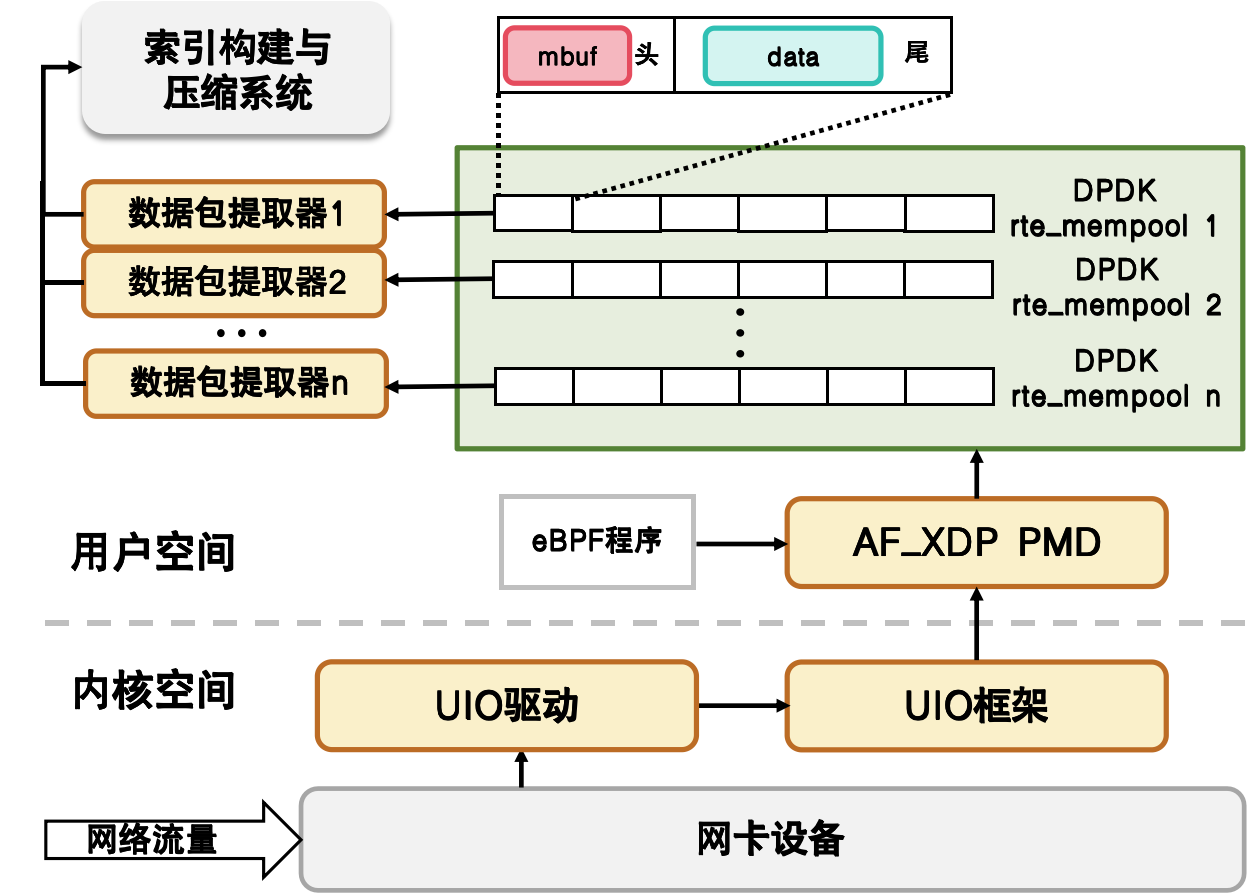
1.流量回溯平台概述

拟实现支持网络全流量捕获、存储和高速检索的流量回溯平台，为安全防御、应用排障和内部监管等流量回溯需求提供支持。该流量回溯平台设计目标在于满足高速流量的全镜像捕获与索引建立，并将流量以高压缩比进行压缩存储，同时支持对用户所需任意网络协议字段的低延迟查询。流量回溯平台预计以四部分组成：**高速流量捕获系统；高并行度索引构建系统；基于相似流聚合的流量数据压缩与存储系统；支持自定义索引建立与检索的用户交互系统。**

* 1. 高速流量捕获系统

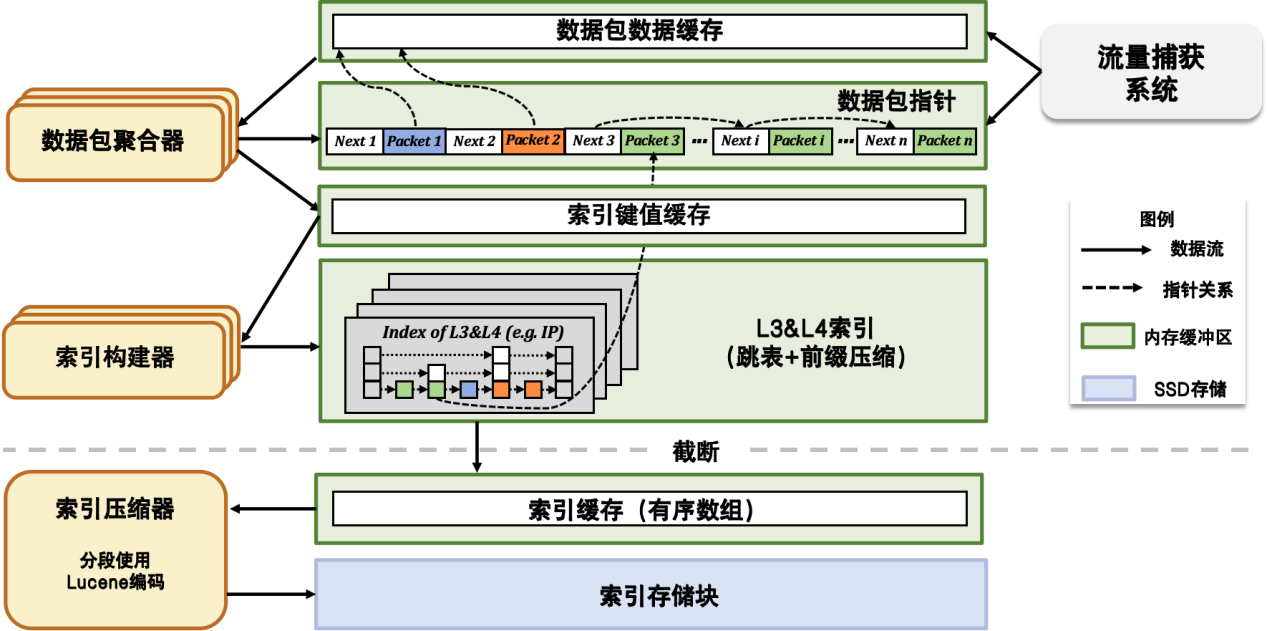


**图1 高速流量捕获系统**

该系统需要满足以下两个需求：(1)对高吞吐量链路流量的全镜像捕获；(2)对流量的初步处理，如对明确无需抓取的流量进行过滤。由于网络流量抓取的主要性能开销来自内核协议栈的内存拷贝开销和系统调用开销（约占75%），而抓取网络流量时不需要在内核栈对数据包进行协议解析，选择**使用DPDK技术实现内核旁路抓包**。同时，DPDK自19.05版本以上实现了AF\_XDP PMD，使得可以在DPDK中直接调用AF\_XDP机制，拟**使用AF\_XDP进行数据包的初步处理**。

（AF\_XDP与XDP存在区别，其是否支持XDP全部功能需要进一步确定；如果DPDK与XDP数据包处理功能不能联合使用，拟在DPDK将数据包传入用户态后再通过用户态程序进行过滤等处理。）

* 1. 高并行度索引构建及压缩系统

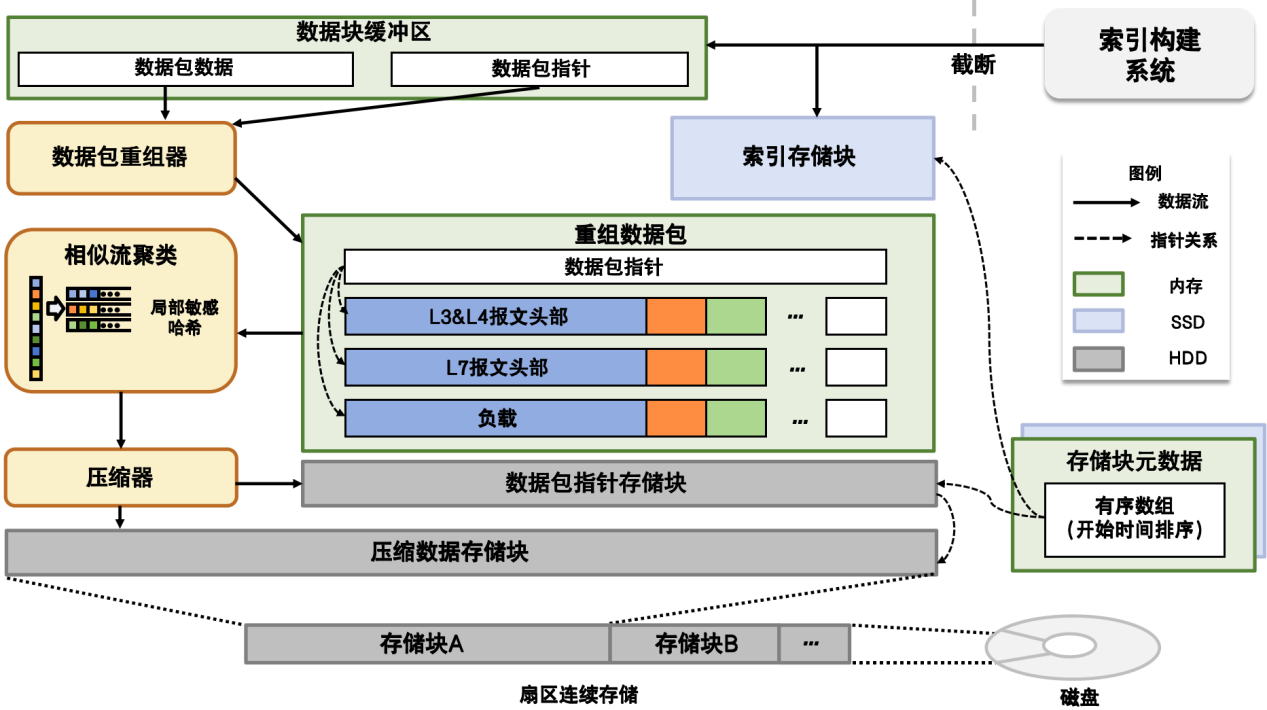


**图2 高并行度索引构建及压缩系统**

由于单一线程难以满足100Gbps流量索引（特别是数据包粒度索引）的实时构建，采用可高度并行化的数据结构完成数据包的初步聚合和索引构建。其步骤为：(1)通过特殊设计的**链表结构**，将IP与端口相同的数据包**在负载零拷贝的前提下完成聚合**；（2）基于流量索引写入后不会发生修改的特征，**通过跳表建立索引**，由于跳表在插入操作时只需修改一个节点，相较于基于平衡树的索引可支持高并行度索引构建。这一设计将数据包与聚合结构连续存储到内存中，尽可能最大限度的利用了内存缓冲区（相较于FLOWSIS对每一个流分配一块固定长度缓冲区），同时对一段时间内的所有抓取流量进行并行索引构建（相较于VAST将数据包分配到不同的节点）。

在流量数据整体截断（见1.3）后，对应的索引也会被固定，并输出到连续内存空间，进而存储到SSD。考虑到用户存在自定义索引的需求（见1.4），为避免索引占据过多的昂贵的SSD存储资源，需要对索引进行压缩。索引压缩可以在两个阶段进行：(1)**索引构建同时进行压缩**，这样可以节省缓冲区使用的内存空间，但是由于不断有新数据进入，为保证索引构建速率与并行度，只能使用固定格式的压缩方式（如**定长前缀压缩**，参考Compact-Index），压缩比受到限制；（2）**索引固定后，在存入SSD前进行整体压缩**，此时由于索引不再发生变化，可以使用**类似于Lucene编码**等针对有序数组的方式进行编码，在保持数组查询复杂度的同时获得更高的压缩比（Lucene编码可改进点：扩展到变长索引；数据段最优划分位置选择，从而使得流量索引适配原本用于递增数据的编码方式）。

* 1. 基于相似流聚合流量数据压缩与存储系统



**图3 基于相似流聚合的流量数据压缩与存储系统**

由于网络流量数据为二进制数据，通用的针对文本数据的压缩方法难以在流量数据中获得较高的压缩比。为此，压缩与存储系统设计基于以下两个见解：(1)网络流量存在大量内容相似的流，如不同用户对同一网页的多次请求等，对其中的相同部分进行整体替换可获得较大的压缩比；（2）更大的数据块可以找到更多的流间相同部分，从而获得更大的压缩比。

因此，我们通过以下步骤对流量数据进行压缩：(1)将1.2中初步聚合的流量进行**整体截断**，形成连续的流量数据块（预计大小在GB级），以流量数据块为单位进行压缩；（2）将流量数据块中**相同流的数据包重新组合**，将路由层、传输层、应用层包头相同字段进行单独压缩，将应用层报文重组为负载数据块；（3）**将协议相同的负载数据块初步聚类，通过局部敏感哈希算法将内容相似的流进一步聚类**；(4)**将流间相同部分使用字典进行整体替换**；（5）将压缩后的数据块**整体写入磁盘**中，**将数据块第一个包和最后一个包的时间戳作为数据块索引键**。

当从磁盘中读取数据包时，读操作包括：（1）根据索引结果读取数据包指针（2）读取数据块压缩表；（3）根据数据包指针，获得压缩后数据包头部、负载所在位置并读取。为降低数据读取延迟，应尽可能减少上述数据读取时的磁盘寻道开销。由于压缩后数据块预计在十兆至百兆量级，远超磁盘读写块的大小（4KB），因此针对小数据块排布的优化（如Hyperion）是不适用的。所以需要在数据块形成时便考虑其内部数据排布位置，以确保：（1）避免同一**数据包指针**跨扇区存储；（2）尽可能避免同一**数据包数据**跨扇区存储。这要求数据块压缩时需要注意上述数据与扇区间的对齐（数据包指针为定长变量易于实施，数据包还需要进一步细化，比如采用特殊编码或排列等）。此外，在写入数据块时，应要求**数据块存储的扇区连续**，这需要**隔离出用于存储数据块的扇区，并直接使用ioctl等操作直接对扇区读写**。

针对硬盘可能的数据损坏，拟采用通用纠删码解决，不作为平台设计点。

* 1. 支持自定义索引建立与检索的用户交互系统

除用于初步聚合流数据的IP地址和端口外，在不同场景需求下，用户还可能对http或DNS协议域名、数据包长度等字段有着快速查询的需求。但是，如果对数据包中每一个协议字段均建立索引，其带来的协议解析时间开销和协议存储开销是难以接受的。为此，我们支持用户对需要的协议进行自定义协议解析与索引建立。**用户可通过交互系统提供的API定义需要解析的协议字段（包括协议层、协议格式等，支持条件判断以实现协议可选字段解析），进而形成lua脚本定义协议解析过程，进而建立自定义索引**。由于应用层报文需要在数据包重组（1.3）后才可获得，自定义应用层索引建立过程也在这一步进行，路由层、传输层索引则与IP、端口索引一同建立。

此外，**考虑到网络流量的协议层-协议-字段的分级特征，拟在查询阶段对查询表达式进行归并优化**（如udp&&ntp等价于ntp）。同时，查询表达式应通过逻辑运算**消除无法索引的“!”运算符**。