FD.IO/VPP和DPDK Cryptodev，会产生什么样的化学反应？

原创 DPDK开源社区 2017-05-12

作者 张帆

什么是FD.IO/VPP

DPDK能干什么我们这里就不用多说了。但FD.io是什么？和DPDK又有什么关系呢？FD.io是 Fast data – Input/Output 的简称，是Cisco提供的若干数据处理的Linux Foundation下开源项目和库的集合。FD.io中包含大名鼎鼎的VPP (Vector Packet Processing) ，是一套基于DPDK的网络帧处理完整解决方案，也就是俗称的懒人包：如果你的应用需求不是特别苛刻，VPP可谓是即插即用。一个make run命令你的系统瞬间就支持以下这些让人眼花缭乱的功能，而你所要做的只是输入命令来配置即可。

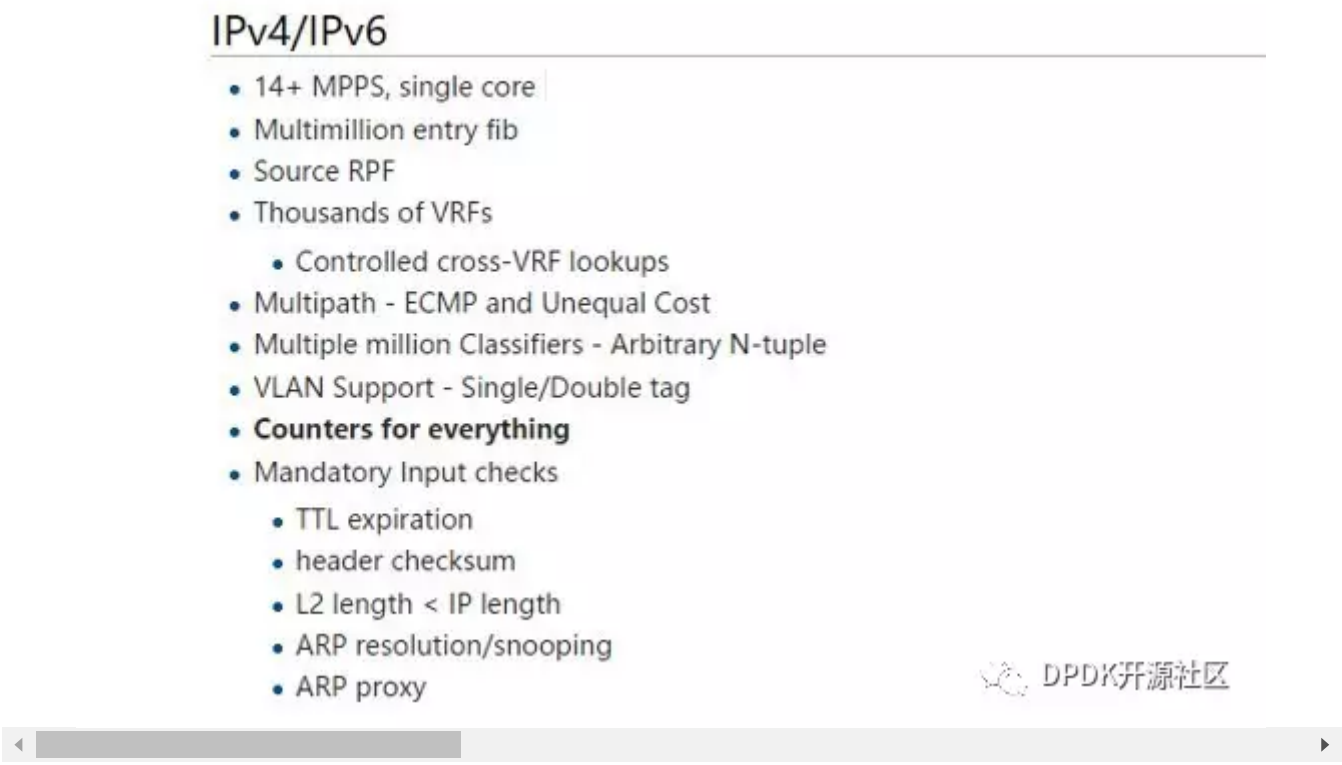

图1. VPP特性列表（可滑动），图片来自https://wiki.fd.io/view/VPP/Features

VPP是个延展性很高的应用，一个应用在VPP里以连起来的若干节点 (Graph Node) 组成，每个节点包含上述的一个或多个功能。网络帧在VPP中被储存在网络帧向量Packet Vector中，它也是节点的唯

一处理对象，节点也会根据处理结果来决定网络帧的下一个目的地节点。你要是觉得这些功能还不够用，你也可以开发自己的插件Plugin来增加自己的节点。VPP是Run-to-completion模式的，这样能更有效地利用Cache。哦，对了，和DPDK一样，VPP也是运行于用户态的，对虚拟化也有很好的支持。

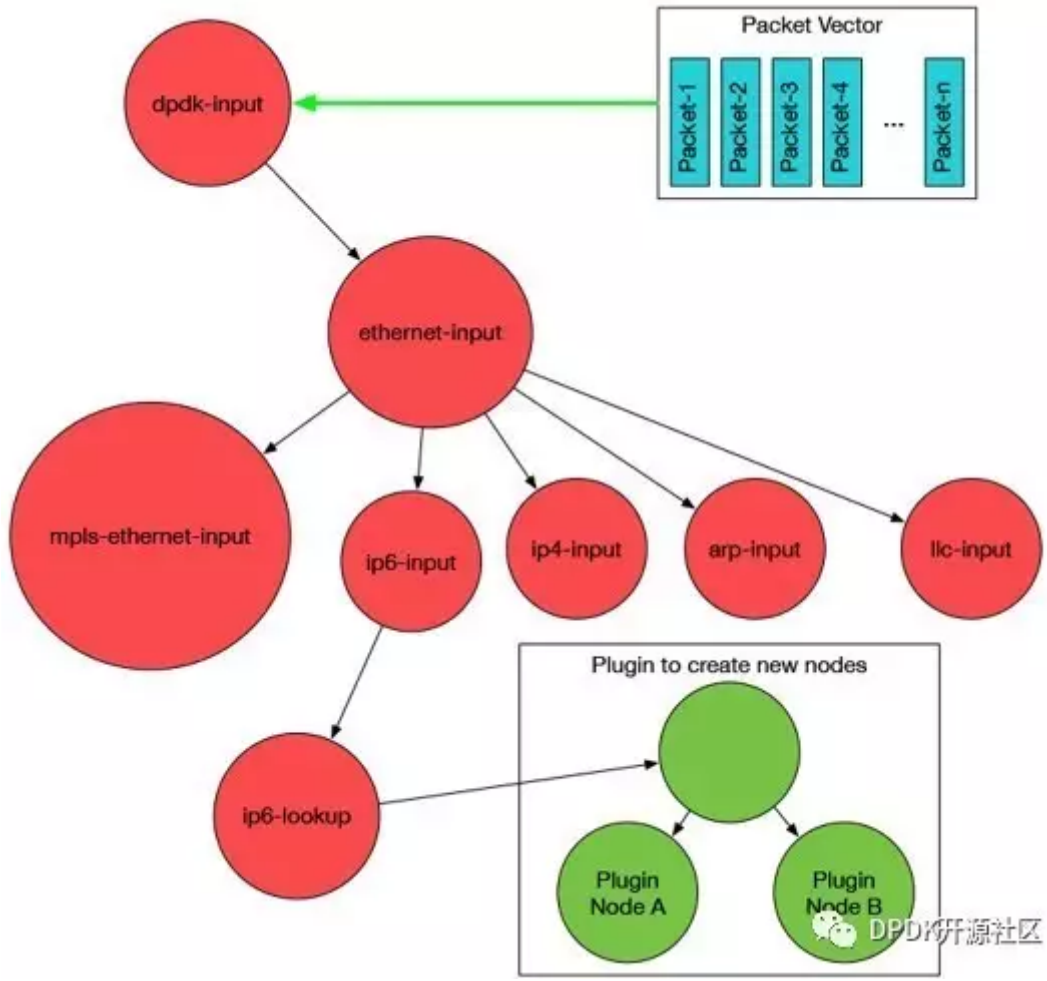

图2. VPP的节点结构图

DPDK Cryptodev和VPP IPSec的化学反应

VPP的功能很强大，里面甚至提供了完整的IPsec协议栈支持：ESP， Transport和Tunnel模式，以及IKEV2。然而，VPP IPsec所使用的Crypto引擎是Openssl而非DPDK Cryptodev。这么强大的功能怎么能没有DPDK Cryptodev的加持呢？因此，我们将VPP IPsec和DPDK Cryptodev进行了融合，在VPP IPsec中成功启用了DPDK Cryptodev来负责所有的Crypto工作，让两者之间有了强烈的化学反应。

首先，我们看看原有的VPP IPsec的节点组成：

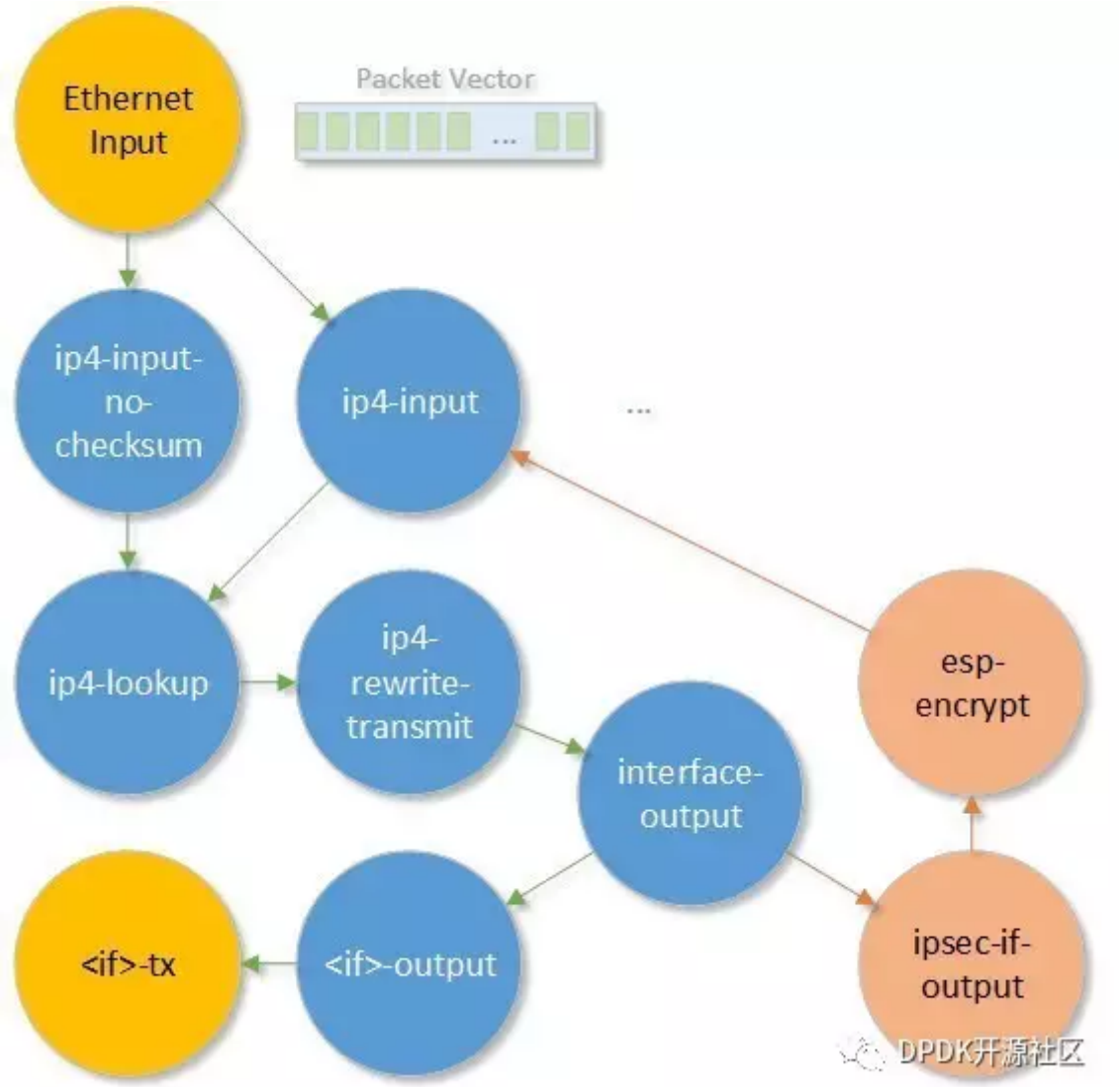

图3. VPP IPsec工作流

可以看到，在interface-output节点处，程序发现网络帧是ESP协议帧后，将其组成 Packet Vector传递给ipsec-if-output节点，再将其交给esp-encrypt/esp-decrypt做Crypto工作。esp-encrypt和decrypt节点的工作复杂度最高。而正因为VPP的run-to-completion模式，这个节点将制约整个系统的效率。

因此，我们所做的第一件事，就是将这两个节点换成DPDK Cryptodev加持的dpdk-esp-encrypt和dpdk-esp-decrypt。为了适应DPDK Cryptodev的enqueue和dequeue模式，我们还增加了两组节点：

- **dpdk-crypto-input：**轮询节点，用来dequeue 处理完的工作流。
- **dpdk-esp-encrypt-post/dpdk-esp-decrypt-post：**网络帧的再封装。

这样，VPP DPDK IPsec的工作流成了下图所示了：

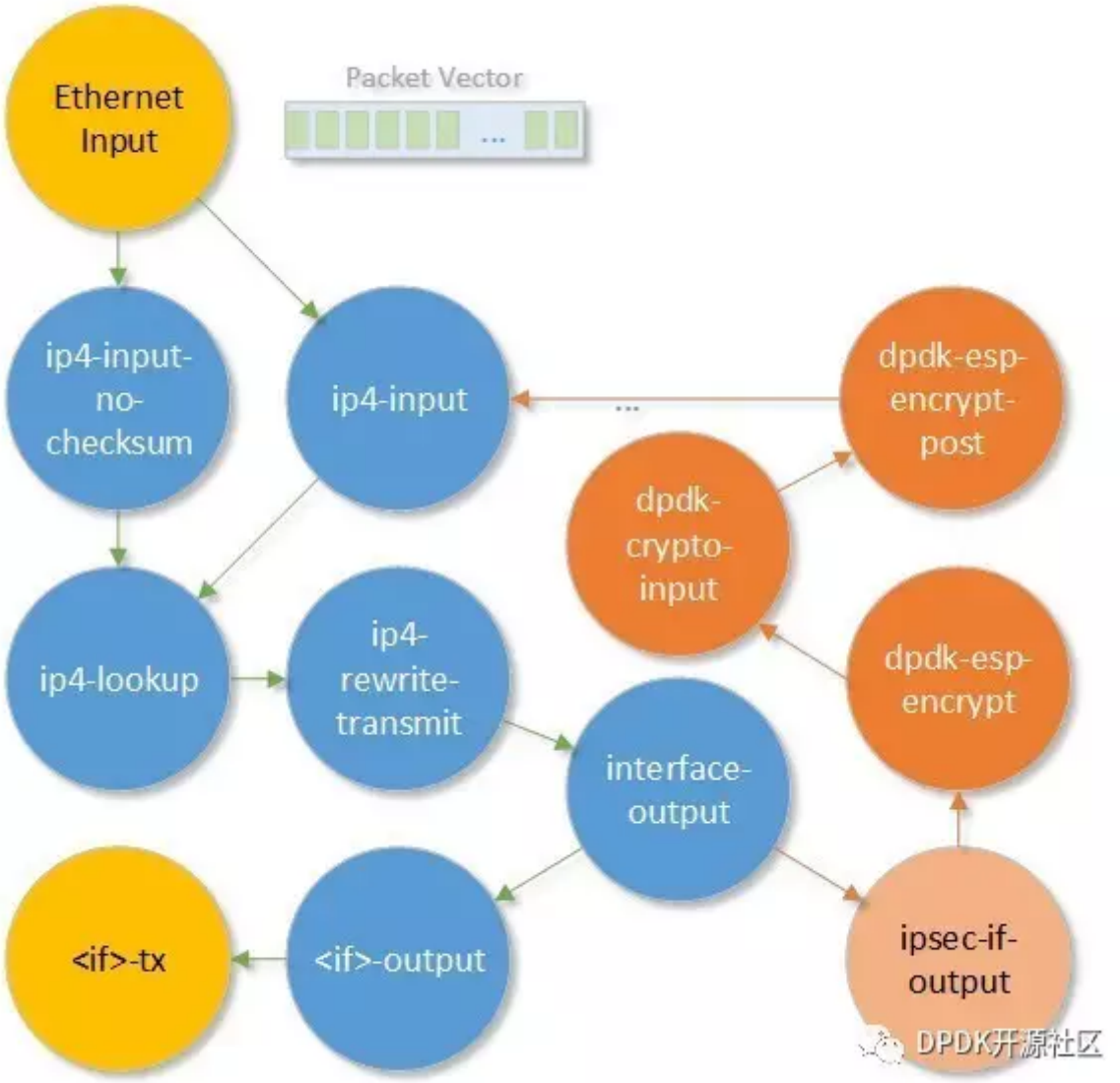

图4. VPP DPDK IPsec工作流

那该怎么启动这新奇的化学反应呢？首先，VPP里的DPDK配置需要启用所需的Cryptodev PMD驱动的编译，同时在VPP的环境配置文件里将以下行修改成

vpp_use_dpdk_cryptodev=yes

编译一次VPP，然后，VPP的startup.conf启动配置文件里也加上你要用的Cryptodev设备描述：

```
dpdk {
  socket-mem 1024,1024
  num-mbufs 131072
  dev 0000:81:00.0
  dev 0000:81:00.1
  dev 0000:85:01.0
  dev 0000:85:01.1
  vdev cryptodev_aesni_mb_pmd,socket_id=1
  vdev cryptodev_aesni_mb_pmd,socket_id=1
}
```


图5. VPP Startup.conf申明DPDK Cryptodev的范例

其中0000:85:0x.x是QAT设备的PCI地址，cryptodev_aesni_mb_pmd是软件AESNI设备的描述符。当然这些cryptodev不是随意配置的，系统对包含的Cryptodev数量还是有点小要求的，这个我们下一节介绍。

然后...就行了，是不是很简单？

化学反应过程浅析

能够让VPP IPsec用上DPDK Cryptodev还是需要一个先决条件的。VPP的运行模式是Run-to-completion，每一个线程都是一个Worker，它会将所有分配给它的网络帧在图4的每个节点中都过一遍。因为每个Worker需要有2个可用的DPDK Cryptodev Queue Pair，一个用于处理用于输出的数据流，另一个用于处理输入的数据流。因此，为了让每个Worker都有必要的Cryptodev资源可用，我们为VPP DPDK Cryptodev作了如下限定：可用的DPDK Cryptodev的Queue Pair总数必须大于等于VPP Worker数量的2倍，而且将优先配置Intel® QuickAssist (QAT)资源。

这里就需要稍微提及一下DPDK Cryptodev的知识了。QAT设备原生支持所有IPsec标准要求的Crypto算法，DPDK QAT 驱动所支持的DH895xCC设备又能提供最多32个VF (Virtual Functions)，每个VF拥有2个Queue Pair，正好提供给一个Worker使用。而DPDK的每个软件Cryptodev仅支持一种或几种Crypto算法，但单个Cryptodev就支持8个Queue Pair。假设我们像图5一样申明要用的DPDK Cryptodev设备，那么VPP DPDK IPsec节点初始化时就能侦测到2个QAT的VF和2个AESNI_MB软件设备。2个VF能提供4个Queue Pair，而2个AESNI_MB则能提供16个，那么所有的Queue Pair加起来够10个Worker使用，而且第1,2个Worker将分得速度更快，时延更低的QAT资源。如果你的系统有超过10个Worker，那么不好意思，Queue Pair不够分的，系统只能拉响警报并drop所有网络帧。

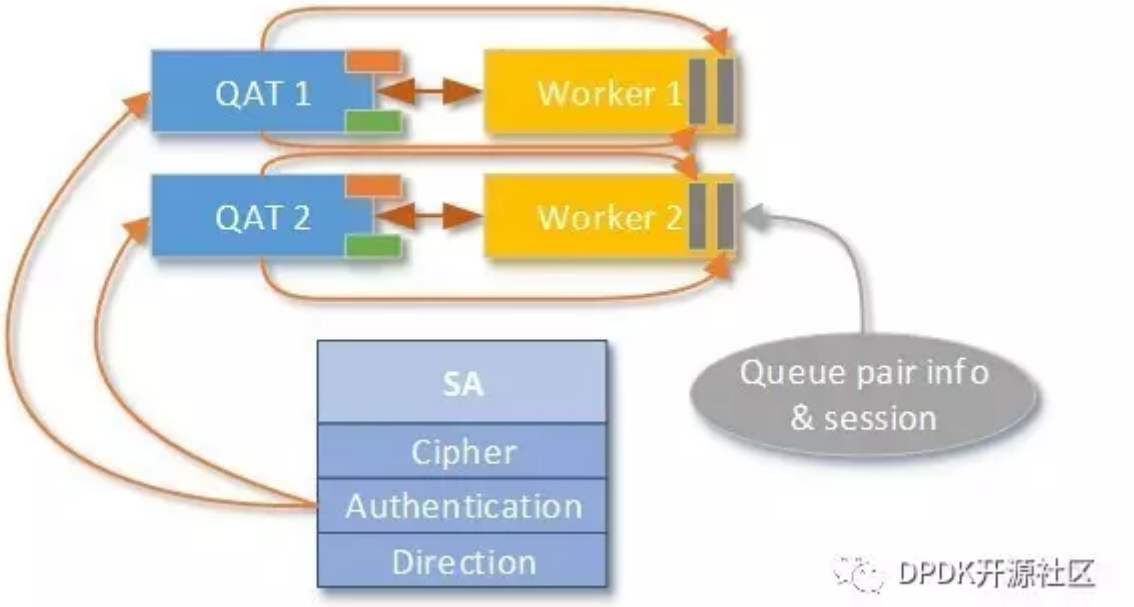

图6. VPP DPDK IPsec初始化流程

初始化时，系统将每个Worker配置2个Queue Pair。在用户要设置一个SA (Security Associations) 规则时，系统将根据SA所界定的Crypto算法和方向（加密/解密）来向Cryptodev为每个Worker申请对应的Session，来确保系统的延展性。如图6所示。

在dpdk-esp-encrypt/decrypt节点获得一个Packet Vector后，因为Session在每个Core中的索引号和SA在系统中的索引号一致，这样在一个IPSec操作时仅进行一次查表操作就能获得目标Cryptodev的ID，Queue Pair指针，以及Session指针。这时系统就能根据其为每个网络帧搭建DPDK Cryptodev所能识别的Crypto op，并将其enqueue来进行Crypto操作。如图7所示。

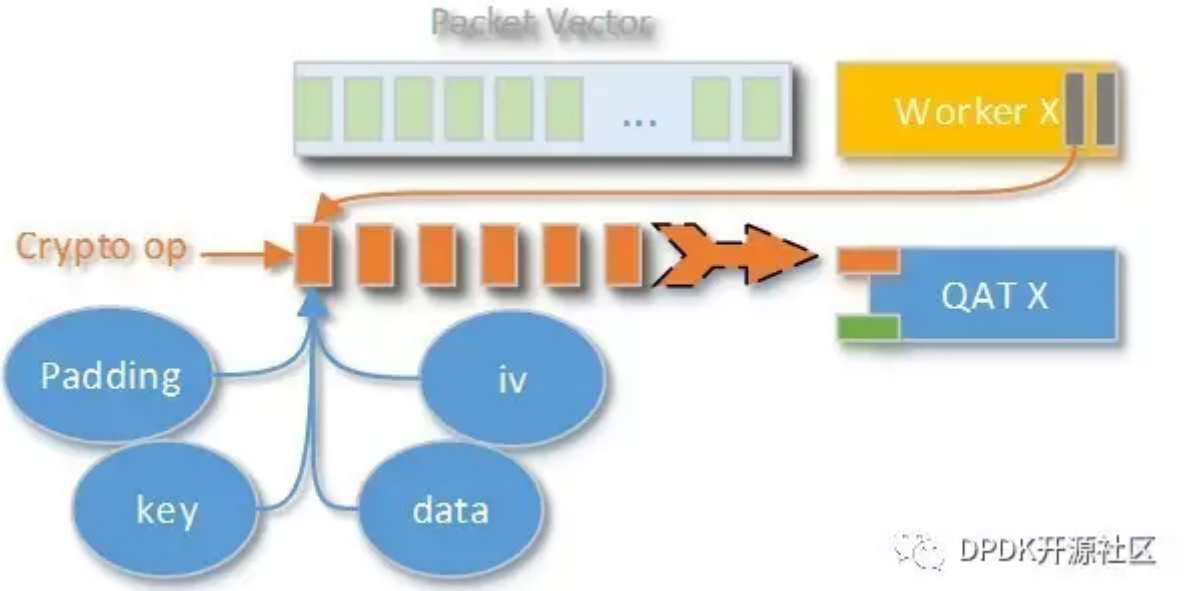

图7. Crypto Enqueue工作流

在上一篇DPDK Cryptodev介绍中我们提及过，DPDK Cryptodev的enqueue和dequeue操作是非对称的，这样能减少enqueue/dequeue本身的开销并最大化QAT资源的利用。然而，这个操作方式

和VPP的Run-to-completion模式相悖，因为假若我们必须在enqueue完后等待Crypto操作全部完成再进行下一步的话，CPU只能原地空转而浪费掉大量的Cycle。

为此我们新加入了一个节点dpdk-crypto-input，该节点负责在dpdk-esp-encrypt/decrypt节点执行完成后直接进行dequeue操作，来获取Cryptodev中处理好了的Crypto Op，并在dpdk-esp-encrypt/decrypt-pos节点中将处理好的Crypto OP重新封装成网络帧向量并传递给下一个节点，如图8所示。

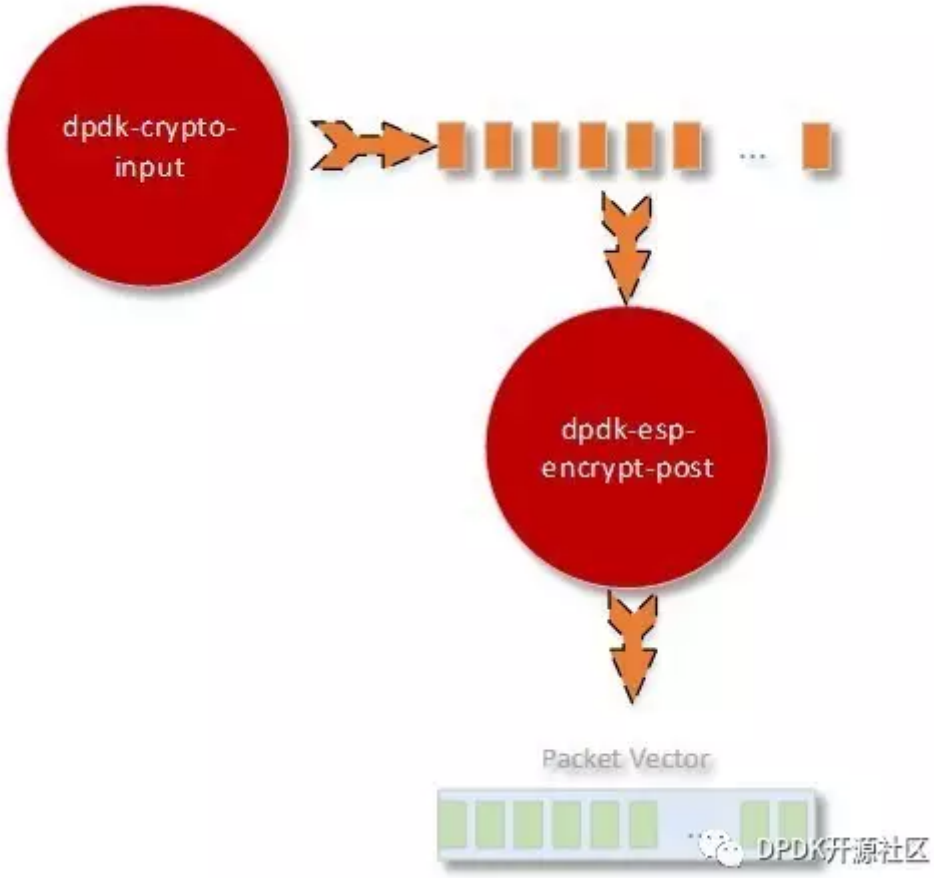

图8. Crypto Dequeue工作流

以上就是VPP DPDK IPsec的简单介绍，怎么样，这个化学反应还不错吗？如果您对DPDK和DPDK Cryptodev有什么疑问或者建议，请登录dpdk.org网站或者发邮件至dev@dpdk.org或者user@dpdk.org参与讨论，关于VPP DPDK IPsec更详细的介绍，请登录 https://docs.fd.io/vpp/17.01/dpdk_crypto_ipsec_doc.html 网站，相关的配置和命令请登录 https://wiki.fd.io/view/VPP/IPSec_and_IKEv2 查询。感谢阅读！

作者

简介

张帆，爱尔兰利莫里克大学计算机网络信息学博士，现为英特尔公司爱尔兰分部网络软件工程师。近年专著有*Comparative Performance and Energy Consumption Analysis of Different AES Implementations on a Wireless Sensor Network Node*等。发表SCI/EI 检索国际期刊及会议论文3 篇。目前主要从事英特尔DPDK 在SDN应用方面的扩展研究工作。

往期文章精选

DPDK

- Hyperscan与Snort的集成方案
- DPDK开源社区加入Linux基金会大家庭了
- DPDK中的memcpy性能优化及思考
- Hyperscan的模式选择
- 关于DPDK Cryptodev，你不得不明白的几点！
- 玩物志 | 什么！DPDK在盒子里？
- 基于virtio-user的新exception path方案
- T-Rex技术，炫到没朋友！
- DPDK Community Update & Fast Data, FD.io, Project
- 技术分享-DPDK编程实例-HelloWorld

DPDK开源社区

最权威的DPDK社区

长按二维码关注

投诉