A Survey on Internet Traffic Archival Systems

Zhen Chen, Lin-yun Ruan, Jun Li, Shuai Ding, Fu-ye Han, Hang Li, Wen-yu Dong
1)(Research Institute of Information Technology, Tsinghua University, Beijing, 100084)
2)(Tsinghua National Laboratory for Information Science and Technology, Beijing, 100084)

Abstract- With the popularity of Internet applications and widespread use of mobile Internet, the Internet traffic maintains a rapid growth over the past decades. Internet traffic archival system (ITAS) for packets or flow records becomes more and more widely used in network monitor, network troubleshooting, user behavior and experience analysis etc. In this paper, we survey the design and implementation of several typical traffic archival systems. We analyze and compare the architectures and key technologies backing up Internet traffic archival system, and summarize the key technologies which include packet/flow capturing, packet/flow storage and bitmap index encoding algorithm, and dive into the packet/flow capturing technologies. Then, we propose the design and implementation of TiFaflow traffic archival system. Finally, we summarize and discuss the future direction of Internet traffic archival systems.

Keywords: Internet Traffic; Big Data; Traffic archival; Network Security; Traffic Acquisition; Packet Capturing; Columnar Storage; Bitmap Index; Bitmap Encoding, algorithm.

1. Introduction

1.1 Big data

Massive, high-speed, dynamic data appear in major areas of the application. In the field of Internet search, Google improves the users’ search experiences by personalized search technology [1]. In this technique, the user’s search behavior information, including Web access path and each page access time, records to a huge database in real time. As search engine will continue to query these information which processes 4200 request per second. When users search for some information. In scientific experiments, the experiments of the European Large Hadron Collider (LHC) produced 15PB data which rated up to 1.5GBps [2].

Network monitor, communication services, sensor networks, and financial services generate unlimited, continuous, rapid, real-time streaming data. Streaming data is characterized by massive, continuous and real-time, it is not feasible that using single linear scanning for random access and storing the stream data locality completely, as the on-line analysis process requires fast, and real-time analysis system resources are limited. Once the flow of data management in emerging overload, it will store, query, handling a significant impact. Once the data flow emerges overload, it will have a great influence on storage, query and analysis process. Traditional database storage is a collection of static relational data records for persistent data storage and complex queries. Query operations are more frequently than insert, update, delete and other operations, and the results
reflect the current state of the database, and its linear complexity algorithm has been unable to meet the requirements of the storage, query and analysis of massive data streams.

It is critical to query and analyze data for making decisions timely from a large stream. In order to monitor and analyze data streams in real time, handling large data streams is important within a limited time and space. The collection, archiving, indexing, analysis of data streams have become a significant trend in the field of storage and database.

1.2 Internet traffic data

With the popularity of the Internet applications and mobile wireless networks for large-scale commercial, huge amounts of information content greatly enriches the users. The outbreak of the mobile Internet which allows users from anywhere and anytime access to any content of the network, results in generating more traffic data. The entire Internet traffic maintains a rapid growth as a normal Internet company generates and accumulates users traffic data is quite large, which cannot use gigabit (G) or trillion (T) to measure. Cisco’s report [3] predicts that the Internet traffic data will grow four-fold from 2011 to 2016, and reach 1.3 ZB in 2016. Internet traffic data is a typical large data stream.

The information is divided into several data blocks which are transmission units in the Internet, and each piece may be transmitted along different paths in one or more networks which restructure at the destination. These data blocks are called "packet". The Internet transmission is based on TCP/IP protocol, which divided the network packets into IP packets, TCP/UDP packets and application packets because packets contain different information.

Figure 1. Pcap file format.

Figure 1 shows the format of the original data acquired from a physical circuit on the Internet, which generally store as Pcap files. Raw network packets store sequentially in Pcap files. Pcap file header describes the overall properties of Pcap files, such as file size. Packet header contains the description information, such as timestamps, size, etc., and packet payload includes the contents of the complete TCP / IP network packets, see Figure 2.

Figure 2a shows an IP packet format, including packet header and payload, mainly has the following fields: version (4 bits), header length (4 bits), type of Services (8 bits), total length (16 bits), identification (16 bits), flags (3 bits), fragment Offset (13 bits), TTL (8 bits), protocol (8 bits), header checksum (16 bits), source IP address (32 bits), destination IP address (32 bits).
Figure 2b shows a TCP header formats, including packet header and payload two parts, mainly has the following fields: source port number (16 bits), destination port number (16 bits), sequence number (32 bits), acknowledgement number (32 bits), data offset (4 bits), reserved (6 bits), emergency bits URG, confirm bit ACK, reset bit RST, synchronization bits SYN, termination bits FIN, window size (16 bits), checksum (16 bits), urgent pointer (16 bits), option field and padding field.

Network flow records care about characteristics and arrival process, and the record could be a UDP connection or a TCP connection. The flow usually refers to a quintuplet of the source IP address, source port, destination IP address, destination port, protocol.

Table 1. Network flow record structure (Router)

<table>
<thead>
<tr>
<th>Number</th>
<th>Field</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SrcIP</td>
<td>Source IP address</td>
</tr>
<tr>
<td>2</td>
<td>SrcPort</td>
<td>Source Port</td>
</tr>
<tr>
<td>3</td>
<td>DstIP</td>
<td>Destination IP</td>
</tr>
<tr>
<td>4</td>
<td>DstPort</td>
<td>Destination Port</td>
</tr>
<tr>
<td>6</td>
<td>Proto</td>
<td>Layer 3 protocol (e.g., TCP UDP)</td>
</tr>
<tr>
<td>8</td>
<td>TCPflags</td>
<td>Cumulative TCP flags (e.g., SYN ACK FIN)</td>
</tr>
<tr>
<td>12</td>
<td>SrcAS</td>
<td>Source Autonomous System Number</td>
</tr>
<tr>
<td>11</td>
<td>DestAS</td>
<td>Destination Autonomous System Number</td>
</tr>
<tr>
<td>9</td>
<td>Octs</td>
<td># Bytes exchanged in the flow</td>
</tr>
<tr>
<td>10</td>
<td>Pkts</td>
<td># Packets exchanged in the flow</td>
</tr>
<tr>
<td>11</td>
<td>Start</td>
<td>Flow start time</td>
</tr>
<tr>
<td>12</td>
<td>Duration</td>
<td>Flow duration</td>
</tr>
<tr>
<td>13</td>
<td>others</td>
<td></td>
</tr>
</tbody>
</table>

A network flow record structure obtained from an Internet router is shown in Table 1, mainly has the following fields: source AS number, destination AS number, start and end time, the number of network packets. Netflow V5 flow record format is more popular which is developed by CISCO.

Netflow determines the packets whether belong to a certain flow based on seven fields: source IP address, destination IP address, source port number, target port number, the third layer protocol type, TOS byte (DSCP), network equipment input (or output) logical network port (ifindex). CISCO’s Netflow V9 has also been identified as the IPFIX (IP Flow Information Export) standard from five candidates by IETF.
Mobile Internet operators obtain traffic record format (CDR), as shown in Table 2, mainly includes terminal attributes, location area code, CI number, the service gateway, start time, end time, TCP/IP information, HTTP application information, etc.

Table 2. Flow record format (in wireless base station)

<table>
<thead>
<tr>
<th>Number</th>
<th>Field</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cell phone number</td>
<td>Not contain a prefix such as +86,0086，86</td>
</tr>
<tr>
<td>2</td>
<td>Location Area Code</td>
<td>LAC</td>
</tr>
<tr>
<td>3</td>
<td>CI number</td>
<td>Select the first CI when a network switches</td>
</tr>
<tr>
<td>4</td>
<td>Terminal</td>
<td>IMEI/IMSI</td>
</tr>
<tr>
<td>6</td>
<td>Start time</td>
<td>YYYY-MM-DD HH:MM:SS.1234567</td>
</tr>
<tr>
<td>8</td>
<td>Duration (in seconds)</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>RAT Type</td>
<td>1 represents 3G, 2 represents 2G</td>
</tr>
<tr>
<td>11</td>
<td>traffic (in bytes)</td>
<td>Upstream/Downstream/total</td>
</tr>
<tr>
<td>9</td>
<td>Terminal IP</td>
<td>traffic (in bytes)</td>
</tr>
<tr>
<td>10</td>
<td>Source port/ Destination port</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>APN</td>
<td>3gwap,3gnet,uniwap,uninet,cmwap,cmnet</td>
</tr>
<tr>
<td>12</td>
<td>SGSN/GGSN IP</td>
<td>The first access IP.</td>
</tr>
<tr>
<td>13</td>
<td>Http protocol</td>
<td>User Agent/Content-type/URL/Status Code</td>
</tr>
<tr>
<td>14</td>
<td>others</td>
<td></td>
</tr>
</tbody>
</table>

1.3 Network flow archiving and retrieval

Network monitoring has been the core function of network management, network fault diagnosis and network security. In addition to the real-time firewalls, intrusion detection systems, traffic archiving system is an important complement to the network protection systems.

Internet Traffic archiving system is passive to capture packet or flow records, index and storage that can be seen as a large flow of a specific retrieval system which has many important applications.

The development of the Internet has made many companies have their own corporate networks. There are also many companies have their own server hosting to the IDC (Internet Data Center). The number of such local networks is too large and network bandwidth is generally 100 ~ 1000Mbps. How to manage the huge data traffic, improve the production efficiency, diagnose network faults and maintain normal operation is a very realistic and challenging issue.

In addition, the data provided by China Unicom[4][44] shows that current mobile Internet users daily generate over 30 billion records and about 8.4TB data in the telecommunications business, monthly reach to trillions of records and storage capacity of PB grade. As the traffic and mobile Internet users are growing rapidly (double about every six months), the Internet records will further increase [4].

With the development of computer network technology, network security issues become prominent because of its openness. Computer networks have to suffer a lot of known or unknown attacks. In this context a lot of network security technologies appear such as intrusion detection systems, signature detection, and security scanning technology. However many attacks cannot be
timely detected and prevented which need network packets capture to achieve information collection for facilitate later analysis and utilize.

The paper is organized as follows: The structure and function of traffic archive system is introduced in Section 2, and the typical evolution and classification of Internet traffic archiving system in Section 2. Section 3 describes the key technologies of packets capture and compress bitmap index encoding of network flow archiving system. We conclude the paper and the future work of traffic archiving systems in Section 4.

2. ITAS and its categories

2.1 Structures and functions in ITAS

Internet traffic archival system, referred to as ITAS, typically includes flow data acquisition, index storage and query processing. Formats of flow data acquisition have the following two categories:

(1) Handling packet level network traffic data;
(2) Handling flow-level network traffic data.

A typical traffic archiving system structure is shown in Figure 3.

![Figure 3. The structure of Internet traffic archiving system](image)

Figure 3 describes a typical traffic archiving system. After being captured, network data can be real time data transmitted to the index module, or be historical data stored to disk waiting for a call. Index module updates the index after getting traffic data, and takes an appropriate index compression method according to the index structure. The indices store by rows or columns after indices compression, in order to reduce storage overhead, we can adopt the method of storage compression in the process of storage. When a query arrives, the query processor calls the current index to deal with the query and returns corresponding result.
2.2 Key technologies in ITAS

2.2.1 High performance packet/flow capture technologies

The general features of Internet data is huge amount, fast speed. In a 10Gbps link, for example, the network will reach 14 million packets per second as 64 bytes per packet. Even after polymerization, there are also millions of flow records. Though only deal with flow data, a central point of an operator would generate millions of records per second and the amount of data generated over 30 billion per day.

Therefore, how to capture and store high arriving speed of packets and streams in real time is a challenging problem.

2.2.2 High performance packet/flow storage technologies

Traditional packets and flow records use the relational database to store records by rows, which consumes a huge storage space and has slow retrieve speed. Currently, in order to efficiently store the packet/flow information, storing in columns and compression is a way to save the storage space overhead. It is normal to adopt LZO compression method, or use new methods based on the optimization of the flow of information, such as RasterZip, to deal with the network information.

2.2.3 High performance index technologies

We need to create an efficient index for such great data. Inverted index which is a general data retrieval technology is widely used in text retrieval, such as search engines. Inverted index is not as efficient as bitmap index to deal with the network flow queries, especially value retrieval.

Create huge amount indices for such great data retrieval, which is called the “index space” explosive issue. Big companies such as google have paid attention to the compression of the index space [41-42]. In order to index efficiently, the general index space will be fragmented (shard or segmentation), and compress the index file at the same time. Bitmap compression encoding algorithm is an effective method for the bitmap index to solve the index space explosion issue.

2.3 Categories of ITAS

We compare and summarize the features of Internet traffic archiving system from data format, data index structure and storage, which is shown in Table 3.

Raw data storage can be divided into 2 types, by rows or by columns. Bitmap index usually stores by columns, as NET-FLi use LZO compression algorithm to reduce the storage size.

All in all, traffic archiving contains packet capture, storage, index and query. Different traffic archiving system focus different aspects, therefore, the paper makes the packet capture and index...
compression algorithm as a starting point to understand traffic archiving system.

### Table 3. The comparison of different traffic archiving systems

<table>
<thead>
<tr>
<th>Data format</th>
<th>Indexing</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet</td>
<td>Flow record</td>
<td>bitmap</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hash</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tree</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Row</td>
</tr>
<tr>
<td>Telegraph CQ</td>
<td>✓</td>
<td>PostgreSQL(DBS)</td>
</tr>
<tr>
<td>Gigascope</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>MIND</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hyperion</td>
<td>✓</td>
<td>BloomFilter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multi-level</td>
</tr>
<tr>
<td></td>
<td></td>
<td>indexing</td>
</tr>
<tr>
<td>FastBit+TelegraphCQ</td>
<td>✓</td>
<td>WAH, PostgreSQL(DBS)</td>
</tr>
<tr>
<td>Time Machine</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>NET-FLi</td>
<td>✓</td>
<td>COMPAX</td>
</tr>
<tr>
<td>NetStore</td>
<td>✓</td>
<td>Segmenta</td>
</tr>
<tr>
<td>RASTERZip</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>PcapIndex</td>
<td>✓</td>
<td>COPMAX</td>
</tr>
<tr>
<td>TiFaFlow</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

### 2.4 The advancement of Internet traffic archiving systems

The development of Internet brings a surge in network traffic which contains amount of information, especially for known or unknown attacks in network security. Many attacks cannot be detected and prevented timely which need capture packet to realize the data collection for future use and analysis. Network monitoring is generally divided into two kinds, one is real-time monitoring, which implement real-time capture and discard the packet header and payload after checking, such as TelegraphCQ [5], Gigascope [6] etc.; another one not only can capture packets with line speed, but also can store, query, efficient retrieval and other features, such as NET-FLi[7], NetStore[8], Hyperion [9] etc. The systems above mentioned mostly input in the flow record format, which compared to the packet form simplify the operation and protect personal privacy.

![Figure 4. The advancement of the traffic archiving systems](image)
Using the conventional relational database in storage and indexing part in the early stage, but for the detection systems which need to accept new data in high speed connections, conventional database can’t meet this requirement. Therefore, a lot of traffic archiving systems creates indices on conventional file systems such as Hyperion [9] which use multi-level indexing architecture to achieve quick search and with the rising of bitmap indexing technology, more and more traffic archiving systems use bitmap index. Figure 4 shows the development of traffic archiving system.

### 2.4.1 TelegraphCQ (flow record)

TelegraphCQ[5] is a data flow system for processing continuous queries based on a novel, highly adaptive architecture. TelegraphCQ can filter, sort and aggregate flow records according to continuous queries and generates reports regularly. TelegraphCQ accepts continuous queries in the CQL query language which can realize window join and merge. TelegraphCQ reuses some portions of the PostgreSQL codes which retain some extended features such as supporting user-defined types, user-defined aggregates, storage management and application APIs.

![TelegraphCQ Architecture](image)

**Figure 5.** TelegraphCQ architecture

The challenges of implementing the TelegraphCQ in conventional database code are Shared processed and adaptivity which are key technologies to carry out the continuous queries. Figure 5 shows an overview of the TelegraphCQ. Users establish a connection with TelegraphCQ Front End (FE) and queries are sent to TelegraphCQ Back End (BE) which load them dynamically to running query and return the results to FE. User-defined wrappers are in TelegraphCQ wrapper ClearingHouse process which sends to BE through a shared buffer pool.

### 2.4.2 Gigascope (flow record)

Due to the network decentralization, network protocols and services become increasingly complex and network attacks are common which make network difficult to manage and analysis. Therefore, the Gigascope designed by Chuck Cranor et al. [6] provides a structured querying environment to simplify analysis and can be applied in many occasions: traffic analysis, intrusion
detection, router analysis, performance monitoring and debugging. Gigascope is a stream database which uses the GSQL (SQL stream database language) to support selection, join, aggregation and stream merge, and to deal with real-time data requests. Stream manager is the central part of Gigascope which tracks each query node. The GSQL processor is a code generator that translates the GSQL query into C or C++ code. Gigascope can process 1.2 million packets per second using an inexpensive dual 2.4 Ghz CPU server can continuously run for 3 months without stop.

2.4.3 MIND (flow record)

Detection and analysis of collaborative attacks require monitoring framework for distributed query system, which needs distributed monitors and distributed query system. MIND (multi-dimensional indices for network diagnosis) which designed by Xin Li et al. [10] is a distributed index system to support range query that contains a series of nodes forms a hypercube overlay. The nodes are logically distributed in monitors and can create a dynamic multi-dimensional directory which uses a simple SQL statement to deal with queries, and utilize multi copies to achieve robustness.

2.4.4 Hyperion (packet)

It must be able to accept new data in a high speed connection to the detection system, as a single connection has Gbit speed per second that can generate millions of packet header and MB data, and each detection system needs to record amount of such connections. Therefore, conventional database cannot meet the requirements.

However, the GigaScope[6] and MIND[10] use SQL interface to deal with the network data queries, while MIND is a distributed index system which can only extract and record a single flow information rather than the raw packet header information, and GigaScope is a stream database without considering the data storage.

P. Desnoyers et al. propose a system for archiving, indexing and online retrieval of high-volume data stream. It mainly contains a stream file system that adopts log structured write allocation of LFS and distributed multi-level Signature Indices to achieve storing millions of packets based on general hardware. The speed of indexing and storage can reach up to 200000 packets per second when processing real-time data requests.

The block length is fixed in Hyperion stream file system and the block contains block header and multiple records. The block header identifies the stream to which the block belongs and separates the record from last block. Each record length is variable and the records in the same block belong to the same stream. The record header contains stream ID, length, and a hash value for security considerations. Every N blocks form a zone while Nth is used as a block map to establish a mapping between the stream ID and blocks. Stream FS looks like a general dictionary that contains all of stream information such as each stream head or tail pointers and size.

Hyperion divides each stream into several intervals and computes one or more signatures to each interval. Therefore, the interval indexing is maintained rather than stream indexing while the contents of storage are signatures. The data will be recovered only when the signature exactly
In order to determine trends, patterns and anomalies, applications for data streams query need to compare with live stream data and historical data. But the overhead of searching the historical data in real time is too expensive while the main bottleneck is the indices update costs. Reiss F[11] et al. use bitmap indexing technique to reduce the update costs, and in order to implement real-time query use the TelegraphCQ[5] stream query processor which can be efficient to get the exact query results. Similarly, it can expand the query capability through parallelism and approximation.

![Diagram of the prototype system combining FastBit and TelegraphCQ](image)

Figure 6. The prototype system combines FastBit and TelegraphCQ

Ingress Manager is used to merge flow records and convert the data into formats that TelegraphCQ and FastBit support. The Controller is used to receive query results from TelegraphCQ, and generate reports for users. The Controller performs the analysis which mainly contains three parts: a TelegraphCQ query template, a FastBit query template and application logic.

### 2.4.6 Time Machine (packet)

TimeMachine (TM) system is proposed by Gregor Maier and Robin Sommer [12] which is a typical open source traffic archiving system. Through the analysis of the network traffic which can get the actual distribution of network streams size, and get further information about high volume data always after the control instructions which means the front part of a large stream is more likely to be control instructions while the back part is probably just raw data. Therefore, storing only a small fraction of the large stream is to preserve the original information of the stream in efforts with the cost of abandoning the data information, and can greatly save storage space as the system can deal with the 10-100 times than before.
Figure 7. Architecture of TimeMachine

Figure 7 shows the overall of the TimeMachine system which consists of user interface, capture and classification, flow table maintenance, storage classes and indexing 5 parts. Users interact with system through user interface and set the listening NIC, filter rules and classification rules to implement the queries. The system captures the network packet by network interface (NIC) and the packet is assigned to the corresponding storage class after classification. The storage class maintains a buffer queue in memory which exports data to disk when the queue is full. The main functions of flow tables are to accelerate the classification and realize storing the small fraction of the large stream. Indexing gets the packet information from the storage classes and establishes the relevant indices which provide to query module for using.

2.4.7 NET-FLi(flow record)

NET-FLi[7] (NETwork Flow Index) achieves high-speed storage, indexing and query based on the dynamic bitmap compression and indexing. It can process 50,000 -100,000 insertions per second on current commodity hardware and provide interactive query response times that enable administrators to perform complex analysis tasks.

The NET-FLI system consists of 2 logical parts, archiving backend and the compressed index. Each flow record uses LZO[13] compressor while the index uses the COMPAX bitmap compression method. In addition, there is an online preprocessing step by using the technique of oLSH which is optional operation. Although it reduces the rate of flow records processing in some way, it can get better compression rate and faster response time. The implementation of system also contains the query processor which can decompress the data when gives a search query and the index structure by matching a given historical flow record.
2.4.8 NetStore(flow record)

Paul Giura and Nasir Memon present NetStore which is based on Internet traffic semantics. It retained the advantage of column based storage. In the process of traffic recording, the tuple is increasing all the time. In the default situation, data values are not classified. So that when using column based storage concept, the compression result and retrieve result can be improved. But in the reordering process, the system cost cannot be ignored. In NetStore, it uses anchor column to decrease the system cost and achieves better result.

2.4.9 RasterZip(flow record)

Francesco Fusco and Michail Vlachos presented RasterZip[14] which is a compressor that mainly focus on the net traffic archiving. It can be used in data block compression of homogeneous network by using common prefix and other related information. The new solution is RLE encoding and partial-decompression. It accelerates highly selective queries targeting a small portion of the dataset. And it can achieve high-speed on-the-fly compression of more than half a million traffic records per second. This solution improves the state-of-the-art both in terms of compression ratios and query response times without introducing penalty in any other performance metric.

2.4.10 PcapIndex(packet)

Francesco Fusco and Xenofontas Dimitropoulos et al [15] develop the packet capture on Libpcap and expand tracking analysis to support the rapid filtration using compressed bitmap index. PcapIndex that they designed is the first indexing scheme based on compressed bitmap index and is backward to compatible with Libpcap. It can significantly reduce the query response time and disk overhead through using PcapIndex.

PcapIndex consists of two components, packet indexing which can capture series of packets from disk or interface for encoding and processing, and query processor which can merge relevant data by using efficient index filter, BPF filter and Pcap tracker to generate a fast and flexible query workflow. Through evaluating the performance of 3 bitmap index compression encoding schemes: WAH, PLWAH, and COMPAX, COMPAX is selected to be default encoding scheme which has the minimal processing overhead.

2.4.11 TiFaflow(packet)

TiFa Internet traffic archiving system [16] combines TimeMachine and FastBit which uses TimeMachine[12] to capture and record the flow information and uses FastBit[17] to index and query the flow record.

In order to facilitate the storage and query, TiFaflow network traffic archiving system [18-19] implements the storage and query of flow records based on Tifa system. TiFaflow contains a lot of
functions, such as flow recording, flow indexing, storage and query. The specific process is shown in Figure 8.

In the process of network data, network security events such as phishing websites, need network forensic. Tifa and TiFaflow systems can be used as a module of UTM or cooperate with UTM to realize network forensic.

![Figure 8. The process of TiFaflow system](image)

TiFaflow-ng[20] is the next generation of network traffic archiving system which can enhance the encoding ability of Fastbitbitmap index and choose a variety of bitmap encoding methods to save storage space and accelerate the speed of query.

3. Key techniques in ITAS

3.1 Packet Capturing

3.1.1 Libpcap

At present, there are two ways to capture packets from the network, one is to use customized hardware which has better performance, but the price is expensive and poor scalability, another is to use software which has worse performance than dedicated hardware, but the cost is low and better scalability. Many of these tools are developed on Libpcap which can realize the capture and analysis of stream data.

Libpcap is an open source library for packet capture and analysis, and an API interface function library on independent platform at the same time which can access to data in link layer independently and make program easy to compile and transplant for packet capture in user level. Except for capturing data, it can also use BPF filter scheme which can filter out uninteresting network packets, and store packets in a particular file format that let read information of the packets at any time.
3.1.2 NAPI

When receiving an Ethernet frame, device driver produces an interrupt event for dealing with the packet timely. But in the case of high traffic load, CPU will spend a lot of time on interrupt handling that can result in wasting CPU resources. L. Rizzo [21] therefore propose device polling to solve the livelock problem of interrupt. Device polling is deployed in FreeBSD version 4.5 while Linux introduces NAPI (New API).

NAPI combines interruption and polling to instead interrupt handling model. Due to less times of interrupt handling, the load of CPU reduces greatly, and makes devices fairer by means of polling devices.

NAPI still uses the interrupt notification system when a device has just received a new frame. NAPI registers the device into polling queue and disables interruption. Poll the queue with registered devices and read data from packets. It adopts the method of quota to ensure fair scheduling for each device. If the quota is not enough to handle the buffer data, the device will be moved to the end of the polling queue, if the quota is enough for dealing with the buffer, the device is cancelled from polling queue and open response of the interruption. The main process of NAPI is shown in figure 9.

![Diagram of NAPI process](image.png)

Figure 9. The main process of NAPI

Although MAPI can reduce the CPU resource waste, only a few devices employment NAPI which has some serious drawbacks, such as closing interruption make system cannot handle new coming packets timely, and the cumulative packets will cost a lot with the increase of transmission speed, etc. Therefore NAPI only adapts to the circumstance of high processing speed and short packets.

3.1.3 PF-RING

CPU spends a lot of time on sending the packets captured from NIC to user space through kernel functions when using Libpcap to capture streams. The whole process is from NIC to kernel and from kernel to user space which greatly reduces the CPU processing efficiency, although using
Luca Deri [22] proposed PF_RING technology that stores the packets received from NIC in a ring buffer. The ring buffer has two interfaces, one for NIC to write packets, another for the application layer to provide interface to read packets. The interface of reading packets can be implemented through mmap which is mapping the kernel memory and user space memory and the return value is the head address pA of user space memory. In this way, we can call the recvfrom() to send the captured packets from NIC to pA that may reduce copies and other operations which reduce the CPU processing time and improve the efficiency of the network packets capturing.

![Diagram of PF-RING technology](attachment:image.png)

**Figure 10. Explanation of PF-RING technology**

### 3.1.4 Netmap(fast packet I/O)

R. Lizzoet et al [23] raise three aspects of huge costs in packet processing: per-packet dynamic memory allocations, overheads of system call, and memory copies. In view of these three aspects, R. Lizzoet al design netmap framework that is closely linked with existing OSes and does not need special hardware support. Netmap is also simple to use and easy to maintain. The experimental data verifies that netmap framework is at least one order of magnitude faster than the general standard interface while it does not improve performance in price of security and ensures the kernel will never collapse. Netmap framework uses pre-allocating resources to remove the per-packet dynamic memory allocations, large batches to share system call overheads, and sharing buffers and metadata between kernel and user space, similar to RF-RING, to avoid memory copies.

### 3.1.5 Scap(Stream capture library)

Stream capture library [24] (referred to as Scap) based on a kernel module, can trace and reassemble TCP stream directly, and support itself in the multi-core parallel processing
architecture. Scap delivers flow level statistics to user space and utilizes subzero-copy to minimize copies and operations. In order to improve the performance, it discards uninterested packets in the early stages. The capture rate of Scap is twice than other flow reassembly library, and when the eight cores used for pattern matching application to do stream processing, it can handle 5 times traffic load than before.

3.2 Key techniques of bitmap index encoding

Efficient indexing of network packet or flow is another core technology of network archive. The purpose of traffic records is for later retrieval and search, and the network events can be identified with traffic records.

The information of Index has the following features:

1. Massive data: The number of index messages is massive and number of index messages can be produced for several million or even billions in one day.
2. High speed for data processing: In a 10Gbps link for example, if the package is 64 bytes, the peak of speed will reach 14 million package per second, which requires high speed for data processing to create the index.
3. Fixed data structure: The index information for each network packet has a fixed format and a fixed length.
4. Only increase without change: Network packet index information will increase only. When the information generates, it can’t be made a change.
5. High repeatability: There are millions of duplication of data in a domain.

Due to the characteristics of data like this, the efficiency of using a relational database to process the network flow data is low. The traditional relational database-oriented changes, and the data stored in the database requires frequent changes. The traditional database usually use B or B + tree, which is not applicable for index on network flow information, and it needs additional high-speed and efficient indexing technology.

Currently, a common technology of large data retrieval is inverted index, which is widely used in the retrieval system such as search engines. The core data structure of Inverted index is Posting List. Sequence of integers of a KEY stored in inverted lists, such as a timestamp and offset, and the most typical KEY is the position list, which shows the position where a word appears. Due to the characteristics of network flow data, the net flow archiving system use Bitmap Indexing widely instead of Inverted Index.

Table 4. An example of bitmap index

<table>
<thead>
<tr>
<th>RowID</th>
<th>Column</th>
<th>Bitmap index</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>=1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>
3.2.1 Brief introduction of Bitmap Index

Bitmap indexing [25-29] was proposed by P’ONeil in 1987, and deployed in a commercial database system called Model 204 for the first time [30]. Bitmap indexing use a bit vector or a sequence of bits to indicate the value of the index whether exists in the indexed data. Bitmap indexing can efficiently use bit logical operations (AND / OR / NOT / XOR, etc.) to answer the complex queries.

Bitmap index designed for scientific data and database, the scientific data is usually generated by scientific instruments or scientific simulation, and it is characterized by an extremely large amount of data without change. Bitmap index database solve the problem how to quickly identify a small amount of data in a mass of scientific data, while the traditional relational database is not suitable for this work.

The technology used in Bitmap index database are bitmap indexing, bitmap compression and classification. In Bitmap index database, the data is columnar storage, the data is stored in a column and create a corresponding bitmap index. A bitmap index example is shown in Table 4.

Currently, the representative bitmap index compression algorithm are BBC [31], WAH [32-33], PLWAH [34], EWAH [35], CONCISE [36], PWAH [37] and COMPAX [7], UCB [45], VLC [46], VAL-WAH [47], RoaringBitmap [48] etc. PLWAH has improved version (PLWAH with adaptive counter), COMPAX also has subsequent improved version (COMPAX + oLSH and COMPAX2). Figure 11 shows the various bitmap index coding algorithm appear in chronological order.

![Figure 11. The advancement of bitmap index compression encoding algorithms](image)

4. Conclusion and future works

We introduce and analysis the traffic archiving systems and the key technologies. First, we make an introduction about classical traffic archiving systems in recent years; then we compare the data format, indexing and storage to analysis different traffic archiving systems, and introduce
the implementation of TiFaflow; finally, the key technologies of packet (or flow) capture are declared, and we illustrate the advantages and disadvantages of each technologies, and scenarios for different technologies.

With the rapid growth of Internet traffic, traffic archiving system and the research of bitmap index compression schemes will face new challenges and risks which need to design a high speed and more efficient indexing technologies. First of all is the performance demands, traffic archiving systems that can support PB traffic will be a hotspot which may meet the requirements of the future network backbones and datacenter network. Second is the demand of the technical support, more efficient bitmap index compression schemes will have important research value, which will provide the powerful technical support for high performance network traffic archiving systems.

Acknowledgement

We are grateful to the teachers and students both in NSLAB and QoSlab, especially thanks to yuhao wen, wenxunzheng, guodong peng, and yinjunwu.

References

techniques for wide-area network monitoring, 21st International Conference on Data Engineering Workshops, pp. 1184-1184, April 2005.


[20] Zhen Chen et al., TiFaflow-ng: a next generation of TiFa flow traffic archival system, 2014.


[42] Jeff Dean, Challenges in Building Large-Scale Information Retrieval Systems, Web Search and Data Mining Conference (WSDM) keynote, February 2009.