ntop: Beyond ping and traceroute

Luca Deri and Stefano Suin
1
Centro SERRA
University of Pisa, Italy.

Owing to the increasing number of networked computers running different operating systems and speaking various network protocols, the task of network management is becoming increasingly complex. 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 solved 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 various network media, ease of use and lightweight cpu utilisation make it suitable for people who want to manage their network without having to adopt a sophisticated yet expensive management platform.

Keywords: Internet, Web-based Network Management, TCP/IP, Open Source Software.

Draft Document
Do not Distribute

1. Background and Motivation

Popular tools such as ping, traceroute [Stevens98] and have been used for years now for monitoring and debugging simple network connectivity issues. Although these tools often are sufficient for tackling simple problems, they have been created for monitoring network activities between two hosts. In cases where the network problem to solve is due to the interaction of traffic originated by various hosts, then these tools show their limits. Network sniffers such as tcpdump [Jacobson89] or snoop [snoop] are quite useful for analysing network traffic but off-line applications are often necessary for correlating captured data hence identify the network flows. Many commercial network sniffers are usually able to analyse data while capturing traffic but still these tools are quite primitives because they focus mainly on the packet and not on general network activities. In other words, operators can know virtually everything about the content of a single network packet whereas it is very difficult to extract information concerning the whole network status while a network problem arose.
Similarly, network probes such as RMON agents [Waldbusser95] are quite powerful tools but unfortunately need sophisticated SNMP managers able to configure them properly, and retrieve and represent collected network statistics. Due to this complexity and also to the cost of such probes, RMON agents are often used uniquely by advanced network managers in large institutions.
Other tools for network monitoring such as NeTraMet [Brownlee98] and NFR [Ranum97] 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.

Despite operating systems evolve quite fast, 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

[1.] Centro SERRA, Lungarno Pacinotti 43, Pisa, Italy, {deri,stefano}@unipi.it,http://www-serra.unipi.it/~ntop/.

have access to simple yet powerful network monitor tools able to give answer to questions such as:

- Why the local network performance is so poor?
- Who is using most of the available network bandwidth?
- What hosts are currently killing the performance of the local NFS server?
- What is the percentage bandwidth usage of my computer?
- What are the hosts contacted and the amount of network traffic produced by each of the processes running on my local computer?
- What are the hosts that periodically contact a multicast address?

ntop has been written for giving a positive answer to all of the above questions. It has been initially written by the authors for tackling performance problems of the network backbone. Similar to the Unix top 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 bandwidth. ntop then evolved in 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 an open-source software[2] (OSS) [Raymond98] application written using the C language available free of charge under the GNU public licence[3]. 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 that has been strongly influenced by the Webbin 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 (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 represented is rather simple.

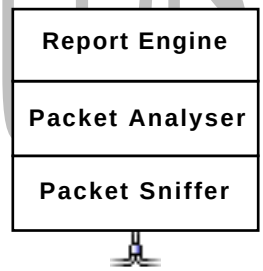

Figure 1 - ntop Architecture

The packet sniffer collects network packets that are then passed to the packet analyser for process-

[2] OSS homepage is http://www.opensource.org/.

[3] ntop can be downloaded free of charge from either the ntop home page (http://www-serra.unipi.it/~ntop/) or from other mirrors on the Internet. Some distributions of the Linux operating system come with ntop preinstalled.

Draft Document
Do not Distribute

ing. 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 threads, there is not a portable library for packet capture. Under Unix the *libpcap* [McCanne94] 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 [Microsoft96] 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 to be processed by the packet analyser. Packet filtering is based on the BPF filter [McCanne93] facility part of libpcap. Filters are specified using simple expressions as those accepted by tcpdump.
Packet capture libraries have small internal buffers that prevents applications from being able to handle burst traffic. In order to overcome this problem hence reduce packet loss, ntop buffers captured packets and it allows the packet analyser to be decoupled by the packet capture.
It is worth to remember that ntop can operate on switched networks. In fact, modern switches allow global network traffic to be mirrored to a switch port. ntop needs then to be activated on a host that is attached to such switch port.

2.2 Packet Analyser

The packet analyser processes one packet at time. The packet headers are analysed according to the network interface being used. This is because headers are different depending on the network interface. Hosts information is stored in a large hash table whose key is the hardware (MAC) address. Each entry contains several counter that keep track of the data sent/received by the host according to the various network protocols. For each packet, the hash entry corresponding to source and packet 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 those hosts. In order to avoid exhausting all the available memory and creating huge tables that decrease the overall performance, ntop when necessary (e.g. periodically or if there are no entries left) purges the host table by removing hosts that have not sent/received data for a long period of time. This guarantees that the ntop memory usage does not grow indefinitely and that packet processing time does not increase with the number of active hosts.

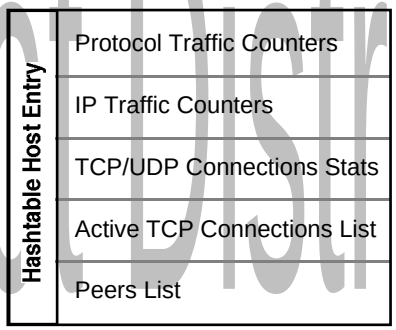

Figure 2 - Host Hashtable Entry

If the received packet is a non-IP packet, the entry counters are updated and the packet is discarded. Instead if the received packet is an IP packet, then further processing is performed.
The host entry contains a counter for each of the user-specified IP protocols. For each IP packet, the appropriate protocol counter is updated. If the (either TCP or UDP) 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 store in the hash table using the fragmentId as hash key. Such information is removed when the last fragment has been received. Since it might happen that some packet (including fragments) get lost, in order to avoid keeping outdated information the fragment table is periodically analysed and outdated information is purged

Draft Document
Do not Distribute

from it. The host entry also contains a list (initially empty) of the host's pending TCP connection. ntop maintains the state of each TCP connection analysing the IP flags. Due to this, if the received packet is a TCP packet, then the host TCP connection list needs to be updated.

Although host traffic counters can be profitably used to analyse the network traffic, in some cases it might be necessary to study specific traffic that flows through some specified hosts. ntop allows people to specify *network flows*, where a flow is a stream of packets that matches a user-specified rule. Rules are specifies on the command line when ntop is started 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.

Draft Document

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 the 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 in order to be independent of the way traffic reports are created, hence 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 application remains unchanged.

3. ntop at Work

3.1 Interactive Mode

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

Do not Distribute

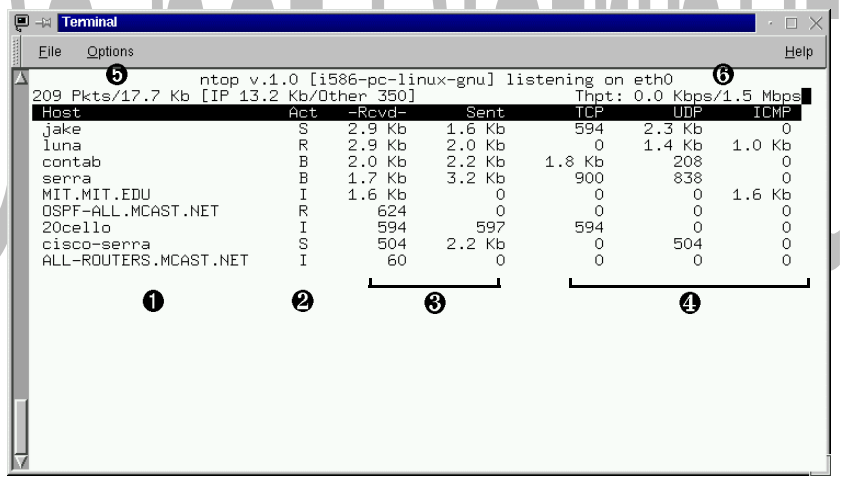

Figure 3 - ntop: Interactive Mode

The column ❶ contains the list of hosts that have sent/received data, the column ❷ specifies the actual host state (S=send, R=receive, B=send/receive and I=idle). The column ❸ contains the total data sent/received by each host, whereas the column ❹ is detailed view of the previous column. Us-

ers can change the sort order or the shown protocols 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 what hosts are decreasing the overall performance). Instead, the web-mode turns ntop into a full fledged web-based management application [Jander96] as shown in the figure below.

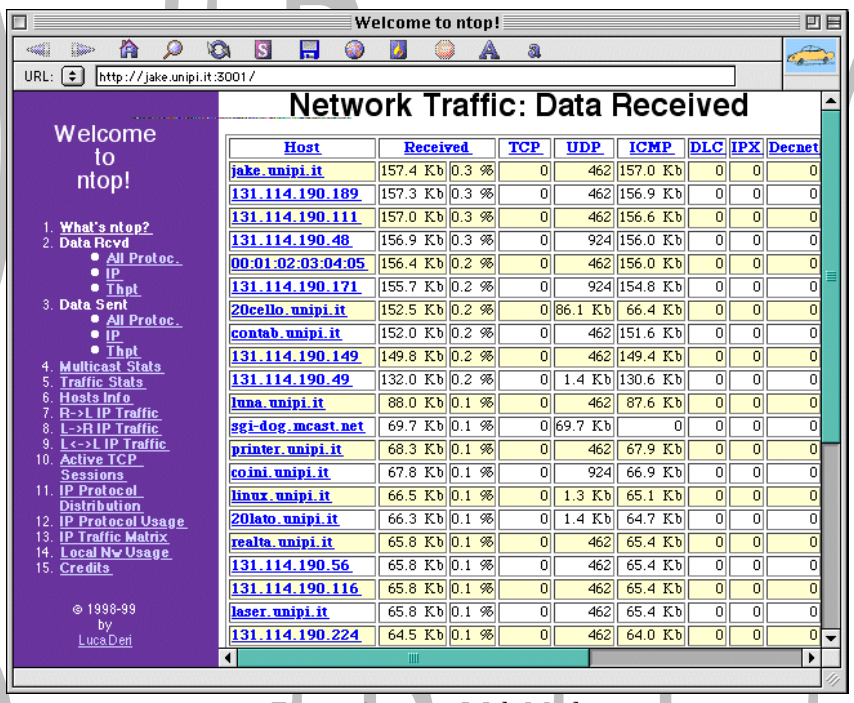

Figure 4 - ntop: Web Mode

The web-mode has been designed as a long standing statistics gathering application that is 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. Users connect their web browsers directly to ntop that acts as an HTTP server. The web view is divided in two frames: the left one is used for navigating through traffic information that will be displayed in the right frame. All the relevant table columns are sortable simply 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, currently active TCP sessions, TCP/UDP session history, provided/used IP services, used bandwidth percentage.
- Traffic Statistics: local (subnet) traffic, local vs. remote (outside specified/local subnets), remote vs. local, packet statistics (similar to RMON), network throughput (actual, peak, aver-

Draft Document
Do not Distribute

age).

- Currently active TCP sessions.
- IP/non-IP Protocol Distribution: distribution of the observed protocols according to source/destination (local vs. remote).
- Local subnet traffic matrix.
- Network Flows: traffic statistics for each user specified 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* [lsof] in order to calculate 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/receive data to 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 ① there is no portable way to retrieve the list of open IP ports for each running process and ② 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 Performance Issues

ntop performance is quite good basically for three 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 users have tested 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 packet loss on a 100Mbit 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 to have to drop some packets. Due to the way ntop works, if a packet gets lost major problems may arise. In fact suppose to loose a 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 the sender wants to close the TCP connection (three way handshake).

In order to overcome these problems, ntop implements internal timeouts and periodical garbage collection in order to purge old data and speculate about the status active connections. This allows ntop to recover whenever some packets get lost.

4. Lessons Learned

ntop has been a great exercise for many reasons:

- performance: it is very difficult to process packet efficiently and at the same time have 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 would had serious performance problems that prevented it

Draft Document
Do not Distribute

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 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 a tool named *queso* [Apostols98] exploits peculiarities of IP stack implementation in order to guess the running operating system.
- open source software: the adoption of OSS not only allowed ntop to be extensively tested on a very large number of different systems but also deeply influenced ntop design. In fact, many ntop features have been implemented because some ntop 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 terminate active TCP connections (even those that do not include the host where ntop runs) and kill them when the actual network throughput is too low. 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. The positive solution to this problem is the definition of a clean programming interface that allows users to write software components [Deri95] able to solve a specific problem such as periodically graph or store in a database the network utilisation.
- 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 the support of specific parts of existing MIBs) and the support of the SNMP protocol. In that way ntop could act as a SNMP agent able to both reply to incoming request and emit traps when some user-specified threshold is exceeded.

6. Final Remarks

This work attempted to demonstrate that it is possible to analyse network traffic without the need to purchase expensive management platforms or network probes usable only by highly skilled people. 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 the actual 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 (bare operating system), 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.

Draft Document
Do not Distribute

7. 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.

8. References

[Apostols98]	E. Apostols, *queso*, http://www.apostols.org/, 1998.
[Brownlee98]	N. Brownlee, *NeTraMet v.4.2 Users' Guide*, http://www.auckland.an.nz/net/Accounting/, 1998.
[Deri95]	L. Deri, *Droplets: Breaking Monolithic Applications Apart*, IBM Research Report RZ 2799, September 1995.
[Deri96]	L. Deri, *Surfin' Network Management Applications Across the Web*, Proceedings of 2nd Int. IEEE Workshop on System and Network Management, June 1996.
[Jacobson89]	V. Jacobson, C.Leres and S. McCanne, *tcpdump*, Lawrence Berkeley National Labs, ftp://ftp.ee.lbl.gov/, June 1989.
[Jander96]	M. Jander, *Web-based Management: Welcome to the Revolution*, Data Communications, November 1996.
[lsof]	*lsof*, ftp://vic.cc.purdue.edu/pub/tools/unix/lsof/ **(????)**
[McCanne93]	S. McCanne and V. Jacobson, *The BSD Packer Filter: A New Architecture for User-level Packet Capture*, Proc. of 1993 Winter USENIX Conference, 1993.
[McCanne94]	S. McCanne, C.Leres and V. Jacobson, *libpcap*, Lawrence Berkeley National Labs, ftp://ftp.ee.lbl.gov/, 1994.
[Microsoft96]	Microsoft Corporation, *NDIS Packet Driver 3.0*, 1996.
[Ranum97]	M. Ranum 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, October 1997.
[Raymond98]	E. Raymond, *The Cathedral and the Bazaar*, http://www.tuxedo.org/~esr/, 1998.
[snoop]	snoop
[Stevens98]	R. Stevens, *UNIX Network Programming*, Volume 1, 2nd Edition, 1998.
[Waldbusser95]	S. Waldbusser, *Remote Network Monitoring Management Information Base*, RFC 1757, February 1995.

Draft Document
Do not Distribute