Ntop: beyond Ping and Traceroute

Luca Deri[1] [2] and Stefano Suin[2]

The task of network management is becoming increasingly complex due to the increasing number of networked computers running different operating systems and speaking various network protocols. Most of network monitoring and diagnostic tools such as ping and traceroute are suitable just for tackling simple connectivity problems. Complex network problems often need to be addressed using rather expensive management tools or probes affordable only by mid-large companies.

This paper covers the design and the implementation of ntop, an open-source web-based network usage monitor that enables users to track relevant network activities including network utilisation, established connections, network protocol usage and traffic classification. ntop's portability across various platforms, its support of many network media, ease of use and lightweight CPU utilisation makes it suitable for people who want to monitor their network without having to adopt a sophisticated yet expensive management platform.

1 Background and Motivation

Popular tools such as ping and traceroute [16] have been used for years now for monitoring and debugging simple network connectivity issues. Although these tools are often sufficient for tackling simple problems, they have been created for monitoring network activities between two hosts. In cases where the network problem to address is due to the interaction of traffic originated by multiple hosts, these tools show their limits. Network sniffers such as tcpdump [9] or snoop are quite useful for analysing network traffic but off-line applications are often necessary for correlating captured data and identifying the network flows. Many commercial network sniffers are usually able to analyse data while capturing traffic but still these tools are quite primitive because they focus mainly on the packet and not on global network activities. In other words, operators are able to virtually know everything about the content of a single network packet whereas it is very difficult to extract information concerning the whole network status when a network problem appears.

Similarly, network probes such as RMON agents [17] are quite powerful but unfortunately need sophisticated SNMP managers that are able to configure them properly, and analyse collected network statistics. Due to this complexity and also the cost of such probes, RMON agents are basically used uniquely by advanced network managers in large institutions.

Other tools for network monitoring such as NeTraMet [4] and NFR [14] offer advanced programming languages for analyzing network flows and building statistical event records. Nevertheless those tools have been designed as instrumentable network daemons suitable for monitoring networks in a mid/long time period whereas in some cases it is necessary to have a very simple tool able to show the actual network status in human-readable format on a character-based terminal.

Even though operating systems have evolved rapidly, software companies did not pay enough attention to network management. Due to this, the latest releases of popular operating systems still offer no more than ping and traceroute. This is because companies often believe that if a network problem is due to network connectivity then ping and traceroute are enough, whereas if the problem is more complicated then a costly and complex network management tool has to be used.

The authors believe that this statement does not hold. In the Internet age, computer users need to have access to simple yet powerful network monitor tools able to give answer to questions such as:

- Why is the local network performance so poor?
- Who is using most of the available network bandwidth?
- Which are the hosts currently decreasing the performance of the local NFS server?
- What is the bandwidth percentage actually used of my computer?

[1] Finsiel S.p.A., Via Matteucci 34/b, 56124 Pisa. Email l.deri@finsiel.it.

[2] Centro Serra, University of Pisa, Lungarno Pacinotti 43, Pisa, Italy. Email {deri, stefano}@unipi.it.

- Which are the contacted peers and the amount of network traffic produced by each of the processes running on my local computer?
- Which are the hosts that produce multicast traffic?

ntop has been written to give a positive answer to all of the above questions. It has been initially written by the authors for tackling performance problems of the campus network backbone. Similar to the Unix *top* [3] tool that reports processes CPU usage, authors needed a simple tool able to report the network top users (hence the term ntop) for quickly identifying those hosts that were currently using most of the available network resources. ntop then evolved into a more flexible and powerful tool as people over the Internet downloaded it and reported problems and suggestions. The following sections cover architecture, the adopted design solutions and the inner details of the current ntop implementation.

2 Inside ntop

ntop is an open-source software (`http://www.opensource.org/`) [15] application written using the C language available free of charge under the GNU public licence. This statement does not just mean that ntop's source code is freely available on the Internet, but also that many requirements came directly from early ntop adopters. The authors designed the first version of ntop and then accommodated new requirements and extensions on the original architecture, strongly influenced by the *Webbin* [7] architecture. ntop's main design goals are:

- portability across Unix and non-Unix (e.g. Win32) platforms;
- simple and efficient application kernel with low resource (both memory and CPU) usage;
- minimal requirements (bare operating system) but capable of exploiting platform features, if present (e.g. kernel threads);
- ability to present data both in a character-based terminal and a web browser;
- the network analysis output should be rich in content and easy to read.

The ntop architecture is shown in the following figure.

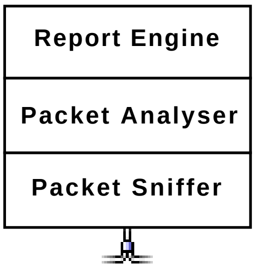


Figure 1 - ntop Architecture

The packet sniffer collects network packets that are then passed to the packet analyser for processing. Whenever traffic information has to be displayed, the report engine renders the requested information appropriately.

2.1 Packet Sniffer

The packet sniffer is the ntop component that potentially has more portability issues. In fact, unlike other facilities such as threads, there is not a portable library for packet capture. Under Unix the *libpcap* [12] library provides a portable and unified packet capture interface, whereas other operating systems provide proprietary capture facility. Due to good design of libpcap and its relatively portable interface, the authors decided to use it as unified capture interface and then wrapped platform-specific packet capture libraries (e.g. NDIS [13] on Win32) around pcap-like interface. This has the advantage that the ntop code is unique whereas the platform-specific code is limited only to a file. The packet sniffer supports different network interface types including PPP, Ethernet and Token

Ring and allows captured packets to be filtered before being processed by the analyser. Packet filtering is based on the BPF filter [11] facility part of libpcap. Filters are specified using simple expressions as those accepted by tcpdump.

Packet capture libraries have small internal buffers that prevent applications from being able to handle burst traffic. In order to overcome this problem hence reduce packet loss, ntop buffers captured packets. This allows the packet analyser to be decoupled by the packet sniffer and not to loose packets due to bursty traffic. It is worth remembering that ntop can operate on switched networks (e.g. an Ethernet network that makes use of switches) as well as on traditional networks. This is because modern switches allow global network traffic (or virtual LANs) to be mirrored to a specified switch port. ntop can then be activated on a host that is attached to such a port.

2.2 Packet Analyser

The packet analyser processes one packet at time. Packet headers are analysed according to the network interface being used. This is because headers are different depending on the network interface (e.g. the Token Ring header is different from the Ethernet one). Hosts information is stored in a large hash table whose key is the 48 bit hardware (MAC) address that guarantees its uniqueness and allow different network protocols other than IP to be handled (e.g. TCP/IP addresses are meaningless in non-IP networks). Each entry contains several counters that keep track of the data sent/received by the host, sorted according to the supported network protocols. For each packet, the hash entry corresponding to packet source and destination is retrieved or created if not yet present. Because it is not possible to predict the number of different hosts whose packets will be handled by ntop, it would be almost impossible to have a hash table large enough to accommodate all the possible hosts. When is necessary (e.g. periodically or if there are no entries left) ntop purges the host table in order to avoid exhausting all the available memory and creating huge tables that decrease the overall performance. Purged entries correspond to hosts that have not sent/received data for a long period of time. This guarantees that ntop's memory utilisation does not grow indefinitely and that packet processing time does not increase linearly with the number of active hosts. If the received packet is a non-IP packet, the protocol entry counters are updated and the packet discarded. Instead if the received packet is an IP packet, then further processing is performed.

Caching is performed in two steps. First level caching is semi-persistent and based upon GNU *gdbm*[18].

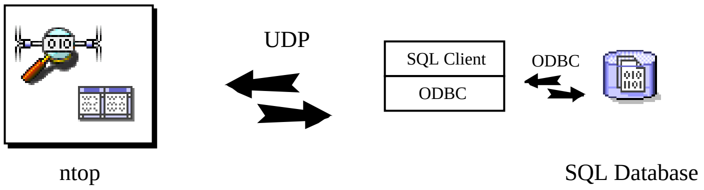


Fig. 2. ntop caching

Second level caching is implemented using a SQL database. ntop caches locally semi-persistent information such as IP address resolution (mapping numeric/symbolic IP address) and remote host operating system (computed using the *queso* [2] tool). Network events (e.g. TCP sessions), performance data and other relevant information is stored permanently into the database. Storage happens either periodically or whenever the garbage collector has to purge some data. ntop talks with the database by means of a client application. Such a client dialogues with ntop via UDP and communicates with the database using ODBC (Open DataBase Connectivity protocol). Whenever some network information has to be stored into the database, ntop sends the client one or more UDP packets containing valid SQL statements. The client, currently implemented in both Perl and Java, receives the packets and executes the statement on the local database via Perl DBI (DataBase Interface) or Java JDBC (Java DataBase Connectivity) depending on the implementation language. This architecture allows ntop to be decoupled from a specific database and able to communicate with remote database (e.g. the main company database) while having a very simple and light database client.

The host entry shown below contains a counter for each of the user-specified IP protocols.

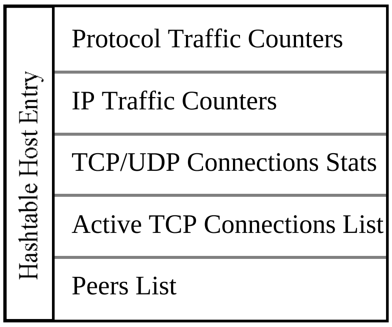

Hashtable Host Entry
Protocol Traffic Counters
IP Traffic Counters
TCP/UDP Connections Stats
Active TCP Connections List
Peers List

Fig. 3. Host Hashtable Entry

For each IP packet, the appropriate protocol counter is updated. If the packet is an IP fragment, ntop retrieves information such as source and destination port from the fragment hash table. Whenever the first packet fragment is encountered, fragment information is stored in the hash table using the packet `fragmentId` as hash key. Fragment information is removed as soon as the last fragment has been received. Because it might happen that some packets (including fragments) have been dropped, the fragment table is periodically analysed and outdated information is purged from it. The host entry also contains a list (initially empty) of the host's active TCP connections. ntop maintains the state of each TCP connection analysing the IP flags. Hence if the received packet is a TCP packet, then the host TCP connection list also needs to be updated.

Although host traffic counters can be profitably used to analyse network traffic, in some cases it might be necessary to study specific traffic that flows through some specified hosts. ntop allows users to specify network flows. A network flow is a stream of packets that matches a user-specified rule. Rules are specified using BPF expressions. Similar to NeTraMet flows, ntop network flows can be used for specifying traffic of particular interest. For instance a simple network flow could be the "total traffic NFS traffic between host A, B and C", whereas a more complex flow is "the total number of TCP connections rejected by the host D". Network flows can be very useful for debugging network problems, gathering statistical data or tracking suspicious access to some specified network resources.

2.3 Report Engine

The actual version of ntop can be started in two ways:

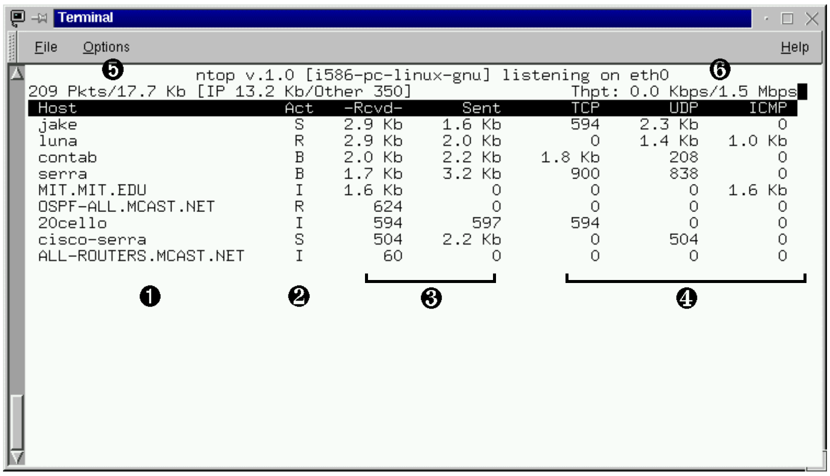
- interactive mode
  ntop runs in a character-based terminal and users can interact using keyboard keys.
- web-mode
  ntop acts as an HTTP server and allows remote users to analyse traffic statistics by means of a web browser.

ntop has been designed for being independent of the way traffic reports are created. The current report engine contains two emitters for both text-based terminals and HTML. Independence from the way reports are created is very important in order to guarantee application evolution. In fact if a new mark-up language such as XML has to be supported, only the report engine needs to be extended whereas the rest of the application remains unchanged. It is worth noting that custom reports and statistics can also be generated using data stored by ntop into the SQL database.

3 ntop at Work

3.1 Interactive Mode

When ntop is started in interactive mode, traffic information is shown in a character-based terminal window as shown below.



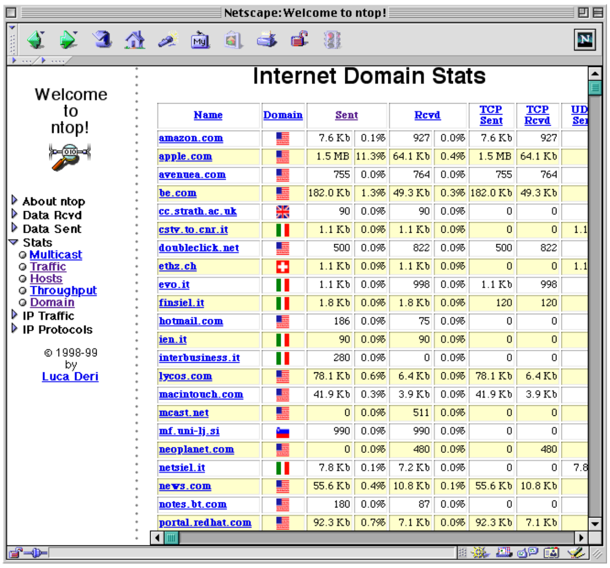
```
                   ntop v.1.0 [i586-pc-linux-gnu] listening on eth0
209 Pkts/17.7 Kb [IP 13.2 Kb/Other 350]            Thpt: 0.0 Kbps/1.5 Mbps
 Host                                Act   -Rcvd-      Sent      TCP     UDP     ICMP
 jake                                 S    2.9 Kb    1.6 Kb      594   2.3 Kb       0
 luna                                 R    2.9 Kb    2.0 Kb        0   1.4 Kb  1.0 Kb
 contab                               B    2.0 Kb    2.2 Kb   1.8 Kb      208       0
 serra                                B    1.7 Kb    3.2 Kb      900      838       0
 MIT.MIT.EDU                          I    1.6 Kb         0        0        0  1.6 Kb
 OSPF-ALL.MCAST.NET                   R       624         0        0        0       0
 20cello                              I       594       597      594        0       0
 cisco-serra                          S       504    2.2 Kb        0      504       0
 ALL-ROUTERS.MCAST.NET                I        60         0        0        0       0
```

Fig. 4. ntop: Interactive Mode

Column ❶ contains the list of hosts that have sent/received data, column ❷ specifies the actual host state (S=send, R=receive, B=send/receive and I=idle). Column ❸ contains the total data sent/received by each host, whereas column ❹ is a detailed view of the previous column. Users can change the sort order or the shown protocols simply by pressing the appropriate keys. The terminal is updated periodically as specified by the user. ❺ indicates the total observed traffic (packets and bytes) since the time ntop has been started, whereas the actual and maximum network throughput is shown in ❻.

3.2 Web Mode

The ntop interactive mode has been conceived as a quick network diagnostic tool for users who need to have a quick look at the actual network traffic (e.g. when the network is slow and it is necessary to find out which hosts are decreasing the overall performance). Instead, the web-mode turns ntop into a full fledged web-based management application [10] as shown in the following figure.



Internet Domain Stats

Name	Domain	Sent		Rcvd		TCP Sent	TCP Rcvd	UD Ser
amazon.com	🇺🇸	7.6 Kb	0.1%	927	0.0%	7.6 Kb	927	
apple.com	🇺🇸	1.5 MB	11.3%	64.1 Kb	0.4%	1.5 MB	64.1 Kb	
avenuea.com	🇺🇸	755	0.0%	764	0.0%	755	764	
be.com	🇺🇸	182.0 Kb	1.3%	49.3 Kb	0.3%	182.0 Kb	49.3 Kb	
cc.strath.ac.uk	🇬🇧	90	0.0%	90	0.0%	0	0	
cstv.to.cnr.it	🇮🇹	1.1 Kb	0.0%	1.1 Kb	0.0%	0	0	1.1
doubleclick.net	🇺🇸	500	0.0%	822	0.0%	500	822	
ethz.ch	🇨🇭	1.1 Kb	0.0%	1.1 Kb	0.0%	0	0	1.1
evo.it	🇮🇹	1.1 Kb	0.0%	998	0.0%	1.1 Kb	998	
finsiel.it	🇮🇹	1.8 Kb	0.0%	1.8 Kb	0.0%	120	120	
hotmail.com	🇺🇸	186	0.0%	75	0.0%	0	0	
ien.it	🇮🇹	90	0.0%	90	0.0%	0	0	
interbusiness.it	🇮🇹	280	0.0%	0	0.0%	0	0	
lycos.com	🇺🇸	78.1 Kb	0.6%	6.4 Kb	0.0%	78.1 Kb	6.4 Kb	
macintouch.com	🇺🇸	41.9 Kb	0.3%	3.9 Kb	0.0%	41.9 Kb	3.9 Kb	
mcast.net	🇺🇸	0	0.0%	511	0.0%	0	0	
mf.uni-lj.si	🇸🇮	990	0.0%	990	0.0%	0	0	
neoplanet.com	🇺🇸	0	0.0%	480	0.0%	0	480	
netsiel.it	🇮🇹	7.8 Kb	0.1%	7.2 Kb	0.0%	0	0	7.8
news.com	🇺🇸	55.6 Kb	0.4%	10.8 Kb	0.1%	55.6 Kb	10.8 Kb	
notes.bt.com	🇺🇸	180	0.0%	87	0.0%	0	0	
portal.redhat.com	🇺🇸	92.3 Kb	0.7%	7.1 Kb	0.0%	92.3 Kb	7.1 Kb	

Fig. 5. ntop: Web Mode

The web-mode has been designed as a long standing statistics gathering application able to provide users a detailed view of the current and past network activities. The web interface has been selected because it guarantees client independence and allows multiple users to be served. However, in order to prevent unauthorised users from accessing sensitive data as traffic information, ntop implements the standard HTTP password protection scheme. Administrators can specify at user level what information can be presented to remote users in order to avoid exposing sensitive information to potential hackers while giving the chance to show all the network statistics to selected people. Users connect their web browsers directly to ntop that acts as an HTTP server. The entry page is divided in two frames: the left frame is used for navigating through traffic information displayed in the right frame. Users can fully customise the layout and change the menu content/position as needed. All the relevant table columns are sortable simply by clicking on the column name. Whenever appropriate hyperlinks are used for correlating information. HTML pages are periodically refreshed automatically or on user request. Beside the information also shown in interactive mode, the web mode contains additional statistics including:

- IP multicast.
- Host information
  Data sent/received, contacted peers, active TCP sessions, TCP/UDP session history, provided/used IP services, bandwidth currently in use.
- Traffic Statistics
  Local (subnet) traffic, local vs. remote (outside specified/local subnets), remote vs. local, packet statistics (similar to RMON), network throughput (actual, peak, average).
- Currently active TCP sessions.
- IP/non-IP Protocol Distribution
  Distribution of the observed traffic according to both protocol and source/destination (local vs. remote).
- Local subnet traffic matrix.
- Network Flows
  Traffic statistics for each user-defined flow.
- Local network usage
  Detailed statistics about open sockets, data sent/received, and contacted peers for each process running on the host where ntop is active.

ntop makes use of a tool named *lsof* [1] for calculating the local network usage. lsof is used at start-up by ntop for getting the list of open IP ports for each of the running processes. ntop runs lsof periodically or whenever a remote host sends/receives data to/from a local port that was not active when lsof was last executed. Although the use of lsof is not very elegant, it is justified by the fact that there is no portable way to retrieve the list of open IP ports for each running process, and even if ntop would implement that functionality, ntop has to periodically poll the kernel because there is no way to be notified when a port is open/closed.

3.3 When to use ntop?

ntop can be profitably used to both monitor the network when some problems arise or just to analyse the overall network status including but not limited to:

- Protocol monitor
  Determine what protocols are used and identify those computers that speak unnecessary protocols. For instance, the Windows™ operating system install by default protocols such as NetBeui and IPX while most of the people use just TCP/IP.
- Network service usage
  Services such as DNS and NFS can be easily monitored. This allows network administrators to both analyse the impact of selected protocols on the overall network performance and identify those applications (e.g an FTP server) that have been silently installed in the network without authorisation.
- Network utilisation
  ntop is able to identify what computers are using most of the network resources, as well as graph network bandwidth usage over the time.
- Security

Portscan, denial of service and other security flaws are traced by ntop and once stored on the database can be used to identify those hosts that violated the overall network security.

In general, ntop combines features otherwise present in various tools not always easy to integrate. Its unique user interface allows administrators to immediately take advantage of ntop without the need to purchase and manage client applications that are necessary for tools such as RMON or NeTraMet. In addition, database support makes ntop suitable not only for network problem debugging but also for long standing network monitoring.

3.4 Performance Issues

ntop performance is quite good basically for five reasons:

- libpcap (or NDIS on Win32) performance is excellent;
- packet loss is very low (if any) because captured packets are buffered twice both inside the kernel and ntop;
- potentially long running actions (e.g. IP address resolution) are implemented asynchronously;
- ntop spawns several threads that prevent user interaction (e.g. HTTP user requests) from interfering with data collection;
- ntop makes extensive use of hash tables whose indexes are easy to compute yet fast during information retrieval due to the nature of network addresses (e.g. they are unique and already in 32/48 bit numeric format).

Users have tested ntop extensively on various network media running at different speeds. In general, ntop performance is greatly influenced by the other running processes because some CPU-greedy applications may take up the whole CPU cycles for a few seconds causing packet loss. Supposing to run ntop on an average loaded host, tests shown that ntop can work with very low (if any) packet loss on a 100 Mbit ethernet.

Nevertheless, performance is strongly influenced by per-packet processing. In fact the more network flows are defined, the more processing time is required hence the higher is the probability of dropping some packets. Due to the way ntop works, if a packet gets lost major problems may arise. In fact suppose to loose the first fragment of a TCP packet containing the FIN flag. In this case there are two problems:

- the fragment entry for the packet is not created, hence the following packets cannot be handled properly;
- ntop does not know that a peer intends to close the TCP connection (three way handshake).

In order to overcome the above mentioned problems, ntop implements internal timeouts and periodical garbage collection in order to purge old data and speculate about the state of active connections. For instance, if there is no data flowing on a connection for a very long period of time, then the connection might have been closed. In this case ntop assumes that the connection has been closed and then the connection entry is purged. This allows ntop to recover whenever some packets get lost and not to get stuck waiting for some lost packet to arrive.

4 Lessons Learned

ntop has been a great exercise in many respects:

- ntop performance
  It is a challenge to process packet efficiently while having rich traffic statistics. That is why the C language has been preferred to other languages such as Java. In fact, the current ntop version runs on hosts with very limited memory whereas an early prototype written in Java had serious performance problems and needed a lot of memory (due to the use of JIT compilers) that prevented it from running on average loaded networks.
- IP Protocol Stack
  Almost every operating system uses IP flags differently, and some protocols (e.g. HTTP)

make extensive use of IP flags for performance optimisation. This pushed the authors to update the ntop TCP protocol engine (used to keep the status of the TCP connections) several times before to reach the actual version. It is worth to note that tools like *queso* and *nmap* [8] exploit peculiarities of IP stack implementation in order to guess the running operating system.

- Open Source Software
  The adoption of OSS allowed both ntop to be extensively tested on a very large number of different systems and deeply influenced ntop's design. In fact, many ntop features have been implemented because some users asked for them and several problems have been fixed because somebody studied the code, tackled the problem and sent back the code patch.

5 Future Work

Although ntop already contains many features that were not planned at the beginning, a few enhancements are necessary in order to increase its flexibility and make it open to extensions. Planned enhacements include, but are not limited to:

- Operating System Integration
  It is unknown to the authors why modern operating systems handle network communications differently from processes. Processes can be listed, changed of priority, killed. The same should be applied to network communications. For instance, users should be able to list and terminate active TCP connections (even those that do not include the host where ntop runs) as described in [5]. Security issues need to be further investigated.
- Application Extensibility
  As of today ntop is a monolithic application that does not allow users to add new specific features. It is the authors belief that user-specific extensions to the ntop kernel would not make too much sense. A possible solution to this problem is the definition of a clean programming interface that allows users to write software components (plugins) [6] able to solve a specific problem. For instance if a user needs to periodically store in a database the used network bandwidth, then a plugin could be written for this purpose. The use of plugins allows users to extend ntop in a clean way by using specified interfaces without having to extend the ntop core with new peculiar functionality.
- SNMP
  The actual ntop implementation cannot be easily integrated with a management platform. This is because ntop supports HTTP whereas management platforms usually speak SNMP. The natural way to add SNMP support to ntop, would be the definition of a specific MIB (or selected parts of existing MIBs) and the support of the SNMP protocol. In that way ntop could act as a SNMP agent able to both handle incoming request and emit traps when some user-specified thresholds are exceeded.

6 Final Remarks

This work attempted to demonstrate that it is possible to analyse network traffic without having to purchase either expensive management platforms or network probes. Established tools such as ping and traceroute can be profitably used for solving connectivity problems whereas ntop can be used as a magnify lens for analysing global network traffic. The ntop interactive mode has been conceived as a quick network diagnostic whereas the web mode provide users a detailed view of the current and past network activities. ntop's lightweight cpu utilisation, minimal requirements, and support of various network media make it suitable for all those people who want to analyse network traffic without having to afford an expensive management platform.

7 Availability

Both *ntop* and *libpcap for Win32* are distributed under the GPL2 licence and can be downloaded free of charge from both the ntop home page (`http://www-serra.unipi.it/~ntop/`) and other mirrors on the Internet. Some Unix distributions including FreeBSD and Linux, come with ntop pre-installed.

8 Acknowledgments

The author would like to thank all the ntop users and early adopters who deeply influenced the design of the overall architecture with all their comments and suggestions.

9 References

1. Abell V.: lsof, `ftp://vic.cc.purdue.edu/pub/tools/unix/lsof/` (1998).
2. Apostols E.: queso, `http://www.apostols.org/` (1998).
3. Binns R.: top (1993).
4. Brownlee N.: NeTraMet v.4.2 Users' Guide, `http://www.auckland.an.nz/net/Accounting/` (1998).
5. Claerhout B.: IP Spoof (1996).
6. Deri L.: Droplets: Breaking Monolithic Applications Apart, IBM Research Report RZ 2799 (1995).
7. Deri L.: Surfin' Network Management Applications Across the Web, Proceedings of 2nd Int. IEEE Workshop on System and Network Management (1996).
8. Fyodor: Remote OS detection via TCP/IP stack fingerprinting, `http://www.insecure.org/nmap/nmap-fingerprinting-article.txt` (1998).
9. Jacobson V., Leres C., and McCanne S.: tcpdump, Lawrence Berkeley National Labs, `ftp://ftp.ee.lbl.gov/` (1989).
10. Jander M.: Web-based Management: Welcome to the Revolution, Data Communications (1996).
11. S. McCanne and V. Jacobson: The BSD Packer Filter: A New Architecture for User-level Packet Capture, Proc. of 1993 Winter USENIX Conference, 1993.
12. S. McCanne, C.Leres and V. Jacobson: libpcap, Lawrence Berkeley National Labs, `ftp://ftp.ee.lbl.gov/` (1994).
13. Microsoft Corporation: NDIS Packet Driver 3.0 (1996).
14. Ranum M., and others: Implementing a Generalized Tool for Network Monitoring, Proc. of LISA'97, USENIX 11th System Administration Conference, `http://www.nfr.com/forum/publications/LISA-97.htm` (1997).
15. Raymond E.: The Cathedral and the Bazaar, `http://www.tuxedo.org/~esr/` (1998).
16. Stevens R.: UNIX Network Programming, Volume 1, 2nd Edition (1998).
17. Waldbusser S.: Remote Network Monitoring Management Information Base, RFC 1757 (1995).
18. Free Software Foundation, GNU gdbm, `http://www.gnu.org/software/gdbm/` (1999).