

## NETMON: The Network Monitor

Netmon is a compact, easy-to-use network information utility. It displays information pertaining to the IP, TCP, UDP and ICMP protocols. Its main purpose is viewing connections made using TCP and UDP protocols from or to your computer. This information may prove very useful in hunting trojans (or other suspicious activity) present in your system.

Netmon is a graphical conversion of the "netstat" utility shipped with Windows. Its main advantages over the console based version, is the graphical user interface (GUI), the database of common trojan ports and the complete list of well-known ports (the ports that are numbered below 1024 and reserved for different applications).

Users familiar with the netstat utility should feel at home with the GUI and the information presented.

## **FILTER section**

Netmon has, beginning with version 1.5, the ability to filter out unwanted ports. It can also use a host filter, which when configured properly, only displays traffic from and to the specified host. These two features can prove very useful in tracking specific software activity or activity from a given host.

The port filter system is used to add ports that should be left out of Netmon's listing of port activity. This is applicable for both the TCP and the UDP protocol. For instance, if you not interested in HTTP traffic occurring on your network, you would open the Filter settings window, and enter "80" in the editbox below the "Add" button. Pressing "Add" will make Netmon ignore all references to port 80 on both local and remote traffic. Also, it is possible add a range of ports to be ignore. For example, if you are only interested in traffic above port 1024 (ports below 1024 are used for application protocols such as ftp and http), you would enter "1-1024" as filter. Up to 16 filters can be defined.

The syntax for adding filters is as follow: to add a single port as filter just enter it's digits ("23", "12"). To add a range of ports use a dash to separate digits ("122-859")

A new rightclicking command has been implemented ("Watch this host only"), so that hosts in the listview can easily be added as a host filter.

## Frequently asked questions

**Q: Why doesn't Netmon display the processes associated with ports?**

A: If this were an easy thing to implement, I would have done it a long time ago. Netmon only reads lists of connections, which are maintained by the system, and the "owner" of the connection is not specified in these. To view what processes has initiated the connection/socket, one would have to delve deeper within the system, most probably even writing a system specific kernel driver. And as I like to keep Netmon compatible with all systems configuration from Win98 and up, this is not possible as I have limited access to hardware.

**Q: Why is Netmon so much faster at displaying it's output than Netstat?**

A: Netmon actually isn't faster at all, but it utilizes multiple threads for it's operation whenever possible. This means that things like looking up host names will be done in the background, not locking up the main window while doing so.

**Q: Can Netmon terminate connections?**

A: Yes, beginning with version 1.54, Netmon can terminate TCP connections by right clicking on them and selecting the "Terminate connection" option.

# About connection states

This information is taken from RFC 793 (RFC = Request For Comments) and it describes the different stages in a connections lifetime.

**LISTEN** - represents waiting for a connection request from any remote TCP and port.

**SYN-SENT** - represents waiting for a matching connection request after having sent a connection request.

**SYN-RECEIVED** - represents waiting for a confirming connection request acknowledgment after having both received and sent a connection request.

**ESTABLISHED** - represents an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.

**FIN-WAIT-1** - represents waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.

**FIN-WAIT-2** - represents waiting for a connection termination request from the remote TCP.

**CLOSE-WAIT** - represents waiting for a connection termination request from the local user.

**CLOSING** - represents waiting for a connection termination request acknowledgment from the remote TCP.

**LAST-ACK** - represents waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).

# About ports

Each process that wants to communicate with another process identifies itself to the TCP/IP protocol suite by one or more ports. A port is a 16-bit number, used by the host-to-host protocol to identify to which higher-level protocol or application program (process) it must deliver incoming messages.

As some higher-level programs are themselves protocols, standardized in the TCP/IP protocol suite, such as TELNET and FTP, they use the same port number in all TCP/IP implementations. Those "assigned" port numbers are called well-known ports and the standard applications well-known services.

The "well-known" ports are controlled and assigned by the Internet Assigned Numbers Authority (IANA) and on most systems can only be used by system processes or by programs executed by privileged users. The assigned "well-known" ports occupy port numbers in the range 0 to 1023. The ports with numbers in the range 1024-65535 are not controlled by the IANA and on most systems can be used by ordinary user-developed programs.

Confusion due to two different applications trying to use the same port numbers on one host is avoided by writing those applications to request an available port from TCP/IP. Because this port number is dynamically assigned, it may differ from one invocation of an application to the next.

Examples of ports which are "well-known" are 80, which is used for http traffic (the web), and 21, which is used for ftp (the file transfer protocol).

# About trojans

## Introduction

A trojan [horse] is (in the computer virii sense of the word), a program which, in likeness with it's mythological counterpart, infiltrates a system, often disguised in the form of a valid program. As the maker of the trojan, or more often a malicious user, need you to run this program in order for the trojan to be able contaminate your system, this program claims to do something spectacular, in lines with "doubling your internet speed", or perhaps "improving the speed of your computer with 500%". As most people find this interesting, they run the trojan "dropper" (which the program carrying the injection code for the trojan often is called), and thus the damage is done. The trojan copies itself into the system, and make sure that it is started along with the regular operating system files the next time you reboot your computer.

## How do trojans operate?

So, what do the trojan do when it successfully has installed itself in a system? First of all, it opens a port (TCP or UDP) on the host system (would be seen as state LISTENING in netstat or Netmon), and awaits for a client to connect to this port. The client in this case is the person who wants to access your system undetected. This person loads his trojancontrolling client software and enters your IP address, and the port number (which port number varies from trojan to trojan, and is mostly configurable in the trojan server). After he has successfully connected, the possibilities are only limited by the trojan server itself. The ability of dumping dialup/network passwords, downloading files, uploading files, rebooting the computer, deleting files etc etc are all featured in most trojans.

.. more to come ..

## System Requirements

### Software requirements:

- \* Tahoma font (while not really required, some parts of the GUI will look really ugly without it as a default fallback font will be used instead). This can be downloaded from [www.microsoft.com](http://www.microsoft.com).
- \* Windows 98, ME, NT or 2000

### This version of Netmon has been successfully tested on the following systems:

- \* Windows 98 Second Edition
- \* Windows 2000 SP 2
- \* Windows Millenium

If you are using one of the system listed above and still experiencing problems with Netmon, do please contact the author so the matter can be investigated.

## Contacting the author

Contact the author if you have suggestions for improvements, have found a bug or if you're curious about anything concerning the program.

**NOTE:** Please feel free to drop me a line if you found the program useful. Non-commercial software really needs encouragement, so if you feel people should keep developing free software, do at least support them by dropping a short note telling them so. =)

**Johan Samuelson**

Snail mail:  
Johan Samuelson  
Stenbarsvagen 22  
590 54 Sturefors  
SWEDEN

Mail: [adamant@home.se](mailto:adamant@home.se)  
Web: <http://adamant.n3.net>



## Things to implement (wishlist)

- \* The ability to log port traffic.
- \* View the process associated with a specific port.

## History of Netmon

- + = added feature or improvement
- = bug removed, small fix, cosmetic change

### **v1.6 ( 22.10.01 )**

- + Tweaking of the filter code.
- + Fixed error messages on failed connection termination attempts.

### **v1.58 ( 08.10.01 )**

- + Fixed some rather nasty memory leaks that occurred when using filters.
- + Added Alt+1,2,3,4 as hotkeys for quick filtering of sockets. This is really useful I believe.
- + Removed all (!) trojan detection support. This is due to the poor identification ability of Netmon (too many misidentifications). It was designed as a network tool, not a trojan hunter. There are plenty of capable software that can be used for this purpose with better results.
- + Fixed the lagging statusbar.

### **v1.57 ( 15.07.01 )**

- + Added search capability to the country code/trojan/service databases.
- + Added some service definitions.

### **v1.56 ( 08.07.01 )**

- + Main window always on top option added.
- Fixed several typos.
- Minor GUI redesigns.
- Fixed a bug pertaining to the vertical scrollbar during startup.

### **v1.55 ( 22.04.01 )**

- + A new toolbar added. Can be turned off via the "View->Windows->Toolbar" menu.
- + New configuration option; "Automatically pause when entry is selected."
- Fix of minimize/systray icon bug.
- Fix in the filter handling code.
- + Updated the system identification code. Now recognizes subsets of 9x, Windows ME and Whistler.
- + Updated the ip address regions database.
- + Added a few more trojans and services.

### **v1.54 ( 15.04.01 )**

- + Added the ability to terminate connections. This feature is used by rightclicking on a TCP connection and selecting the "Terminate connection" option.
- Fixed the popup menu to be top-left aligned, as this is more logical (so you actually can see what entry is selected =).

### **v1.53 ( 01.03.01 )**

- + Pruned the service database, removing ~400 non Windowish services (no use having them waste good memory while never being used, right?) and instead added a bunch of Windows services operating on ports higher than 1024.
- + Changed into the NSIS installer, which really rocks!
- Added a timeout to webquery function, so that it won't lock the main window anymore.
- Improved the functionality of the lookup engine, so that hostnames which has been marked as unresolvable are retried after some time ( if the first lookup failed for some recoverable network error).

### **v1.52 ( 13.01.01 )**

- + Updated the trojan database. 174 entries now.
- + Added a "Check for trojans" ability. This is used to match all local ports against the list of trojan default ports. If a match is done, this is reported to the user. Beats looking up all ports manually. \*smile\*

### **v1.51 ( 05.01.01 )**

- + Finally fixed the HORRID flickering occurring when refreshing a large list of connections.

### **v1.5 ( 02.01.01 )**

- + Added host and port filtering. Be sure to read the specific filter section in this documentation

### **v1.4 ( 21.12.00 )**

- + Resizing main window now resizes columns accordingly.
- Version 1.4 release. Removed beta status.

### **v1.4 beta 5 ( 14.12.00 )**

- + Added an option to switch my "improved" refresh window code, as it seemed to cause trouble on some systems.
- Began augmenting work on the documentation (network reference section)
- Interface statistics now refreshes automatically when open
- Options->Lookup->Update connection when a resolve.. is now fully functional
- Cosmetic change of the connection states (LISTEN->Listening etc)

#### **v1.4 beta 4 ( 19.11.00 )**

- + Added the "Web Query" capability. This is a new function in the toolmenu accessed via rightclicking on a host in the main view. It determines what webserver the remote host uses.
- Added a minor fix for the Interface statistics on cards using RTL8139.sys.
- A couple of nasty typos fixed (I can't spell for crap it seems)

#### **v1.4 beta 3 ( 02.11.00 )**

- + Reduced flickering in main window
- Removed a HUGE bug; Netmon would crash if no connections at all were present

#### **v1.4 beta 2 ( 28.10.00 )**

- + Added a nice installer for the project
- + Added the Lookup page to configuration
- Configuration remembers last page =)
- Fixed the TCP/UDP filters
- Redesigned almost all dialogboxes and found some new catchy icons by Copland to use
- Updated the help file a bit
- Removed a bug that occurred if not interfaces were available
- Removed a bug in the context help routine

#### **v1.4 beta 1 ( 25.10.00 )**

- + Added the Interface statistics windows
- + Added the ability to automatically lookup ip addresses.
- + Added the "Save snapshot..." ability. This is used to take a "snapshot" of the current tcp/udp connections and save this to a file.
- + Added a cache database entry. This holds all hostnames Netmon has resolved during this session.
- Integrated all the external files in Netmon, thus removing the need of "services.dat" and "cc.dat".
- Redesigned the configuration window into a property sheet.

#### **v1.3 ( 05.06.00 )**

- + Added new option "start as minimized".
- + Updated the trojan database list.
- + Added the small, nice icons for the statusbar.
- + Added logging ability.
- + Added the toolbar.
- + Added the country codes database.
- Improved the handling of services and cc's. More dynamic now.
- No longer needs snmpapi.dll.
- Fixed the shown/total bug.

#### **v1.2 ( 10.02.00 )**

- + The protocol statistics window now update values in realtime
- + Added the ability to pause the automatic update of the list of connections
- + Added automatic refresh of list as an option in the configuration
- + Added the option to display real ip addresses instead of "localhost"
- Fixed the lagging counters in the statusbar

#### **v1.1 ( 25.01.00 )**

- + improved trojan detection, less false alarms from now on when scanning for trojans
- + host details window added
- + GUI was written, real window class, more clarity in interface
- + the help system was rewritten and revised, context help for almost every window now
- + separate configuration window with option to set exact refreshrate
- fixed trayicon bug upon exit

#### **v1.0 ( 12.01.00 )**

first release for the masses

#### **beta ( 12.01.00 )**

- + added more configurability like options to disable trojan/service lookup
- fixed window size/position config code

#### **beta ( 07.01.00 )**

- finally removed the ugly memory leaks in countconnections() and refreshconnections()

**beta ( 01.01.00 )**

happy new year! =)

**beta ( 27.12.00 )**

- + added savable configuration (window pos etc)
- + added configurable refresh interval
- + added clipboard support

**beta ( 21.12.00 )**

- + added nice bars to graph
- + added selective display
- + added udp sockets support
- + added parsing of services (services.dat)
- .. old stuff removed ..

**beta ( 02.11.99 )**

project started

## Legal and distributional status

The software may be freely copied, stored and distributed by any person or organization, providing that the person or organization meets the terms and conditions of the following conditions:

- \* Modification of the software, and storage, distribution and copying of modifications to the software is prohibited.

- \* Redistribution of the software for profit without the authors knowledge and/or permission is prohibited.

Netmon is released as freeware. No registration or fee is needed to use this program. Still, if you find it useful, please feel free to drop me an e-mail telling me so.

**Disclaimer:** The author takes no responsibility for direct or indirect damage caused by the use of this software product.

## Acknowledgements and credits

Thanks to Copland <copland@cybergal.com> for the wonderful main icons.

Thanks to Scrow (<http://members.nbci.com/scrows/>) for the toolbar icons.

Huge thanks to the following people for suggestions, bug reports and/or general feedback. Unfortunately names are missing here, since I haven't been archiving all mails, but you know who you are. Anyway, here's the list (in no particular order):

pmacneil, Deborah Hirshberg, Mike McDaniel, Gokula Krishnan, Namgorf, David Heebner, Andrew Price, Moni Keo, Robert W. Habern, Bill Wainwright, Ken Cox, Luke Hamburg, Denny Church, Karsai Peter, Shawn Giese, Joseph Collins, Thorsten Stocksmeier, Daniel Sudakovskiy, Chaim, Robert Lundblad, Mark Rowlands, Turbo\_Kazzam, Biscut, R00t, R Y, Umesh, Alexander Stan, Ben F, Mark Henry, Marco Kuhnel, Chris Carey, Steve Watson, Jamal Al-Akkad, Oliver Maess, Amir Abbas, Gary A. Mays, Chuck, Lillian Abraham, John DeCola, Björn Harrell, Gabe Rieger, Takashi, AJ Valleskey, Joakim von Braun, Frank Smith, suhoiy, RobFox, lisa madden, Tom Cameron, theyen, Ivan Shelley, Mark Bell, Thomas Hein, Mark J Pappert, Martin Strandh, Stephane Mass, William Muscat, Neil Fitch, Sim Hildreth, Jean-Louis Frissard, David D, Richard Landau, steve, Bern Tengvall, Onur Ilkorur, Thomas Galdamez, Juan Carlos Castro, Zdravo Stoychev, Michael Myers, William Fullerton, Ken Hickman, Garapata

## Configuration Help

Please use the context help available on all configuration options. This is reached by clicking on the questionmark in the upper-right window border and then clicking on the configuration option.

## Main Window Help

**Listview** The main window consists of a listview that displays connections, if using the TCP protocol, or ports, if using the UDP protocol. The UDP protocol does not use "connections", so not remote address will be available. The Status column only display status for TCP connections, UDP uses no status for sockets. Double clicking on entries in this list will bring up the host details window. \*: in any column means any address or any interface.

**Status bar** In the bottom of the main window is a statusbar, in which various information will be shown continuously during execution. The right pane of the statusbar shows how many connections are displayed and the grand total of connections.

**Menus** The statistics menu gives you the possibility to examine some common protocols and details about them. Details won't be given here, all statistics windows have context helps; click the questionmark in the right upper corner, and point on one of the values to display more information about that statistic.

**Context (popup) Menu** Click the right mouse button over an remote address will bring up a context menu. Here you currently have three options:

**Context Menu -> Ping host** Send ICMP Echo request to host. This will be sent asynchronous, so you can continue operations while waiting for reply. A reponse could look something like this, 127.0.0.1 :: Reponse time 1 ms (TTL 128), First we have the host ip address, then an estimated round trip time (the time it took for the packet to travel to that computer and back). Lastly we have the Time To Live value. This is a kind of "passed-hosts" counter, the packet sent out is set with a certain TTL value. As a router on the Internet handles the packet, it decrements this value.

**Context Menu -> Lookup host** This performs a lookup on the selected ip address. It turns the dotted ip (for example 127.0.0.1) into a symbolic form (for example www.altavista.com). Netmon has a built in cache for host names that have been looked up. This can be accessed via the "database" menu.

**Context Menu -> Copy to clipboard** This option copies the selected address to the clipboard, so that you can use it with other programs etc.



This field will display either the general location of the world where the Registry for this network address is placed, or the company who has registered the network address.

This field displays the Web Server used at the remote computer, if one is available and if it exposes its name/type. Note that for this field to be valid, you'll have to right click on a host in the main display and select "Query Webserver".

Example: lcs.mit.edu

Here, lcs.mit.edu is the lowest-level domain name, a subdomain of mit.edu, which again is a subdomain of edu (education) which is called a top-level domain.

There are five classes of IP addresses, A, B, C, D and E, but only A, B and C are currently used for hosting. Class A address can have 16,777,214 hosts, Class B 65534 hosts and Class C 254 hosts. Class D is reserved for multicasting and Class E is reserved for future use.

The early internet configurations required users to use only numeric IP addresses. Very quickly, this evolved to the use of symbolic host names. For example, instead of typing TELNET 128.12.7.14 one could type TELNET eduvms9, and eduvms9 is then translated in some way to the IP address 128.12.7.14. On the internet, this is accomplished using DNS (Domain Name System).

IP addresses are 32-bit numbers usually represented in a dotted decimal form (as the decimal representation of four 8-bit values concatenated with dots). For example 128.2.7.9 is an IP address with 128.2 being the network number and 7.9 being the host number.

The total number of IP datagrams that local IP user-protocols, including ICMP, supplied to IP in requests for transmission.

The total number of input datagrams received from interfaces, including those received in error.



The number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc.

The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (for example, 0.0.0.0) and addresses of unsupported Classes (such as Class E). For entities that are not IP gateways and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.

The number of locally-addressed datagrams received successfully but discarded because of an unknown or unsupported protocol.

The number of outbound IP datagrams discarded because no route could be found to transmit them to their destination. This counter includes any packets counted in ipForwDatagrams that meet this no-route criterion, which includes any datagrams that a host cannot route because all of its default gateways are down.

The indication of whether this entity is acting as an IP gateway in respect to the forwarding of datagrams received by, but not addressed to, this entity. IP gateways forward datagrams, but IP hosts do not (except those source-routed via the host).

The default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity whenever a TTL value is not supplied by the transport layer protocol.

The maximum number of seconds that received fragments are held while they are awaiting reassembly at this entity.

The total number of segments sent, including those on current connections but excluding those containing only retransmitted bytes.



The total number of segments received, including those received in error. This count includes segments received on currently established connections.

The algorithm used to determine the timeout value used for retransmitting unacknowledged bytes.

The limit on the total number of TCP connections that the entity can support. Can be dynamic or fixed value.

The number of times TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state.

The total number of segments retransmitted, that is, the number of TCP segments transmitted containing one or more previously transmitted bytes.

The total number of UDP datagrams sent from this entity.

The total number of UDP datagrams delivered to UDP users.

The total number of received UDP datagrams for which there was no application at the destination port.



The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.

If received from an intermediate router, it means that the router regards the destination IP address as unreachable. If received from destination host, it means that the protocol specified in the protocol number field of the original datagram is not active, or that protocol is not active on this host or the specified port is inactive.

If received from an intermediate router, it means that the time-to-live field of an IP datagram has expired. If received from the destination host, it means that the IP fragment reassembly time-to-live timer has expired while the host is waiting for a fragment of the datagram.

Indicates that a problem was encountered during processing of the IP header parameters.

If received from an intermediate router, it means that the router does not have the buffer space needed to queue the datagrams for output to the next network. If received from the destination host, it means that the incoming datagrams are arriving too quickly to be processed.

If received from an intermediate router, it means that the host should send future datagrams for the network to the router whose IP address is given in the ICMP message. This preferred router will always be on the same subnet as the host which sent the datagram and the router which returned the IP datagram. The router will forward the datagram to its next hop destination. If the router IP address matches the source IP address in the original datagram header it indicates a routing loop.

Echo is used to detect if another host is active on the network.

Echo reply is the response type for Echo requests.



Timestamp (+reply) are used for performance measurements and for debugging. They are not used for clock synchronization.

Timestamp (+reply) are used for performance measurements and for debugging. They are not used for clock synchronization.

An Address Mask Request is used by a host to determine the subnet mask in use on an attached network.

An Address Mask Request is used by a host to determine the subnet mask in use on an attached network.

This setting controls Netmon's behavior when you click on the close window button. If checked, Netmon minimizes itself to the system tray but continues monitoring. If not checked Netmon exits when the main window is closed.

Checking this option will make Netmon start minimized in the system tray. Useful if you want to run it at Windows startup, but don't want the GUI to popup.

Checking this option will make Netmon resolve references to port numbers below 1024 (well-known ports). For example, if there is traffic on port 80, this will be referenced as http.

Checking this option will make Netmon resolve references to trojan default ports. For example, if there is traffic on port 31337, this will be referenced as BackOrifice.



Checking this option will make Netmon display "localhost" for all local addresses, ignoring what interface (PPP/Ethernet card etc) the address is bound to.

This setting controls how Netmon refreshes the list of connections. If enabled, windows updates the list continously according to what is set in the refresh interval slider. If it is disabled, you have to update the list manually with a press of F5 or selecting it from the menu.

This slider controls the amount of time Netmon waits before checking if there is any new connections, or if old connections have been closed. A small value updates often, making it hard to see what's going on, and a large value might result in a loss of report of connections due to the infrequent updates. This setting defaults to 1000 (= 1 second).

If this option is checked, Netmon will display all sockets which utilizes the TCP (Transmission Control Protocol) for data traffic. TCP is connection-oriented and stream-oriented, and is more common than UDP in general network applications.

If this option is checked, Netmon will display all sockets which utilizes the UDP (User Datagram Protocol) for data traffic. UDP is a connectionless protocol, thus Netmon can never show what computer is accessing a certain UDP service.

If this option is checked, Netmon will display all sockets which are currently at status LISTEN. This means they are ready for inbound traffic. In the list of connections, an address or port named "\*" means any host or any port.

If this option is checked, Netmon will display all sockets which are currently at status ESTABLISHED. This means they are connected with another host and ready for transmission.

If this option is checked, Netmon will display all sockets which are currently at a transitional status. This means they are either switching from LISTEN->ESTABLISHED or from ESTABLISHED->CLOSE. Please note that these transitions are so quick that they often don't show up in the list anyhow.



If this option is checked, Netmon will try to resolve all IP addresses it sees into hostnames (for instance 165.178.4.12 would show up as [www.not-a-valid-host.com](http://www.not-a-valid-host.com)). This is done asynchronous and in multiple threads for maximum performance.

If this option is checked, Netmon will refresh the lists of connections whenever an IP address resolution is successful. Please note that this can cause excessive flickering in the main window, if many lookups are successful within a short timespace.

This value controls how many simultaneous threaded lookups Netmon is allowed to perform. Defaults to 10 and the valid range is 1 to 20. There is no optimal value, so try out a few different settings on your system.



