哈爾濱工業大學

毕业设计（论文）中期报告

题 目：基于 dpdk 的高性能的 IPSec VPN

专　　业　信息安全

学　　生　殷悦

学　　号　150120526

班　　号　1504201

指导教师　刘杨

日　　期　2019 年 4 月 28 日

1．论文工作是否按预期进行、目前已完成的研究工作及结果

1.1 论文工作是否按预期进行

（正文　宋体小 4 号字，多倍行距值 1.25，段前 0 行，段后 0 行。字数 3000 字以上。具体的撰写要符合哈尔滨工业大学本科生毕业论文撰写规范的书写规定。）

1.1.1 IPSec 综述

1.1.1.1 IPSec 处理

由于 IP 协议设计之初没有安全保护，数据在传输中有可能被监听或篡改，存在安全隐患。而 IPSec 是一套完整的加密系统，IPSec 有协议提供每个 IP 数据包的认证，完整性校验（保证传输中未被篡改），机密性（数据包加密）

IPSec 技术主要有 AH，ESP，IKE，ISAKMP/Oakley 和算法组成。

AH 协议：有完整性校验功能，但不能加密数据。

ESP 协议：有完整性校验功能和加密功能。

IKE 协议：主用于秘钥管理，完成设备间会话秘钥协商和交换。

加密认证算法：建立连接时需要确定加密和认证的算法，加密常用 AES、3DES 等算法，认证常用 SHA-1 和 MD5 算法。

SA 协议：用于在不同设备间算法协商和秘钥匙交换的概念。

1.1.1.2 ESP(封装安全载荷)与 AH(验证头)

ESP 主要提供数据加密和完整性校验

ESP 协议：

加密前数据包：　IP HDR|Data

加密后数据包：　New IP HDR|Auth(ESP HDR|Enc(IP HDR|Data)|ESP Trailer|ESP Auth)

隧道和传输模式：

传输模式：

IP HDR|Auth(ESP HDR|Enc(Data)|ESP Trailer|ESP Auth)

仅对 IP 头部以上的数据进行加密，不对 IP 头进行加密，主要用于基于 IPSec 的点对点通用路由协议

隧道模式：

New IP Header|Auth(ESP HDR|Enc(IP HDR|Data)|ESP Trailer|ESP Auth)

首先加密原有数据包，然后加入新头部

1.1.1.3 IKE(Internet 密钥交换)

IKE 由三部分组成：

ISAKMP：使用了 UDP 的 500 端口，定义了信息交换的体系结构

SKEME：实现公钥加密认证体制

Oakley：提供了 IPSec 对等体间达成相同加密密钥的基本模式机制。

SA(SecurityAssocation,安全联盟)：用于协商实体通信建立的一种协约，协定了 IPSec 协议、密钥、转码方式、密钥有效期等。IPSec 会构建一个 SA 数据库（SADB）用来维护 IPSec 协议保障数据安全。

SA 是单向的：两设备通过 ESP 进行通信，则设备则需要 SA（IN）和 SA（OUT），SA（OUT）用作处理发出的数据包，SA（IN）用作处理接受的数据包，SA（IN）和对方的 SA（OUT）使用相同的加密参数（密钥等）。

SA 还区分协议，若 AH 和 ESP 同时生效，AH 和 ESP 会产生不同的 SA。

SA 有两种：

IKE（ISAKMP） SA：用于协商 IKE 数据流加密及对等体认证的算法（对密钥加密和 peer 认证），只能有一个。

IPSec SA：用于协商对等体之间的 IP 数据流进行的加密算法，可以有多个。

IKE 交换模式（主模式）

Peer1 Peer2

SA 交换（用于确认对方使用的算法）

发送本地 IKE 策略 ---发起方发送策略-->

接受对端确认的策略 <--接收方确认策略--- 查找匹配的策略

密钥交换（产生密钥）

---发起方的密钥生成信息--> 密钥生成

密钥生成 <--接收方的密钥生成信息---

ID 交换及验证（验证对方身份）

---发起方身份和验证数据--> 身份验证和交换过程验证

身份验证和交换过程验证 <--接收方身份和验证数据---

IKE 的交换模式（野蛮模式）

Peer1 Peer2

SA 交换，密钥生成(确认对方使用的算法，产生密钥)

发送本地 IKE 策略，密钥生成信息

---发起方策略，密钥生成信息-->

查找匹配的策略，密钥生成

<--接收方的密钥生成信息，身份和验证数据---

接收对端确认的策略，密钥生成

ID 交换及验证（验证对方身份）

--发起方身份和验证数据-->

身份验证和交换过程验证

点对点 IPSec VPN 协商过程有两个阶段，两设备间建立安全的传输连接需要先协商使用的加密算法、密钥、封装技术。

第一步：两设备间建立安全的管理连接，用作保护加密第二阶段协商过程。

第二步：协商安全连接的参数，两设备间形成安全连接。

接下来就可以使用该安全连接来传输数据。

阶段一 ISAKMP SA 提供了后续协商的安全，阶段二 IPSec SA 协商在第一阶段加密保护下进行，IPSec SA 为后续传输数据加密。

阶段一：进行 ISAKMP SA 协商

1.协商对等体间认证的方式（共享密钥或数字证书）

2.协商加密使用的算法（DES 或 3DES 等）

3.协商认证使用的算法（MD5 或 SHA）

4.协商 Diffie-Hellman 密钥组

5.协商协商模式（主模式或野蛮模式）

6.协商 SA 生存期

阶段二：进行 IPSec SA 协商

1.协商双方封装技术（ESP 或 AH）

2.协商加密算法

3.协商 HMAC 方式（MD5 或 SHA）

4.协商传输模式（传输模式或隧道模式）

5.协商 SA 生存期

1.1.2 XFRM 框架

XFRM 是 Linux 引入的一种基于策略的高扩展性网络安全架构。在 Linux2.6 内核中包含了 PF_KEY，2.4 内核需要打补丁实现。根据 RFC2367 的定义，内核 PF_KEY 实现了安全联盟(SA)和安全策略(SP)的数据库以及用户空间接口。不同系统的 SA 和 SP 实现管理不同，在 Linux 内核中则通过 xfrm 库来实现。

IPSec 的 SP：

SPD 用来存放 IPSec 哪些流量需要走 IPSec 的规则表，表中包含目的 IP，源 IP，执行协议(AH 或 ESP 或 AH 和 ESP 并存)，源端口，目的端口，工作模式(传输模式或隧道模式)。当主机有数据通过 VPN 发出的时候，数据包会根据 SPD 规则进行匹配，只有匹配到，数据包才会通过 AH 或 ESP 处理。

IPSec 的 SA：

SAD 用来存安全信息，SAD 数据库中包含的信息有 SPI 值，目的端 IP，AH 或 ESP，AH 验证算法，AH 验证加密密钥，ESP 验证算法，ESP 验证加密密钥，ESP 加密算法，ESP 加密密钥，隧道或传输模式。SPI 索引值是双方用于索引数据库，手动指定或随机生成的一个值。

1.1.3 DPDK 平台介绍

1.1.3.1 DPDK 简介

DPDK(Data planedevelopment kit)是一个用于处理和加速网络数据包的软件库，相比于传统的 Linux 操作系统协议栈，DPDK 有五个特点：

1.轮询模式处理数据包：poll-mode 网卡驱动轮训检查是否有数据到达，避免了因为中断导致上下文切换的开销，提升了收发报文的效率。虽然该方式会导致 CPU 一直处于满负荷运行，但却很适合处理不间断大量数据包。当然在特定情况下仍然支持中断。DPDK 无锁队列是通过内核 kfifo 无锁队列完成的。数据使用生产者消费者模型，

一对一，一对多，多对多实现入队和出队，不仅保证了数据的同步，也提高了性能。

2.用户态驱动：传统协议栈在内核态实现，用户态和内核态交换数据需要内存拷贝和系统调用，不必要的消耗很大。使用 DPDK 可以实现用户态驱动，避免了内存拷贝和系统调用。但用户需自行实现协议栈对数据进行处理和解析。

3.独占与亲和性：为避免不同核间线程的频繁切换，可以指定某个核心的特定任务，保证 cache 的更高命中率

4.降低访问存开销：传统协议栈未充分利用 cache 和内存，使用 Hugepage 可以降低 TLB miss，使用内存多通道交错访问提高内存访问有效带宽。大页是 DPDK 通过用户配置，提前在内存预留一块区域，程序可以通过 API 来申请释放大页，大页最小单位可选 2M 和 1G，并且可选择大页的数量。使用 NUMA 来访问尽可能靠近 CPU 的节点内存，跨 NUMA 节点速度相对较慢。

5.软件调优：数据预取利用了程序的时间和空间局部性原理，软件预先取，cache 行对齐及和 burst 批量处理多元数据，可以一次将 8 个、16 个甚至 32 个报文一次处理，这样可以降低访存次数，提高收发包效率。内存池是 DPDK 巧妙的使用大页技术在用户空间完成的。控制层将数据包扔入内存池，通过指针的方式将控制权转交给下个处理，避免了内存拷贝。

1.1.3.2 DPDK 加密加速技术

随着网络带宽的增加，数据的实时加密成为新瓶颈，DPDK 提供了多种加密驱动和加密方案:

AES-NI:AES-NI 是 Intel 和 AMD 微处理器上 x86 架构的扩展,从硬件上提高了 AES 加密速度,目前 PC 端和服务器的 CPU 对 AES-NI 支持普及率很高。DPDK 支持 aesni-gcm 和 aesni-mb 两种方案。Intel 官方公布的数据：开启 AES-NI 性能比纯软件加密速度提高了 5-8 倍。

armv8:在移动端没有 AES 指令时，谷歌推广使用 chacha20 加密算法，此算法在同等配置手机中是 AES 加密算法速度的 4 倍,随后 ARM 在 ARMv8 处理器之后加入了 AES 指令，AES 加密性能反超 chacha20。DPDK 支持多种硬件加密：AMD 加密协处理器 ccp、飞思卡尔的硬件加密（caam-jr、dpaa2-sec、dpaa-sec）、mvsam、octeontx、以及英特尔的 qat 加密加速卡。openssl 加密库。除此之外还支持 kasumi、snow3g、virtio、zuc 等。

1.1.3.3 用户态协议栈

相比于传统的 Linux 内核协议栈，用户态协议栈优势非常大。数据包到达网卡后会不断产生硬中断，而系统调用和软中断优先级都比硬中断低，产生的消耗比较大。

数据包从网卡传送到应用程序需要经历漫长的过程：数据首先要通过 DMA 方式从网卡传到系统内核指定的缓冲区，之后又要将缓冲区接收到的数据拷贝到用户态空间，该过程时间消耗占到 Linux 内核处理数据包整个流程的一多半以上。若将数据和控制分离，用户空间只用于管理内存、处理数据包、调度 CPU，控制指令交由内核管理，就可以解决用户态和内核态频繁切换的问题。除了频繁软硬中断抢占系统调用产生巨量上下文开销之外，多线程间调度和加锁同样会产生大量上下文切换，DPDK 中采用线程与核心对应来减少线程切换，使用无锁队列避免资源竞争。内存页大小为 4K。操作系统可以为了提高访存速度，避免页失效，但若增加了缓存数量，查询速度又会随之降低，可以使用大页减少映射数目来防止跨内存访问。

1.2 目前已完成的研究工作及结果

1.2.1 IPSec 网关设计与实现

本程序主要涉及 IPSec 的 IKE 和 ESP 两部分，IKE 用作协商密钥和传输规则，而 ESP 用来传输数据的。IKE 又分为 IKEv1 和 IKEv2 两个版本。由于 IKE 阶段数据包较少，实现非常复杂，因此可以使用其他 IPSec VPN 实现 IKE 过程，ESP 阶段数据量巨大，实现相对容易，需要使用 DPDK 进行加速。ESP 部分通信数据加解密使用对称加密算法，若使用软加密，则加密速度成为瓶颈，可使用硬加密来提升加密速度。

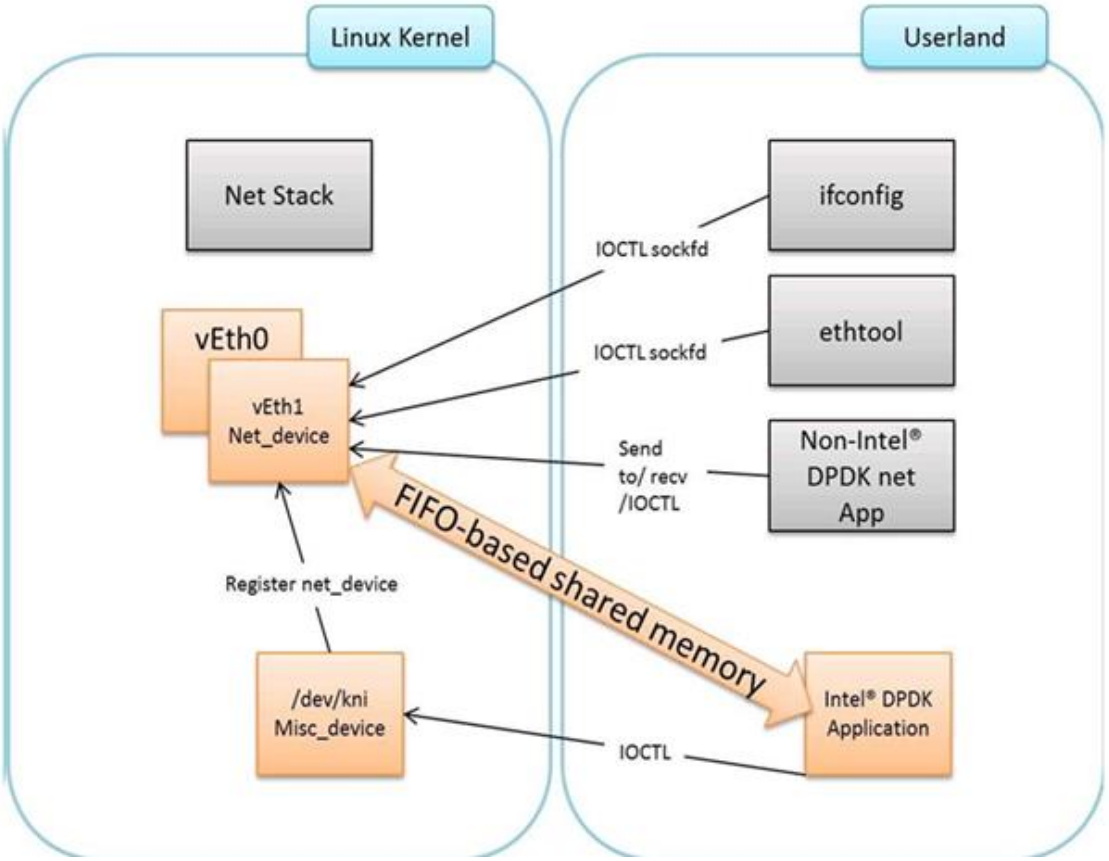
由于 ESP 部分需要加密，而 IKE 部分不需要加密，当数据到达网卡后，需要对数据进行分流，ESP 部分交由 DPDK 进行处理，而 IKE 和其他数据需要传入 Linux 内核协议栈。DPDK 和 Linux 内核态协议栈交换报文有两种模式：KNI 或 TUN/TAP。KNI 比 Linux 现有的 TUN/TAP 接口速度更快，因为和 TUN/TAP 相比，KNI 消除了系统调用和其数据拷贝。本程序采用 KNI 来实现 DPDK 和 Linux 内核态协议栈通信。当数据到达网卡后，DPDK 取到数据，获取数据包 IP 头的协议类型，若是 IPv4 或 IPv6 的 ESP 协议，则交由 DPDK 继续处理，否则交由 Linux 内核进行处理。这样就完成了分流功能。

ESP 部分需要 IKE 协商的 SA 安全联盟和 SP 安全策略。在 Linux 内核 2.6 以后，内核实现了 IPSec 传输部分(ESP 和 AH)和 XFRM 框架。因此 IPSec VPN 有两种方案，第一种方案：IPSec VPN 实现 IKE 部分并使用自己的 ESP / AH 程序来处理数据包。第二种方案：IPSec VPN 实现 IKE 部分，并使用 XFRM 框架将 A 安全联盟和 SP 安全策略传给内核，由内核处理 ESP / AH 数据包。本程序实现类似内核的 ESP 部分。使用 StrongSwan 完成 IKE 部分,配置开启 StrongSwan 的 IKEv1 和 IKEv2,配置 StrongSwan 使用内核 IPSec。使用 NetLink 的 libnl 库监听 IPSec VPN 传给内核的 SA 安全联盟和 SP 安全策略，并将其传给 DPDK 的 ESP 部分程序，而内核虽然收到了 SA 安全联盟和 SP 安全策略，但是由于第一步 ESP 数据被分流给 DPDK 进行处理，内核 IPSec VPN 收不到 ESP 数据。

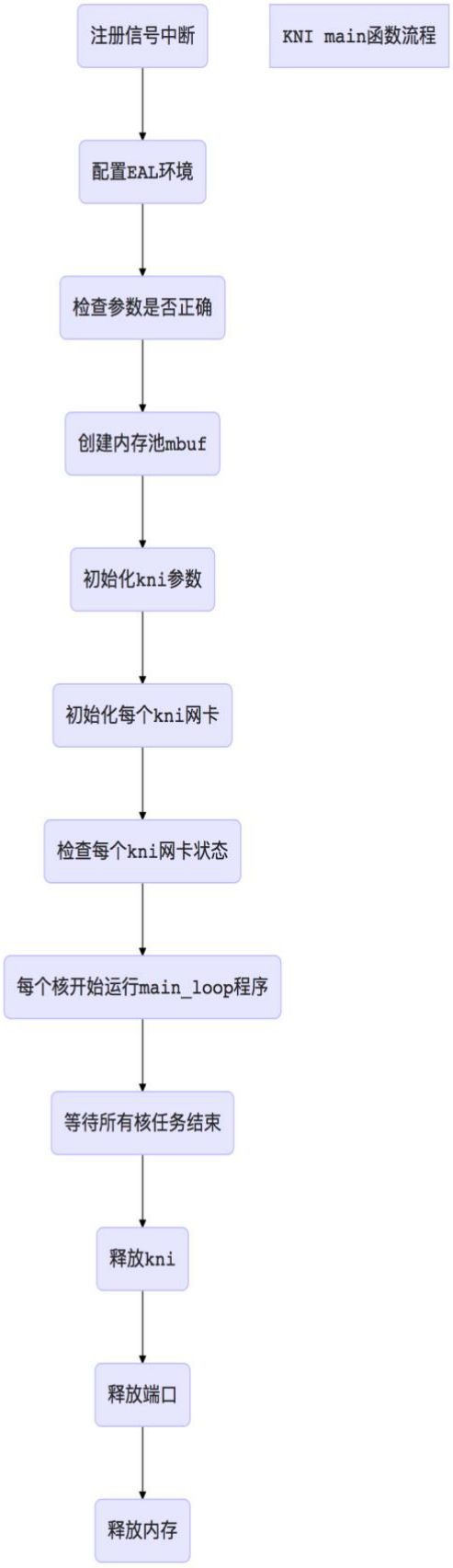
虽然 DPDK 高速转发性能非常出色，但是 DPDK 没有协议栈。Linux 内核使用 ARP 协议 IP 和 MAC 映射关系。而 DPDK 协议栈处理 ESP 数据包时，由于没有 ARP 协议，难以生成以太网头源 Mac 和目的 Mac。当数据的到达网卡后，IKE 部分数据通过 KNI 被分流到内核。本程序使用内核的 ARP 协议解析 MAC 地址，并解析发往内核的 IKE 数据包，解析 IKE 数据包的以太网头和 IP 头，并保存以太网中的 MAC 地址和 IP 头存入数组，用作后续 ESP 协议通过 IP 查询 MAC 地址来生成以太网头。

1.2.2 KNI 内核协议栈转发

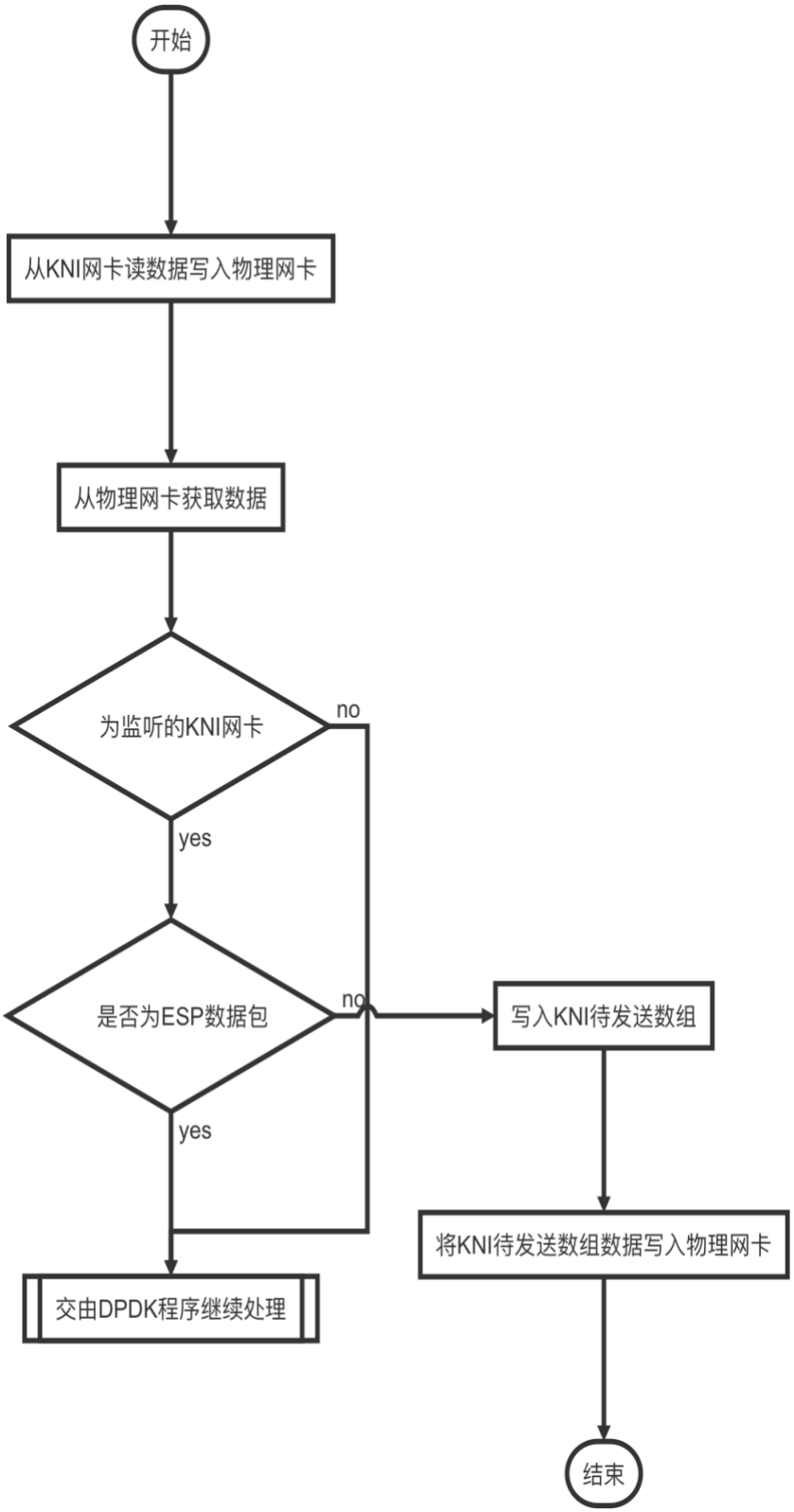
如图所示，KNI 在内核注册一个网卡设备，常见的网卡配置工具可以直接配置该网卡信息，通过基于 FIFO 机制来同步控制信息，网络数据使用共享内存来实现。



如图 KNI 环境初始化流程如图


KNI main函数流程
注册信号中断
配置EAL环境
检查参数是否正确
创建内存池mbuf
初始化kni参数
初始化每个kni网卡
检查每个kni网卡状态
每个核开始运行main_loop程序
等待所有核任务结束
释放kni
释放端口
释放内存

数据的处理流程如图

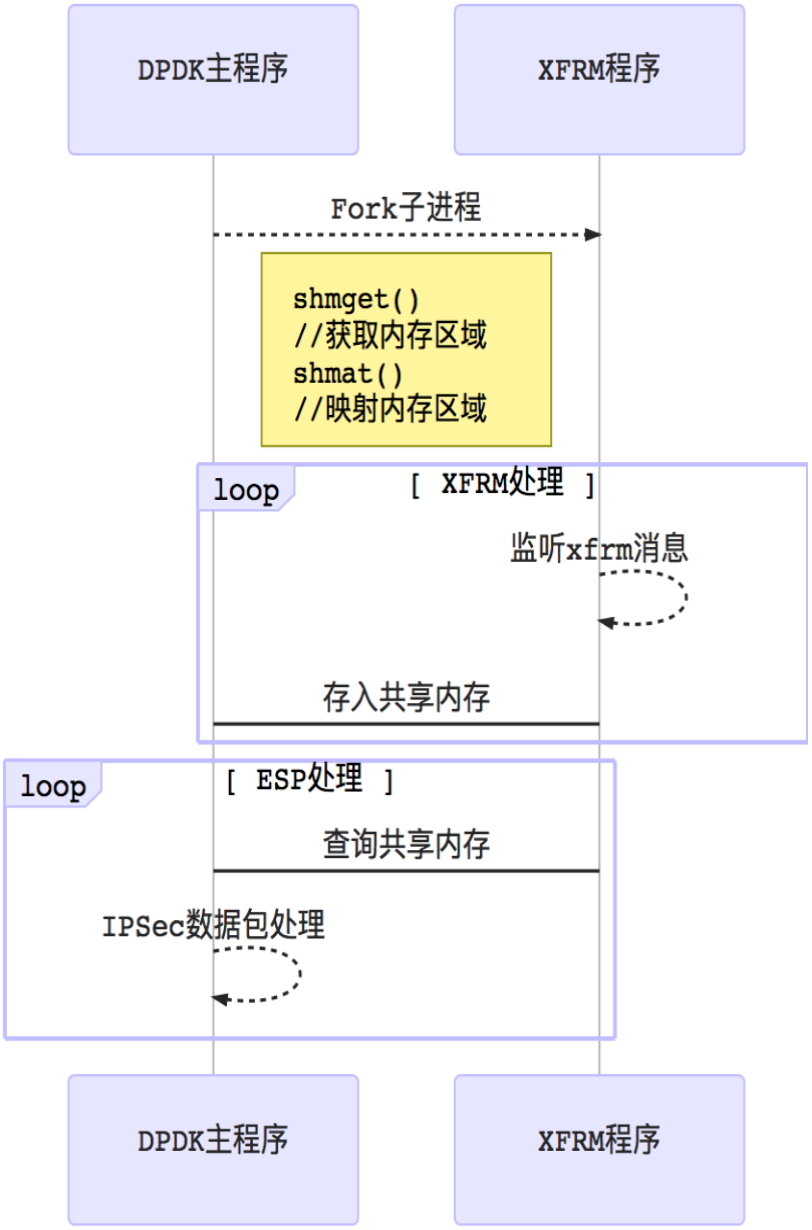

开始
从KNI网卡读数据写入物理网卡
从物理网卡获取数据
为监听的KNI网卡
no
yes
是否为ESP数据包
no
yes
写入KNI待发送数组
将KNI待发送数组数据写入物理网卡
交由DPDK程序继续处理
结束

1.2.3 XFRM 监听设计

DPDK 处理数据包和 XFRM 获取 IPSec 协商结果为两个并行操作，可通过多进程或多线程的方式实现并行操作。若使用多进程，进程间通信方式有：管道，信号，消息队列，信号量，共享内存，原始套接字。

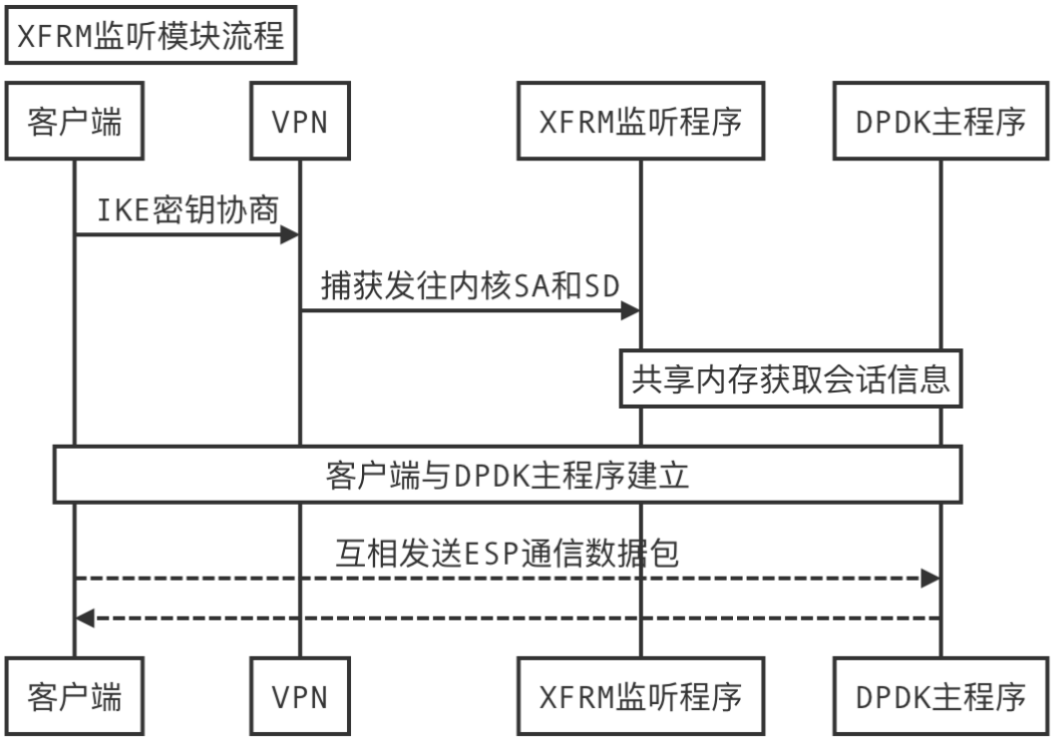
在 Linux 系统中，进程是资源分配的最小单位，线程是调度的最小单位。多进程数据共享复杂，需要 IPC(进程间通信)，但数据同步复杂，而多线程数据在同一个线程中，共享简单，数据同步复杂。和多线程相比，多进程完全复制内存空间，占用空间多，切换复杂，CPU 利用率低，创建销毁速度慢，切换复杂。多进程编程和调试相对简单。和多线程比，多进程间不会相互影响，而多线程中一个线程挂掉可能导致整个进程挂掉。多进程适合多核心多机分布，扩展到多台机器比较简单，而多线程适合多核分布。结合上述有缺点，本程序只需要两个并行任务，不需要频繁创建销毁，任务间通信数据量不大，因此使用多线程。

由于进程间资源隔离，通常进程间不可互相访问。但很多情况下进程通信不可避免，需要进程通过内核或其他进程间通信来完成。常见的进程间通信应用场景有：数据传输、事件通知、共享数据、进程控制、资源共享。管道有三种：无名管道 PIPE、流管道、有名管道 FIFO。无名管道只能父子间通信，并且只能单向流动。流管道可以在父子间双向传输。有名管道可以在多个不相关进程间通信。管道是内核管理的一块缓冲区，空间不大，数据结构为环形以便重复利用。若管道中没有数据，则读取进程会一直等待，直到写进程写入数据；若管道写满，写进程会一直等待，直到读进程取出数据。当两个进程终止之后，管道也会消失。信号可以发给进程，无需知道进程状态，进程若被挂起，当进程恢复执行时才传给进程。若进程阻塞信号，信号传递讲被延迟，知道取消阻塞才继续被传递。信号是一种异步通信方式，来源有硬件和软件。套接字常用于网络间通信，也可用于进程间通信。由于管道，FIFO，消息队列等都使用内核进行通信，而系统调用是用户空间和内核空间的唯一接口，并且需用户态和内核态进行数据复制，消耗比较大，而本程序对实施性要求比较高，因此这几种方案不可取。而共享内存是通过将同一块内存区映射到不同进程地址空间中，不经过内核，因此共享内存是 IPC 中速度最快的，但共享内存需要用户来操作并且同步也需要用户来完成。因此本程序采用最复杂的 mmap 共享内存完成，并使用 CAS 无锁技术来避免加锁，提高性能。



xfrm 使用 netlink 机制来实现 ipsec 用户态程序和内核通信。netlink 是 Linux 内核与用户空间进程通信的一种机制，类似于 UDP 协议，也是网络应用程序与内核通信最常见的接口。netlink 是一种异步通信机制，在内核和用户态之间，传递消息存不用等待存在在 socket 缓冲队列中即可，而系统调用和 ioctl 是同步通信机制。本程序采用 libnl 库来实现捕获应用程序发往内核的 ipsec 的 sa 和 sd。

用户态 VPN->内核(被捕获)



1.2.4 IP-Mac 映射表

Mac 地址获取如图，当数据不为 ESP 数据包时，其他数据(如 ARP,ICMP,TCP,UDP)走 KNI 网卡，若数据为 UDP 协议，目标地址为 KNI 网卡 IP，端口为 500 时，此数据包为 IKE 通信数据包，可将该数据包的 IP 和端口作为一组 IP 和 MAC 映射表存入数组以备查询使用，以太网头、IP 头、UDP 头如下：

UDP 数据封装

以太网头:目的地址(6)|源地址(6)|帧类型(2)

IP 头:

版本(4)|首部长度(4)|服务类型(8)|总长度(16)|

标识(16)|标志(3)|片偏移(13)|

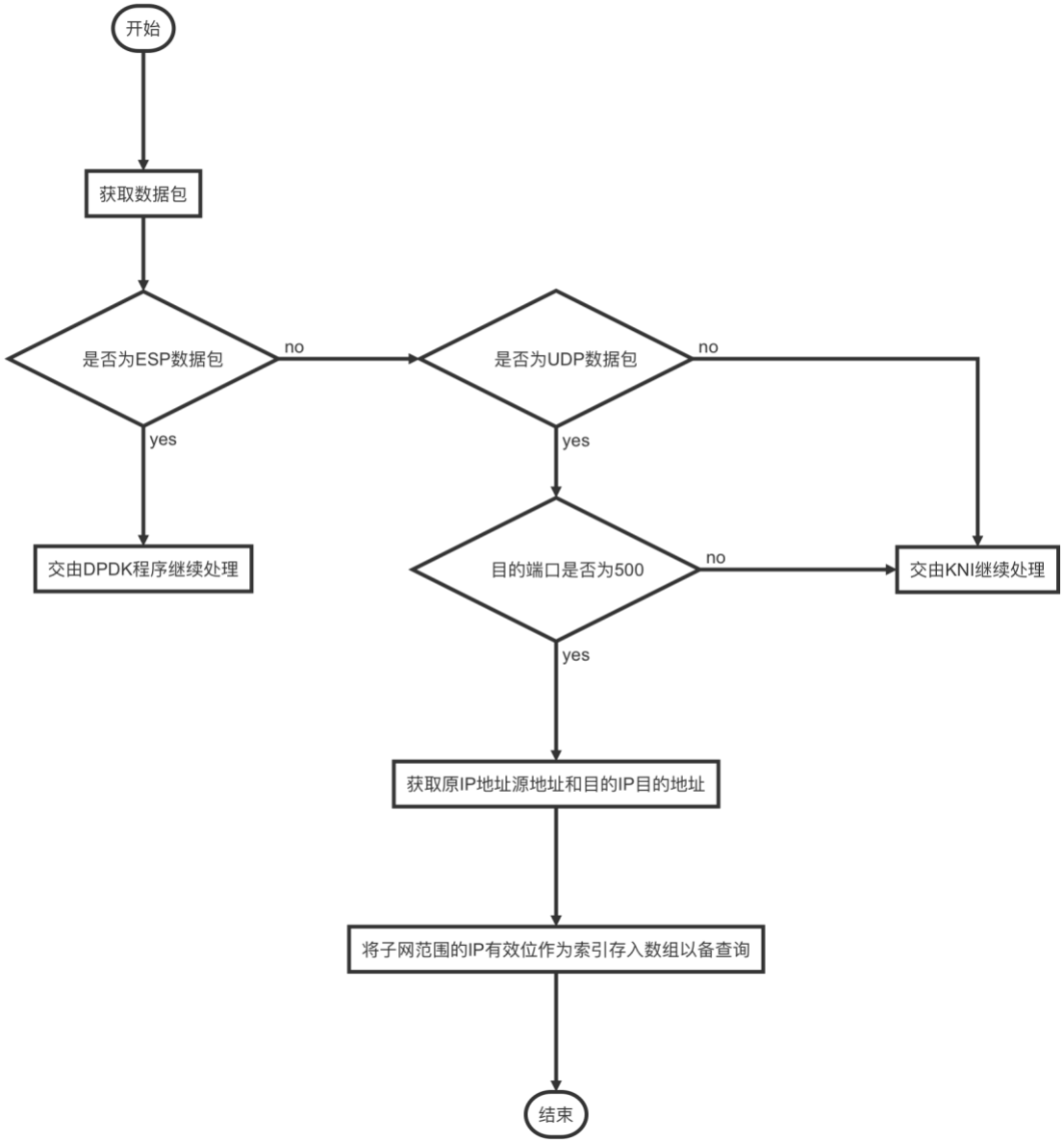
TTL(8)|协议(8)|首部校验和(16)|

源 IP 地址(32)|

目的 IP 地址(32)|

UDP 头:

源端口(8)|目的端口(8)|包长度(8)|校验和(8)



2. 后期拟完成的研究工作及进度安排

2.1 后期拟完成的研究工作

1. 完成 DPDK 数据转发到 tun 网卡

2. 完成 ipsec 客户端，DPDK 程序，Linux 内核网卡的对接

3. 搭建 DPDK-pktgen 发包工具环境并完成性能测试

4.如果有时间和设备，完成使用 QAT 加密卡、AES-NI CPU 指令集加密加速及其性能提升

5.使用类似内核自旋锁模型的 CAS 操作来替换多进程共享内存数据同步安全问题

2.2 后期进度安排

5 月 6 日-5 月 10 日 完成 tun 功能

5 月 10 日-5 月 13 日 完成对接工作

5 月 13 日-5 月 20 日 完成性能测试

3. 存在的问题与困难

问题 1：DPDK 程序和 pktgen 性能测试工具同时启动，程序崩溃

解决方案：机器 CPU 性能和内存不足，发包机和测试机不能使用同一台机器

问题 2：内核无法收到来自 KNI 网卡的数据

解决方案：

1. rte_kni 的驱动不带入参数，直接 insmod 即可，例子中的 insmod=lo_mode 的意思是仅仅把报文发给回环网卡而不处理，lo_mode 是 disable 状态时，数据才经过内核协议栈
2. 使用 kni 设备的 iptables 关闭，在 centos7.3 中，需要将 iptables 先启动在停止才生效，iptables 无法处理数据

问题 3：升级内核版本后 DPDK 无法启动

解决方案：DPDK 程序依赖内核驱动，DPDK 17.02 版本必须使用 Linux 3.10.0-514 版本的内核，不能随便使用 yum update 升级内核，可手动从官网下载对应版本内核的 kernel、kernel-tools、kernel-tools-libs，然后先安装旧版本内核程序，再强制卸载新版本内核

问题 4：安装 AES-NI CPU 加速加密环境后，程序崩溃

解决方案：CPU j1900 不支持 AES 加密加速 AES-NI 指令集，需更使用支持 AES-NI 指令集设备

问题 5:数据必须攒满缓冲区才会发送到物理网卡

解决方案：程序默认缓冲区满时发送数据包。可设置定时器，定期发送缓冲区中数据到物理网卡

问题 6:内核物理网卡无法接受 DPDK 物理网卡发出的数据

解决方案：将目的 MAC 地址设置为内核目的物理网卡 MAC 地址并添加 ARP 映射表，内核仍无法处理数据包，拟在后期使用将数据直接通过 tun 虚拟网卡使用异步非阻塞模型发送到内核。

4．论文按时完成的可能性

如果不改需求，不加需求，有充足性能的测试机，勉强能吧。

5．参考文献

[1] RFC4301, Security Architecture for the Internet Protocol [S]. USA: S. Kent,K. Seo 2015.12

[2] RFC4303, IP Encapsulating Security Payload (ESP) [S]. USA: S. Kent. 2015.12

[3] RFC3602, The AES-CBC Cipher Algorithm and Its Use with IPsec [S]. USA: S. Frankel,R. Glenn,NIST,S. Kelly,Airespace 2003,9

[4] RFC2404, The Use of HMAC-SHA-1-96 within ESP and AH [S]. USA: C. Madson,Cisco Systems Inc,R. Glenn,NIST 1998,11

[5] RFC2367, PF_KEY Key Management API, Version 2 [S]. USA: D. McDonald,C. Metz,B. Phan 1998,7

[6] 阚闯,栾新,戚玮玮. 基于 Linux 的 XFRM 框架下 IPSec VPN 的研究[J]. 计算机工程. 2008(20)

[7] 孙云霄. 面向企业用户的高性能 VPN 系统的设计与实现[D]: [硕士毕业论文]. 威海: 哈尔滨工业大学(威海). 2015.

[8] 穆瑞超. 基于DPDK的高性能VPN网关的研究与实现[D]:[硕士毕业论文]. 威海: 哈尔滨工业大学(威海). 2017.

[9] 吴承. 用户态 IPSec 协议栈的研究与实现[D]: [硕士毕业论文]. 西安: 西安电子科技大学. 2014.

[10] 唐宏,柴桌原,任平,王勇. DPDK 应用基础[J]. 电信科学. 2016(09)

[11] 朱河清. 深入浅出 DPDK[M]. 机械工业出版社. 2016

[12] 王冠群. IPSec 中间人攻击检测方法与防御策略[D]: [硕士毕业论文]. 威海: 哈尔滨工业大学(威海). 2018.

[13] 廖悦欣. IPSec 协议实现技术研究[D]: [硕士论文]. 华南理工大学. 2013.

指导教师评语：_____

_____

_____

_____

_____

指导教师签字：_____　　检查日期：_____