse_ipv4
   packet.extract(hdr.ipv4);
   transition accept;

state parse_esp
   packet.extract(hdr.esp);
   transition accept;
```

(**a**) The process of PARSER

```
/*****INGRESS*****/
extern void conversion&encrypt( );
extern void restoration&decrypt( );
actions :Con&Enc();
          Res&Dec();
          Add_spd_flag();
          L3_forward() ;
          Protocol-related forward();
          Drop();
table spd
   key = hdr.mf.dest_guid: exact;
   action = Add_spd_flag; Drop;

table enc
   key = hdr.mf.dest_guid: exact;
   actions = Con&Enc; Drop;

table dec
   key = hdr.ipv4.srcAddr: exact;
         hdr.ipv4.dstAddr: exact;
   action = Res&Dec; Drop;

table forward_1
    key =  hdr.ipv4.dstAddr: lpm;
    actions = L3_forward; Drop;
```

(**b**) The process of INGRESS

```
table forward_2
    key =  hdr.mf.dest_guid: exact;
    actions = Protocol-related forward; Drop;

apply
   if(hdr.esp.isValid())
     dec.apply();
     forward_2.apply
   else if(hdr.mf.isValid())
     spd.apply();
     if(metadata.spd_flag == 0)
        BYPASS;
     else if(metadata.spd_flag == 1)
        enc.apply();
        forward_1.apply();


/****DEPARSER****/
control MyDeparser()
   apply
      packet.emit(hdr.ethernet);
      packet.emit(hdr.ipv4);
      packet.emit(hdr.mf);
      packet.emit(hdr.esp);
```

(**c**) The process of DEPARSER

Figure 7. Analysis and processing flow of a MF packet.

Packets running different protocols have different packet structures and numbers of headers. To store and locate the parsed packet headers at each level, we design a packet header caching method by introducing a packet descriptors table and a header descriptors table, as shown in Figure 8. While parsing the packet, the parser maps packet information to the packet descriptors table. It contains every state of the parser (header of the packet) and records the total length of all parsed headers. Each state information is mapped to the second level table, called the header descriptor table, which records the header type, location, and length of each header. The location and total packet length are automatically updated with each new header parsed. In this mechanism, all headers are located by the packet descriptors.

Compared with the traditional protocol conversion, MultiSec is designed to process any protocol. How to obtain the original protocol number when the security gateway B (shown in Figure 3) performs protocol restoration is a key issue in the design process of MultiSec. To save bandwidth, we redefine the function of the **NextHead** field of the ESP trailer. Take the application scenario shown in Figure 3 as an example. After parsing the packet, Security Gateway A saves the EtherType field value of the Ethernet header in the NextHead field, which is sent to the Internet along with the packet. Security Gateway B receives the packet, parses the ESP trailer to obtain the original protocol number, and then fills in the Ethertype field of the newly encapsulated Ethernet header, restoring the IP protocol to the original protocol.

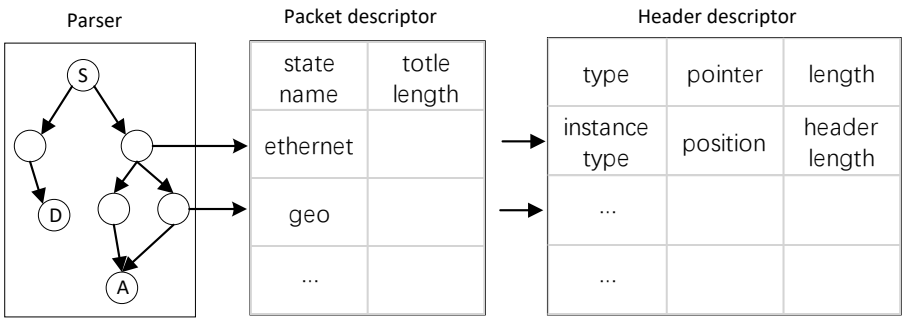

Figure 8. The packet header caching method.

3.4. The Integration of Protocol Conversion and Secure Forwarding

3.4.1. Recirculation Mechanism

To guarantee the reliability of data transmission and minimize the system overhead, we add a security mechanism to the protocol conversion mechanism. The key to integrating

protocol conversion and secure forwarding is the integration of the decryption of encrypted packets and the parsing of the decrypted packets, that is, the problem of secondary parsing of the same packet in a P4 pipeline. Developing a decryption mechanism in the parser module may be able to decrypt the ESP packet and parse the original packet at the same time. However, this method needs to delete the parsed header in the parser, which is currently not supported by P4. We develop a recirculation mechanism based on the v1model architecture. The mechanism introduces the standard metadata **recirculate_flag** in the R&D module and determines whether the packet needs to be parsed again by the flag bit value. In the definition of the P4 language, the standard metadata is used to carry the configuration information of the switch itself. We provide the algorithm for the recirculation mechanism to process encrypted packets in Algorithm 1.

Algorithm 1 Recirculation mechanism

Input: encrypted packet
Output: original packet
1: apply Res&Dec action
2: recirculate_flag ← 1
3: send to the Deparer
4: **if** recirculate_flag = 0 **then**
5: send to the output ports
6: **else**
7: send to the Parser
8: extract original packet
9: **if** ESP header is valid OR recirculate_flag = 1 **then**
10: recirculate_flag ← 0
11: send to R&D module
12: send to Protocol related forwarding module
13: send to the Deparer
14: **if** recirculate_flag =0 **then**
15: send to the out ports

Lines 1 to 3 indicate that a packet enters the R&D module for the first time and uses the Red&Dec action to perform the decryption and protocol restoration. Then, the recirculate_flag is set to 1, and the data packet is forwarded to the Deparser module. Lines 4 to 7 indicate that the Deparser module selects the packet sending path according to the value of recirculate_flag. If recirculate_flag is 0, the packet will be sent to the output ports, and if it is 1, the packet will be sent to the recirculate port for secondary parsing. Lines 8 to 11 indicate that the decrypted data packet re-enters the Parser module through the recirculate port for parsing. The parser sends the packet to the R&D module based on the recirculate_flag value of 1 and then sets the recirculate_flag to 0. Lines 12 to 13 indicate that the decrypted packet is directly sent to the protocol-related forwarding module to perform the forwarding operation. Finally, the packet is sent to the output ports.

3.4.2. Automatically Distribute and Dynamically Update SA

In the security mechanism, the keys used to encrypt and authenticate packets need to be updated in time to prevent them from being leaked. In order to configure and automatically update SAs, we add the security gateway configuration files to the controller, as shown in Figure 9. This file can be manually defined by an administrator or generated by other software components. The security gateway configuration files are the basis for establishing a secure connection between programmable gateways. Files contain essential information such as switch identification, endpoint IP, and network resources. The SDN controller generates SA configuration data for the data plane based on the security gateway configuration files and interacts with the switching devices through P4runtime. For the SM4-CBC-HMAC-SM3 cipher suite, the SA data includes keys for SM4 and Hash-based Message Authentication Code(HMAC), register indices, and configuration data for SPD. To

periodically update the key material, we set a limited lifetime for SAs. If an SA needs to be updated, the P4 switch will send the packet that triggers the update to the SDN controller to update the SA information. This process is implemented using packet counters in the register. When the SDN controller receives the SA expiration notification, it generates a new SA identified by the new SPI and updates the table entry following three steps. First, the new SA is installed in the R&D MAT. Then, the existing SA in the C&E MAT is replaced by the new SA via a modified operation. Last, as it can be ensured that no packets are encrypted using the previous SA, its entry can be removed from the R&D MAT.

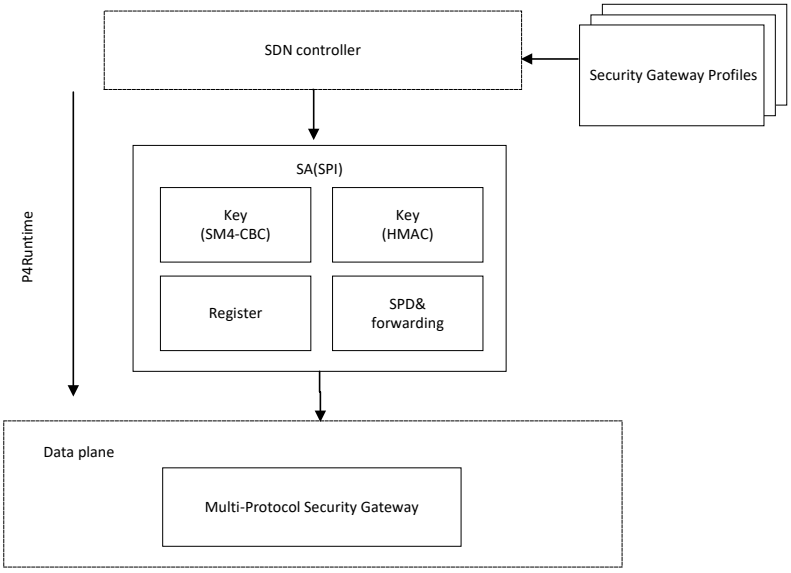

Figure 9. Application of security gateway profiles.

4. Security Analysis and Experimental Evaluation

In this section, we first analyze the security of MultiSec, then evaluate its functionality and performance on the testbed, and finally compare it with IPsec.

4.1. Security Analysis

Data confidentiality: The MultiSec uses SM4-CBC and HMAC-SM3 encryption modes to encrypt and authenticate data before transmission. The controller uses a random number to construct the initialization vector of the SM4 encryption algorithm and SM3 authentication algorithm and sends it to the security gateways. The security gateway encrypts the first group of packet data using initialization vectors. Because random numbers are used in encryption and verification and are generated by the controller, the attacker cannot calculate the key stream and random numbers by capturing data packets, which ensures the confidentiality of data transmission in the core network. Moreover, the content and format of data packets sent by heterogeneous networks are different, making it difficult to analyze the corresponding laws of ciphertext data, and it is difficult to carry out attacks such as replay and hijacking to ensure confidentiality and authenticity of the data.

Data integrity: The sender first encrypts the data block to generate the ciphertext. Then, the ciphertext, ESP head, and ESP tail are calculated with the group key to obtain the authentication summary, which is added to the ESP authentication field and sent with the ciphertext. Since random numbers are introduced into SM4 and SM3 algorithms, attackers cannot construct valid data validation information. Additionally, the receiver can not only detect unexpected errors in the data by verifying and then decrypting, but also judge whether the data packet is maliciously tampered with or destroyed, which has more robust security in terms of data integrity. The key in the system is generated by the controller and then delivered to the switches, and the process of key transmission, storage, and decryption

is performed in hardware and updated according to the corresponding rules. Unless the attacker obtains the decryption algorithm and private key, the key is safe.

4.2. Prototype Implementation

On the data plane, we extend the v1model P4 forwarding model to be able to run P4 programs that describe the functionality of the data plane. The PPK based on Data Plane Development Kit (DPDK) includes a P4 compiler, which generates platform-independent C codes from the P4 codes, and then compiles them into target-related underlying code through the back-end compiler. The underlying codes are tied to the DPDK, which allows P4 programs to bypass the system kernel and be executed in the user space of the CPU-based software system, increasing runtime speed.

Our goal is to implement a lightweight, simple-to-understand controller. We use p4runtime_lib to program the interface of the P4 switch and the central controller, which leverages the P4Runtime API for communication. Compared to using a controller architecture, this approach reduces the potential for errors caused by version conflicts. We implement the P4 processing pipeline as a p4-16 program and execute it on a P4 switch running PPK.

4.3. Testbed Setup

The evaluation setup shown in Figure 10 is built using two heterogeneous networks. Each heterogeneous network contains a sending or receiving device, an access switch, a programmable security gateway, and a network switch. MultiSec is applied to the programmable security gateways s1 and s2. The security gateway implements the prototype system on a switch based on the PPK. PPK is a compilation environment of the programmable data plane, which can configure the operation of the data plane and is compatible with the P4 language. The switches are equipped with a Ubuntu20.04 operating system, a Zhaoxin KH-20000 processor, and 64GB of memory.

We perform performance evaluation experiments to investigate the goodput and latency. We vary the frame lengths of packets sent from h1 and measure goodput and latency for 74 to 1518 frame sizes. For each evaluation experiment, we consider three scenarios. In the first scenario, we implement MultiSec on the security gateways to protect all packets with SM4-CBC. In the second scenario, we enable MultiSec but bypass SM4-CBC encryption and decryption in the P4 extern, where the plaintext payloads are sent within the MultiSec packets. We use this scenario to measure the effect of the security function and the overhead it brings by comparing it to network packet forwarding with a P4 extern. In the third scenario, we implement MultiSec and offload encryption and decryption to an encryption card to improve the performance defects of the P4 software.

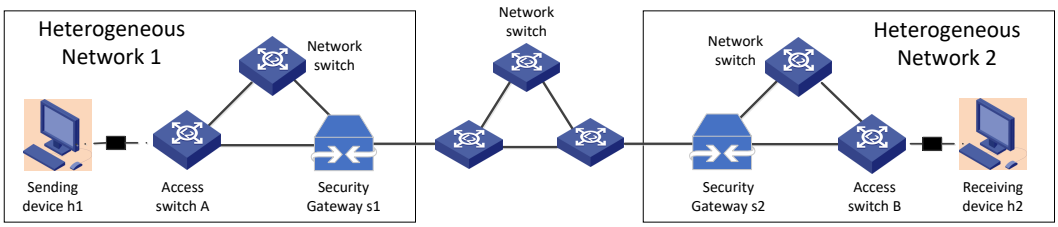

Figure 10. Evaluation testbed of MultiSec.

4.4. Functional Verification

In the interconnection test between heterogeneous subnets, we mainly test encryption, decryption, and the conversion between multi-protocol packets and IP packets. We take MF and Geo packets as test cases. We first configure the flow table in the programmable security gateway s1 and s2 and send packets into the security gateway s1 from h1. By capturing packets on s1 and s2, we can see the change of packet structure in Figures 11 and 12.

Figure 11 shows the processing flow of the MF packet. The MF protocol number is 0X27C0, which is displayed in the Ethertype field of the MAC header. The mftype indicates

the type of the MF packet. The packet payload is transmitted in plaintext before encryption. After being processed by the security gateway s1, the MF packet is converted into an IP packet encrypted by the ESP protocol, in which the original MF header and payload are encrypted as the new payload of the IP packet, and the information security is guaranteed. The IP packet is forwarded into the security gateway s2 through the Internet. It can be seen that the IP packet is decrypted and restored to the MF packet after passing through the security gateway s2.

Figure 12 shows the processing flow of the Geo packet. The Geo header shall be comprised of a **Basic Header**, **Common Header**, and an optional **Extended Header**. The composition of the **Basic Header** and **Common Header** equals for all packet transport types and differs for the **Extended Header**. **Extended Header** is determined by the Header type field in **Common Header**. In this case, the **Extended Header** is a GBC, which is used to provide the transport of packets in the ITS ad hoc network. Similar to the processing flow of the MF packet, the Geo packet is first converted into an IP packet encrypted by the ESP protocol and then restored to the MF packet by the security gateway s2.

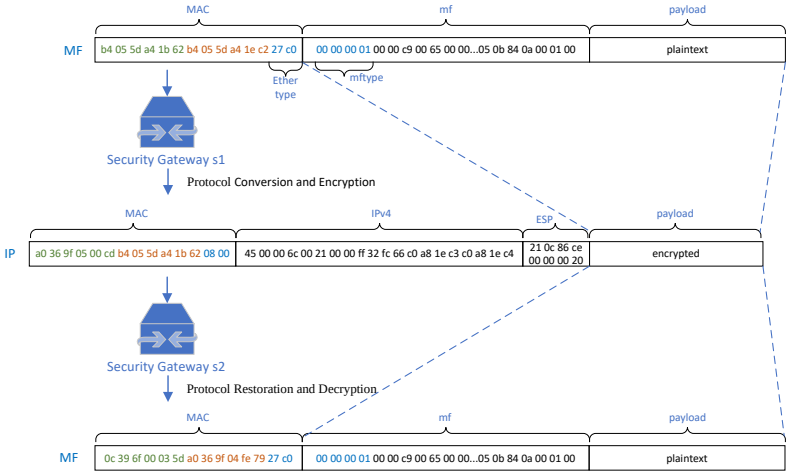

Figure 11. The processing flow of a MF packet.

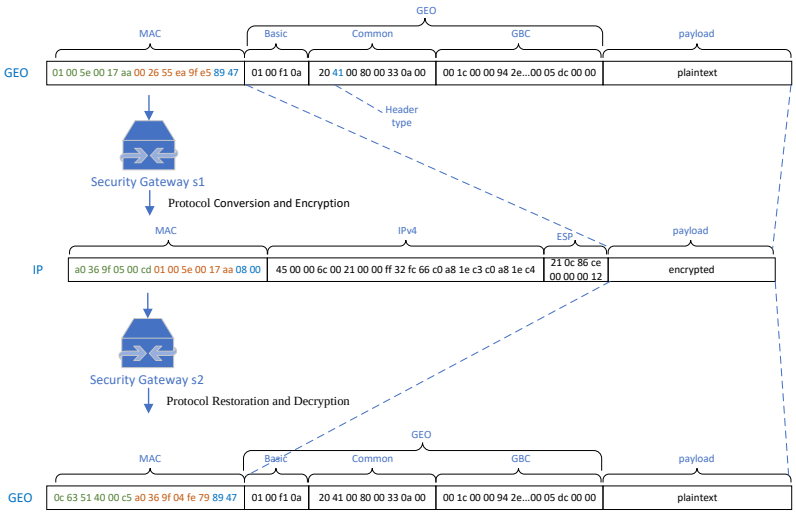

Figure 12. The processing flow of a Geo packet.

4.5. TCP Goodput

We investigate the TCP goodput in MultiSec in our setting with security gateways. To that end, we use h1 to randomly send three different types of packets of IPv4, MF, and Geo and measure TCP transmissions between h1 and h2 with Spirent. Each experiment

comprises 10 runs, each with a duration of 30 s and a Maximum Transmission Unit (MTU) set to 1518 B. Figure 13 shows the results calculated as average over the 10 runs. The goodput increases with the frame size, resulting from decreased packet number due and, in particular, lower packet loss rate. We can see that MultiSec achieves a significantly larger throughput with the encryption card than in the other two scenarios. The encryption card could reduce packet processing delay. However, the experimental results show similar TCP goodput rates for MultiSec and MultiSec without encryption. Therefore, we conclude that the usage of the encrypted card increases the TCP goodput, and the encryption only has minimal impact on throughput.

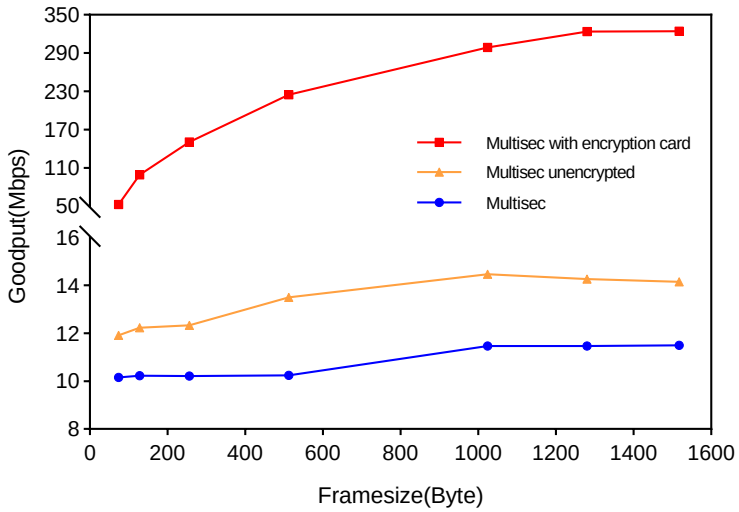

Figure 13. TCP Goodput evaluation.

4.6. Latency

In the second experiment, we investigate the latency between the two network hosts connected by two security gateway. Figure 14 depicts the latency calculated as an average over the 10 runs. The evaluation results are similar to those of the experiment for TCP goodput. The latency increases with the frame size. MultiSec with the encryption card significantly reduces latency. Again, applying or omitting SM4-CBC in the P4 extern does not cause significant differences in the latency. However, their latency is very high, resulting from software switches having limited processing capability in the face of complex operations.

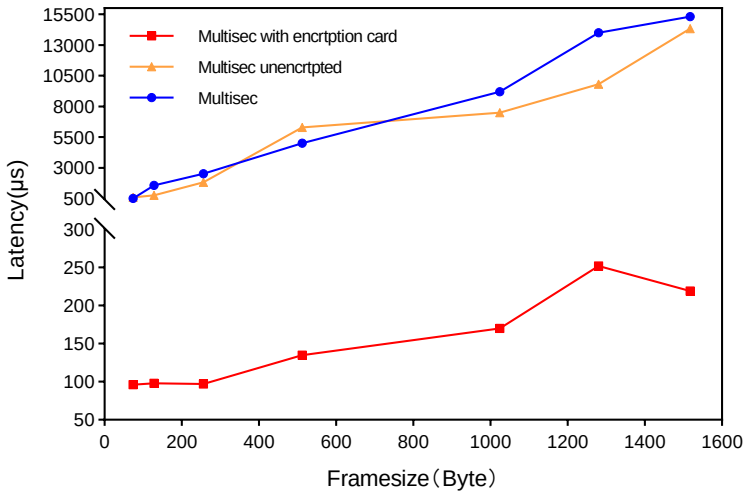

Figure 14. Latency evaluation.

4.7. Comparison and Analysis with IPsec

We compared the proposed MultiSec with IPSec in function and performance. We build the experimental topology shown in Figure 15. The device h1 sends a packet, and the P4 switch P1 encrypts the packet after receiving it and establishes a secure connection with the P4 switch P2. After transmission over the core network, P2 receives and decrypts the packet and finally sends it to the receiving device h2. IPsec uses the AES-CTR-HMAC-MD5 cipher suite to protect the internal IP network (Local Area Network, LAN) by establishing a virtual tunnel.

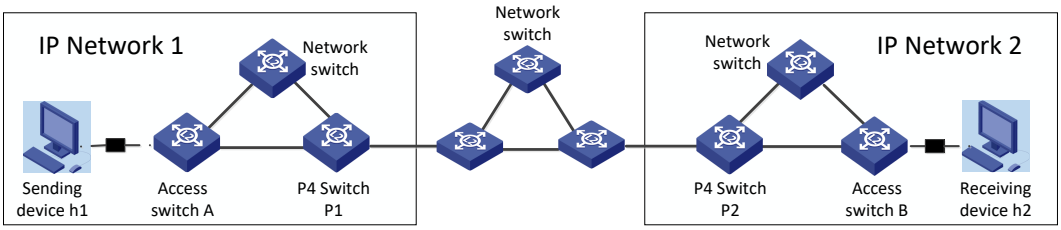

Figure 15. Evaluation testbed of IPsec.

The comparison results are shown in Table 2. The encryption and decryption speed of MultiSec is faster than that of IPsec, but the authentication speed is slower than that of IPsec. Because the key scheduling algorithm of SM4 is simpler, and AES considers integer alignment during the polling process, the key generation is more complicated. Since the length of the output summary value of the SM3 algorithm is 256 bits, which is higher than 128 bits of MD5, the processing time is longer than that of MD5, but the security performance is much higher than that of MD5. There is little difference in goodput, with MultiSec slightly higher than IPsec. IPsec provides protection for terminals in the IP subnet, but only IP protocol access is allowed. MultiSec provides protection for heterogeneous subnets and allows multi-protocol access.

Table 2. The comparison of MultiSec and IPsec.

Mechanism	Algorithm	Latency (μs)	Goodput (Mbps)	Protection Range	Access Protocol
MultiSec	SM4-DEC	79.55	101.38	Heterogeneous subnets (LAN)	Multi-protocol
	SM3-AUT	12.63			
IPsec	AES-DEC	104.24	98.27	IP LAN	IP protocol
	MD5-AUT	6.10			

5. Conclusions

In this paper, we propose MultiSec, a multi-protocol secure forwarding mechanism based on the programmable data plane, which provides encryption protection for heterogeneous protocols transported across the core network. Benefiting from the flexible and definable advantages of the programmable network architecture, MultiSec supports the flexible extension of protocols to facilitate the convergence of various heterogeneous networks. Meanwhile, we design a recirculation mechanism in the data plane so that MultiSec can implement encryption, decryption, and protocol conversion on one switch, improving the completeness of functions. Finally, we realize the automatic distribution and update of SA information by adding the security gateway configuration file in the controller. Experiments show that MultiSec supports multi-protocol parsing and conversion, and provides security protection for packets forwarded on the core network. Compared with software switches, the goodput increases by 70 times, and transmission latency is reduced by two orders of magnitude through hardware acceleration.

Currently, some new network protocols lack a security transmission mechanism, or their security transmission mechanism is imperfect. However, due to the number of protocols, designing a set of security mechanisms for each protocol is not only burdensome but also inflexible. In the next step, we will study a general-purpose encrypted transmission scheme deployed in programmable network equipment, which takes effect for all protocols to improve the forwarding efficiency and data security.

Author Contributions: Conceptualization, P.C. and Y.D.; Data curation, Z.L.; Formal analysis, Y.D.; Funding acquisition, P.C.; Investigation, Z.L. and L.X.; Methodology, Z.L. and Y.H.; Supervision, Y.H.; Writing – original draft, Z.L., P.C., Y.D., L.X. and Y.H. All authors have read and agreed to the published version of the manuscript

Funding: This work is supported by the National Key Research and Development Project of China (2019YFB1802501).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, L.; Afanasyev, A.; Burke, J.; Jacobson, V.; Claffy, K.; Crowley, P.; Papadopoulos, C.; Wang, L.; Zhang, B. Named data networking. *ACM Sigcomm Comput. Commun. Rev.* **2014**, *44*, 66–73.
2. Seskar, I.; Nagaraja, K.; Nelson, S.; Raychaudhuri, D. Mobilityfirst future internet architecture project. In Proceedings of the 7th Asian Internet Engineering Conference, Bangkok, Thailand, 9–11 November 2011; pp. 1–3.
3. Kuhlmorgen, S.; Llatser, I.; Festag, A.; Fettweis, G. Performance evaluation of etsi geonetworking for vehicular ad hoc networks. In Proceedings of the 2015 IEEE 81st Vehicular Technology Conference (VTC Spring). IEEE, Glasgow, UK, 11–14 May 2015; pp. 1–6.
4. Shuang, W.; Han, Y. Development and application of intelligent transportation system. *J. Inst. Eng. Ser. B* **2019**, *102*, 1191–1200.
5. *IEEE Std 1888.2-2014*; IEEE Standard for Ubiquitous Green Community Control Network: Heterogeneous Networks Convergence and Scalability. IEEE: NY, USA, 2014; pp. 1–48. doi: 10.1109/IEEESTD.2014.6823069.
6. Hauser, F.; Häberle, M.; Schmidt, M.; Menth, M. P4-IPsec: Site-to-site and host-to-site VPN with IPsec in P4-based SDN. *IEEE Access* **2020**, *8*, 139567–139586.
7. Bosshart, P.; Gibb, G.; Kim, H.S.; Varghese, G.; McKeown, N.; Izzard, M.; Mujica, F.; Horowitz, M. Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN. *ACM Sigcomm Comput. Commun. Rev.* **2013**, *43*, 99–110.
8. Chole, S.; Fingerhut, A.; Ma, S.; Sivaraman, A.; Vargaftik, S.; Berger, A.; Mendelson, G.; Alizadeh, M.; Chuang, S.T.; Keslassy, I.; et al. drmt: Disaggregated programmable switching. In Proceedings of the Conference of the ACM Special Interest Group on Data Communication, Los Angeles, CA, USA, 21–25 August 2017; pp. 1–14.
9. Kaljic, E.; Maric, A.; Njemcevic, P.; Hadzialic, M. A survey on data plane flexibility and programmability in software-defined networking. *IEEE Access* **2019**, *7*, 47804–47840.
10. Tibor, S.; Kuljic, B.; Dukan, P.; Mathe, K.; Peter, O. Realization of protocol conversion for high speed data acquisition system. In Proceedings of the 2009 7th International Symposium on Intelligent Systems and Informatics. Subotica, Serbia, 25–26 September, 2009; pp. 341–344.
11. Kenneally, E. Whos's liable for insecure networks? *Computer* **2002**, *35*, 93–95.

12. He, Q.; Yuan, S.; Zhu, L. Modulation domain encrypted transmission based on chaotic sequence for satellite communication. In Proceedings of the 2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Rome, Italy, 27–29 November 2019; pp. 1–6.
13. Eum, S.h.; Hong, S.k. Development of user protocol converter about Modbus and NMEA0183. *J. Korea Inst. Inf. Commun. Eng.* **2015**, *19*, 2584–2589.
14. Ananda, M.; Nagarathna, R.; Krishna, L.; Rajeswari, P. Implementation of low cost protocol converter using common media device. In Proceedings of the 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bengaluru, India, 21–23 February 2017, pp. 780–783.
15. Sinha, R. Conversing at many layers: Multi-layer system-on-chip protocol conversion. In Proceedings of the 2015 20th International Conference on Engineering of Complex Computer Systems (ICECCS), Gold Coast, Australia, 9–11 December 2015; pp. 170–173.
16. Nie Yongjun, Xu Guanghui, F.X. Design of FPGA-based ethernet and G. SHDSL protocol converter. *Inf. Technol.* **2017**, *4*, 117–120.
17. Yukai, L. FC-AE Multi-Protocol Heterogeneous Network Bridge and Related Software Design. PhD Thesis, Zhejiang University, Zhejiang, China.
18. Zhiling, W. Research and Design of Multi-Protocol Conversion System Based on FPGA. PhD Thesis, University of Electronic Science and Technology, Sichuan, China, 2018.
19. Deng, A.; Wang, H. An efficient protocol conversion mechanism between profinet network and IPv6 backhaul network for industrial internet. In Proceedings of the 2021 China Automation Congress (CAC), Beijing, China, 22–24 October 2021, pp. 4792–4796.
20. Xie, J.; Gao, Q. Design and implementation of embedded protocol conversion gateway for intelligent buildings. In Proceedings of the 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 18–20 October 2019; pp. 1–5.
21. Derhamy, H.; Eliasson, J.; Delsing, J. IoT interoperability—On-demand and low latency transparent multiprotocol translator. *IEEE Internet Things J.* **2017**, *4*, 1754–1763.
22. Lixin, M. Research on Key Technologies of Software-Defined Protocol Conversion. PhD Thesis, Strategic Support Force Information Engineering University, Zhengzhou, China, 2020.
23. Hellwig, M.; Beyer, H.G. A matrix adaptation evolution strategy for constrained real-parameter optimization. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 10–13 June 2019; pp. 1–8.
24. Huasheng, L.; Wendong, W.; Yu, L.; Shiduan, C. A scheme to provide QoS levels in MPLS VPN. *J. Beijing Univ. Posts Telecommun.* **2004**, *27*, 98–102.
25. Jin, Z. Research on High-Performance SSL VPN System Based on DPDK. Master's Thesis, Huazhong University of Science and Technology, Wuhan, China.
26. Guanhua, R. SSL Protocol Research and Optimization Based on TCM. PhD Thesis, Beijing University of Technology, Beijing, China.
27. Yuelei, X.; Junsheng, W.; Zhixiang, Z. MACSec security association scheme in trusted computing environment. *Comput. Appl. Res.* **2019**, *36*, 6.
28. Shepherd, S. Continuous authentication by analysis of keyboard typing characteristics. In Proceedings of the European Convention on Security and Detection, Brighton, UK, 16–18 May 1995; pp. 111–114.
29. de Marcos, L.; Martínez-Herráiz, J.J.; Junquera-Sánchez, J.; Cilleruelo, C.; Pages-Arévalo, C. Comparing machine learning classifiers for continuous authentication on mobile devices by keystroke dynamics. *Electronics* **2021**, *10*, 1622.
30. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. Ramhu: A new robust lightweight scheme for mutual users authentication in healthcare applications. *Secur. Commun. Netw.* **2019**, *2019*. https://doi.org/10.1155/2019/3263902.
31. Giri, D.; Maitra, T.; Amin, R.; Srivastava, P. An efficient and robust rsa-based remote user authentication for telecare medical information systems. *J. Med. Syst.* **2015**, *39*, 1–9.
32. Kumar, N.; Kaur, K.; Misra, S.C.; Iqbal, R. An intelligent RFID-enabled authentication scheme for healthcare applications in vehicular mobile cloud. *Peer-Peer Netw. Appl.* **2016**, *9*, 824–840.
33. Liu, Y.; Wu, N.; Zhang, X.; Zhou, F. A new compact hardware architecture of S-Box for block ciphers AES and SM4. *Ieice Electron. Express* **2017**, *14*, 20170358–20170358.
34. Abdullah, A.M.; et al. Advanced encryption standard (AES) algorithm to encrypt and decrypt data. *Cryptogr. Netw. Secur.* **2017**, *16*, 1–11.
35. Ge, G.; Feng, H.; Liu, B.; Zhang, J. Research on video surveillance key management scheme based on identification password. In Proceedings of the 2019 4th International Conference on Mechanical, Control and Computer Engineering (ICMCCE), Hohhot, China, 25–27 October 2019; pp. 741–7415.
36. Kong, D.; Wang, X.; Zhong, L.; Sha, Y.; Li, Z. Dynamic password token based on SM3 algorithm. In Proceedings of the 2016 International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, 11–12 July 2016; pp. 180–184.
37. Liu, D.; Wang, R.; Zhang, H.; Chen, J.; Liu, X.; Ma, L. Research on terminal security technology of ubiquitous power Internet of Things based on PUF and SM3. In Proceedings of the 2019 IEEE 3rd Conference on Energy Internet and Energy System Integration (EI2), Changsha, China, 8–10 November 2019; pp. 910–915.

38. Scholz, D.; Oeldemann, A.; Geyer, F.; Gallenmuller, S.; Carle, G. Cryptographic hashing in P4 data planes. In Proceedings of the 2019 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), Cambridge, UK, 24–25 September 2019.
39. Chen, X. Implementing AES encryption on programmable switches via scrambled lookup tables. In Proceedings of the SIGCOMM '20: Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication, Virtual Event, USA, 10–14 August 2020. .
40. Hauser, F.; Schmidt, M.; Haberle, M.; Menth, M. P4-MACsec: Dynamic topology monitoring and data layer protection with MACsec in P4-based SDN. *IEEE Access* **2020**, *8*, 58845-58858. doi: 10.1109/ACCESS.2020.2982859.
41. Hauser, F.; Häberle, M.; Schmidt, M.; Menth, M. P4-IPsec: Implementation of IPsec gateways in P4 with SDN control for host-to-site scenarios. *arXiv* **2019**. Available online: https://www.arxiv-vanity.com/papers/1907.03593/ (accessed date 18 June 2022).
42. Moghaddam, H.M.; Mosenia, A. Anonymizing masses: Practical light-weight anonymity at the network level. *arXiv* **2019**, arXiv:1911.09642.
43. Datta, T.; Feamster, N.; Rexford, J.; Wang, L. {spine}: Surveillance protection in the network elements. In Proceedings of the 9th USENIX Workshop on Free and Open Communications on the Internet (FOCI 19), Santa Clara, CA, USA, 13 August 2019.
44. *IEEE Std 1017.1-2021 (Revision of IEEE Std 1017.1-2013)*; IEEE Recommended Practice for Testing of Electric Submersible Pump Cable. IEEE Standards Association: NJ, USA, 2022; pp. 1–67.
45. Suo, S.; Cui, C.; Jian, G.; Kuang, X.; Yang, Y.; Zhao, Y.; Huang, K. Implementation of the high-speed sm4 cryptographic algorithm based on random pseudo rounds. In Proceedings of the 2020 IEEE International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China, 6–8 November 2020; Volume 1, pp. 112–117.