A Network Monitoring System with a Peer-to-Peer Architecture

Paulo Salvador, Rui Valadas

University of Aveiro / Institute of Telecommunications Aveiro
E-mail: salvador@av.it.pt; rv@det.ua.pt

Abstract

The ever growing complexity of modern data networks requires versatile and scalable network monitoring architectures. In this paper we propose a network monitoring system with a peer-to-peer (P2P) architecture, allowing for high tolerance to failures and distributed storage of measured data. We describe the main features of the architecture, namely the system elements and its hierarchical organization, the protocols for handshaking, promoting and demotion of system elements, and distributing control information, the algorithm for system startup, addition of new elements and failure recovery, and the procedures for storing, replicating, searching and downloading measurement data. The proposed architecture is shown to be flexible in adapting to various network conditions and available resources.

Keywords: *Traffic monitoring, peer-to-peer, distributed control, distributed storage, distributed access, probing.*

1 Introduction

The complexity of data networks is growing very fast due to the diversity of technologies, applications and user behaviors. In such environments, a complete and updated knowledge of the network status is of fundamental importance for network administrators. Traffic monitoring systems provide administrators with a tool to detect and respond to network events or behaviors that can have a significant impact on the network performance.

Existing traffic monitoring systems either rely on a single probe [1, 2, 3, 4] or in a centralized architecture where a set of probes are controlled by a single manager [5, 6, 7, 8, 9]. The single probe system only monitors the traffic at one location and, therefore, is not flexible enough for medium and large size networks. The centralized architecture is able to provide an accurate view of the network status. However, it relies on a single manager, which makes it vulnerable to failures. Moreover, it stores measured data at a single collector, which may consume significant bandwidth when downloading data from probes.

In this paper we propose a versatile, scalable and easily manageable traffic monitoring system based on a peer-to-peer (P2P) hierarchical architecture. The adoption of a P2P architecture allows high tolerance to failures and distributed storage of measured data. This architecture is also advantageous for traffic monitoring in wide area network environments. Moreover, access and querying of measured data can be performed using traditional P2P file sharing schemes.

The system consists of two main entities: the probe and the client. The probe performs the measurements and stores the results. The client is the interface between the monitoring system and the user. It is used to configure the system and to retrieve the measured data. A client can connect to any probe and use this connection to configure measurements and retrieve measured data on any other probe. Thus access to the monitoring system is completely distributed. The addition of a new probe to the monitoring system is transparent: the system automatically detects and integrates a new probe.

To improve the scalability of the system, the probes are organized in groups, and each group is responsible for monitoring a particular network area. Within a group, one or more probes, called super-probes, are responsible for controlling other probes and communicating with other groups.

There are two types of measured data files: light data and heavy data files. The light files store the monitoring system parameters and summary statistics of the traffic, and are broadcasted to all probes. They provide a coarse, but updated and fully available, view of the network status. The heavy files store the results of all scheduled measurements, and can include detailed packet or flow information, and statistics of packet delays and losses measured over a period of time, obtained through active [5, 10] or passive [6] monitoring techniques. These files are stored at the probe that created them and are possibly replicated at other probes. The replication improves the system reliability, since data can be retrieved even if the probe that made the measurements becomes inactive or inaccessible. The search and retrieval methodologies for measured data stored at the probes are similar to the ones used in P2P file

sharing applications.

Peer-to-peer based measurement systems have been proposed by Srinivasan and Zegura [11] and by Liu *et al.* [12]. Both systems have rudimentary storage and search capabilities, which makes it difficult to handle large data files and restricts the type of measurements that can be carried out. Moreover, the architectures are not hierarchical and, therefore, do not scale well with the number of measuring probes.

The paper is organized as follows. Section 2 gives a detailed view of the proposed system architecture, namely the system elements and its hierarchical organization, the protocols for handshaking, promoting probes, demoting super-probes and distributing control information, the algorithm for system startup, addition of new elements and failure recovery, and the procedures for storing, replicating, searching and downloading measurement data. Finally, in Section 3 we present our conclusions.

2 System architecture

2.1 System elements and hierarchy

Given that the monitoring elements may have different computational resources (e.g. processing capabilities, storage space or network connections) and the availability of those resources may vary drastically over time, we propose an hierarchical architecture for the monitoring system similar to Gnutella 0.6 [13, 14]. In the first versions of Gnutella the nodes were considered identical and connected to each other randomly. In order to accommodate elements with distinct characteristics and increase the network scalability, version 0.6 of Gnutella introduced a more structured network where the elements can operate in ultrapeer or leaf mode [14]. A leaf establishes only connections to ultrapeers which work as gateways to the Gnutella network.

In the proposed monitoring system, probes in the same network area are associated in groups. A probe can also operate in super-probe mode. A probe performs measurement operations and stores measurement results. When in super-probe mode is also responsible for managing the set of probes connected to it and to establish interconnections with the other super-probes. A monitoring element can alternate between the two modes in order to adjust to different network conditions and resource's availability. In Figure 1 we represent the hierarchical relationship between the system elements. The communication between groups is performed by the super-probes. A super-probe connects to all other super-probes in the monitoring system (Figure 2).

The other system element is the client which is the interface between the monitoring system and the user. The client is used to configure the system (e.g. schedule measurements, change the mode of a probe and update the default probe list) via the COMMAND message and to retrieve the measured data. A client can connect to any probe and use this connection to configure measurements and retrieve measured data on any network probe.

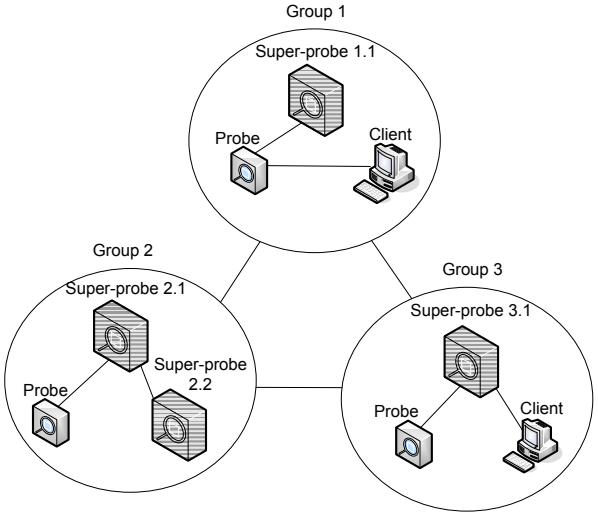

Figure 1. System hierarchy.

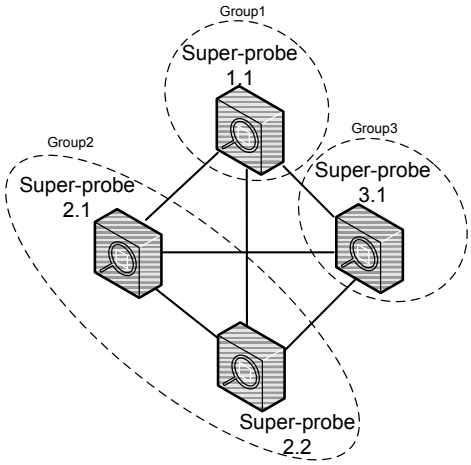

Figure 2. Groups connections.

2.2 Handshaking

The handshaking between the elements of the monitoring system (probe or super-probe) is performed by establishing a TCP connection and exchanging a set of control messages. The handshaking procedure differs from the one implemented by Gnutella. In Gnutella networks, the handshaking is performed initially with two ASCII messages (GNUTELLA CONNECT and GNUTELLA OK) to assure compatibility between different versions, followed by binary control messages (PING and PONG). We note that

in order to establish a connection over a reliable channel such as the one provided by TCP only two messages are required. Thus, in the proposed system we suppress the ASCII messages in order to simplify the process and reduce the number of exchanged messages. The handshaking is performed using only an adaptation of the Gnutella binary messages PING and PONG.

A monitoring element that initializes the handshaking with another element, sends a PING message requesting a connection, over the TCP channel previously established, which differs from the Gnuttela PING. It includes a field indicating the mode (probe or super-probe) of the element in question. The response to a PING is a PONG message, which also differs from the Gnuttela PONG. It includes the mode of the element in question, a flag indicating the acceptance or refusal of the connection request, the address of another super-probe (if known) and, in case of an accepted connection, includes the light data file. The usage of this messages for system startup, probe addition or failure recovery will be detailed in subsection 2.5.

A client can connect to any probe (or super-probe) using the same handshaking process used between probes.

2.3 Probe promotion and super-probe demotion

A super-probe can promote automatically a probe to super-probe if, according to a predetermined criterion, it is overloaded. To determine the overload condition, the probe takes into account the mean CPU usage, the mean memory usage, the free storage space, the number of connect probes and mean available bandwidth at the network interface. When a super-probe becomes overloaded, it tries first to distribute its probes to other super-probes with available resources; if not possible, promotes one of its probes to the super-probe mode and assigns some of the probes to it. A super-probe can demote itself to probe mode, when it is the least loaded super-probe and perceives that there are enough available resources in other super-probes. To start the promotion and reassignment process the super-probe sends a PONG message to the future super-probe with the `promotion` flag activated. Then the super-probe sends to each of the probes to be reassigned, a PONG message with the `bye` flag activated and with the address of the new super-probe. It is now up to the reassigned probes to initiate the handshaking with the new super-probe. In Figure 3 we show an example of the probe promotion and reassignment process. Using this process the system always accepts new elements and distributes its load such that the number of super-probes is minimized but the system performance is not degraded.

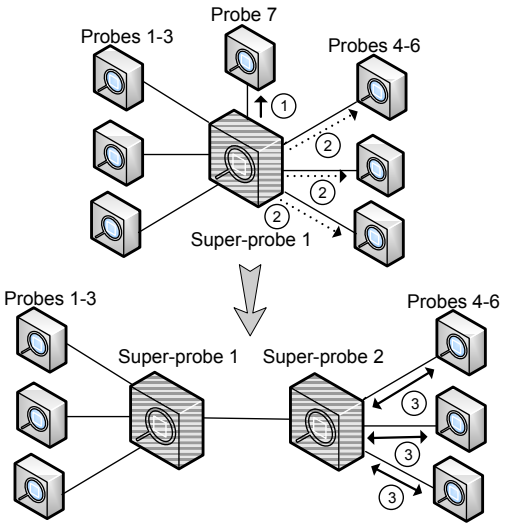

Figure 3. Example of probe promotion to super-probe, when probe 7 becomes super-probe 2. ① PONG message with `promotion` flag active, ② PONG message with `bye` flag active and with the address of super-probe 2 (probe 7) and ③ PING/PONG handshaking.

2.4 System information distribution

The monitoring system parameters and summary statistics of the measured traffic, which comprises the so-called light data file, are broadcasted to all probes within a group.

All PING messages sent by a probe (or super-probe) include the available resources of its sender. The available resources reported are the CPU clock, memory, mean CPU usage, mean memory usage, free storage space, mean available bandwidth at the network interface, and number of allowed probe connections (in the case of super-probes).

The light data file (in binary format) is broadcasted using the data field of PONG messages. The informations included in a light data file are: the addresses of all super-probes and their group IDs, the addresses of the probes within the group to which the light data file belongs and its group ID, the available resources in each probe or super-probe within the group and the summary traffic statistics measured within the group. The summary statistics of the traffic include the average probe to probe delays and average throughput at each probe interface.

The light data file is identified by an unique digital fingerprint (SHA1 hash value). The SHA1 (Secure Hash Algorithm) [15] is also used in Gnutella 0.6 networks to identify identical files [14].

2.5 System startup, probe addition and failure recovery

In Figure 4 we depict the flow diagram of the algorithm used to manage the monitoring system. This algorithm applies at system startup, when a new probe must be added to the system and after a system failure. The algorithm describes the necessary actions of an individual probe to enter an already established system or to initiate a system. In both cases the unconnected probe requires a list of potential super-probes. At startup this list must be provided by the system administrator in the so-called default light data file and when recovering from a system failure the probe should use the last light data file received.

A probe can be in super-probe mode when joining the network, if it was in that mode before a system failure or when set administratively. The joining element (probe or super-probe) starts by identifying the potential super-probes in its group and tries to establish a connection with the first one in the list. If the TCP connection was successful, the joining element sends a PING message; otherwise it tries another super-probe in the list. After sending a PING message, it will wait for a response; if no response is received within a predefined timeout it will retransmit the PING message a number of times. When the maximum number of attempts is reached, the joining element resets the TCP connection and tries the next probe in list. There are two possible responses to a PING: a PONG from a probe or a PONG from a super-probe. If the joining element receives a PONG from a probe, verifies if the message contains the address of a super-probe. If it does, it resets the TCP connection and tries to establish a connection with that super-probe; otherwise it will try the next probe in list. If the joining element is a probe and the PONG response is from a super-probe, it reads the light data file from the PONG message, sends to the super-probe the list of available heavy data files using the FILE message and starts to interact with the network. Otherwise, if the joining element is a super-probe and receives a PONG message from a super-probe, both elements decide whether or not they maintain themselves in the super-probe mode, using the information on available resources included in the PING and PONG message. The probes use a unique criterion known to all probes to make this decision. In case of demotion, the super-probe will first reassign its probes to other super-probes; it will send a PONG message to each of the probes that need to be reassigned with the `bye` flag activated and with the address of the new super-probe. When the joining element is a probe and has tried all address in its list without success, it promotes itself to super-probe and starts the procedure all over again. Also, when the joining element is a super-probe and has no more addresses to try, stops the joining process, waits connections from the probes of its group and starts establishing inter-group connections.

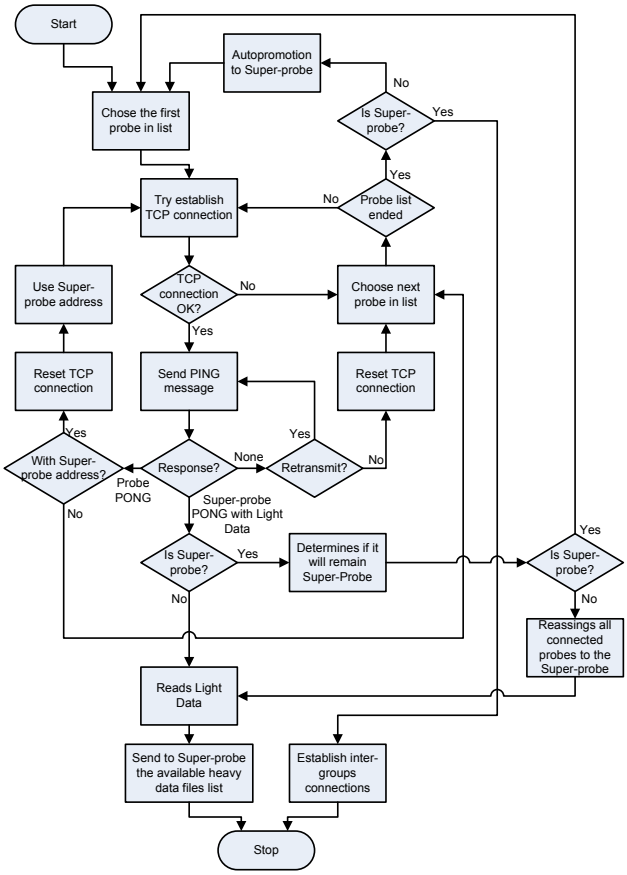

Figure 4. System startup and recovery.

2.6 Results storing and retrieval

2.6.1 Data storage and replication

The system uses a distributed data archive system. The probes store locally the results of their measurements in the heavy data files but also replicate these files in some other probes. This approach stands between two major archive philosophies: centralized and completely distributed. Replicating the data in some probes guarantees that data remains available even if the probe that made the measurements becomes inactive. Having multiple data sources also allows a faster and more reliable data retrieval using P2P file sharing techniques namely the Partial File Sharing Protocol [14].

The proposed system also implements the Gnutella 0.6 protocol meta-data approved extension [14, 16], where the data files are assigned a set of meta-data tags. The meta-data tags associated with the heavy data files allow the user to search for a particular measured data. The meta-data tags used in the heavy data files are: <NAME>,

<TYPE>, <DATE>, <ADDRESS1>, <ADDRESS2>, <M_PROBE> , <GROUP_ID>, <FILE_HASH> and <OBS>. The <NAME> tag defines the file name and the <TYPE> tag the type of measurement (i.e. passive measured delay, active measured losses, throughput, ...). The <DATE> tag records the measurement date. The <ADDRESS1> and <ADDRESS2> tags are used to identify the measurement location. For example, tag <ADDRESS1> can be used to identify the origin and tag <ADDRESS2> the destination in one-way delay measurements [10]; or <ADDRESS1> can be used to identify the interface address where a passive throughput measurement is carried out. The <M_PROBE> and <GROUP_ID> define respectively the address and the measurement group of the probe that performed the measurement. The <FILE_HASH> tag defines the file SHA1 hash. The <OBS> tag is reserved to any comment or additional information provided by the system administrator. A probe must notify its super-probe when a new heavy data file is created, changed or deleted. This process is carried out using the FILE message.

The data replication process is depicted in Figure 5. A probe, after the end of a particular measurement, generates an heavy data file that is stored locally and notifies its super-probe (step 1). The super-probe can then (immediately or not) send a REPLICATION message to the probe indicating one (or more) locations for storing the data (step 2). Afterwards, the probe opens a TCP connection to each location and transfers the heavy data file (step 3). The destination probe, as soon it receives the replica, signals the super-probe acknowledging that the new replica is available for download (step 4). The number of replicas must be related to the available storage space. In an extreme situation, when the system detects that it has no longer any storage space available in the super-probes and/or probes attached, it deletes some of the replicas assuring that at least one copy of each data file remains in the system. This process is controlled by the super-probe, which sends a FILE message to the probes indicating the delete order and the particular file to be deleted. The super-probe must start the replica elimination by the files that have more replicas. The system administrator must receive an e-mail alerting him of the event. A probe must signal the super-probe when its storage space becomes lower than a specified threshold using a PING message with the `out of space` flag active, to which the super-probe should respond with an order to delete some of the probe data replicas.

2.6.2 Results search

The heavy data files stored within the system can be searched using a P2P architecture approach, specifically the one used in Gnutella 0.6 [14]. A client connected to an element of a particular group can perform global and local searches. A global search is performed in all groups of the monitoring system while a local search is restricted to the group where the client is attached.

The client initiates a search by ending a QUERY message to its probe. The QUERY message will then be broadcasted to super-probes. In case of a global search the QUERY message is sent to all super-probes of the monitoring system. Each super-probe has a list of all heavy data files stored in its probes. When a super-probe receives a QUERY message with a file descriptor matching one (or more) of its heavy data files, it responds with a QUERYHIT message. The QUERYHIT message includes, for each heavy data file, the address of the probe where it is located and its meta-tags. In Figure 6 we give an example of a heavy data file search.

2.6.3 Data download

The data download is performed directly between the client and the probes where the files are stored using HTTP. This

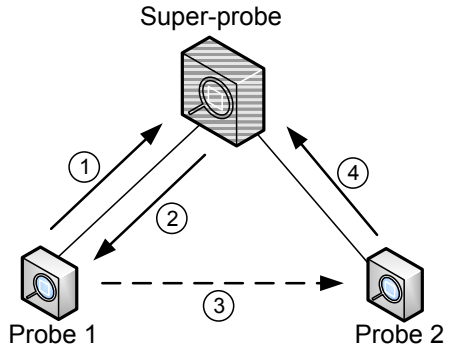

Figure 5. Heavy data replication process: ① new data available, ② locations to replica data, ③ data replication and ④ new replica available.

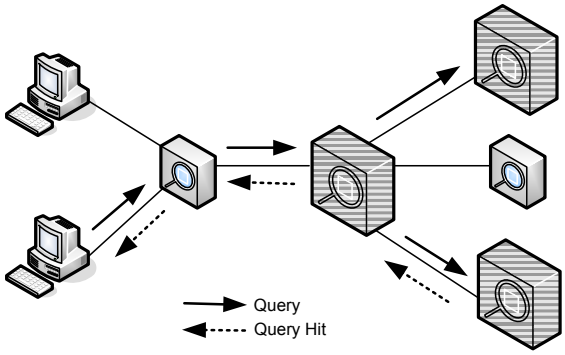

Figure 6. Data query.

download can be made from multiple sources resorting to the Partial File Sharing Protocol. The Partial File Sharing Protocol is used by the Gnutella v0.6 [14], BitTorrent [17] and EDonkey2000 [18]. This protocol prevents file corruption as data is transferred over the system. Each data segment has its own hash. When an individual segment has finished downloading, it is rehashed and checked against the reported hash value from the host. If the hashes differ, the downloaded segment is discarded and the client downloads again that segment. This protocol was chosen because it greatly enhances file sharing speed and availability.

In addition to the Partial File Sharing Protocol, the Gnutella 0.6 link compression extension [14] can be used. This methodology can reduce the amount of information transmitted and additional load in the network, but also increases the computational requirements in the probes. This feature is optional and should be activated only when there is available processing power at the probes.

2.7 System Messages Format

The system has a set of control messages: PING, PONG, QUERY, QUERYHIT, COMMAND, FILE and REPLICATION. All messages have a common header with 4 bytes . The first byte of the message header is used to define the type of message:

- 0x00 : PING
- 0x01 : PONG
- 0x80 : QUERY
- 0x81 : QUERYHIT
- 0x20 : COMMAND
- 0x30 : FILE
- 0x31 : REPLICATION

The first two bits of the second byte header (md) are used to code the type of the system element that is sending the message: 00 for probe, 01 for super-probe and 10 for client. The remaining 6 bits of the header second byte are used as control flags:

- `Connection accepted` : Used to acknowledge a connection acceptance
- `Bye` : Used by super-probes to disconnect probes (PONG message only)
- `Promotion` : Used by super-probes to promote probes (PONG message only)
- `Light data present` : Signals that the message contains light data (PONG message only)
- `Out of space`: Used by probes to signal the lack of available storage space
- `Super-probe address present` : Signals that the message contains a super-probe address (PONG message only)

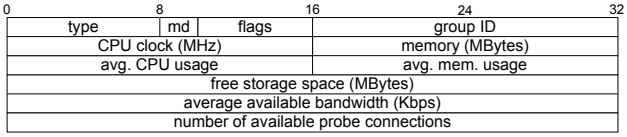

Figure 7. PING message format.

The last 2 bytes of the header are used to identify the measurement group of the message sender.

The PING message format is depicted in Figure 7. Additionally to the common header the PING message has 7 fields used to advertise the resources available at the sender.

The PONG message (Figure 8) has a field to broadcast the address of the super-probe. The SHA1 field is used to identify the specific light data file broadcasted. The remaining message fields are used to transmit the light data itself. If no light data is being broadcasted the data field length value should be zero.

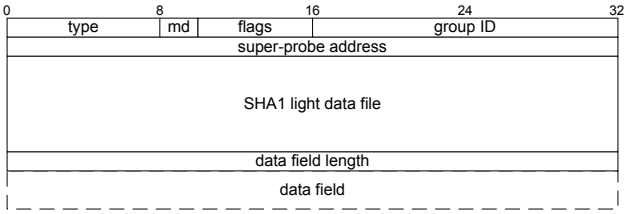

Figure 8. PONG message format.

The QUERY message has only two additional fields, the search block size and the search block, in which it is possible to transmit the querying parameters, specifically the meta-tag values (Figure 9).

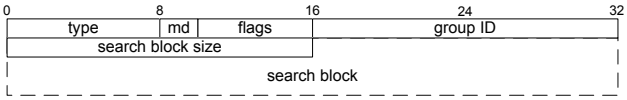

Figure 9. QUERY message format.

The first additional field of the QUERYHIT message (Figure 10) is the number of query hits. The QUERYHIT message contains as many result blocks as the number of hits. Each result block consists of the IP address of the file source, the size of the file meta-data descriptor and the file descriptor.

The COMMAND message (Figure 11) has only two fields, the command size and the command. The command is transmitted in ASCII format according to predetermined rules. The command should include the requested action and the action parameters.

The FILE message (Figure 12) includes a field indicating the number of file descriptors and the respective meta-

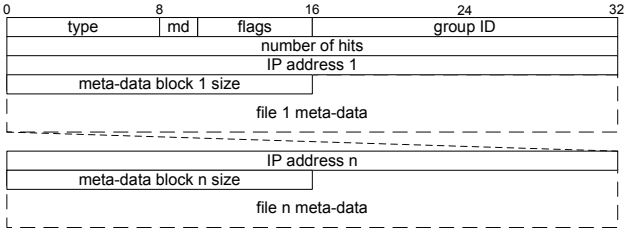

Figure 10. QUERYHIT message format.

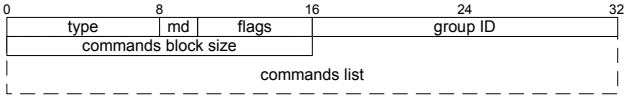

Figure 11. COMMAND message format.

data file descriptors. The FILE message control fields are used to determine the action to be performed with each file descriptor, and can have the following values:

- 0x00 : New
- 0x01 : Deleted
- 0x02 : Changed
- 0x10 : Delete

The New, Changed and Deleted values are used by probes to signal the super-probe of changes in its heavy data file archive. The Delete value is used by super-probes to signal the deletion of a particular heavy data file.

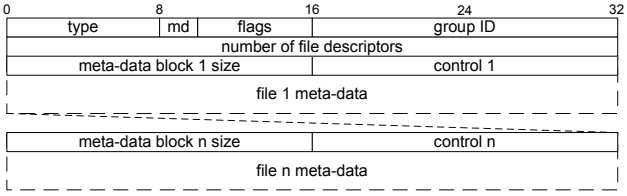

Figure 12. FILE message format.

The REPLICATION message (Figure 13) has a field with the meta-data that describes the file to be replicated, fields indicating the number of probes where the file should be replicated and their IP addresses, and a field (type of transference) indicating the protocol to be used in the file transfer (e.g. HTTP or FTP).

3 Conclusion

In this paper we proposed a network monitoring system with a peer-to-peer (P2P) architecture, allowing for high tolerance to failures and distributed storage of measured data. We have described the main features of the architecture, namely the system elements and its hierarchical organization, the protocols for handshaking and distributing

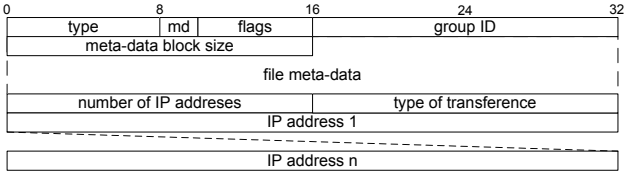

Figure 13. REPLICATION message format.

control information, the algorithm for system startup, addition of new elements and failure recovery, and the procedures for storing, replicating, searching and downloading measurement data. The proposed architecture was shown to be flexible in adapting to various network conditions and available resources.

Acknowledgments

This research was supported by Fundação para a Ciência e a Tecnologia, project POSI/EIA/60061/2004, and European Commission, Network of Excellence EuroNGI (Design and Engineering of the Next Generation Internet).

References

[1] "TCPdump," http://www.tcpdump.org/.

[2] "Ethereal - A Network Protocol Analyzer," http://www.ethereal.com/.

[3] "NTOP - Network TOP," http://www.ntop.org/.

[4] "MRTG - Multi Router Traffic Grapher," http://mrtg.hdl.com/.

[5] S. Shalunov and B. Teitelbaum, "RFC 3763 - One-way Active Measurement Protocol (OWAMP) Requirements," .

[6] J. Quittek, T. Zseby, B. Claise, and S. Zander, "RFC 3917 - Requirements for IP Flow Information Export (IPFIX)," .

[7] S. Waldbusser, "RFC 1757 - Remote Network Monitoring Management Information Base," .

[8] Inc. NetScout System, "RMON, RMON2, and Beyond," .

[9] "Network Monitoring Tools," http://www.slac.stanford.edu/xorg/nmtf/nmtf-tools.html/.

[10] H. Veiga, T. Pinho, J. L. Oliveira, R. Valadas, P. Salvador, and A. Nogueira, "Active traffic monitoring for heterogeneous environments," *Proceedings of 4th International Conference on Networking (ICN05), Reunion Island*, vol. 2005, April.

[11] S. Srinivasan and E. Zegura, "Network measurement as a cooperative enterprise," *Lecture Notes In Computer Science*, vol. 2429, pp. 166–177, March 2002.

[12] W. Liu, R. Boutaba, and J. W. Hong, "pMeasure: A tool for measuring the internet," in *Proceedings of the 2nd Workshop on End-to-End Monitoring Techniques and Services (E2EMON)*, October 2004.

[13] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-like P2P systems scalable," in *Proceedings of ACM SIGCOMM 2003*, April 2003.

[14] "Rfc-gnutella 0.6," http://rfc-gnutella.sourceforge.net/.

[15] "Secure hash standard," Federal Information Processing Standards Publication 180-1, April 1995.

[16] S. Thadani, "Meta information searches on the Gnutella network," LimeWire LLC, 2001.

[17] "BitTorrent," http://bittorrent.com/.

[18] "eDonkey2000," http://www.edonkey2000.com/.