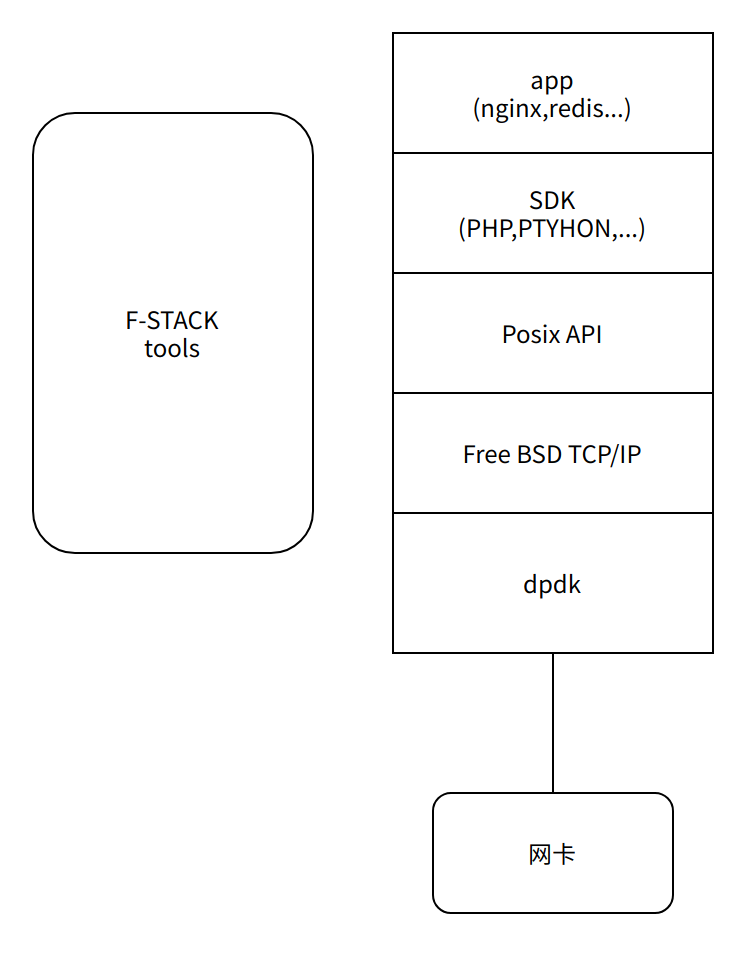
**F-STACK**

**f-stack架构**



f-stack是一种基于DPDK的用户空间协议栈，它旨在提供更高性能和更低延迟的网络数据包处理方式。与传统的内核空间协议栈相比，f-stack将协议栈移植到了用户空间，从而避免了内核空间和用户空间之间频繁的上下文切换，同时还提供了更大的自由度和灵活性。

f-stack架构的设计具有以下几个特点：

1. 基于DPDK：f-stack使用DPDK作为基础框架，利用其提供的高性能数据包I/O功能和优化的内存管理机制，实现了高效的数据包处理。
2. 用户空间协议栈：f-stack将协议栈移植到了用户空间，通过直接访问网卡设备的方式，避免了内核空间和用户空间之间的上下文切换，从而提高了处理性能和降低了延迟。
3. 支持多协议：f-stack支持TCP、UDP、ICMP等传输层协议以及IPv4/IPv6网络层协议，并提供了完整的协议栈实现，可以方便地扩展和定制。
4. 简单易用：f-stack提供了简单易用的API，使得开发者可以快速地构建自己的高性能网络应用程序。

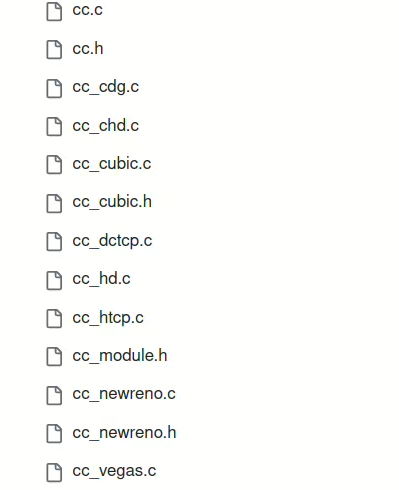
**拥塞控制：**

**freeBSD已有的拥塞控制算法**

freeBSD支持多种拥塞控制协议（f-stack/freebsd/netinet/cc）：

BBR、DCTCP、CUBIC、vegas、newreno

通过配置conf.ini文件中net.inet.tcp.cc.algorithm字段选择拥塞控制协议



**freeBSD拥塞控制结构体**

|  |
| --- |
| C struct cc\_algo {  char name[TCP\_CA\_NAME\_MAX];  /\* Init global module state on kldload. \*/  int (\*mod\_init)(void);  /\* Cleanup global module state on kldunload. \*/  int (\*mod\_destroy)(void);  /\* Init CC state for a new control block. \*/  int (\*cb\_init)(struct cc\_var \*ccv);  /\* Cleanup CC state for a terminating control block. \*/  void (\*cb\_destroy)(struct cc\_var \*ccv);  /\* Init variables for a newly established connection. \*/  void (\*conn\_init)(struct cc\_var \*ccv);  /\* Called on receipt of an ack. \*/  void (\*ack\_received)(struct cc\_var \*ccv, uint16\_t type);  /\* Called on detection of a congestion signal. \*/  void (\*cong\_signal)(struct cc\_var \*ccv, uint32\_t type);  /\* Called after exiting congestion recovery. \*/  void (\*post\_recovery)(struct cc\_var \*ccv);  /\* Called when data transfer resumes after an idle period. \*/  void (\*after\_idle)(struct cc\_var \*ccv);  /\* Called for an additional ECN processing apart from RFC3168. \*/  void (\*ecnpkt\_handler)(struct cc\_var \*ccv);  /\* Called for {get|set}sockopt() on a TCP socket with TCP\_CCALGOOPT. \*/  int (\*ctl\_output)(struct cc\_var \*, struct sockopt \*, void \*);  STAILQ\_ENTRY (cc\_algo) entries; }; struct cc\_var {  void \*cc\_data; /\* Per-connection private CC algorithm data. \*/  int bytes\_this\_ack; /\* # bytes acked by the current ACK. \*/  tcp\_seq curack; /\* Most recent ACK. \*/  uint32\_t flags; /\* Flags for cc\_var (see below) \*/  int type; /\* Indicates which ptr is valid in ccvc. \*/  union ccv\_container {  struct tcpcb \*tcp;  struct sctp\_nets \*sctp;  } ccvc;  uint16\_t nsegs; /\* # segments coalesced into current chain. \*/ }; |

**tcp协议控制块**

[tcp内核代码](https://uestc.feishu.cn/docx/JZOtdKlSgocutdxflllctD0lnzg)

**tcp源码阅读**

