Université du LUXEMBOURG

Faculté des Sciences, de la Technologie et de la Communication
Campus du Kirchberg

June 2005

Development of a framework for performance testing of intrusion detection systems

Stage de pratique professionnelle

Yannick Loth
3rd year, Industrial Engineer in Applied Computer Sciences

Organisation:
Royal Military Academy (RMA)
Department of Communication, Information Systems and Sensors
Brussels
Belgium

Tutor UL:
Pr. Dr. Thomas Engel

Tutor RMA:
Dr. Ir. Wim Mees

Table of Contents

Illustration Index

Preface

The growth of information networks and of their importance in everyday life led many organizations and people to develop various security tools to monitor and control information flow through networks. But such tools are not always well documented or their behavior under specific circumstances may be unknown. For this reason, there is a need for some tools which help determine these marginal behaviors, which in fact may indicate security flaws even on devices well configured and efficient for everyday network usage.

The fact is that network architects and administrators do have to assume that if a private network is attacked, it will in fact be the worst case. Network architects have to build a security architecture which would deal best with heavy attacks. But how to take effective decisions about architecture if the behavior of security devices under heavy attacks is not well known?

This report describes basic concepts and tools for the development of an application framework which may be used to evaluate and illustrate the performance of Intrusion Detection Systems. Nevertheless, the results and techniques described may be applied to other network devices, such as firewalls or even web servers.

Thanks

I'd like to thank any people who permitted me to do this training in the best conditions, beginning with my parents who have been supporting me all my life, Major Célestin Herten and Professor Martin Timmermann, who were my two first contacts at the RMA and who accepted me as a trainee, Major Wim Mees, who permitted me to choose a subject which interests me a lot and who helped me with Captain Olivier Thonnard during the training. Both of them led me in my training to a good end and helped me to attain the goals that were established.

I also address a lot of thanks to the whole staff from the Computer Sciences Department of the RMA for the way they accepted me and for their sympathy during these fifteen weeks.

1 Context of the training

This training takes place during the second semester of the second cycle (3rd year) in Industrial Engineering in Applied Informatics at the University of Luxembourg, Faculté des Sciences, de la Technologie et de la Communication.

This training should permit the students to participate to industrial projects and to practice general industrial engineering techniques learned in the common courses. This training should also be in relation with the specializations proposed at the University of Luxembourg.

These requirements are fulfilled here, as the training involves:

- a complete study of networking techniques and network programming;
- the development of an application meeting usability requirements and performance requirements;
- the tests to prove that the application in fact verifies these requirements and to find its limits;
- the use of this application to study performance issues of at least one software security devices (Snort IDS) and if enough time of a hardware security device (Cisco IDS);
- the subject itself is in concordance with the specialization in Networking and Distributed Systems proposed by the University of Luxembourg.

2 Working place

1 Royal Military Academy – CISS Info

The training took place in the department of communication, information systems and sensors of the Royal Military Academy. This computer sciences group of this department is composed of about one dozen of researchers or professors and is headed by Pr. Martin Timmerman.

This department permits primarily officers to work on Master and Ph.D. Theses. These people are generally also charged of some teaching to the military students.

Most of them are Polytechnician Civil Engineers.

This training was headed by Dr. Ir. Wim Mees and should result in performance measures which are useful for Captain Olivier Thonnard's Master Thesis in Applied Computer Sciences at the Vrije Universiteit Brussel: *Network IDS Performance Analysis – Snort Evaluation and Profiling*.

In fact, the application framework developed for this training can be used to confirm some of the results found by Captain Thonnard.

It is obvious that Major Mees and Captain Thonnard were the two persons who guided me and decided which work I had to do.

My workplace was in the laboratory, as I continuously had to test the performance of my code and to test IDS performance.

3 Purpose of the training

The purpose of the training was to develop applications and scripts to test the performance of intrusion detection systems. This application framework should be able to generate high network loads of various kinds of data. Depending on the bit rate and the type of data that was sent to an intrusion detection system (IDS), its performances in fact does change.

Mainly two conclusions did interest Major Mees and Captain Thonnard:

- How does the IDS react to some data at different speeds?
- How reliable is the IDS when some type of data is sent to it at different speeds?

This permits to see how the IDS reacts in the worst case (i.e. when the data type sent to the IDS needs most processing power) and to see whether or not the profiling of Captain Thonnard is concordant to reality.

These tests were realized on the open source software IDS Snort (version 2.3.0), as installed by default. It is obvious that using the framework developed here, one can test performance of other IDSs like for example Cisco IDS 4215.

As there was no obvious way to realize such a work, the first weeks of this training were used to find, read and analyze documentation and consequently try code in little programs to find out if it was efficient and how difficult it would be to use it.

Also, there were issues we had to deal with during the implementation. This implies that it was quite not possible to establish a precise calendar during the training and that the application framework can easily be optimized. This will be discussed later.

Briefly:

- Build an application framework to test IDS;
- Do some tests with Snort IDS and interpret the values measured.

4 Intrusion detection systems

The following lines are resumed from Rebecca Bace and Peter Mell's paper *Intrusion Detection Systems*[1].

1 Concepts

Intrusion detection systems (IDS) are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analyzing them for signs of security problems.

Intrusion is defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network.

Intrusion detection allows organizations to protect their systems from the threats that come with increasing network connectivity and reliance on information systems.

Good reasons to acquire and uses IDSs:

1. They increase the perceived risk of discovery and punishment for attackers;
2. They may detect attacks that are not prevented by other security measures;
3. They detect preambles to attacks;
4. They may be useful to document the existing threat to an organization;
5. They may be used as quality control for security design and administration;
6. They provide useful information about intrusions that do take place.

Most IDSs are composed of the three following fundamental components:

1. Information sources – the sources of event information used to determine wether an intrusion has taken place;
2. Analysis – the part of intrusion detection that actually organizes and makes sens of the events derived from the information sources, deciding when those events indicate that intrusions are occurring or have already taken place. The most common analysis approaches are misuse detection (pattern-matching) and anomaly detection ("intelligent" IDSs);
3. Response – the passive and active measures taken once the system detects intrusions.

1**Rebecca Bace and Peter Mell**, *Intrusion Detection Systems*, NIST Special Publication on Intrusion Detection Systems.

2 Variety of existing systems

Control strategy describes how the elements of an IDS is controlled, and furthermore, how the input and output of the IDS is managed. Control strategy may be centralized, partially distributed or fully distributed.

Timing – measured as the elapsed time between the events that are monitored and the analysis of those events – is a critical property of IDS. Timing of IDSs may be interval-based or real-time.

Interval based IDSs feature a discontinuity in the information flow from monitoring points to analysis engines. Such IDSs cannot perform active responses.

Real-time IDSs operate on continuous information feeds from information sources. Detection performed by real-time IDSs yields quickly enough to allow the IDS to take action that affects the progress of the detected attack.

The most common way to **classify IDSs** is to group them **by information source:**

IDS may be network-based, host-based or application-based.

Network-based IDSs can monitor the network traffic affecting multiple hosts that are connected to the network segment, thereby protecting those hosts.

Host-based IDSs operate on information collected from within an individual computer. They can see the outcome of an attempted attack, as they can directly access and monitor the data files and system processes usually targeted by attacks. Their usual information sources are operating system audit trails and system logs.

Application-based IDSs – a subset of host-based IDSs – analyze the events transpiring within a software application. The ability to interface with applications directly allows those IDSs to detect suspicious behavior due to authorized users exceeding their authorization.

IDSs – like any other security system – do not provide an absolute and certain protection against attacks and intrusions. They are to be complemented by other systems, like vulnerability analysis systems, file integrity checkers, honey pots, firewalls and anti-virus systems.

3 Snort – an open-source software IDS

The following lines are taken from the Snort Users Manual 2.3.2[2] and from Rebecca Bace and Peter Mell's paper *Intrusion Detection Systems*.[3]

Snort is a lightweight network intrusion detection system, which can perform a variety of traffic logging and analysis functions on IP networks. It is a freeware product, available under the terms of the GNU General Public License as published by the Free Software Foundation. Snort has an extensive database of attack signatures. Both Snort and the attack signature database are found at http://www.snort.org.

Snort can be configured to run in different modes:

- *Sniffer mode*, which simply reads the packets off of the network and displays them for you in a continuous stream on the console;
- *Packet logger mode*, which logs the packets to disk;
- *Network Intrusion Detection System (NIDS) mode*, the most complex and configurable configuration, which allows Snort to analyze network traffick for matches against a user-defined rule set and performs several actions based upon what it sees.

2 *Snort™ Users Manual*, http://www.snort.org, March 2005
3 Cfr. infra

- *Inline mode*, which obtains packets from iptables instead of libpcap and then causes iptables to drop or to pass packets based on Snort rules that use inline-specific rule types.

For this work, we've used the NIDS mode of Snort.

As for almost every application, improving user interface and configurability of Snort does have negative impacts on its performance. For example, Snort makes the use of the Perl Compatible Regular Expression (PCRE) library to realize pattern-matching during detection. This permits to define very easily detection rules, but this also has a high impact on performance.

Snort is the IDS used throughout the training, because it's widely used, open source and the subject of the Master thesis of Olivier Thonnard.

To this training the use of Snort is not significant in the sense that the framework that was developed and the experience acquired during the training will be applicable to other IDS and network devices.

5 Application Requirements

1 Requirements

The application framework had to be able to send different types of data over networks at different speeds reliably. The maximum speed should be as high as possible. Reliably is not defined by a constant value, but it means that when a given .

The main idea is to be able to send various inoffensive data intermixed with various malicious data to the IDS in a way such that the ratio between the quantity of inoffensive data packets and the quantity of malicious data packets can be fixed by the user.

The application must accept one or two packet files as argument.

The network interface over which to send the packets may be imposed by the user.

The total quantity of packets sent over the network also may be imposed by the user. Provided the knowledge of the ratio and the quantity of packets to send, one can determine the quantity of malicious packets sent as well as the quantity of inoffensive packets.

The bit rate at which to send data over the network may be imposed by the user. In this case, imposed is not absolute, as there is no guarantee that the operating system, the application or the hardware are able to reach speeds desired by the user.

The scheduling priority of the application may be imposed by the user.

The application should be implemented to run on x86 PC computers.

It was decided that a Linux 2.6+ operating system would be used to implement the framework and the tests, thus the requirements are adapted to this fact.

2 Existing software

There already exists some software (for example *Packet Excalibur* or *tcpreplay*) which does about the same as asked these requirements, but the timing these applications use to determine when to send data rely on signals, which depend on the PC clock. On x86 PC's, signals are delivered at best every 1 or 10 ms, which is much too slow for our use. Thus it was decided to develop a new application and to write shell scripts to make its use easy.

Let's do some maths :

> If one wants to send packets of size 125 bytes (=1000 bits) at a bitrate of 100Mbit/s, the application must be able to send a 1000bit-sized packet every 10 microseconds. This is 1000 faster than what is permitted with signal-using implementations.

6 Stressnet

1 General concepts

The main application developed here and used for the tests is called stressnet.

2 Design decisions

2.a Architecture

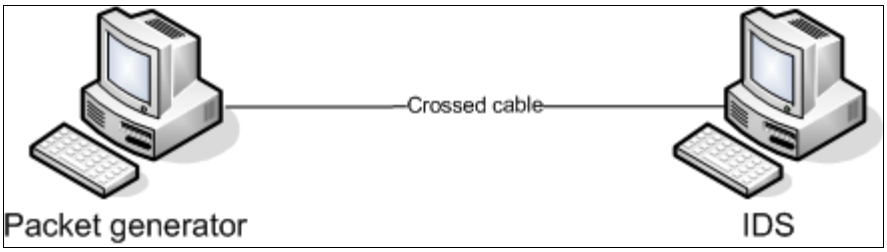


Illustration 1: Network architecture used to do the tests

The computers used to develop the application and to stress the IDSs are three HP Proliant ML110 computers with Intel 2.6 Ghz P4-based Celeron processors and 768 Mb physical memory.

The network devices on these PCs were Broadcom NetXtreme BCM5705_2 Gigabit Ethernet cards. During the tests, two computers were connected together directly with a crossed cable, thus there was no interference coming from external networks. Illustration 1 shows the network configuration used to do the tests.

The operating system installed on these computers and used to do the tests is SuSE Linux 9.2 Professional.

2.b Conception Logic

2.b.1 General concepts

It has been decided that the packets would be written directly to the data link layer and as such they had to contain every header information as well as the data payload, from the Ethernet header to higher level protocol headers. Because of the success of tcpdump and libpcap and their ease of use, it has been decided that the application has to be able to read packets from tcpdump-files. This format is easily read by libpcap, which is present on most if not on all modern Linux operating

systems.

As sending-performance is the most critical issue of Stressnet and more generally of the tests done, the kernels of both computers used for the tests were recompiled with the minimum set of features working and necessary for this use. Working means that because it is not obvious to determine which features to drop and which to keep, there may still be some unnecessary features.

Recompiling the kernel with minimalist features permits us to guarantee that the program will not be interrupted too much during its critical loop. For the same reason, we limited the applications running on the platforms to the minimum required for the tests.

Details about the features left in the kernel as well as about how to build such a minimalist kernel are given in *Appendix 1 – Custom Kernel*.

2.b.2 Stressnet command line arguments

The code used to parse arguments is generated with the application gengetopt[4]. Arguments may be given to Stressnet in the command line or as a configuration file. There is one argument which must be given to Stressnet: the path and filename of one dump file containing the packets to send.

If a ratio r is given as argument (this has no effect if only one file is passed to Stressnet as argument), then Stressnet will alternatively send r packets from file M then 1 packet from file N, this repeated until the total quantity of packets is reached.

Illustration 2: Ordering of packets in memory and for sending

The argument '--help' lists details about the usage of Stressnet and its various arguments.

2.b.3 Reading file stats

The packets that will be sent, when read from files, are stored in a common buffer in the sequence they will be sent. Another buffer contains the length of each of those packet. As tcpdump-files do not contain statistics about the packets it contains, it is necessary to iterate through the packets in the file and read their length to determine the size of the buffer used to store them. Once this is done, Stressnet allocates memory for the buffers. Then Stressnet iterates once again through the files to copy the packets into the buffers.

For each packet which has to be sent, there is a common buffer for all packets which contains its length, one which later will contain timing information and one which contains pointers to the begin of the packet considered.

Illustration 3 illustrates the buffers created.

4 ***GNU** Gengetopt,* http://www.gnu.org/software/gengetopt/gengetopt.html

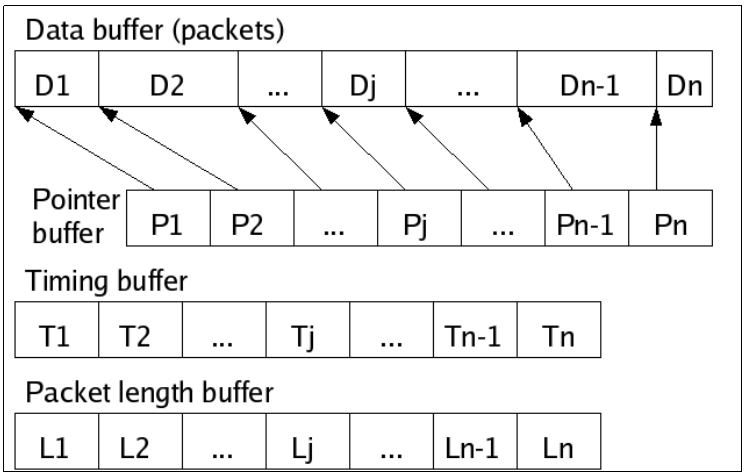


Illustration 3: Buffers created to contain information about packets. Illustrated with a total amount of n packets.

Once the buffers are created, they are locked into physical memory, to avoid swapping. This should lead to better performance, as the packets are directly read from physical memory.

2.b.4 Prepare the socket connection

As the datagrams to be sent are read including Ethernet header, there is a need to directly write them on the data link layer. For this purpose, we decided to use the sockets API with the PF_PACKET family and SOCK_RAW socket type. We could have used libnet, which tends to be a portable networking API, but using libnet may involve some overhead which is not present with the sockets API.

Once the socket is opened, it is bound to the interface given as argument. Thus, datagrams will be sent through this interface.

Note that on usual systems this only works if Stressnet is run as root.

2.b.5 Change the process scheduling policy and priority

Immediately before sending out the packets, Stressnet's scheduling policy and priority are set to FIFO (first-in first-out) and to the value given by the user.

FIFO is the real time scheduling policy proposed by the Linux operating system.

To change the scheduling priority, Stressnet must be run as root.

2.b.6 Send packets

If all packets were sent continuously, without interruption, the bit rate would be close to the maximum bit rate permitted on the line or would be limited by the sending computer's performance.

Thus the only way to send packets at different bit rates is to wait some little time before sending each of them. This time to wait depends on the size of the packet and increases linearly with the size of the packet.

Assume we have to send one packet of size L bits at a given bit rate D. Then the packet will have to be sent after a time T given by: $D = \frac{L}{T} \Leftrightarrow T = \frac{L}{D}$

Let's assume we have to send packets of size 1000, 1200, 400, 800 bytes at a bit rate of 70 Mbit/s.

Before sending packet 1, we have to wait $T_1 = \frac{1000*8}{70*10^6} = 114\,\mu s$

Then after packet 1 has been sent, we have to wait $T_2 = \frac{1200*8}{70*10^6} = 137\,\mu s$ before sending packet 2.

Before sending packet 3 we have to wait $T_3 = \frac{400*8}{70*10^6} = 45\,\mu s$.

And finally before sending packet 4 we have to wait $T_4 = \frac{800*8}{70*10^6} = 91\,\mu s$.

This implies that during the $387\,\mu s$ from the emission time of packet 1, we send 3400 bytes.

Let's verify: $\frac{3400*8}{387*10^{-6}} = 70.2\,Mbit/s$, which roughly corresponds to the desired bit rate.

Using the `signal()` system call to implement this is simply not applicable, as it cannot wait times smaller than 1 to 10 ms. Thus the decision we made was to implement a busy-waiting loop, which continuously asks for the system time until the time fixed to send the packet is reached. Busy-waiting means that the program does not give resources to other applications during waiting. Thus the system has to be dedicated to the application during the emission of the packets. This looks clumsy, but as we'll explain later, it's quite accurate enough for our purpose.

The system call used to send packets to the network has been chosen such that it does not drop packets if they are sent to the interface faster than the interface is able to send them to the network.

2.b.7 Terminate the program

Once the packets are all sent, the program unlocks buffers from physical memory, sets the scheduling priority to 0 and deallocates all buffers before exiting.

2.c Source code

The commented source code can be found at the end of this paper. This documentation was generated with Doxygen[5].

3 The complete framework

To be able to automate the tests being done with Stressnet, some more tools and shell scripts were necessary.

First of all, we used a program which is able to send or receive UDP packets to or from the network. This program was necessary to synchronize the applications throughout the whole tests.

The device which was tested was a PC with Snort installed on it. Thus this PC worked as an IDS device.

Let's illustrate the concepts.

5 *Doxygen*, http://www.stack.nl/~dimitri/doxygen/

PACKET GENERATOR PC	IDS PC

testsettings.sh:

```
#!/bin/bash
#This file contains the configuration settings of the tests.  This is the only file to
be modified through the tests.
#echo "Loading settings into memory."

export dumpdir="/home/yannick/dumps"

export stressnetdir="/home/yannick/Coding/eclipse_workspace/stressnet/Rp3"
export bitrates="10000 20000 30000 40000 50000 60000 70000 80000 90000 100000 110000
120000 140000 150000 160000 170000 180000 190000 200000"
export priority="0"
export fileM="$dumpdir/smtp_outside.dump"
export fileN="$dumpdir/UDP_port_zero.dump"
export interface="eth0"
export ratio="999"
export quantity="200000"
#DO NOT FORGET TO SET nbkbytes TO THE RIGHT VALUE!
export nbkbytes="65000"

export testdir="/tests"
export snortdir="/snort-2.3.0/src"
export sleepbeforekill="60"
```

export netbenchdir=”/netbench”

gen.sh:

```
#!/bin/bash
#This is the script for the generator.
Don't modify this. Change settings in
test_settings.sh

. test_settings.sh
for ratio in $ratio
do
  for ((repeat=1;repeat<=20;repeat++))
  do
    for bitrate in $bitrates
    do
      sleep 3
      echo "Bitrate: $bitrate kbit/s"

      $stressnetdir/stressnet -i$interface
-m$fileM -n$fileN -r$ratio -p$priority -q
$quantity -b$bitrate

      $netbenchdir/prec_raw_block -i eth0 >
/dev/null
      echo "Bitrate done: $bitrate kbit/s"
    done
  done
done
```

killsnort.sh:

```
#!/bin/bash
. test_settings.sh

#sleep $sleepbeforekill
sleep $* #the star * means that the
arguments of the calling script are passed
to this application (sleep)
echo "    ___killing all snort processes if
any___"
killall snort
```

parse.sh:

```
. test_settings.sh
export nofalerts=`less
$testdir/snortstderr.txt | grep ALERTS |
cut -d ' ' -f 2`
export nofreceived=`less
$testdir/snortstderr.txt | grep Snort\
received | cut -d ' ' -f 3`
export nofanalyzed=`less
$testdir/snortstderr.txt | grep Analyzed |
cut -d ' ' -f 6 | cut -d '(' -f 1`
export nofdropped=`less
$testdir/snortstderr.txt | grep Dropped |
cut -d ' ' -f 6 | cut -d '(' -f 1`
echo "$bitrate $nofreceived $nofalerts
$nofanalyzed $nofdropped" >> result.txt

echo "Don't forget to copy the results file
or to rename it, or it will be overwritten
during the next execution!"
```

ids.sh:

```
#!/bin/bash
. test_settings.sh
```

PACKET GENERATOR PC	IDS PC
	echo "___removing old results___" rm -f $testdir/snortstd*.txt rm -f $testdir/result.txt echo "___killing existing snort processes if any___" killall snort killall sleep killall bash\ killsnort.sh for ratio in $ratio do for ((repeat=1;repeat<=20;repeat++)) do for bitrate in $bitrates do echo "___Bitrate: $bitrate kbit/s___" export sleepbeforekill=`echo "8*$nbkbytes/$bitrate+20" | bc` #20 SHOULD be enough. If syncronization fails, try a higher value bash killsnort.sh $sleepbeforekill & sleep 3 echo "___Launch snort___" $snortdir/snort -c $snortdir/../etc/snort.conf -l $snortdir/../log -i eth0 >$testdir/snortstdout.txt 2>$testdir/snortstderr.txt echo "___Parsing snort output files___" . parse.sh echo "___Generate releasing packet___" ../netbench/pgen_raw -i eth0 -c 1 -s 5 -l 0 -d "ff:ff:ff:ff:ff:ff" -b 100000 done done done killall sleep killall bash\ killsnort.sh echo "___Finished the test.___"

Let's explain this:

The file `testsettings.sh` is the only file which has to be changed from one test to the other. It contains all required settings, like the directories in which the programs are, the different bit rates at which the tests have to be done or the amount of bytes that will be sent. The same version of the file `testsettings.sh` must be copied on both computers used to do the test, to avoid synchronization issues.

This amount has to be entered by hand and can be read during the first execution of Stressnet. Either one puts an amount big enough or one launches Stressnet with the option -s to read it and

then to enter it in the `testsettings.sh`.

This value is critical, as it's the key value for the timing. If you enter a too big value, the tests may last for a too long time, if the value is too short, the tests will loose synchronization and the values measured won't be valid for interpretation.

The script `killsnort.sh` does wait the time given to it as argument and then kills all processes named snort.

The script `parse.sh` parses Snort's output and puts the interesting values in the comma (here space) separated values file `result.txt`. The saved values are organized like this:

Bit rate	Received packets	Number of alerts	Number of analyzed	Number of dropped

The script `ids.sh` does kill all snort occurrences, then iterates through the various bit rates given as well as through the different ratios given. In every loop it launches `killsnort.sh` as a parallel process, then launches `snort`. `killsnort.sh` waits a certain time then kills snort. This unblocks `ids.sh` which then launches `parse.sh` and finally executes `netbench`, which sends an UDP packet to the generator to unblock it and permit the execution of the next loop (i.e. to launch the test for the next bit rate). The waiting time is calculated to be longer than the time the generator needs to send all its packets at the desired bitrate. Once

The script `gen.sh` iterates through the various bit rates given as well as through the different ratios given. In every loop it tries to send its packets at the desired bit rate and then waits for a UDP packet from the IDS PC. This UDP packet provokes the program Netbench to quit and to unblock the script, thus it provokes the script to continue through the next loop.

The following picture illustrates this algorithm:

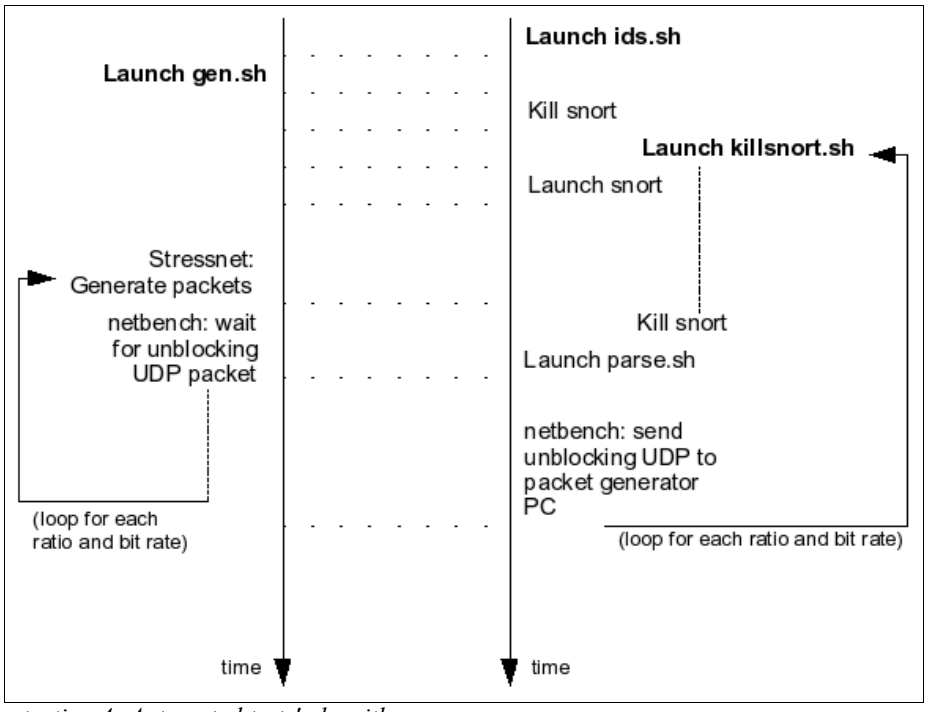


Illustration 4: Automated tests' algorithm

The tool Netbench is a general packet generator and reader developed by Major Mees and is not described in this paper. The only use of Netbench here is to synchronize the computers during the tests.

4 Implementation decisions

4.a Programming language

As the Linux source code is written in C language, it was obvious to use C or C++ for Stressnet. Nevertheless, a first implementation in C++ revealed to be too slow, probably because of the overhead induce by C++ dereferencing. Thus, the whole program has been rewritten in C language and is in fact much closer to our expectations. The difference was such that where the C code could generate 70 Mbit/s bit rate, the C++ code only reached about 7 Mbit/s!

Note that it would be sufficient to develop the program in C++, provided the critical loop – the one where sending packets occurs – is completely written in C and there happens the least possible dereferencing. This means that data access should be direct, even for class attributes. Those attributes which are to be access to in the sending loop should be accessed directly.

4.b API used to read files

As we decided to use tcpdump formatted files, it became obvious to use the libpcap[6] library to read those files.

4.c Alternative API to write to the data link

We had the possibility to use some other data link layer access API than the sockets API. Saying this, it was very attractive to use the libnet[7] library to ensure portability of data link layer access. But like for any other application, a trade has to be done between usability and performance. As our main interests were to achieve some high bit rates with a quite good accuracy, it was obvious that no solution which potentially generates overhead compared to the sockets API should be used. Nevertheless, this does not mean that libnet is not efficient. We simply didn't try to use it. Maybe some interesting work would be to evaluate the performance limitations of libnet and to compare them to the limits of the socket API.

5 Issues

5.a Slow bit rate – packets too small

When the packets to send are too small, the repeated calls to write() generate too much overhead and slows down the send bit rate. To avoid this, we found a solution at the very end of the training, thus there was no more time to implement it in Stressnet. Nevertheless we implemented it in a simple program to have an idea of the performance rise it could give us.

The solution is to use the system call `writev()` (or `sendmsg()`), which permits the application to pass more than one packet to the kernel using only one system call. The results we've found are astonishing and we highly recommend to implement an algorithm that uses this function if Stressnet is to be developed further.

It's worth saying that this also permits the application to reach much higher bit rates, as sending

6 *Libpcap*, http://www.tcpdump.org/
7 *Libnet*, http://libnet.sourceforge.net/

normal sized packets at very high bit rates or sending small packets at high bit rates do lead to the same issue.

6 Further possible improvements

- Port to other OS: This may be quite easy if libnet reveals to be fast enough. Don't forget that porting to different architectures may impact on speed and accuracy.
- Permit use of other networking devices if this is not possible using raw sockets;
- Optimization of reading files stats and copying packets into memory (this could be done simultaneously);
- Optimization of managing the packets which have to be hold in memory, for example load a given packet only once into memory, even if it has to be sent more often;
- Use of sendmsg() or writev() instead of write() or sendto();
- Extend stressnet to 2 applications which calculate TCP streams and the corresponding header values before sending , to be able to test 'real networks', one occurrence of stressnet on each side of the firewalls etc :
  stressnet1 – firewall, ids, dmz – private network with one computer having stressnet;
- Take into account the specific coding in the physical layer and the data link layer to calculate timing with the complete quantity of bits sent to the network, for example add the 4 bytes used for the FCS/CRC at the end of ethernet packets;
- Permit to set values in packets, like the MAC address or TCP/UDP ports;
- Permit to receive packets and to output some data to files.

7 The tests

This part will discuss the tests we've done and the results we've found. Once we had a working version of Stressnet, as it was quite late in the training, we immediately implemented some performance measurements of Snort, and we only did one test to evaluate the accuracy of Stressnet. Remember that the goal of this training was to establish how IDSs, particularly Snort, behave when they are under heavy attack. Only later we did some tests to measure the accuracy and reliability of Stressnet.

As Olivier Thonnard's work for his Master Thesis is to profile Snort, he already found some packets which generate a high overhead in Snort: the packets for which Snort uses the Perl Compatible Regular Expression (PCRE) library. The first test we did was to send SMTP packets with strings which have a rule in Snort needing PCRE. These packets all contained the triggering string repeatedly, such that the PCRE was called very often for each of them.

We also implemented a test using a file containing 3 standard HTTP sessions to compare it to the previously described PCRE-test.

Each time the second file used by Stressnet is a one packet file with an packet containing UDP zero signature, which is interpreted by Snort as an intrusion attempt.

The version of Snort that was used throughout the tests was Snort 2.3.0 for Linux. We installed Snort with the default out-of-the-box configuration and did not change this configuration.

1 Test 1: SMTP with PCRE

The bitrates tested were 10, 20, ..., 90 Mbit/s.

The ratio of normal packets over malicious packets was 999:1, thus 1/1000 packet was an attack.

The total quantity of packets sent is 200.000, and for each bit rate, the test was repeated 20 times.

Illustration 5 shows the results we obtained:

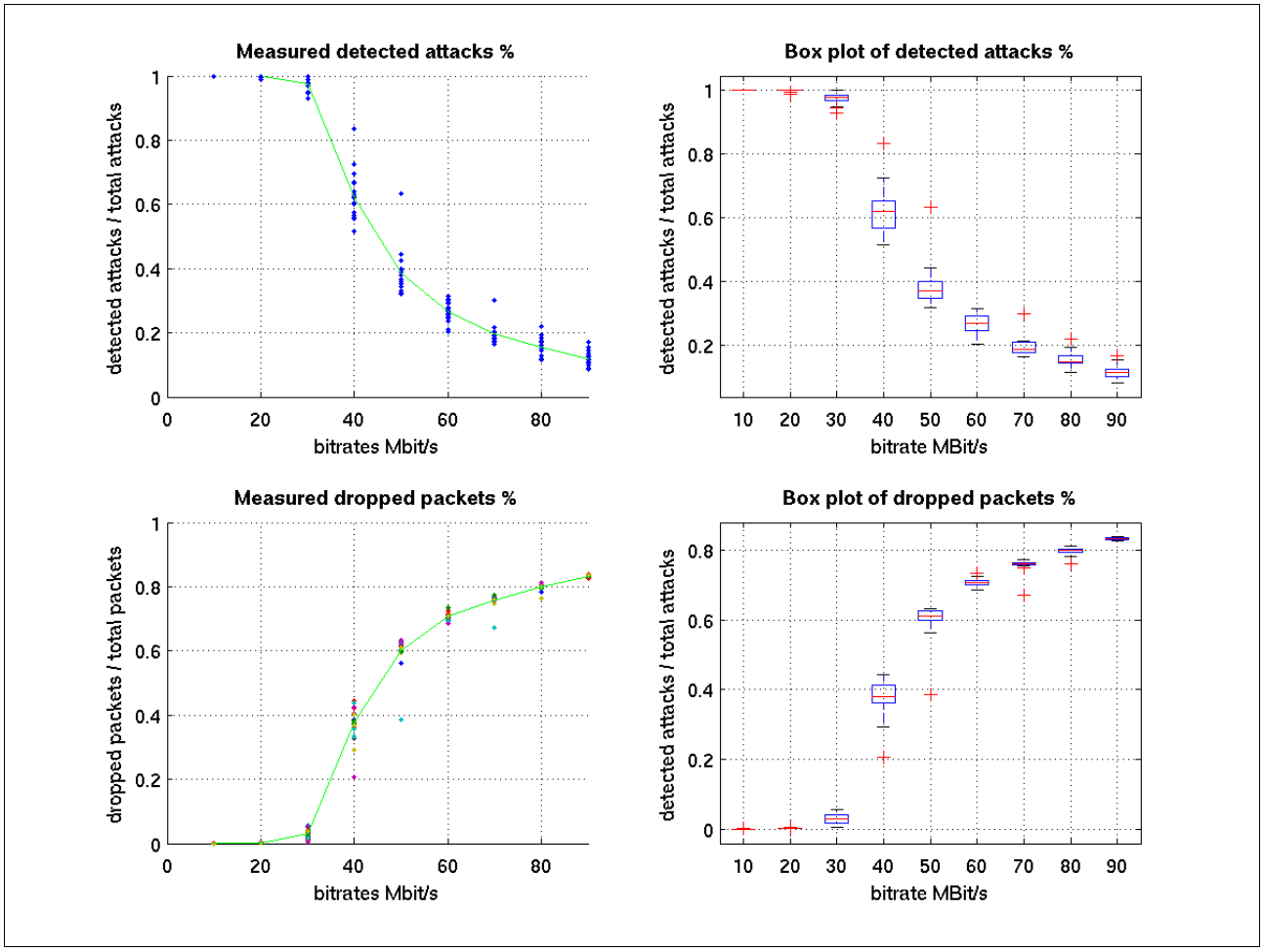


Illustration 5: Test results: 999 SMTP-PCRE : 1 UDP_ZERO

On both left figures the proportions of dropped packets and detected attacks respectively raises and drops quite fast once the bit rate is higher that 30 Mbit/s.

Both right figures show error box plots. The plot is drawn around the median value and its borders reach the 25th and 75th percentiles. The outer lines drawn show the 10th and 90th percentiles.

Don't forget that these values are only valid because Stressnet in fact is reliable at these bit rates.

It's obvious that even for common bit rates – in fact 35 Mbit/s is not so rare – Snort is not reliable any more. And remember: the whole overhead is generated by sane packets which only contain the trigger to launch the detection, but provoke no alert! Only the UDP packet contains an alert.

Now let's view the distribution of values at a fixed bit rate (Illustration 6). We see at bitrates 40, 50, 70, 80 and 90 Mbit/s that there is one outsider which corresponds to better attack detection. As it's quite far away from the median value, this distribution does not seem to behave like a Gaussian distribution. Thus we've tried to fit a Gaussian distribution to the $\log_{10}$ of our values, which tends to reduce the impact of the outsiders and takes them nearer to the median. Illustration 6 shows that even with the logarithmic values the distribution is not Gaussian, but rather a t-distribution. This fitting has been realized with the Matlab R14[8] fitting tool.

8 *MATLAB*, http://www.mathworks.com/

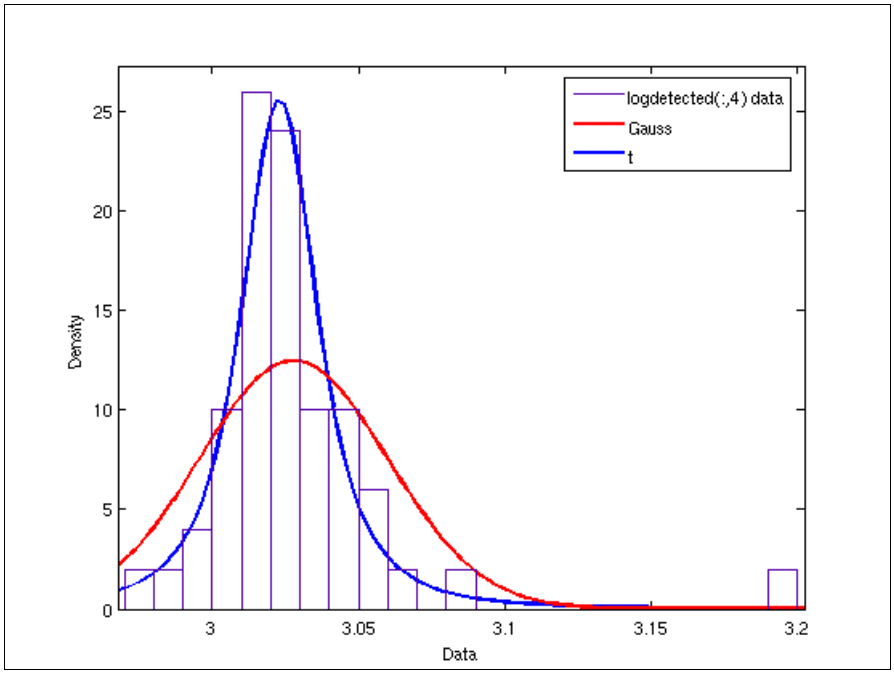


Illustration 6: Fitting of logarithmic values

As the Gaussian does not fit well, we conclude that the outsider value is *probably* not a pure accident (at this bit rate!).

As we find a distribution that is not Gaussian, we only show box error plots (Illustration 5) - not the standard deviation. If the distribution was Gaussian, we could have concluded that for a given bit rate, it was possible to guarantee that the percentage of detected packets would be higher a certain value with a probability of 0.98. Doing so would be an error! (For example, one could have been tempted to say that at 40 Mbit/s, there is a probability of 0.98 that the percentage of detected attacks is always higher than 55%, but this is not true!

2 Test 2: Standard complete HTTP sessions

The bit rates tested were 10, 20, ..., 90 Mbit/s.

The ratio of normal packets over malicious packets was 999:1, thus 1/1000 packet was an attack.

The total quantity of packets sent is 200.000, and for each bit rate, the test was repeated 20 times.

Illustration 7 shows the results we obtained:

It's obvious that at common bit rates, Snort performs well and finds almost every attack. This, compared to test 1, shows the impact of the pattern matching done with the PCRE library on Snort's performances. Thus there is a need to develop better algorithms, to implement some part of the pattern matching in hardware or to use a subset of PCRE patterns in Snort rules. But the last solution implies a trade between usability and performance.

As UDP needs less processing than HTTP (because it is stateless, thus it needs no reassembly preprocessing), it's not necessary to make tests with UDP packets.

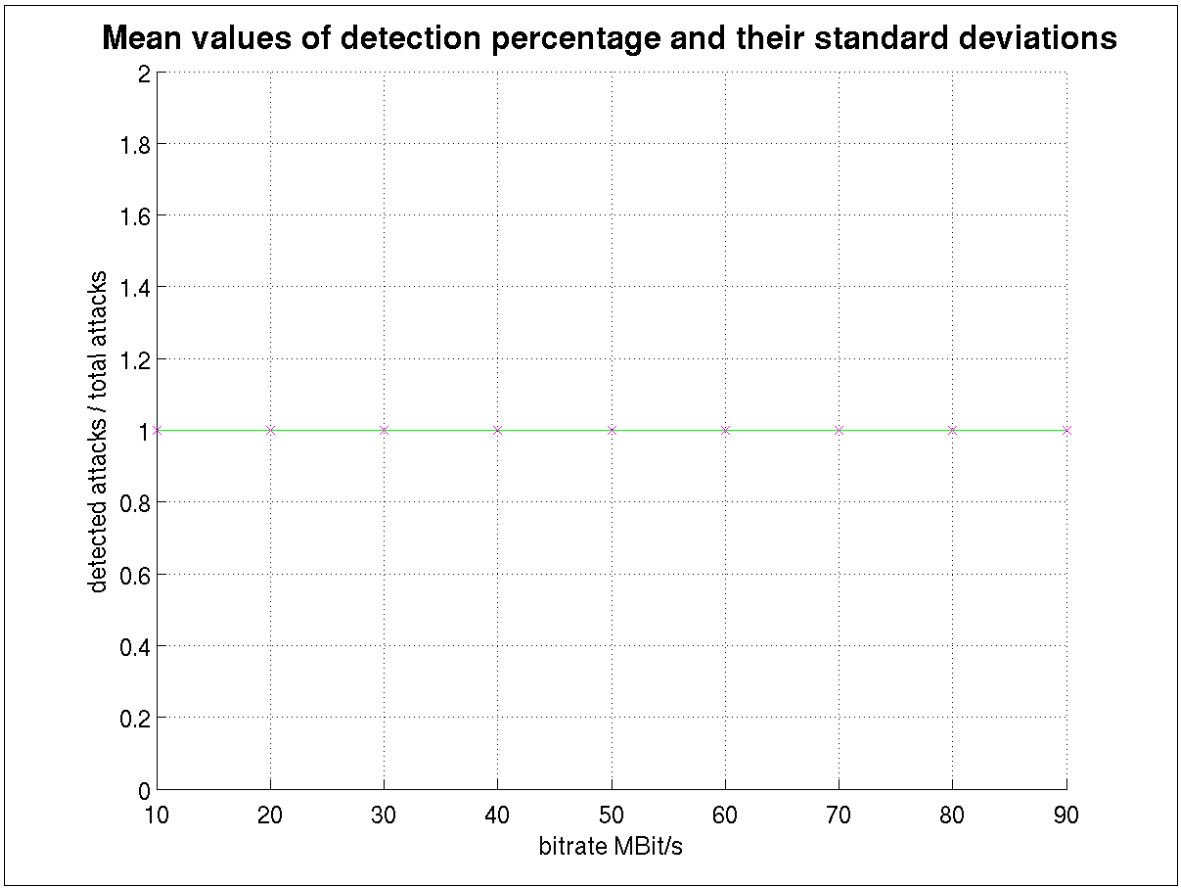


Illustration 7: Test results: 999 HTTP : 1 UDP_ZERO

3 Test 3: Stressnet's reliability

We finally made one test using the file with the three HTTP sessions we used in the previous test to estimate the reliability of Stressnet. This test has been done for bit rates from 5 to 150.000 Mbit/s, with steps of 5 Mbit.

Illustration 8 shows the results of this test.

The first picture shows the relative difference between the desired bit rate and the obtained bit rate $D_{rel} = \frac{B_{desired} - B_{measured}}{B_{desired}}$. For each bit rate, the test has been done 20 times and the $B_{measured}$ used in the formula is the mean measured bit rate for a given desired bit rate.

We can see that the absolute value of this difference reaches 10% at bit rates close to 130 Mbit/s and higher.

We also see that the absolute value of this difference seems to rise up quite fast once the bit rate is over 125 Mbit/s.

The lower picture shows a plot of the desired bit rate (line) against the measured bit rate.

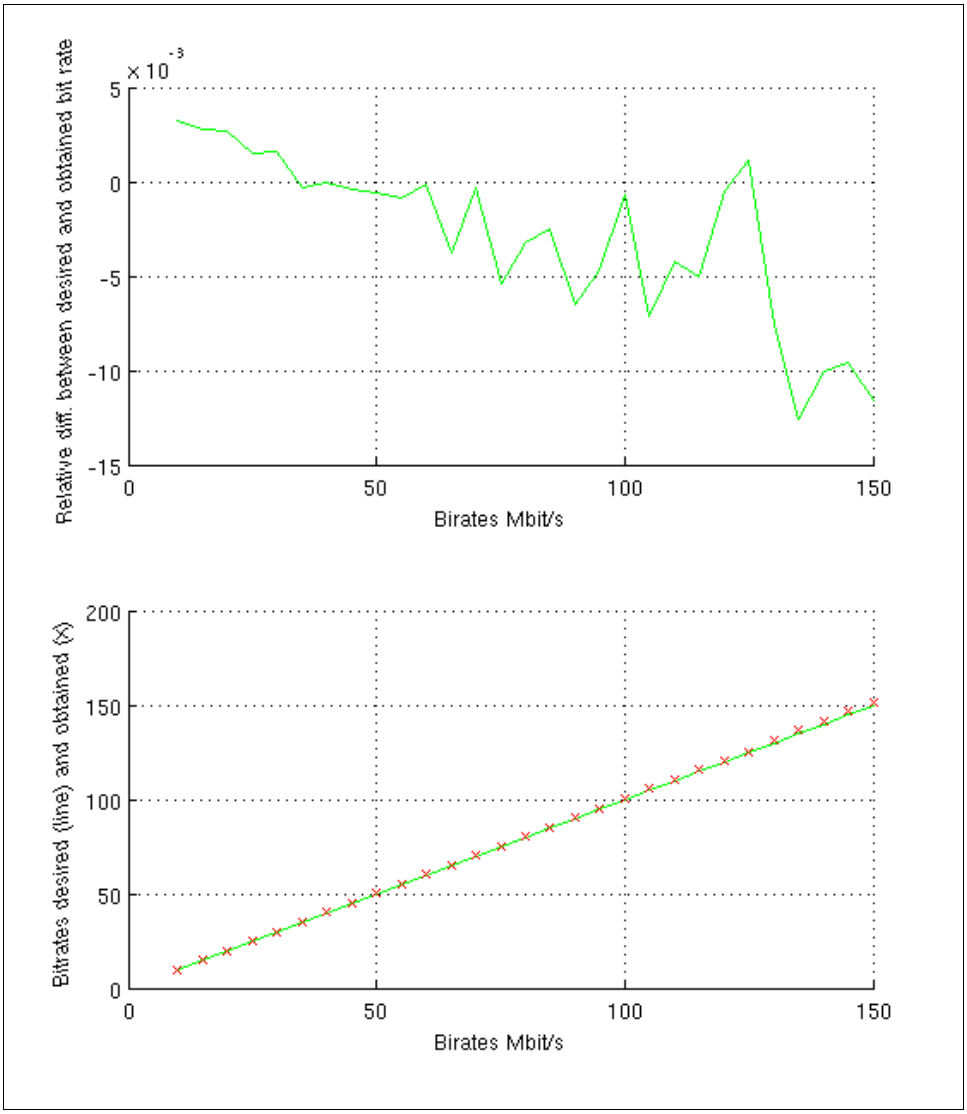


Illustration 8: Stressnet's reliability test: HTTP packets

8 Conclusion

Concerning the project of developing a framework to test IDSs, we've got a tool that permits to test reliably devices at bit rates up to 130 Mbit/s and we've shown how to use it. In fact, we have found that Snort is not reliable when the rules containing calls to the PCRE library are enabled.

But more than having a tool that already works, we now have the basis knowledge and experience to further develop quite better tools to test various network devices at bit rates much higher than 130 Mbit/s.

We also have basic examples about how to interpret results found using such benchmarking tools.

Concerning the goals of the training, it permitted me to actively participate to the development of a Unix-like command line tool using the sockets networking API and various Linux system calls. I also did have to think about how to validate this program and thus to find out how to optimize the code to ensure reliability.

To reach these goals, the first work to do, and this is probably the most important part, was to read a lot of documentation about Linux network programming and about general networking – including protocols and security issues. This took a lot of time, but once most ideas have been assimilated, it became easy for me to define efficient requirements and to develop the tools that were be needed.

Probably the most important thing I learned during the training happened at the end of the training, when exchange of ideas and discussions with both the Major Mees and the Captain Thonnard permitted to see that sharing information is important as the whole team benefits from discussed topics.

Now that basic work has been done and that basic knowledge about reliable high speed network programming has been collected, all doors are opened to design and implement a high performance tool to test various network devices.

Appendixes

1 Custom Kernel[9]

The Linux kernel is the heart of Linux operating systems. It's the place where memory management, device access and program scheduling takes place. The Linux kernel is based on a monolithic design – this means that it's mainly composed of one principal bloc. Nevertheless it is possible to extend the kernel with external modules. Modern Linux distributions often enable almost every part of the kernel – which means that it is often bigger than needed and that unnecessary modules might be loaded.

The fact that the Linux kernel is open source permits everyone to tailor the kernel to his needs. For example, someone who doesn't use USB devices can completely disable USB support from the kernel.

The kernel source code distributions (*http://www.kernel.org*) do provide a tool to configure which parts to enable or to disable. The configuration is saved into a text file and is read during compilation of the kernel.

The performance achieved by Stressnet is very sensitive to miscellaneous interrupts due to device support or program scheduling. In fact, when used to send little datagrams at high speeds, Stressnet produces a big overhead due to processing and the write() or sendto() system call. To avoid concurrency between Stressnet and other applications, we decided to rebuild a minimalist kernel with only the features necessary for our job.

The kernel permits us to configure a large amount of features, so it would not be surprising to find smaller configurations than those we used. A very critical point here is to which extent the user knows his hardware and knows which hardware features he'll needs. We did in fact try multiple configurations and finally kept the one we preferred. It's also worth saying that not every build worked and gave us a bootable configuration.

Let's explain the steps we went through to configure, build and launch our minimalist kernel.

For our tests, we used SuSE Linux 9.2 Professional, which comes with Linux kernel 2.6.8-24.

1. The first step is to install the source code corresponding to the kernel version we wanted to compile.
   Either you download it from a http://www.kernel.org in form of a tarball, or you install it using the package management tool your distribution provides. It will usually be installed in `/usr/src/linux/` or `/usr/src/linux-source-2.6.8-24/`.
2. Now we'll configure the `/etc/inittab` file to enable booting by default till runlevel 3 (usually the runlevel with networking and multiuser enabled – thus no graphical user

9 *Linux Kernel*, http://www.kernel.org

interface as is in runlevel 5). Note that this is not working on Ubuntu, which defaults multiuser and networking on runlevel 2 and seems to obsolete the higher runlevels. Replace the lines:

```
# The default runlevel.
id:5:initdefault:
```

With:

```
# The default runlevel.
id:3:initdefault:
```

3. Now we'll deactivate in runlevel 3 all services we don't need for our application. For this we use the `runlevel config editor` provided with Yast (SuSE Linux) (in Redhat we can use the command-line tool `chkconfig`). There we'll disable almost everything except networking and keyboard key arrangement.
4. So far it's time to reboot and to check wether we're right till here. After reboot, the system boots till runlevel 3. To change back to runlevel 5 and to X, we need to enter as root the command `/sbin/init 5`. Again, when we are in runlevel 5 and want to go back to runlevel 3, there is no need to reboot, simply type as root: `/sbin/init 3`.
5. In X, we open as root the file `/etc/inittab` and disable 4 tty terminals from the 6 provided (the reason is still to keep resources for our application). To activate the new inittab file, simply enter `/sbin/init q` as root.
6. The file `/etc/modules.conf` permits us to disable some modules that may not be necessary.
7. Now let's configure the kernel. In a terminal, as root, change to the source directory `/usr/src/linux/` and launch `make menuconfig` or `make xconfig` if in command-line or X respectively.
8. Now go through the tree and disable or enable features. Enabling can be done in two ways: either statically loaded in the kernel or dynamically loaded as a module. Once you've done your work, go to General Setup and add a local version (or extra version) to avoid overwriting the default compiled modules and kernel files with yours. We used version number 27.
9. Then save your configuration as `.config`, quit and launch: `make && make modules_install` (note that this is only valid for kernel 2.6+). After a while, depending on your hardware and the number of features you've enabled, the compilation is ended. Now you still need some steps till being able to boot your new kernel (which of course will function immediately...).
10. If needed, let's create the initial ramdisk file: `initrd`. This file is needed to load external modules needed to enable booting the computer. For example, if you use reiserfs partitions but you didn't enable the reiserfs support into the kernel but as a module, you'll need to load this module before booting. This is what the `initrd` is meant to do. We use the application `mkinitrd` to create this initial ramdisk file:

    ```
    mkinitrd -k /boot/kernel-2.6.8-27 -i /boot/initrd-2.6.8-27
    ```

11. Now we need to copy the generated files to the /boot directory:

    ```
    cp /usr/src/linux-source-2.6.10/arch/i386/boot/bzImage /boot/kernel-
    2.6.10-27

    cp /usr/src/linux-source-2.6.10/System.map /boot/System.map-2.6.10-27

    cp /usr/src/linux-source-2.6.10/.config /boot/config-2.6.10-27
    ```

12. Now we only need to edit the Grub configuration file. Grub is the bootloader installed by default with SuSE Linux 9.2 Professional. Other distributions may provide Lilo instead of Grub. In /boot/grub/menu.lst, add the following lines:

```
title        SuSE Linux 9.2 Pro MINIMALIST
root         (hd0,4)
kernel       /boot/kernel-2.6.8-27 root=/dev/hda5 ro quiet splash
initrd       /boot/initrd-2.6.8-27
savedefault
boot
```

13. Now you're ready to reboot and try your new kernel. Reboot your computer and in the boot menu, select the newly added menu option.
14. Now you've done all these steps one, you've got three possibilities:
    1. It works and you're satisfied
    2. It works, but you still want a finer tuned kernel. Redo these steps and change the configuration.
    3. It doesn't work, the screen shows kernel panic, there is no console or whatever else: reset your computer, boot to the default kernel and spend some more time with configuring and building again the kernel.

2 Various functions and system calls used in Stressnet

All the functions described here are completely specified on http://www.opengroup.org[10]. The following descriptions contain information from this site and is sometimes completed with comments about their use in Stressnet.

2.a assert()

```
#include <assert.h>
```

The <*assert.h*> header defines the *assert()* macro. It refers to the macro *NDEBUG* which is not defined in the header. If *NDEBUG* is defined as a macro name before the inclusion of this header, the *assert()* macro is defined simply as:

```
#define assert(ignore)((void) 0)
```

otherwise the macro behaves as described in *assert()*.

The *assert()* macro is implemented as a macro, not as a function. If the macro definition is suppressed in order to access an actual function, the behaviour is undefined.

2.b bind()

```
#include <sys/socket.h>
int bind(int socket, const struct sockaddr *address,
     socklen_t address_len);
```

The *bind()* function assigns an *address* to an unnamed socket. Sockets created with *socket()* function are initially unnamed; they are identified only by their address family.

The socket in use may require the process to have appropriate privileges to use the *bind()* function.

10 The Open Group, http://www.opengroup.org

2.c close()

```
#include <unistd.h>

int close(int fildes);
```

The *close()* function will deallocate the file descriptor indicated by *fildes*. To deallocate means to make the file descriptor available for return by subsequent calls to *open()* or other functions that allocate file descriptors. All outstanding record locks owned by the process on the file associated with the file descriptor will be removed (that is, unlocked).

2.d div()

```
#include <stdlib.h>

div_t div(int numer, int denom);
```

The *div()* function computes the quotient and remainder of the division of the numerator *numer* by the denominator *denom*. If the division is inexact, the resulting quotient is the integer of lesser magnitude that is the nearest to the algebraic quotient. If the result cannot be represented, the behavior is undefined; otherwise, *quot* * *denom* + *rem* will equal *numer*.

Remark : This function is used here because it produces less instructions than the integer division with the operator '/'.

2.e gettimeofday()

```
#include <sys/time.h>

int gettimeofday(struct timeval *tp, void *tzp);
```

The *gettimeofday()* function obtains the current time, expressed as seconds and microseconds since 00:00 Coordinated Universal Time (UTC), January 1, 1970, and stores it in the **timeval** structure pointed to by *tp*. The resolution of the system clock is unspecified.

If *tzp* is not a null pointer, the behavior is unspecified.

2.f mlock()

```
#include <sys/mman.h>

int mlock(const void * addr, size_t len);

int munlock(const void * addr, size_t len);
```

The function *mlock()* causes those whole pages containing any part of the address space of the process starting at address *addr* and continuing for *len* bytes to be memory resident until unlocked or until the process exits or *exec*s another process image. The implementation may require that *addr* be a multiple of {PAGESIZE}.

The function *munlock()* unlocks those whole pages containing any part of the address space of the process starting at address *addr* and continuing for *len* bytes, regardless of how many times *mlock()* has been called by the process for any of the pages in the specified range. The implementation may require that *addr* be a multiple of the {PAGESIZE}.

If any of the pages in the range specified to a call to *munlock()* are also mapped into the address spaces of other processes, any locks established on those pages by another process are unaffected by the call of this process to *munlock()*. If any of the pages in the range specified by a call to *munlock()* are also mapped into other portions of the address space of the calling process outside the range specified, any locks established on those pages via the other mappings are also unaffected by this call.

Upon successful return from *mlock()*, pages in the specified range will be locked and memory resident. Upon successful return from *munlock()*, pages in the specified range will be unlocked with

respect to the address space of the process. Memory residency of unlocked pages is unspecified.

The appropriate privilege is required to lock process memory with *mlock()*.

Remark : The use of this function in stressnet permits to keep in physical memory the buffers which are used in the critical loop, to avoid the overhead due to page swapping.

2.g munlock()

```
#include <sys/mman.h>
int munlock(const void * addr, size_t len);
```

Refer to *mlock()* for a description.

2.h sendto()

```
#include <sys/socket.h>
ssize_t sendto(int socket, const void *message, size_t length, int flags,
      const struct sockaddr *dest_addr, socklen_t dest_len);
```

The *sendto()* function sends a message through a connection-mode or connectionless-mode socket. If the socket is connectionless-mode, the message will be sent to the address specified by *dest_addr*. If the socket is connection-mode, *dest_addr* is ignored.

2.i setsockopt()

```
#include <sys/socket.h>
int setsockopt(int socket, int level, int option_name, const void
                *option_value, socklen_t option_len);
```

The *setsockopt()* function sets the option specified by the *option_name* argument, at the protocol level specified by the *level* argument, to the value pointed to by the *option_value* argument for the socket associated with the file descriptor specified by the *socket* argument.

Remark : This function is used to allocate a larger buffer size to the socket.

2.j socket()

```
#include <sys/socket.h>
int socket(int domain, int type, int protocol);
```

The *socket()* function creates an unbound socket in a communications domain, and returns a file descriptor that can be used in later function calls that operate on sockets.

The process may need to have appropriate privileges to use the *socket()* function or to create some sockets.

2.k write()

```
#include <unistd.h>
ssize_t write(int fildes, const void *buf, size_t nbyte);
ssize_t pwrite(int fildes, const void *buf, size_t nbyte,
    off_t offset);
#include <sys/uio.h>
```

```
ssize_t writev(int fildes, const struct iovec *iov, int iovcnt);
```

The *write()* function attempts to write *nbyte* bytes from the buffer pointed to by *buf* to the file associated with the open file descriptor, *fildes*.

Remark : This function can be used with any file descriptor, thus, it also works with socket file descriptors.

References and bibliography

3 Linux

Andrew Tanenbaum, *Réseaux, 4ème édition*, Pearson Education, ISBN 2-7440-7001-7

Andrew Tanenbaum, *Systèmes d'exploitation, 2ème édition*, Pearson Education, ISBN 2-7440-7002-5

Various Quick References, http://www.digilife.be/quickreferences/quickrefs.htm

CodeSourcery LLC, *Advanced Linux Programming*, New Riders Publishing, http://www.advancedlinuxprogramming.com/.

The Linux Documentation Project, http://www.tldp.org/

Linux Online, http://www.linux.org/

The Open Group, http://www.opengroup.org/

4 Bash Shell

Bash Programming, Introduction, http://www.tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html

5 C

Peter Aitken & Bradley L. Jones, *Le langage C*, Campus Press, ISBN 2-7440-0838-9

Claude Delannoy, *Programmer en Langage C*, Eyrolles, ISBN 2-212-08985-6

How to Optimize C/C++ Source – Performance Programming, Borland Developer Network, http://www.borland.com

AMD AthlonTM Processor x86 Code Optimization Guide, Advanced Micro Devices, http://www.amd.com

6 C++

Claude Delannoy, *Programmer en Langage C++*, Eyrolles, ISBN 2-212-09138-9

Bruce Eckel, *Thinking in C++*, Prentice Hall, ISBN 0-13-917709-4

7 Kernel

Linux Kernel, http://www.kernel.org

Kwan Lowe, *Kernel Rebuild Guide*, Digital Hermit, http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html

8 Network programming

W. Richard Stevens, *UNIX Network Programming, Volume 1, Second Edition*, Prentice Hall, ISBN 0-13-490012-X

Libpcap, http://www.tcpdump.org/

Libnet, http://libnet.sourceforge.net/

9 IDS - Snort

Snort Users Manual, http://www.snort.org

Rebecca Bace and Peter Mell, *Intrusion Detection Systems*, NIST Special Publication on Intrusion Detection Systems

Marcus J. Ranum, *Experiences Benchmarking Intrusion Detection Systems*, NFR Security, http://www.nfr.com

10 Doxygen

Doxygen, http://www.stack.nl/~dimitri/doxygen/

Custom kernel .config file

```
#
# Automatically generated make config: don't edit
# Linux kernel version: 2.6.8--
# Wed Jun  1 18:59:22 2005
#
CONFIG_X86=y
CONFIG_MMU=y
CONFIG_UID16=y
CONFIG_GENERIC_ISA_DMA=y

#
# Code maturity level options
#
CONFIG_EXPERIMENTAL=y
CONFIG_CLEAN_COMPILE=y
CONFIG_BROKEN_ON_SMP=y

#
# General setup
#
CONFIG_LOCALVERSION=""
# CONFIG_SWAP is not set
CONFIG_SYSVIPC=y
CONFIG_POSIX_MQUEUE=y
CONFIG_BSD_PROCESS_ACCT=y
CONFIG_BSD_PROCESS_ACCT_V3=y
CONFIG_SYSCTL=y
CONFIG_AUDIT=y
CONFIG_AUDITSYSCALL=y
CONFIG_LOG_BUF_SHIFT=14
CONFIG_HOTPLUG=y
CONFIG_IKCONFIG=y
CONFIG_IKCONFIG_PROC=y
# CONFIG_EMBEDDED is not set
CONFIG_KALLSYMS=y
# CONFIG_KALLSYMS_EXTRA_PASS is not set
CONFIG_FUTEX=y
CONFIG_EPOLL=y
CONFIG_IOSCHED_NOOP=y
CONFIG_IOSCHED_AS=y
CONFIG_IOSCHED_DEADLINE=y
CONFIG_IOSCHED_CFQ=y
# CONFIG_CC_OPTIMIZE_FOR_SIZE is not set
CONFIG_SHMEM=y
# CONFIG_TINY_SHMEM is not set

#
# Loadable module support
#
CONFIG_MODULES=y
```

```
CONFIG_MODULE_UNLOAD=y
CONFIG_MODULE_FORCE_UNLOAD=y
CONFIG_OBSOLETE_MODPARM=y
CONFIG_MODVERSIONS=y
CONFIG_KMOD=y

#
# Processor type and features
#
CONFIG_X86_PC=y
# CONFIG_X86_ELAN is not set
# CONFIG_X86_VOYAGER is not set
# CONFIG_X86_NUMAQ is not set
# CONFIG_X86_SUMMIT is not set
# CONFIG_X86_BIGSMP is not set
# CONFIG_X86_VISWS is not set
# CONFIG_X86_GENERICARCH is not set
# CONFIG_X86_ES7000 is not set
# CONFIG_M386 is not set
# CONFIG_M486 is not set
# CONFIG_M586 is not set
# CONFIG_M586TSC is not set
# CONFIG_M586MMX is not set
# CONFIG_M686 is not set
# CONFIG_MPENTIUMII is not set
# CONFIG_MPENTIUMIII is not set
# CONFIG_MPENTIUMM is not set
CONFIG_MPENTIUM4=y
# CONFIG_MK6 is not set
# CONFIG_MK7 is not set
# CONFIG_MK8 is not set
# CONFIG_MCRUSOE is not set
# CONFIG_MWINCHIPC6 is not set
# CONFIG_MWINCHIP2 is not set
# CONFIG_MWINCHIP3D is not set
# CONFIG_MCYRIXIII is not set
# CONFIG_MVIAC3_2 is not set
CONFIG_X86_GENERIC=y
CONFIG_X86_CMPXCHG=y
CONFIG_X86_XADD=y
CONFIG_X86_L1_CACHE_SHIFT=7
CONFIG_RWSEM_XCHGADD_ALGORITHM=y
CONFIG_X86_WP_WORKS_OK=y
CONFIG_X86_INVLPG=y
CONFIG_X86_BSWAP=y
CONFIG_X86_POPAD_OK=y
CONFIG_X86_GOOD_APIC=y
CONFIG_X86_INTEL_USERCOPY=y
CONFIG_X86_USE_PPRO_CHECKSUM=y
# CONFIG_HPET_TIMER is not set
# CONFIG_SMP is not set
# CONFIG_PREEMPT is not set
CONFIG_X86_UP_APIC=y
CONFIG_X86_UP_IOAPIC=y
CONFIG_X86_LOCAL_APIC=y
CONFIG_X86_IO_APIC=y
CONFIG_X86_TSC=y
CONFIG_X86_MCE=y
# CONFIG_X86_MCE_NONFATAL is not set
CONFIG_X86_MCE_P4THERMAL=y
CONFIG_TOSHIBA=m
CONFIG_I8K=m
CONFIG_MICROCODE=m
CONFIG_X86_MSR=m
```

```
CONFIG_X86_CPUID=m

#
# Firmware Drivers
#
# CONFIG_EDD is not set
# CONFIG_NOHIGHMEM is not set
CONFIG_HIGHMEM4G=y
# CONFIG_HIGHMEM64G is not set
CONFIG_HIGHMEM=y
CONFIG_PROC_MM=y
CONFIG_HIGHPTE=y
# CONFIG_MATH_EMULATION is not set
CONFIG_MTRR=y
CONFIG_REGPARM=y

#
# Power management options (ACPI, APM)
#
# CONFIG_PM is not set
# CONFIG_PM_DEBUG is not set

#
# ACPI (Advanced Configuration and Power Interface) Support
#
# CONFIG_ACPI is not set

#
# CPU Frequency scaling
#
# CONFIG_CPU_FREQ is not set

#
# Bus options (PCI, PCMCIA, EISA, MCA, ISA)
#
CONFIG_PCI=y
# CONFIG_PCI_GOBIOS is not set
# CONFIG_PCI_GOMMCONFIG is not set
# CONFIG_PCI_GODIRECT is not set
CONFIG_PCI_GOANY=y
CONFIG_PCI_BIOS=y
CONFIG_PCI_DIRECT=y
CONFIG_PCI_MSI=y
# CONFIG_PCI_LEGACY_PROC is not set
# CONFIG_PCI_NAMES is not set
# CONFIG_ISA is not set
# CONFIG_MCA is not set
# CONFIG_SCx200 is not set

#
# PCMCIA/CardBus support
#
# CONFIG_PCMCIA is not set

#
# PCI Hotplug Support
#
# CONFIG_HOTPLUG_PCI is not set

#
# Executable file formats
#
CONFIG_BINFMT_ELF=y
# CONFIG_BINFMT_AOUT is not set
```

```
# CONFIG_BINFMT_MISC is not set

#
# Device Drivers
#

#
# Generic Driver Options
#
# CONFIG_STANDALONE is not set
CONFIG_PREVENT_FIRMWARE_BUILD=y
CONFIG_FW_LOADER=m

#
# Memory Technology Devices (MTD)
#
# CONFIG_MTD is not set

#
# Parallel port support
#
# CONFIG_PARPORT is not set

#
# Plug and Play support
#

#
# Block devices
#
# CONFIG_BLK_DEV_FD is not set
# CONFIG_BLK_CPQ_DA is not set
# CONFIG_BLK_CPQ_CISS_DA is not set
# CONFIG_BLK_DEV_DAC960 is not set
# CONFIG_BLK_DEV_UMEM is not set
CONFIG_BLK_DEV_LOOP=y
CONFIG_BLK_DEV_CRYPTOLOOP=m
# CONFIG_BLK_DEV_NBD is not set
# CONFIG_BLK_DEV_SX8 is not set
CONFIG_BLK_DEV_RAM=y
CONFIG_BLK_DEV_RAM_SIZE=64000
CONFIG_BLK_DEV_INITRD=y
# CONFIG_LBD is not set
# CONFIG_CIPHER_TWOFISH is not set

#
# ATA/ATAPI/MFM/RLL support
#
CONFIG_IDE=y
CONFIG_BLK_DEV_IDE=y

#
# Please see Documentation/ide.txt for help/info on IDE drives
#
# CONFIG_BLK_DEV_IDE_SATA is not set
# CONFIG_BLK_DEV_HD_IDE is not set
CONFIG_BLK_DEV_IDEDISK=y
CONFIG_IDEDISK_MULTI_MODE=y
CONFIG_BLK_DEV_IDECD=m
CONFIG_BLK_DEV_IDETAPE=m
CONFIG_BLK_DEV_IDEFLOPPY=y
CONFIG_BLK_DEV_IDESCSI=m
# CONFIG_IDE_TASK_IOCTL is not set
# CONFIG_IDE_TASKFILE_IO is not set
```

```
#
# IDE chipset support/bugfixes
#
CONFIG_IDE_GENERIC=y
CONFIG_BLK_DEV_CMD640=y
CONFIG_BLK_DEV_CMD640_ENHANCED=y
CONFIG_BLK_DEV_IDEPCI=y
CONFIG_IDEPCI_SHARE_IRQ=y
CONFIG_BLK_DEV_OFFBOARD=y
CONFIG_BLK_DEV_GENERIC=y
CONFIG_BLK_DEV_OPTI621=y
CONFIG_BLK_DEV_RZ1000=y
CONFIG_BLK_DEV_IDEDMA_PCI=y
# CONFIG_BLK_DEV_IDEDMA_FORCED is not set
CONFIG_IDEDMA_PCI_AUTO=y
CONFIG_IDEDMA_ONLYDISK=y
CONFIG_BLK_DEV_ADMA=y
CONFIG_BLK_DEV_AEC62XX=y
CONFIG_BLK_DEV_ALI15X3=y
# CONFIG_WDC_ALI15X3 is not set
CONFIG_BLK_DEV_AMD74XX=y
CONFIG_BLK_DEV_ATIIXP=y
CONFIG_BLK_DEV_CMD64X=y
CONFIG_BLK_DEV_TRIFLEX=y
CONFIG_BLK_DEV_CY82C693=y
CONFIG_BLK_DEV_CS5520=m
CONFIG_BLK_DEV_CS5530=m
CONFIG_BLK_DEV_HPT34X=y
CONFIG_HPT34X_AUTODMA=y
CONFIG_BLK_DEV_HPT366=y
CONFIG_BLK_DEV_SC1200=y
CONFIG_BLK_DEV_PIIX=y
CONFIG_BLK_DEV_NS87415=y
CONFIG_BLK_DEV_PDC202XX_OLD=y
CONFIG_PDC202XX_BURST=y
CONFIG_BLK_DEV_PDC202XX_NEW=y
CONFIG_PDC202XX_FORCE=y
CONFIG_BLK_DEV_SVWKS=y
CONFIG_BLK_DEV_SIIMAGE=y
CONFIG_BLK_DEV_SIS5513=y
CONFIG_BLK_DEV_SLC90E66=y
CONFIG_BLK_DEV_TRM290=y
CONFIG_BLK_DEV_VIA82CXXX=y
# CONFIG_IDE_ARM is not set
CONFIG_BLK_DEV_IDEDMA=y
# CONFIG_IDEDMA_IVB is not set
CONFIG_IDEDMA_AUTO=y
# CONFIG_BLK_DEV_HD is not set

#
# SCSI device support
#
CONFIG_SCSI=m
CONFIG_SCSI_PROC_FS=y

#
# SCSI support type (disk, tape, CD-ROM)
#
CONFIG_BLK_DEV_SD=m
CONFIG_CHR_DEV_ST=m
CONFIG_CHR_DEV_OSST=m
CONFIG_BLK_DEV_SR=m
# CONFIG_BLK_DEV_SR_VENDOR is not set
```

```
CONFIG_CHR_DEV_SG=m
CONFIG_CHR_DEV_SCH=m

#
# Some SCSI devices (e.g. CD jukebox) support multiple LUNs
#
CONFIG_SCSI_MULTI_LUN=y
# CONFIG_SCSI_CONSTANTS is not set
# CONFIG_SCSI_LOGGING is not set

#
# SCSI Transport Attributes
#
# CONFIG_SCSI_SPI_ATTRS is not set
# CONFIG_SCSI_FC_ATTRS is not set

#
# SCSI low-level drivers
#
# CONFIG_BLK_DEV_3W_XXXX_RAID is not set
# CONFIG_SCSI_3W_9XXX is not set
# CONFIG_SCSI_ACARD is not set
# CONFIG_SCSI_AACRAID is not set
# CONFIG_SCSI_AIC7XXX is not set
# CONFIG_SCSI_AIC7XXX_OLD is not set
# CONFIG_SCSI_AIC79XX is not set
# CONFIG_SCSI_DPT_I2O is not set
# CONFIG_MEGARAID_NEWGEN is not set
# CONFIG_MEGARAID_LEGACY is not set
# CONFIG_SCSI_SATA is not set
# CONFIG_SCSI_BUSLOGIC is not set
# CONFIG_SCSI_DMX3191D is not set
# CONFIG_SCSI_EATA is not set
# CONFIG_SCSI_EATA_PIO is not set
# CONFIG_SCSI_FUTURE_DOMAIN is not set
# CONFIG_SCSI_GDTH is not set
# CONFIG_SCSI_IPS is not set
# CONFIG_SCSI_INIA100 is not set
# CONFIG_SCSI_SYM53C8XX_2 is not set
# CONFIG_SCSI_LPFC is not set
# CONFIG_SCSI_IPR is not set
# CONFIG_SCSI_QLOGIC_ISP is not set
# CONFIG_SCSI_QLOGIC_FC is not set
# CONFIG_SCSI_QLOGIC_1280 is not set
CONFIG_SCSI_QLA2XXX=m
# CONFIG_SCSI_QLA21XX is not set
# CONFIG_SCSI_QLA22XX is not set
# CONFIG_SCSI_QLA2300 is not set
# CONFIG_SCSI_QLA2322 is not set
# CONFIG_SCSI_QLA6312 is not set
# CONFIG_SCSI_QLA6322 is not set
# CONFIG_SCSI_DC395x is not set
# CONFIG_SCSI_DC390T is not set
# CONFIG_SCSI_NSP32 is not set
# CONFIG_SCSI_DEBUG is not set

#
# Multi-device support (RAID and LVM)
#
# CONFIG_MD is not set

#
# Fusion MPT device support
#
```

```
# CONFIG_FUSION is not set

#
# IEEE 1394 (FireWire) support
#
# CONFIG_IEEE1394 is not set

#
# I2O device support
#
CONFIG_I2O=m
CONFIG_I2O_CONFIG=m
CONFIG_I2O_BLOCK=m
CONFIG_I2O_SCSI=m
CONFIG_I2O_PROC=m

#
# Networking support
#
CONFIG_NET=y

#
# Networking options
#
CONFIG_PACKET=m
CONFIG_PACKET_MMAP=y
CONFIG_NETLINK_DEV=m
CONFIG_UNIX=y
CONFIG_NET_KEY=m
CONFIG_INET=y
CONFIG_IP_MULTICAST=y
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_IP_MULTIPLE_TABLES=y
CONFIG_IP_ROUTE_MULTIPATH=y
CONFIG_IP_ROUTE_TOS=y
CONFIG_IP_ROUTE_VERBOSE=y
# CONFIG_IP_PNP is not set
CONFIG_NET_IPIP=m
CONFIG_NET_IPGRE=m
CONFIG_NET_IPGRE_BROADCAST=y
# CONFIG_IP_MROUTE is not set
# CONFIG_ARPD is not set
# CONFIG_SYN_COOKIES is not set
# CONFIG_INET_AH is not set
# CONFIG_INET_ESP is not set
# CONFIG_INET_IPCOMP is not set
CONFIG_INET_TUNNEL=m
# CONFIG_IPV6 is not set
CONFIG_IPV6_NDISC_NEW=y
# CONFIG_NETFILTER is not set
CONFIG_XFRM=y
# CONFIG_XFRM_USER is not set

#
# SCTP Configuration (EXPERIMENTAL)
#
# CONFIG_IP_SCTP is not set
# CONFIG_SCTP_HMAC_NONE is not set
# CONFIG_SCTP_HMAC_SHA1 is not set
# CONFIG_SCTP_HMAC_MD5 is not set
# CONFIG_ATM is not set
# CONFIG_BRIDGE is not set
# CONFIG_VLAN_8021Q is not set
# CONFIG_DECNET is not set
```

```
# CONFIG_LLC2 is not set
# CONFIG_IPX is not set
# CONFIG_ATALK is not set
# CONFIG_X25 is not set
# CONFIG_LAPB is not set
# CONFIG_NET_DIVERT is not set
# CONFIG_ECONET is not set
# CONFIG_WAN_ROUTER is not set
# CONFIG_NET_HW_FLOWCONTROL is not set

#
# QoS and/or fair queueing
#
# CONFIG_NET_SCHED is not set
# CONFIG_NET_SCH_CLK_JIFFIES is not set
# CONFIG_NET_SCH_CLK_GETTIMEOFDAY is not set
# CONFIG_NET_SCH_CLK_CPU is not set
# CONFIG_NET_CLS_ROUTE is not set

#
# Network testing
#
CONFIG_NET_PKTGEN=m
# CONFIG_NETPOLL is not set
# CONFIG_NET_POLL_CONTROLLER is not set
# CONFIG_HAMRADIO is not set
# CONFIG_IRDA is not set
# CONFIG_BT is not set
CONFIG_NETDEVICES=y
# CONFIG_DUMMY is not set
# CONFIG_BONDING is not set
# CONFIG_EQUALIZER is not set
# CONFIG_TUN is not set
# CONFIG_ETHERTAP is not set

#
# ARCnet devices
#
# CONFIG_ARCNET is not set

#
# Ethernet (10 or 100Mbit)
#
# CONFIG_NET_ETHERNET is not set

#
# Ethernet (1000 Mbit)
#
# CONFIG_ACENIC is not set
# CONFIG_DL2K is not set
# CONFIG_E1000 is not set
# CONFIG_NS83820 is not set
# CONFIG_HAMACHI is not set
# CONFIG_YELLOWFIN is not set
# CONFIG_R8169 is not set
# CONFIG_SK98LIN is not set
CONFIG_TIGON3=m
CONFIG_NET_BROADCOM=m

#
# Ethernet (10000 Mbit)
#
# CONFIG_IXGB is not set
# CONFIG_S2IO is not set
```

```
#
# Token Ring devices
#
# CONFIG_TR is not set

#
# Wireless LAN (non-hamradio)
#
# CONFIG_NET_RADIO is not set

#
# Wan interfaces
#
# CONFIG_WAN is not set
# CONFIG_FDDI is not set
# CONFIG_HIPPI is not set
# CONFIG_PPP is not set
# CONFIG_SLIP is not set
# CONFIG_NET_FC is not set
# CONFIG_SHAPER is not set
# CONFIG_NETCONSOLE is not set

#
# ISDN subsystem
#
# CONFIG_ISDN is not set

#
# Telephony Support
#
# CONFIG_PHONE is not set

#
# Input device support
#
CONFIG_INPUT=y

#
# Userland interfaces
#
CONFIG_INPUT_MOUSEDEV=y
CONFIG_INPUT_MOUSEDEV_PSAUX=y
CONFIG_INPUT_MOUSEDEV_SCREEN_X=1024
CONFIG_INPUT_MOUSEDEV_SCREEN_Y=768
# CONFIG_INPUT_JOYDEV is not set
# CONFIG_INPUT_TSDEV is not set
CONFIG_INPUT_EVDEV=m
# CONFIG_INPUT_EVBUG is not set

#
# Input I/O drivers
#
# CONFIG_GAMEPORT is not set
CONFIG_SOUND_GAMEPORT=y
CONFIG_SERIO=y
CONFIG_SERIO_I8042=y
# CONFIG_SERIO_SERPORT is not set
# CONFIG_SERIO_CT82C710 is not set
# CONFIG_SERIO_PCIPS2 is not set
# CONFIG_SERIO_RAW is not set

#
# Input Device Drivers
```

```
#
CONFIG_INPUT_KEYBOARD=y
CONFIG_KEYBOARD_ATKBD=y
# CONFIG_KEYBOARD_SUNKBD is not set
# CONFIG_KEYBOARD_LKKBD is not set
# CONFIG_KEYBOARD_XTKBD is not set
# CONFIG_KEYBOARD_NEWTON is not set
CONFIG_INPUT_MOUSE=y
CONFIG_MOUSE_PS2=y
# CONFIG_MOUSE_SERIAL is not set
# CONFIG_MOUSE_VSXXXAA is not set
# CONFIG_INPUT_JOYSTICK is not set
# CONFIG_INPUT_TOUCHSCREEN is not set
# CONFIG_INPUT_MISC is not set

#
# Character devices
#
CONFIG_VT=y
CONFIG_VT_CONSOLE=y
CONFIG_HW_CONSOLE=y
CONFIG_ECC=m
# CONFIG_SERIAL_NONSTANDARD is not set

#
# Serial drivers
#
CONFIG_SERIAL_8250=y
CONFIG_SERIAL_8250_CONSOLE=y
CONFIG_SERIAL_8250_NR_UARTS=4
CONFIG_SERIAL_8250_EXTENDED=y
CONFIG_SERIAL_8250_MANY_PORTS=y
CONFIG_SERIAL_8250_SHARE_IRQ=y
# CONFIG_SERIAL_8250_DETECT_IRQ is not set
CONFIG_SERIAL_8250_MULTIPORT=y
CONFIG_SERIAL_8250_RSA=y

#
# Non-8250 serial port support
#
CONFIG_SERIAL_CORE=y
CONFIG_SERIAL_CORE_CONSOLE=y
CONFIG_UNIX98_PTYS=y
CONFIG_LEGACY_PTYS=y
CONFIG_LEGACY_PTY_COUNT=256

#
# IPMI
#
CONFIG_IPMI_HANDLER=m
CONFIG_IPMI_PANIC_EVENT=y
CONFIG_IPMI_PANIC_STRING=y
CONFIG_IPMI_DEVICE_INTERFACE=m
CONFIG_IPMI_SI=m
CONFIG_IPMI_WATCHDOG=m
CONFIG_IPMI_POWEROFF=m

#
# Watchdog Cards
#
# CONFIG_WATCHDOG is not set
CONFIG_HW_RANDOM=m
CONFIG_NVRAM=m
CONFIG_RTC=y
```

```
# CONFIG_DTLK is not set
# CONFIG_R3964 is not set
# CONFIG_APPLICOM is not set
# CONFIG_SONYPI is not set

#
# Ftape, the floppy tape device driver
#
# CONFIG_FTAPE is not set
# CONFIG_AGP is not set
# CONFIG_DRM is not set
# CONFIG_MWAVE is not set
# CONFIG_RAW_DRIVER is not set
# CONFIG_HANGCHECK_TIMER is not set
# CONFIG_VTUNE is not set

#
# Linux InfraRed Controller
#
CONFIG_LIRC_SUPPORT=m
CONFIG_LIRC_MAX_DEV=2
CONFIG_LIRC_BT829=m
CONFIG_LIRC_IT87=m
CONFIG_LIRC_SERIAL=m
# CONFIG_LIRC_HOMEBREW is not set
CONFIG_LIRC_PORT_SERIAL=0x3f8
CONFIG_LIRC_IRQ_SERIAL=4
CONFIG_LIRC_SIR=m
CONFIG_LIRC_PORT_SIR=0x3f8
CONFIG_LIRC_IRQ_SIR=4

#
# I2C support
#
CONFIG_I2C=m
CONFIG_I2C_CHARDEV=m

#
# I2C Algorithms
#
CONFIG_I2C_ALGOBIT=m
CONFIG_I2C_ALGOPCF=m
CONFIG_I2C_ALGOPCA=m

#
# I2C Hardware Bus support
#
CONFIG_I2C_ALI1535=m
CONFIG_I2C_ALI1563=m
CONFIG_I2C_ALI15X3=m
CONFIG_I2C_AMD756=m
CONFIG_I2C_AMD8111=m
CONFIG_I2C_I801=m
CONFIG_I2C_I810=m
CONFIG_I2C_ISA=m
CONFIG_I2C_NFORCE2=m
CONFIG_I2C_PARPORT_LIGHT=m
CONFIG_I2C_PIIX4=m
CONFIG_I2C_PROSAVAGE=m
CONFIG_I2C_SAVAGE4=m
CONFIG_SCx200_ACB=m
CONFIG_I2C_SIS5595=m
CONFIG_I2C_SIS630=m
CONFIG_I2C_SIS96X=m
```

```
CONFIG_I2C_VIA=m
CONFIG_I2C_VIAPRO=m
CONFIG_I2C_VOODOO3=m
CONFIG_I2C_PCA_ISA=m

#
# Hardware Sensors Chip support
#
CONFIG_I2C_SENSOR=m
CONFIG_SENSORS_ADM1021=m
CONFIG_SENSORS_ADM1025=m
CONFIG_SENSORS_ADM1031=m
CONFIG_SENSORS_ASB100=m
CONFIG_SENSORS_DS1621=m
CONFIG_SENSORS_FSCHER=m
CONFIG_SENSORS_GL518SM=m
CONFIG_SENSORS_IT87=m
CONFIG_SENSORS_LM75=m
CONFIG_SENSORS_LM77=m
CONFIG_SENSORS_LM78=m
CONFIG_SENSORS_LM80=m
CONFIG_SENSORS_LM83=m
CONFIG_SENSORS_LM85=m
CONFIG_SENSORS_LM90=m
CONFIG_SENSORS_MAX1619=m
CONFIG_SENSORS_SMSC47M1=m
CONFIG_SENSORS_VIA686A=m
CONFIG_SENSORS_W83781D=m
CONFIG_SENSORS_W83L785TS=m
CONFIG_SENSORS_W83627HF=m

#
# Other I2C Chip support
#
CONFIG_SENSORS_EEPROM=m
CONFIG_SENSORS_PCF8574=m
CONFIG_SENSORS_PCF8591=m
CONFIG_SENSORS_RTC8564=m
# CONFIG_I2C_DEBUG_CORE is not set
# CONFIG_I2C_DEBUG_ALGO is not set
# CONFIG_I2C_DEBUG_BUS is not set
# CONFIG_I2C_DEBUG_CHIP is not set

#
# Dallas's 1-wire bus
#
# CONFIG_W1 is not set

#
# Misc devices
#
# CONFIG_IBM_ASM is not set

#
# Multimedia devices
#
# CONFIG_VIDEO_DEV is not set

#
# Digital Video Broadcasting Devices
#
# CONFIG_DVB is not set

#
```

```
# Graphics support
#
CONFIG_FB=y
CONFIG_FB_MODE_HELPERS=y
# CONFIG_FB_CIRRUS is not set
# CONFIG_FB_PM2 is not set
# CONFIG_FB_CYBER2000 is not set
# CONFIG_FB_ASILIANT is not set
# CONFIG_FB_IMSTT is not set
CONFIG_FB_VGA16=m
CONFIG_FB_VESA=y
CONFIG_VIDEO_SELECT=y
# CONFIG_FB_HGA is not set
# CONFIG_FB_RIVA is not set
# CONFIG_FB_I810 is not set
# CONFIG_FB_MATROX is not set
# CONFIG_FB_RADEON_OLD is not set
# CONFIG_FB_RADEON is not set
# CONFIG_FB_ATY128 is not set
CONFIG_FB_ATY=m
CONFIG_FB_ATY_CT=y
CONFIG_FB_ATY_GX=y
CONFIG_FB_ATY_XL_INIT=y
# CONFIG_FB_SIS is not set
# CONFIG_FB_NEOMAGIC is not set
# CONFIG_FB_KYRO is not set
# CONFIG_FB_3DFX is not set
# CONFIG_FB_VOODOO1 is not set
# CONFIG_FB_TRIDENT is not set
# CONFIG_FB_VIRTUAL is not set

#
# Console display driver support
#
CONFIG_VGA_CONSOLE=y
CONFIG_DUMMY_CONSOLE=y
CONFIG_FRAMEBUFFER_CONSOLE=y
# CONFIG_FONTS is not set
CONFIG_FONT_8x8=y
CONFIG_FONT_8x16=y

#
# Logo configuration
#
# CONFIG_LOGO is not set

#
# Bootsplash configuration
#
CONFIG_BOOTSPLASH=y

#
# Sound
#
# CONFIG_SOUND is not set

#
# USB support
#
# CONFIG_USB is not set

#
# USB Gadget Support
#
```

```
# CONFIG_USB_GADGET is not set

#
# InfiniBand support
#
# CONFIG_INFINIBAND is not set

#
# File systems
#
CONFIG_EXT2_FS=y
CONFIG_EXT2_FS_XATTR=y
CONFIG_EXT2_FS_POSIX_ACL=y
CONFIG_EXT2_FS_SECURITY=y
CONFIG_EXT3_FS=m
CONFIG_EXT3_FS_XATTR=y
CONFIG_EXT3_FS_POSIX_ACL=y
CONFIG_EXT3_FS_SECURITY=y
CONFIG_JBD=m
CONFIG_JBD_DEBUG=y
CONFIG_FS_MBCACHE=y
CONFIG_REISER4_FS=m
CONFIG_REISER4_LARGE_KEY=y
# CONFIG_REISER4_CHECK is not set
CONFIG_REISERFS_FS=m
# CONFIG_REISERFS_CHECK is not set
# CONFIG_REISERFS_PROC_INFO is not set
CONFIG_REISERFS_FS_XATTR=y
CONFIG_REISERFS_FS_POSIX_ACL=y
CONFIG_REISERFS_FS_SECURITY=y
CONFIG_JFS_FS=m
CONFIG_JFS_POSIX_ACL=y
# CONFIG_JFS_DEBUG is not set
CONFIG_JFS_STATISTICS=y
CONFIG_FS_POSIX_ACL=y
CONFIG_XFS_FS=m
CONFIG_XFS_RT=y
CONFIG_XFS_QUOTA=y
CONFIG_XFS_SECURITY=y
CONFIG_XFS_POSIX_ACL=y
CONFIG_MINIX_FS=y
CONFIG_ROMFS_FS=m
CONFIG_QUOTA=y
CONFIG_QFMT_V1=m
CONFIG_QFMT_V2=m
CONFIG_QUOTACTL=y
# CONFIG_SUBFS_FS is not set
CONFIG_AUTOFS_FS=m
CONFIG_AUTOFS4_FS=m

#
# CD-ROM/DVD Filesystems
#
CONFIG_ISO9660_FS=y
CONFIG_JOLIET=y
CONFIG_ZISOFS=y
CONFIG_ZISOFS_FS=y
CONFIG_UDF_FS=m
CONFIG_UDF_NLS=y

#
# DOS/FAT/NT Filesystems
#
CONFIG_FAT_FS=m
```

```
CONFIG_MSDOS_FS=m
CONFIG_VFAT_FS=m
CONFIG_FAT_DEFAULT_CODEPAGE=437
CONFIG_FAT_DEFAULT_IOCHARSET="iso8859-1"
CONFIG_NTFS_FS=m
# CONFIG_NTFS_DEBUG is not set
# CONFIG_NTFS_RW is not set

#
# Pseudo filesystems
#
CONFIG_PROC_FS=y
CONFIG_PROC_KCORE=y
CONFIG_SYSFS=y
# CONFIG_DEVFS_FS is not set
CONFIG_DEVPTS_FS_XATTR=y
CONFIG_DEVPTS_FS_SECURITY=y
CONFIG_TMPFS=y
CONFIG_HUGETLBFS=y
CONFIG_HUGETLB_PAGE=y
CONFIG_RAMFS=y

#
# Miscellaneous filesystems
#
CONFIG_ADFS_FS=m
# CONFIG_ADFS_FS_RW is not set
CONFIG_AFFS_FS=m
CONFIG_HFS_FS=m
CONFIG_HFSPLUS_FS=m
CONFIG_BEFS_FS=m
# CONFIG_BEFS_DEBUG is not set
CONFIG_BFS_FS=m
CONFIG_EFS_FS=m
CONFIG_CRAMFS=m
CONFIG_VXFS_FS=m
CONFIG_HPFS_FS=m
CONFIG_QNX4FS_FS=m
# CONFIG_QNX4FS_RW is not set
CONFIG_SYSV_FS=m
CONFIG_UFS_FS=m
# CONFIG_UFS_FS_WRITE is not set

#
# Network File Systems
#
CONFIG_NFS_FS=y
CONFIG_NFS_V3=y
CONFIG_NFS_ACL=y
# CONFIG_NFS_V4 is not set
CONFIG_NFS_DIRECTIO=y
CONFIG_NFSD=m
CONFIG_NFSD_V3=y
CONFIG_NFSD_ACL=y
CONFIG_NFS_ACL_SUPPORT=y
# CONFIG_NFSD_V4 is not set
CONFIG_NFSD_TCP=y
CONFIG_LOCKD=y
CONFIG_STATD=y
CONFIG_LOCKD_V4=y
CONFIG_EXPORTFS=m
CONFIG_SUNRPC=y
CONFIG_SUNRPC_GSS=y
CONFIG_RPCSEC_GSS_KRB5=y
```

```
CONFIG_RPCSEC_GSS_SPKM3=m
CONFIG_SMB_FS=m
CONFIG_SMB_NLS_DEFAULT=y
CONFIG_SMB_NLS_REMOTE="cp850"
CONFIG_CIFS=m
CONFIG_CIFS_STATS=y
CONFIG_CIFS_XATTR=y
CONFIG_CIFS_POSIX=y
CONFIG_NCP_FS=m
CONFIG_NCPFS_PACKET_SIGNING=y
CONFIG_NCPFS_IOCTL_LOCKING=y
CONFIG_NCPFS_STRONG=y
CONFIG_NCPFS_NFS_NS=y
CONFIG_NCPFS_OS2_NS=y
CONFIG_NCPFS_SMALLDOS=y
CONFIG_NCPFS_NLS=y
CONFIG_NCPFS_EXTRAS=y
CONFIG_CODA_FS=m
# CONFIG_CODA_FS_OLD_API is not set
CONFIG_AFS_FS=m
CONFIG_RXRPC=m

#
# Partition Types
#
CONFIG_PARTITION_ADVANCED=y
# CONFIG_ACORN_PARTITION is not set
CONFIG_OSF_PARTITION=y
# CONFIG_AMIGA_PARTITION is not set
CONFIG_ATARI_PARTITION=y
CONFIG_MAC_PARTITION=y
CONFIG_MSDOS_PARTITION=y
CONFIG_BSD_DISKLABEL=y
# CONFIG_MINIX_SUBPARTITION is not set
CONFIG_SOLARIS_X86_PARTITION=y
CONFIG_UNIXWARE_DISKLABEL=y
CONFIG_LDM_PARTITION=y
# CONFIG_LDM_DEBUG is not set
CONFIG_SGI_PARTITION=y
CONFIG_ULTRIX_PARTITION=y
CONFIG_SUN_PARTITION=y
CONFIG_EFI_PARTITION=y

#
# Native Language Support
#
CONFIG_NLS=y
CONFIG_NLS_DEFAULT="utf8"
CONFIG_NLS_CODEPAGE_437=m
CONFIG_NLS_CODEPAGE_737=m
CONFIG_NLS_CODEPAGE_775=m
CONFIG_NLS_CODEPAGE_850=m
CONFIG_NLS_CODEPAGE_852=m
CONFIG_NLS_CODEPAGE_855=m
CONFIG_NLS_CODEPAGE_857=m
CONFIG_NLS_CODEPAGE_860=m
CONFIG_NLS_CODEPAGE_861=m
CONFIG_NLS_CODEPAGE_862=m
CONFIG_NLS_CODEPAGE_863=m
CONFIG_NLS_CODEPAGE_864=m
CONFIG_NLS_CODEPAGE_865=m
CONFIG_NLS_CODEPAGE_866=m
CONFIG_NLS_CODEPAGE_869=m
CONFIG_NLS_CODEPAGE_936=m
```

```
CONFIG_NLS_CODEPAGE_950=m
CONFIG_NLS_CODEPAGE_932=m
CONFIG_NLS_CODEPAGE_949=m
CONFIG_NLS_CODEPAGE_874=m
CONFIG_NLS_ISO8859_8=m
CONFIG_NLS_CODEPAGE_1250=m
CONFIG_NLS_CODEPAGE_1251=m
CONFIG_NLS_ASCII=m
CONFIG_NLS_ISO8859_1=m
CONFIG_NLS_ISO8859_2=m
CONFIG_NLS_ISO8859_3=m
CONFIG_NLS_ISO8859_4=m
CONFIG_NLS_ISO8859_5=m
CONFIG_NLS_ISO8859_6=m
CONFIG_NLS_ISO8859_7=m
CONFIG_NLS_ISO8859_9=m
CONFIG_NLS_ISO8859_13=m
CONFIG_NLS_ISO8859_14=m
CONFIG_NLS_ISO8859_15=m
CONFIG_NLS_KOI8_R=m
CONFIG_NLS_KOI8_U=m
CONFIG_NLS_UTF8=m

#
# Profiling support
#
# CONFIG_PROFILING is not set

#
# Kernel hacking
#
# CONFIG_DEBUG_KERNEL is not set
# CONFIG_FRAME_POINTER is not set
CONFIG_EARLY_PRINTK=y
# CONFIG_4KSTACKS is not set
CONFIG_X86_FIND_SMP_CONFIG=y
CONFIG_X86_MPPARSE=y

#
# Security options
#
# CONFIG_SECURITY is not set

#
# Cryptographic options
#
CONFIG_CRYPTO=y
CONFIG_CRYPTO_HMAC=y
CONFIG_CRYPTO_NULL=m
CONFIG_CRYPTO_MD4=m
CONFIG_CRYPTO_MD5=y
CONFIG_CRYPTO_SHA1=m
CONFIG_CRYPTO_SHA256=m
CONFIG_CRYPTO_SHA512=m
CONFIG_CRYPTO_DES=y
CONFIG_CRYPTO_BLOWFISH=m
CONFIG_CRYPTO_TWOFISH=m
CONFIG_CRYPTO_SERPENT=m
CONFIG_CRYPTO_AES_586=m
CONFIG_CRYPTO_CAST5=m
CONFIG_CRYPTO_CAST6=m
CONFIG_CRYPTO_TEA=m
CONFIG_CRYPTO_ARC4=m
CONFIG_CRYPTO_KHAZAD=m
```

```
CONFIG_CRYPTO_DEFLATE=m
CONFIG_CRYPTO_MICHAEL_MIC=m
CONFIG_CRYPTO_CRC32C=m
CONFIG_CRYPTO_TEST=m

#
# Library routines
#
CONFIG_CRC_CCITT=m
CONFIG_CRC32=y
CONFIG_LIBCRC32C=m
CONFIG_QSORT=y
CONFIG_ZLIB_INFLATE=y
CONFIG_ZLIB_DEFLATE=m

#
# Build options
#
CONFIG_SUSE_KERNEL=y
CONFIG_CFGNAME="default"
CONFIG_RELEASE="27"
CONFIG_X86_BIOS_REBOOT=y
CONFIG_PC=y
```

Doxygen source code documentation for Stressnet

Stressnet Reference Manual

Generated by Doxygen 1.4.2

Tue Jun 21 21:52:27 2005

Contents

1 Stressnet Data Structure Index

1.1 Stressnet Data Structures

Here are the data structures with brief descriptions:

2 Stressnet File Index

2.1 Stressnet File List

Here is a list of all files with brief descriptions:

3 Stressnet Data Structure Documentation

3.1 gengetopt_args_info Struct Reference

`#include <cmdline.h>`

Data Fields

- long bitrate_arg
- int bitrate_given
- char * configfile_arg
- int configfile_given
- char * destMAC_arg
- int destMAC_given
- int help_given
- char * interface_arg
- int interface_given
- char * packetfileM_arg
- int packetfileM_given
- char * packetfileN_arg
- int packetfileN_given
- int priority_arg
- int priority_given
- int quantity_arg
- int quantity_given
- long ratio_arg
- int ratio_given
- int version_given

3.1.1 Field Documentation

3.1.1.1 long gengetopt_args_info::bitrate_arg

Definition at line 31 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.2 int gengetopt_args_info::bitrate_given

Definition at line 43 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.3 char* gengetopt_args_info::configfile_arg

Definition at line 32 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.4 int gengetopt_args_info::configfile_given

Definition at line 44 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.5 char* gengetopt_args_info::destMAC_arg

Definition at line 35 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.6 int gengetopt_args_info::destMAC_given

Definition at line 47 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.7 int gengetopt_args_info::help_given

Definition at line 37 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.8 char* gengetopt_args_info::interface_arg

Definition at line 27 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.9 int gengetopt_args_info::interface_given

Definition at line 39 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.10 char* gengetopt_args_info::packetfileM_arg

Definition at line 28 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.11 int gengetopt_args_info::packetfileM_given

Definition at line 40 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.12 char* gengetopt_args_info::packetfileN_arg

Definition at line 29 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.13 int gengetopt_args_info::packetfileN_given

Definition at line 41 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.14 int gengetopt_args_info::priority_arg

Definition at line 33 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.15 int gengetopt_args_info::priority_given

Definition at line 45 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.16 int gengetopt_args_info::quantity_arg

Definition at line 34 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.17 int gengetopt_args_info::quantity_given

Definition at line 46 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.18 long gengetopt_args_info::ratio_arg

Definition at line 30 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), and main().

3.1.1.19 int gengetopt_args_info::ratio_given

Definition at line 42 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

3.1.1.20 int gengetopt_args_info::version_given

Definition at line 38 of file cmdline.h.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

The documentation for this struct was generated from the following file:

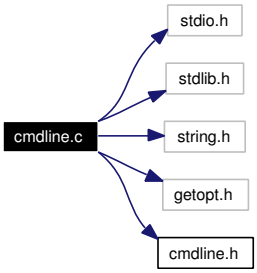
- cmdline.h

4 Stressnet File Documentation

4.1 cmdline.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "getopt.h"
#include "cmdline.h"
```

Include dependency graph for cmdline.c:



Defines

- #define clear_args()
- #define CONFIGPARSERBUFSIZE 1024

Functions

- int cmdline_parser (int argc, char *const *argv, struct gengetopt_args_info *args_info)
- int cmdline_parser_configfile (char *const filename, struct gengetopt_args_info *args_info, int override)
- void cmdline_parser_print_help (void)
- void cmdline_parser_print_version (void)
- static char * gengetopt_strdup (const char *s)

4.1.1 Define Documentation

4.1.1.1 #define clear_args()

Value:

```
{ \
  args_info->interface_arg = gengetopt_strdup("eth0") ;\
  args_info->packetfileM_arg = NULL; \
  args_info->packetfileN_arg = NULL; \
  args_info->ratio_arg = 50 ;\
  args_info->bitrate_arg = 100 ;\
  args_info->configfile_arg = NULL; \
  args_info->priority_arg = 0 ;\
  args_info->quantity_arg = 100 ;\
  args_info->destMAC_arg = gengetopt_strdup("00:11:22:33:44:55") ;\
}
```

Referenced by cmdline_parser().

4.1.1.2 #define CONFIGPARSERBUFSIZE 1024

Definition at line 301 of file cmdline.c.

Referenced by cmdline_parser_configfile().

4.1.2 Function Documentation

4.1.2.1 int cmdline_parser (int *argc*, char *const * *argv*, struct gengetopt_args_info * *args_info*)

Definition at line 105 of file cmdline.c.

References gengetopt_args_info::bitrate_arg, gengetopt_args_info::bitrate_given, clear_args, CMDLINE_PARSER_PACKAGE, cmdline_parser_print_help(), cmdline_parser_print_version(), gengetopt_args_info::configfile_arg, gengetopt_args_info::configfile_given, gengetopt_args_info::destMAC_arg, gengetopt_args_info::destMAC_given, gengetopt_strdup(), gengetopt_args_info::help_given, gengetopt_args_info::interface_arg, gengetopt_args_info::interface_given, gengetopt_args_info::packetfileM_arg, gengetopt_args_info::packetfileM_given, gengetopt_args_info::packetfileN_arg, gengetopt_args_info::packetfileN_given, gengetopt_args_info::priority_arg, gengetopt_args_info::priority_given, gengetopt_args_info::quantity_arg, gengetopt_args_info::quantity_given, gengetopt_args_info::ratio_arg, gengetopt_args_info::ratio_given, and gengetopt_args_info::version_given.

Referenced by main().

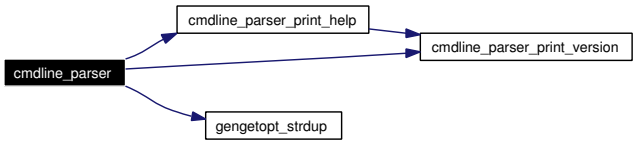
```
106 {
107   int c;        /* Character of the parsed option.  */
108   int missing_required_options = 0;
109
110   args_info->help_given = 0 ;
111   args_info->version_given = 0 ;
112   args_info->interface_given = 0 ;
113   args_info->packetfileM_given = 0 ;
114   args_info->packetfileN_given = 0 ;
115   args_info->ratio_given = 0 ;
116   args_info->bitrate_given = 0 ;
117   args_info->configfile_given = 0 ;
118   args_info->priority_given = 0 ;
119   args_info->quantity_given = 0 ;
120   args_info->destMAC_given = 0 ;
121 #define clear_args() { \
122   args_info->interface_arg = gengetopt_strdup("eth0") ;\
123   args_info->packetfileM_arg = NULL; \
124   args_info->packetfileN_arg = NULL; \
125   args_info->ratio_arg = 50 ;\
126   args_info->bitrate_arg = 100 ;\
127   args_info->configfile_arg = NULL; \
128   args_info->priority_arg = 0 ;\
129   args_info->quantity_arg = 100 ;\
130   args_info->destMAC_arg = gengetopt_strdup("00:11:22:33:44:55") ;\
131 }
132
133   clear_args();
134
135   optarg = 0;
136   optind = 1;
137   opterr = 1;
138   optopt = '?';
139
140   while (1)
141     {
142       int option_index = 0;
143       char *stop_char;
144
145       static struct option long_options[] = {
146         { "help",       0, NULL, 'h' },
147         { "version",    0, NULL, 'V' },
148         { "interface",  1, NULL, 'i' },
149         { "packetfileM",        1, NULL, 'm' },
150         { "packetfileN",        1, NULL, 'n' },
151         { "ratio",      1, NULL, 'r' },
152         { "bitrate",    1, NULL, 'b' },
153         { "configfile", 1, NULL, 'c' },
154         { "priority",   1, NULL, 'p' },
155         { "quantity",   1, NULL, 'q' },
```

```
156         { "destMAC",    1, NULL, 'd' },
157         { NULL, 0, NULL, 0 }
158       };
159
160       stop_char = 0;
161       c = getopt_long (argc, argv, "hVi:m:n:r:b:c:p:q:d:", long_options, &option_index);
162
163       if (c == -1) break;       /* Exit from `while (1)' loop.  */
164
165       switch (c)
166         {
167         case 'h':       /* Print help and exit.  */
168           clear_args ();
169           cmdline_parser_print_help ();
170           exit (EXIT_SUCCESS);
171
172         case 'V':       /* Print version and exit.  */
173           clear_args ();
174           cmdline_parser_print_version ();
175           exit (EXIT_SUCCESS);
176
177         case 'i':       /* network device used to send packets.  */
178           if (args_info->interface_given)
179             {
180               fprintf (stderr, "%s: `--interface' (`-i') option given more than once\n", CMDLINE_PARSE
181               clear_args ();
182               exit (EXIT_FAILURE);
183             }
184           args_info->interface_given = 1;
185           args_info->interface_arg = gengetopt_strdup (optarg);
186           break;
187
188         case 'm':       /* First file of packets.  */
189           if (args_info->packetfileM_given)
190             {
191               fprintf (stderr, "%s: `--packetfileM' (`-m') option given more than once\n", CMDLINE_PAR
192               clear_args ();
193               exit (EXIT_FAILURE);
194             }
195           args_info->packetfileM_given = 1;
196           args_info->packetfileM_arg = gengetopt_strdup (optarg);
197           break;
198
199         case 'n':       /* Second file ofpackets.  */
200           if (args_info->packetfileN_given)
201             {
202               fprintf (stderr, "%s: `--packetfileN' (`-n') option given more than once\n", CMDLINE_PAR
203               clear_args ();
204               exit (EXIT_FAILURE);
205             }
206           args_info->packetfileN_given = 1;
207           args_info->packetfileN_arg = gengetopt_strdup (optarg);
208           break;
209
210         case 'r':       /* ratio M/N: number of packets of file M for every 1 packet of file N.  */
211           if (args_info->ratio_given)
212             {
213               fprintf (stderr, "%s: `--ratio' (`-r') option given more than once\n", CMDLINE_PARSER_PA
214               clear_args ();
215               exit (EXIT_FAILURE);
216             }
217           args_info->ratio_given = 1;
218           args_info->ratio_arg = strtol (optarg,&stop_char,0);
219           break;
220
221         case 'b':       /* desired send bitrate in kbit/s.  */
222           if (args_info->bitrate_given)
```

```
223             {
224               fprintf (stderr, "%s: `--bitrate' (`-b') option given more than once\n", CMDLINE_PARSER_
225               clear_args ();
226               exit (EXIT_FAILURE);
227             }
228           args_info->bitrate_given = 1;
229           args_info->bitrate_arg = strtol (optarg,&stop_char,0);
230           break;
231
232         case 'c':       /* config file containing the command line arguments.  */
233           if (args_info->configfile_given)
234             {
235               fprintf (stderr, "%s: `--configfile' (`-c') option given more than once\n", CMDLINE_PARS
236               clear_args ();
237               exit (EXIT_FAILURE);
238             }
239           args_info->configfile_given = 1;
240           args_info->configfile_arg = gengetopt_strdup (optarg);
241           break;
242
243         case 'p':       /* scheduling priority value.  */
244           if (args_info->priority_given)
245             {
246               fprintf (stderr, "%s: `--priority' (`-p') option given more than once\n", CMDLINE_PARSER
247               clear_args ();
248               exit (EXIT_FAILURE);
249             }
250           args_info->priority_given = 1;
251           args_info->priority_arg = strtol (optarg,&stop_char,0);
252           break;
253
254         case 'q':       /* total quantity of packets to send.  */
255           if (args_info->quantity_given)
256             {
257               fprintf (stderr, "%s: `--quantity' (`-q') option given more than once\n", CMDLINE_PARSER
258               clear_args ();
259               exit (EXIT_FAILURE);
260             }
261           args_info->quantity_given = 1;
262           args_info->quantity_arg = strtol (optarg,&stop_char,0);
263           break;
264
265         case 'd':       /* destination MAC address for ALL packets.  */
266           if (args_info->destMAC_given)
267             {
268               fprintf (stderr, "%s: `--destMAC' (`-d') option given more than once\n", CMDLINE_PARSER_
269               clear_args ();
270               exit (EXIT_FAILURE);
271             }
272           args_info->destMAC_given = 1;
273           args_info->destMAC_arg = gengetopt_strdup (optarg);
274           break;
275
276
277         case 0: /* Long option with no short option */
278
279         case '?':       /* Invalid option.  */
280           /* `getopt_long' already printed an error message.  */
281           exit (EXIT_FAILURE);
282
283         default:        /* bug: option not considered.  */
284           fprintf (stderr, "%s: option unknown: %c\n", CMDLINE_PARSER_PACKAGE, c);
285           abort ();
286         } /* switch */
287     } /* while */
288
289
```

```
290   if (! args_info->packetfileM_given)
291     {
292       fprintf (stderr, "%s: '--packetfileM' ('-m') option required\n", CMDLINE_PARSER_PACKAGE);
293       missing_required_options = 1;
294     }
295   if ( missing_required_options )
296     exit (EXIT_FAILURE);
297
298   return 0;
299 }
```

Here is the call graph for this function:



4.1.2.2 int cmdline_parser_configfile (char *const *filename*, struct gengetopt_args_info * *args_info*, int *override*)

Definition at line 304 of file cmdline.c.

References gengetopt_args_info::bitrate_arg, gengetopt_args_info::bitrate_given, CMDLINE_PARSER_-PACKAGE, gengetopt_args_info::configfile_arg, gengetopt_args_info::configfile_given, CONFIG-PARSERBUFSIZE, gengetopt_args_info::destMAC_arg, gengetopt_args_info::destMAC_given, gengetopt_strdup(), gengetopt_args_info::help_given, gengetopt_args_info::interface_arg, gengetopt_-args_info::interface_given, gengetopt_args_info::packetfileM_arg, gengetopt_args_info::packetfileM_-given, gengetopt_args_info::packetfileN_arg, gengetopt_args_info::packetfileN_given, gengetopt_-args_info::priority_arg, gengetopt_args_info::priority_given, gengetopt_args_info::quantity_arg, gengetopt_args_info::quantity_given, gengetopt_args_info::ratio_arg, gengetopt_args_info::ratio_given, and gengetopt_args_info::version_given.

Referenced by main().

```
305 {
306   FILE* file;
307   char linebuf[CONFIGPARSERBUFSIZE];
308   int line_num = 0;
309   int len;
310   int fnum;
311   char fopt[CONFIGPARSERBUFSIZE], farg[CONFIGPARSERBUFSIZE];
312   char *stop_char;
313   char *str_index, *str_index2;
314   int next_token;
315
316   if ((file = fopen(filename, "r")) == NULL)
317     {
318       fprintf (stderr, "%s: Error opening configuration file '%s'\n",
319                CMDLINE_PARSER_PACKAGE, filename);
320       exit (EXIT_FAILURE);
321     }
322
323   while ((fgets(linebuf, CONFIGPARSERBUFSIZE, file)) != NULL)
324     {
325       ++line_num;
326       len = strlen(linebuf);
327       if (len == CONFIGPARSERBUFSIZE-1)
328         {
```

```
329           fprintf (stderr, "%s: Line longer than %d characters found in configuration file '%s'\n",
330                    CMDLINE_PARSER_PACKAGE, CONFIGPARSERBUFSIZE, filename);
331           exit (EXIT_FAILURE);
332         }
333
334       if (linebuf[0] == '#')
335         continue; /* Line was a comment */
336
337       /* read the option */
338       fnum = sscanf (linebuf, "%s", fopt);
339
340       if (fnum < 0)
341         continue; /* blank line */
342
343       next_token = strlen (fopt);
344       str_index = strchr (&linebuf[next_token], '\"');
345       if (str_index)
346         {
347           str_index2 = strchr (str_index + 1, '\"');
348           if (! str_index2)
349             {
350               fprintf
351                 (stderr,
352                  "%s: unterminated string in configuration file '%s'\n",
353                  CMDLINE_PARSER_PACKAGE, filename);
354               exit (EXIT_FAILURE);
355             }
356
357           strncpy (farg, str_index + 1, str_index2 - str_index - 1);
358           farg[str_index2 - str_index]='\0';
359           ++fnum;
360         }
361       else
362         if (sscanf (&linebuf[next_token], "%s", farg) > 0)
363           ++fnum;
364
365       if (fnum > 0)
366         {
367           if (!strcmp(fopt, "help"))
368             {
369               if (override || !args_info->help_given)
370                 {
371                   args_info->help_given = 1;
372
373                 }
374               continue;
375             }
376           if (!strcmp(fopt, "version"))
377             {
378               if (override || !args_info->version_given)
379                 {
380                   args_info->version_given = 1;
381
382                 }
383               continue;
384             }
385           if (!strcmp(fopt, "interface"))
386             {
387               if (override || !args_info->interface_given)
388                 {
389                   args_info->interface_given = 1;
390                   if (fnum == 2)
391                     args_info->interface_arg = gengetopt_strdup (farg);
392                   else
393                     {
394                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
395                                filename, line_num);
```

```
396                       exit (EXIT_FAILURE);
397                     }
398                 }
399               continue;
400             }
401           if (!strcmp(fopt, "packetfileM"))
402             {
403               if (override || !args_info->packetfileM_given)
404                 {
405                   args_info->packetfileM_given = 1;
406                   if (fnum == 2)
407                     args_info->packetfileM_arg = gengetopt_strdup (farg);
408                   else
409                     {
410                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
411                                filename, line_num);
412                       exit (EXIT_FAILURE);
413                     }
414                 }
415               continue;
416             }
417           if (!strcmp(fopt, "packetfileN"))
418             {
419               if (override || !args_info->packetfileN_given)
420                 {
421                   args_info->packetfileN_given = 1;
422                   if (fnum == 2)
423                     args_info->packetfileN_arg = gengetopt_strdup (farg);
424                   else
425                     {
426                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
427                                filename, line_num);
428                       exit (EXIT_FAILURE);
429                     }
430                 }
431               continue;
432             }
433           if (!strcmp(fopt, "ratio"))
434             {
435               if (override || !args_info->ratio_given)
436                 {
437                   args_info->ratio_given = 1;
438                   if (fnum == 2)
439                     args_info->ratio_arg = strtol (farg,&stop_char,0);
440                   else
441                     {
442                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
443                                filename, line_num);
444                       exit (EXIT_FAILURE);
445                     }
446                 }
447               continue;
448             }
449           if (!strcmp(fopt, "bitrate"))
450             {
451               if (override || !args_info->bitrate_given)
452                 {
453                   args_info->bitrate_given = 1;
454                   if (fnum == 2)
455                     args_info->bitrate_arg = strtol (farg,&stop_char,0);
456                   else
457                     {
458                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
459                                filename, line_num);
460                       exit (EXIT_FAILURE);
461                     }
462                 }
```

```
463               continue;
464             }
465           if (!strcmp(fopt, "configfile"))
466             {
467               if (override || !args_info->configfile_given)
468                 {
469                   args_info->configfile_given = 1;
470                   if (fnum == 2)
471                     args_info->configfile_arg = gengetopt_strdup (farg);
472                   else
473                     {
474                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
475                                filename, line_num);
476                       exit (EXIT_FAILURE);
477                     }
478                 }
479               continue;
480             }
481           if (!strcmp(fopt, "priority"))
482             {
483               if (override || !args_info->priority_given)
484                 {
485                   args_info->priority_given = 1;
486                   if (fnum == 2)
487                     args_info->priority_arg = strtol (farg,&stop_char,0);
488                   else
489                     {
490                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
491                                filename, line_num);
492                       exit (EXIT_FAILURE);
493                     }
494                 }
495               continue;
496             }
497           if (!strcmp(fopt, "quantity"))
498             {
499               if (override || !args_info->quantity_given)
500                 {
501                   args_info->quantity_given = 1;
502                   if (fnum == 2)
503                     args_info->quantity_arg = strtol (farg,&stop_char,0);
504                   else
505                     {
506                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
507                                filename, line_num);
508                       exit (EXIT_FAILURE);
509                     }
510                 }
511               continue;
512             }
513           if (!strcmp(fopt, "destMAC"))
514             {
515               if (override || !args_info->destMAC_given)
516                 {
517                   args_info->destMAC_given = 1;
518                   if (fnum == 2)
519                     args_info->destMAC_arg = gengetopt_strdup (farg);
520                   else
521                     {
522                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
523                                filename, line_num);
524                       exit (EXIT_FAILURE);
525                     }
526                 }
527               continue;
528             }
529
```

```
530
531           /* Tried all known options. This one is unknown! */
532           fprintf (stderr, "%s: Unknown option '%s' found in %s\n",
533                    CMDLINE_PARSER_PACKAGE, fopt, filename);
534           exit (EXIT_FAILURE);
535         }
536     } /* while */
537   fclose(file); /* No error checking on close */
538
539   return 0;
540 }
```

Here is the call graph for this function:

4.1.2.3 void cmdline_parser_print_help (void)

Definition at line 32 of file cmdline.c.

References CMDLINE_PARSER_PACKAGE, and cmdline_parser_print_version().

Referenced by cmdline_parser().

```
33 {
34   cmdline_parser_print_version ();
35   printf("\n"
36   "Purpose:\n"
37   "  stressnet is intended for replaying tcpdump/pcap capture files \n"
38   "  -except the ethernet address- with a given bitrate to stress network \n"
39   "  devices such as intrusion detection systems or firewalls.  It mixes two pcap\n"
40   "  files (A and B) with a ratio r given as argument and meaning: for every 1 \n"
41   "  packet of file B that is sent, there are r packets of file A that are sent. \n"
42   "  This is meant to impose a rate of malformed packets in the dataflow.\n"
43   "  \n"
44   "  THIS SOFTWARE CAN AND SHOULD BE IMPROVED AS MUCH AS POSSIBLE, new ideas and \n"
45   "  critics are always welcome.  You can send them by email to the following \n"
46   "  address:\n"
47   "  \n"
48   "  \tyannick AT loth.be\n"
49   "  \n"
50   "  I would be pleased to discuss about them with you and to exchange ideas about\n"
51   "  further improvements.\n"
52   "  \n"
53   "  This software was initially developed for the Royal Military Academy (RMA) of\n"
54   "  Belgium (Brussels), by a student (Yannick Loth) during a training for his \n"
55   "  studies as Industrial Engineer in Applied Informatics at the  University of \n"
56   "  Luxembourg (2005).\n"
57   "  The aim of stressnet is to provide a reliable tool (i.e. intended for \n"
58   "  engineers etc.) to send packets (well formed and malformed ones) through a \n"
59   "  network at different speeds and ratios of innofensive packets over attacking \n"
60   "  packets to establish the limits of devices when they are seen as 'blackboxes',\n"
61   "  i.e. when the internal functioning of the devices is unknown.\n"
62   "  'Reliable' means that when someone asks for a certain bitrate, it should send\n"
63   "  data at bitrates quite close to the wanted bitrate.\n"
64   "  \n"
65   "  The author of this software thanks Maj. W. Mees (RMA) and Capt. O. Thonnard \n"
66   "  (RMA) as well as Pr. Th. Engel (Uni. Lux.) for their advices.\n"
67   "  \n"
68   "  stressnet should only be used by people knowing what they do, this tool could \n"
69   "  in fact block your computer for a long time if you don't take care of what you\n"
70   "  do.\n"
```

```
71   "  \n"
72   "  You'll (one day, I hope...) find more about this program on \n"
73   "  \thttp://www.loth.be/yannick/stressnet/index.html\n"
74   "\n"
75   "Usage: %s [OPTIONS]...\n", CMDLINE_PARSER_PACKAGE);
76   printf("   -h         --help               Print help and exit\n");
77   printf("   -V         --version            Print version and exit\n");
78   printf("   -iSTRING   --interface=STRING   network device used to send packets (default='eth0')\n");
79   printf("   -mSTRING   --packetfileM=STRING First file of packets\n");
80   printf("   -nSTRING   --packetfileN=STRING Second file ofpackets\n");
81   printf("   -rLONG     --ratio=LONG         ratio M/N: number of packets of file M for every 1 packet
82   printf("   -bLONG     --bitrate=LONG       desired send bitrate in kbit/s (default='100')\n");
83   printf("   -cSTRING   --configfile=STRING  config file containing the command line arguments\n");
84   printf("   -pINT      --priority=INT       scheduling priority value (default='0')\n");
85   printf("   -qINT      --quantity=INT       total quantity of packets to send (default='100')\n");
86   printf("   -dSTRING   --destMAC=STRING     destination MAC address for ALL packets (default='00:11:2
87 }
```

Here is the call graph for this function:

4.1.2.4 void cmdline_parser_print_version (void)

Definition at line 26 of file cmdline.c.

References CMDLINE_PARSER_PACKAGE, and CMDLINE_PARSER_VERSION.

Referenced by cmdline_parser(), and cmdline_parser_print_help().

```
27 {
28   printf ("%s %s\n", CMDLINE_PARSER_PACKAGE, CMDLINE_PARSER_VERSION);
29 }
```

4.1.2.5 char * gengetopt_strdup (const char * *s*) [static]

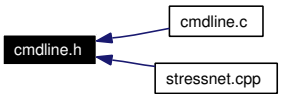
Definition at line 95 of file cmdline.c.

Referenced by cmdline_parser(), and cmdline_parser_configfile().

```
96 {
97   char *result = (char*)malloc(strlen(s) + 1);
98   if (result == (char*)0)
99     return (char*)0;
100   strcpy(result, s);
101   return result;
102 }
```

4.2 cmdline.h File Reference

This graph shows which files directly or indirectly include this file:



Defines

- #define CMDLINE_PARSER_PACKAGE "stressnet"
- #define CMDLINE_PARSER_VERSION "0.1"

Functions

- int cmdline_parser (int argc, char ∗const ∗argv, struct gengetopt_args_info ∗args_info)
- int cmdline_parser_configfile (char ∗const filename, struct gengetopt_args_info ∗args_info, int override)
- void cmdline_parser_print_help (void)
- void cmdline_parser_print_version (void)

4.2.1 Define Documentation

4.2.1.1 #define CMDLINE_PARSER_PACKAGE "stressnet"

Definition at line 18 of file cmdline.h.

Referenced by cmdline_parser(), cmdline_parser_configfile(), cmdline_parser_print_help(), and cmdline_parser_print_version().

4.2.1.2 #define CMDLINE_PARSER_VERSION "0.1"

Definition at line 22 of file cmdline.h.

Referenced by cmdline_parser_print_version().

4.2.2 Function Documentation

4.2.2.1 int cmdline_parser (int *argc*, char ∗const ∗ *argv*, struct gengetopt_args_info ∗ *args_info*)

Definition at line 105 of file cmdline.c.

References gengetopt_args_info::bitrate_arg, gengetopt_args_info::bitrate_given, clear_args, CMDLINE_PARSER_PACKAGE, cmdline_parser_print_help(), cmdline_parser_print_version(), gengetopt_args_info::configfile_arg, gengetopt_args_info::configfile_given, gengetopt_args_info::destMAC_arg, gengetopt_args_info::destMAC_given, gengetopt_strdup(), gengetopt_args_info::help_given, gengetopt_args_info::interface_arg, gengetopt_args_info::interface_given, gengetopt_args_info::packetfileM_arg, gengetopt_args_info::packetfileM_given, gengetopt_args_info::packetfileN_arg, gengetopt_args_info::packetfileN_given, gengetopt_args_info::priority_arg, gengetopt_args_info::priority_given, gengetopt_args_info::quantity_arg, gengetopt_args_info::quantity_given, gengetopt_args_info::ratio_arg, gengetopt_args_info::ratio_given, and gengetopt_args_info::version_given.

Referenced by main().

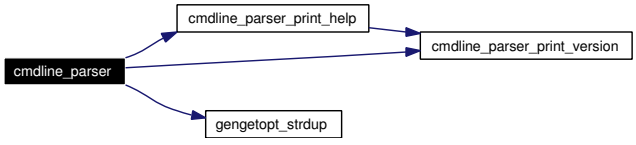
```
106 {
107   int c;        /* Character of the parsed option.  */
108   int missing_required_options = 0;
109
110   args_info->help_given = 0 ;
111   args_info->version_given = 0 ;
112   args_info->interface_given = 0 ;
113   args_info->packetfileM_given = 0 ;
114   args_info->packetfileN_given = 0 ;
```

```
115   args_info->ratio_given = 0 ;
116   args_info->bitrate_given = 0 ;
117   args_info->configfile_given = 0 ;
118   args_info->priority_given = 0 ;
119   args_info->quantity_given = 0 ;
120   args_info->destMAC_given = 0 ;
121 #define clear_args() { \
122   args_info->interface_arg = gengetopt_strdup("eth0") ;\
123   args_info->packetfileM_arg = NULL; \
124   args_info->packetfileN_arg = NULL; \
125   args_info->ratio_arg = 50 ;\
126   args_info->bitrate_arg = 100 ;\
127   args_info->configfile_arg = NULL; \
128   args_info->priority_arg = 0 ;\
129   args_info->quantity_arg = 100 ;\
130   args_info->destMAC_arg = gengetopt_strdup("00:11:22:33:44:55") ;\
131 }
132
133   clear_args();
134
135   optarg = 0;
136   optind = 1;
137   opterr = 1;
138   optopt = '?';
139
140   while (1)
141     {
142       int option_index = 0;
143       char *stop_char;
144
145       static struct option long_options[] = {
146         { "help",       0, NULL, 'h' },
147         { "version",    0, NULL, 'V' },
148         { "interface",  1, NULL, 'i' },
149         { "packetfileM",        1, NULL, 'm' },
150         { "packetfileN",        1, NULL, 'n' },
151         { "ratio",      1, NULL, 'r' },
152         { "bitrate",    1, NULL, 'b' },
153         { "configfile", 1, NULL, 'c' },
154         { "priority",   1, NULL, 'p' },
155         { "quantity",   1, NULL, 'q' },
156         { "destMAC",    1, NULL, 'd' },
157         { NULL, 0, NULL, 0 }
158       };
159
160       stop_char = 0;
161       c = getopt_long (argc, argv, "hVi:m:n:r:b:c:p:q:d:", long_options, &option_index);
162
163       if (c == -1) break;       /* Exit from `while (1)' loop.  */
164
165       switch (c)
166         {
167         case 'h':       /* Print help and exit.  */
168           clear_args ();
169           cmdline_parser_print_help ();
170           exit (EXIT_SUCCESS);
171
172         case 'V':       /* Print version and exit.  */
173           clear_args ();
174           cmdline_parser_print_version ();
175           exit (EXIT_SUCCESS);
176
177         case 'i':       /* network device used to send packets.  */
178           if (args_info->interface_given)
179             {
180               fprintf (stderr, "%s: `--interface' (`-i') option given more than once\n", CMDLINE_PARSE
181               clear_args ();
```

```
182               exit (EXIT_FAILURE);
183             }
184           args_info->interface_given = 1;
185           args_info->interface_arg = gengetopt_strdup (optarg);
186           break;
187
188         case 'm':       /* First file of packets.  */
189           if (args_info->packetfileM_given)
190             {
191               fprintf (stderr, "%s: `--packetfileM' (`-m') option given more than once\n", CMDLINE_PAR
192               clear_args ();
193               exit (EXIT_FAILURE);
194             }
195           args_info->packetfileM_given = 1;
196           args_info->packetfileM_arg = gengetopt_strdup (optarg);
197           break;
198
199         case 'n':       /* Second file ofpackets.  */
200           if (args_info->packetfileN_given)
201             {
202               fprintf (stderr, "%s: `--packetfileN' (`-n') option given more than once\n", CMDLINE_PAR
203               clear_args ();
204               exit (EXIT_FAILURE);
205             }
206           args_info->packetfileN_given = 1;
207           args_info->packetfileN_arg = gengetopt_strdup (optarg);
208           break;
209
210         case 'r':       /* ratio M/N: number of packets of file M for every 1 packet of file N.  */
211           if (args_info->ratio_given)
212             {
213               fprintf (stderr, "%s: `--ratio' (`-r') option given more than once\n", CMDLINE_PARSER_PA
214               clear_args ();
215               exit (EXIT_FAILURE);
216             }
217           args_info->ratio_given = 1;
218           args_info->ratio_arg = strtol (optarg,&stop_char,0);
219           break;
220
221         case 'b':       /* desired send bitrate in kbit/s.  */
222           if (args_info->bitrate_given)
223             {
224               fprintf (stderr, "%s: `--bitrate' (`-b') option given more than once\n", CMDLINE_PARSER_
225               clear_args ();
226               exit (EXIT_FAILURE);
227             }
228           args_info->bitrate_given = 1;
229           args_info->bitrate_arg = strtol (optarg,&stop_char,0);
230           break;
231
232         case 'c':       /* config file containing the command line arguments.  */
233           if (args_info->configfile_given)
234             {
235               fprintf (stderr, "%s: `--configfile' (`-c') option given more than once\n", CMDLINE_PARS
236               clear_args ();
237               exit (EXIT_FAILURE);
238             }
239           args_info->configfile_given = 1;
240           args_info->configfile_arg = gengetopt_strdup (optarg);
241           break;
242
243         case 'p':       /* scheduling priority value.  */
244           if (args_info->priority_given)
245             {
246               fprintf (stderr, "%s: `--priority' (`-p') option given more than once\n", CMDLINE_PARSER
247               clear_args ();
248               exit (EXIT_FAILURE);
```

```
249             }
250           args_info->priority_given = 1;
251           args_info->priority_arg = strtol (optarg,&stop_char,0);
252           break;
253
254         case 'q':       /* total quantity of packets to send.  */
255           if (args_info->quantity_given)
256             {
257               fprintf (stderr, "%s: `--quantity' (`-q') option given more than once\n", CMDLINE_PARSER
258               clear_args ();
259               exit (EXIT_FAILURE);
260             }
261           args_info->quantity_given = 1;
262           args_info->quantity_arg = strtol (optarg,&stop_char,0);
263           break;
264
265         case 'd':       /* destination MAC address for ALL packets.  */
266           if (args_info->destMAC_given)
267             {
268               fprintf (stderr, "%s: `--destMAC' (`-d') option given more than once\n", CMDLINE_PARSER_
269               clear_args ();
270               exit (EXIT_FAILURE);
271             }
272           args_info->destMAC_given = 1;
273           args_info->destMAC_arg = gengetopt_strdup (optarg);
274           break;
275
276
277         case 0: /* Long option with no short option */
278
279         case '?':       /* Invalid option.  */
280           /* `getopt_long' already printed an error message.  */
281           exit (EXIT_FAILURE);
282
283         default:        /* bug: option not considered.  */
284           fprintf (stderr, "%s: option unknown: %c\n", CMDLINE_PARSER_PACKAGE, c);
285           abort ();
286         } /* switch */
287     } /* while */
288
289
290   if (! args_info->packetfileM_given)
291     {
292       fprintf (stderr, "%s: '--packetfileM' ('-m') option required\n", CMDLINE_PARSER_PACKAGE);
293       missing_required_options = 1;
294     }
295   if ( missing_required_options )
296     exit (EXIT_FAILURE);
297
298   return 0;
299 }
```

Here is the call graph for this function:



4.2.2.2 int cmdline_parser_configfile (char *const *filename*, struct gengetopt_args_info * *args_info*, int *override*)

Definition at line 304 of file cmdline.c.

References gengetopt_args_info::bitrate_arg, gengetopt_args_info::bitrate_given, CMDLINE_PARSER_-PACKAGE, gengetopt_args_info::configfile_arg, gengetopt_args_info::configfile_given, CONFIG-PARSERBUFSIZE, gengetopt_args_info::destMAC_arg, gengetopt_args_info::destMAC_given, gengetopt_strdup(), gengetopt_args_info::help_given, gengetopt_args_info::interface_arg, gengetopt_-args_info::interface_given, gengetopt_args_info::packetfileM_arg, gengetopt_args_info::packetfileM_-given, gengetopt_args_info::packetfileN_arg, gengetopt_args_info::packetfileN_given, gengetopt_-args_info::priority_arg, gengetopt_args_info::priority_given, gengetopt_args_info::quantity_arg, gengetopt_args_info::quantity_given, gengetopt_args_info::ratio_arg, gengetopt_args_info::ratio_given, and gengetopt_args_info::version_given.

Referenced by main().

```
305 {
306   FILE* file;
307   char linebuf[CONFIGPARSERBUFSIZE];
308   int line_num = 0;
309   int len;
310   int fnum;
311   char fopt[CONFIGPARSERBUFSIZE], farg[CONFIGPARSERBUFSIZE];
312   char *stop_char;
313   char *str_index, *str_index2;
314   int next_token;
315
316   if ((file = fopen(filename, "r")) == NULL)
317     {
318       fprintf (stderr, "%s: Error opening configuration file '%s'\n",
319                CMDLINE_PARSER_PACKAGE, filename);
320       exit (EXIT_FAILURE);
321     }
322
323   while ((fgets(linebuf, CONFIGPARSERBUFSIZE, file)) != NULL)
324     {
325       ++line_num;
326       len = strlen(linebuf);
327       if (len == CONFIGPARSERBUFSIZE-1)
328         {
329           fprintf (stderr, "%s: Line longer than %d characters found in configuration file '%s'\n",
330                    CMDLINE_PARSER_PACKAGE, CONFIGPARSERBUFSIZE, filename);
331           exit (EXIT_FAILURE);
332         }
333
334       if (linebuf[0] == '#')
335         continue; /* Line was a comment */
336
337       /* read the option */
338       fnum = sscanf (linebuf, "%s", fopt);
339
340       if (fnum < 0)
341         continue; /* blank line */
342
343       next_token = strlen (fopt);
344       str_index = strchr (&linebuf[next_token], '\"');
345       if (str_index)
346         {
347           str_index2 = strchr (str_index + 1, '\"');
348           if (! str_index2)
349             {
350               fprintf
351                 (stderr,
352                  "%s: unterminated string in configuration file '%s'\n",
353                  CMDLINE_PARSER_PACKAGE, filename);
354               exit (EXIT_FAILURE);
355             }
```

```
356
357           strncpy (farg, str_index + 1, str_index2 - str_index - 1);
358           farg[str_index2 - str_index]='\0';
359           ++fnum;
360         }
361       else
362         if (sscanf (&linebuf[next_token], "%s", farg) > 0)
363           ++fnum;
364
365       if (fnum > 0)
366         {
367           if (!strcmp(fopt, "help"))
368             {
369               if (override || !args_info->help_given)
370                 {
371                   args_info->help_given = 1;
372
373                 }
374               continue;
375             }
376           if (!strcmp(fopt, "version"))
377             {
378               if (override || !args_info->version_given)
379                 {
380                   args_info->version_given = 1;
381
382                 }
383               continue;
384             }
385           if (!strcmp(fopt, "interface"))
386             {
387               if (override || !args_info->interface_given)
388                 {
389                   args_info->interface_given = 1;
390                   if (fnum == 2)
391                     args_info->interface_arg = gengetopt_strdup (farg);
392                   else
393                     {
394                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
395                                filename, line_num);
396                       exit (EXIT_FAILURE);
397                     }
398                 }
399               continue;
400             }
401           if (!strcmp(fopt, "packetfileM"))
402             {
403               if (override || !args_info->packetfileM_given)
404                 {
405                   args_info->packetfileM_given = 1;
406                   if (fnum == 2)
407                     args_info->packetfileM_arg = gengetopt_strdup (farg);
408                   else
409                     {
410                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
411                                filename, line_num);
412                       exit (EXIT_FAILURE);
413                     }
414                 }
415               continue;
416             }
417           if (!strcmp(fopt, "packetfileN"))
418             {
419               if (override || !args_info->packetfileN_given)
420                 {
421                   args_info->packetfileN_given = 1;
422                   if (fnum == 2)
```

```
423                     args_info->packetfileN_arg = gengetopt_strdup (farg);
424                   else
425                     {
426                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
427                                filename, line_num);
428                       exit (EXIT_FAILURE);
429                     }
430                 }
431               continue;
432             }
433           if (!strcmp(fopt, "ratio"))
434             {
435               if (override || !args_info->ratio_given)
436                 {
437                   args_info->ratio_given = 1;
438                   if (fnum == 2)
439                     args_info->ratio_arg = strtol (farg,&stop_char,0);
440                   else
441                     {
442                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
443                                filename, line_num);
444                       exit (EXIT_FAILURE);
445                     }
446                 }
447               continue;
448             }
449           if (!strcmp(fopt, "bitrate"))
450             {
451               if (override || !args_info->bitrate_given)
452                 {
453                   args_info->bitrate_given = 1;
454                   if (fnum == 2)
455                     args_info->bitrate_arg = strtol (farg,&stop_char,0);
456                   else
457                     {
458                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
459                                filename, line_num);
460                       exit (EXIT_FAILURE);
461                     }
462                 }
463               continue;
464             }
465           if (!strcmp(fopt, "configfile"))
466             {
467               if (override || !args_info->configfile_given)
468                 {
469                   args_info->configfile_given = 1;
470                   if (fnum == 2)
471                     args_info->configfile_arg = gengetopt_strdup (farg);
472                   else
473                     {
474                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
475                                filename, line_num);
476                       exit (EXIT_FAILURE);
477                     }
478                 }
479               continue;
480             }
481           if (!strcmp(fopt, "priority"))
482             {
483               if (override || !args_info->priority_given)
484                 {
485                   args_info->priority_given = 1;
486                   if (fnum == 2)
487                     args_info->priority_arg = strtol (farg,&stop_char,0);
488                   else
489                     {
```

```
490                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
491                                filename, line_num);
492                       exit (EXIT_FAILURE);
493                     }
494                 }
495               continue;
496             }
497           if (!strcmp(fopt, "quantity"))
498             {
499               if (override || !args_info->quantity_given)
500                 {
501                   args_info->quantity_given = 1;
502                   if (fnum == 2)
503                     args_info->quantity_arg = strtol (farg,&stop_char,0);
504                   else
505                     {
506                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
507                                filename, line_num);
508                       exit (EXIT_FAILURE);
509                     }
510                 }
511               continue;
512             }
513           if (!strcmp(fopt, "destMAC"))
514             {
515               if (override || !args_info->destMAC_given)
516                 {
517                   args_info->destMAC_given = 1;
518                   if (fnum == 2)
519                     args_info->destMAC_arg = gengetopt_strdup (farg);
520                   else
521                     {
522                       fprintf (stderr, "%s:%d: required <option_name> <option_val>\n",
523                                filename, line_num);
524                       exit (EXIT_FAILURE);
525                     }
526                 }
527               continue;
528             }
529
530
531           /* Tried all known options. This one is unknown! */
532           fprintf (stderr, "%s: Unknown option '%s' found in %s\n",
533                    CMDLINE_PARSER_PACKAGE, fopt, filename);
534           exit (EXIT_FAILURE);
535         }
536     } /* while */
537   fclose(file); /* No error checking on close */
538
539   return 0;
540 }
```

Here is the call graph for this function:

4.2.2.3 void cmdline_parser_print_help (void)

Definition at line 32 of file cmdline.c.

References CMDLINE_PARSER_PACKAGE, and cmdline_parser_print_version().

Referenced by cmdline_parser().

```
33 {
34   cmdline_parser_print_version ();
35   printf("\n"
36   "Purpose:\n"
37   "  stressnet is intended for replaying tcpdump/pcap capture files \n"
38   "  -except the ethernet address- with a given bitrate to stress network \n"
39   "  devices such as intrusion detection systems or firewalls.  It mixes two pcap\n"
40   "  files (A and B) with a ratio r given as argument and meaning: for every 1 \n"
41   "  packet of file B that is sent, there are r packets of file A that are sent. \n"
42   "  This is meant to impose a rate of malformed packets in the dataflow.\n"
43   "  \n"
44   "  THIS SOFTWARE CAN AND SHOULD BE IMPROVED AS MUCH AS POSSIBLE, new ideas and \n"
45   "  critics are always welcome.  You can send them by email to the following \n"
46   "  address:\n"
47   "  \n"
48   "  \tyannick AT loth.be\n"
49   "  \n"
50   "  I would be pleased to discuss about them with you and to exchange ideas about\n"
51   "  further improvements.\n"
52   "  \n"
53   "  This software was initially developed for the Royal Military Academy (RMA) of\n"
54   "  Belgium (Brussels), by a student (Yannick Loth) during a training for his \n"
55   "  studies as Industrial Engineer in Applied Informatics at the  University of \n"
56   "  Luxembourg (2005).\n"
57   "  The aim of stressnet is to provide a reliable tool (i.e. intended for \n"
58   "  engineers etc.) to send packets (well formed and malformed ones) through a \n"
59   "  network at different speeds and ratios of innofensive packets over attacking \n"
60   "  packets to establish the limits of devices when they are seen as 'blackboxes',\n"
61   "  i.e. when the internal functioning of the devices is unknown.\n"
62   "  'Reliable' means that when someone asks for a certain bitrate, it should send\n"
63   "  data at bitrates quite close to the wanted bitrate.\n"
64   "  \n"
65   "  The author of this software thanks Maj. W. Mees (RMA) and Capt. O. Thonnard \n"
66   "  (RMA) as well as Pr. Th. Engel (Uni. Lux.) for their advices.\n"
67   "  \n"
68   "  stressnet should only be used by people knowing what they do, this tool could \n"
69   "  in fact block your computer for a long time if you don't take care of what you\n"
70   "  do.\n"
71   "  \n"
72   "  You'll (one day, I hope...) find more about this program on \n"
73   "  \thttp://www.loth.be/yannick/stressnet/index.html\n"
74   "\n"
75   "Usage: %s [OPTIONS]...\n", CMDLINE_PARSER_PACKAGE);
76   printf("   -h         --help               Print help and exit\n");
77   printf("   -V         --version            Print version and exit\n");
78   printf("   -iSTRING   --interface=STRING   network device used to send packets (default='eth0')\n");
79   printf("   -mSTRING   --packetfileM=STRING First file of packets\n");
80   printf("   -nSTRING   --packetfileN=STRING Second file ofpackets\n");
81   printf("   -rLONG     --ratio=LONG         ratio M/N: number of packets of file M for every 1 packet
82   printf("   -bLONG     --bitrate=LONG       desired send bitrate in kbit/s (default='100')\n");
83   printf("   -cSTRING   --configfile=STRING  config file containing the command line arguments\n");
84   printf("   -pINT      --priority=INT       scheduling priority value (default='0')\n");
85   printf("   -qINT      --quantity=INT       total quantity of packets to send (default='100')\n");
86   printf("   -dSTRING   --destMAC=STRING     destination MAC address for ALL packets (default='00:11:2
87 }
```

Here is the call graph for this function:

4.2.2.4 void cmdline_parser_print_version (void)

Definition at line 26 of file cmdline.c.

References CMDLINE_PARSER_PACKAGE, and CMDLINE_PARSER_VERSION.

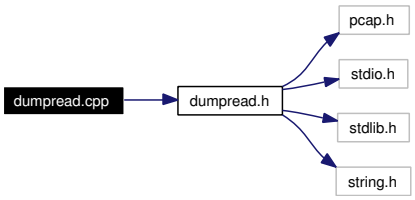
Referenced by cmdline_parser(), and cmdline_parser_print_help().

```
27 {
28   printf ("%s %s\n", CMDLINE_PARSER_PACKAGE, CMDLINE_PARSER_VERSION);
29 }
```

4.3 dumpread.cpp File Reference

```
#include "dumpread.h"
```

Include dependency graph for dumpread.cpp:



Functions

- void readDumpfilePackets (char ∗const fileM, char ∗const fileN, int totalQuantity, int quantityN, int ratio, char ∗const dataBuffer, struct timeval ∗const timingArray, int ∗const lengthArray)

  readDumpfilePackets() iterates through the packets in the tcpdump formatted file to copy the packets which will be sent into the buffers

- void readDumpfileStats (char ∗const file, int quantity, int ∗const length)

  readDumpfileStats() iterates through the packets in the tcpdump formatted files to determine which sizes will be allocated to the buffers

4.3.1 Function Documentation

4.3.1.1 void readDumpfilePackets (char ∗const *fileM*, char ∗const *fileN*, int *totalQuantity*, int *quantityN*, int *ratio*, char ∗const *dataBuffer*, struct timeval ∗const *timingArray*, int ∗const *lengthArray*)

readDumpfilePackets() iterates through the packets in the tcpdump formatted file to copy the packets which will be sent into the buffers

Definition at line 74 of file dumpread.cpp.

Referenced by main().

```
76 {
77         /*Locals*/
78         pcap_t *descrM;
79         pcap_t *descrN;
80         char errbuf[ PCAP_ERRBUF_SIZE ];
81         int i;
82         struct pcap_pkthdr *packetHeader;
83         const u_char *packetData;
84         int bytes;
```

```
85          int result;
86          int counter;
87          div_t divisionResult;
88          div_t divisionResult10th;
89          /*Initialise buffers and variables*/
90          memset( errbuf, 0, PCAP_ERRBUF_SIZE ); //initialise the buffer to all zeros
91          i = 0;
92          bytes = 0;
93          result = 0;
94          counter = 0;
95          descrM = NULL;
96          descrN = NULL;
97
98          /*divide totalQuantity/10 to print dots each time 10% of packets have been processed*/
99          divisionResult10th=div(totalQuantity,10);
100
101          /*Printing a message to indicate the the files are being read*/
102          if(quantityN!=0)
103                  printf("The files %s and %s are being processed,\nthis may take a while.\n",fileM, fil
104          else
105                  printf("The file %s is being processed,\nthis may take a while.\n",fileM);
106
107          /*Begin reading the file*/
108          descrM = pcap_open_offline( fileM, errbuf );
109          if ( descrM == NULL )
110          {
111                  printf( "%s\n", errbuf );
112                  exit( 1 );
113          }
114          if ( quantityN != 0 )
115          {
116                  descrN = pcap_open_offline( fileN, errbuf );
117                  if ( descrN == NULL )
118                  {
119                          printf( "%s\n", errbuf );
120                          exit( 1 );
121                  }
122          }
123          else descrN = NULL;
124
125          while ( counter < totalQuantity )
126          {
127                  for ( i = 0;i < ratio;++i )  //read ratio packets from M
128                  {
129                          result = pcap_next_ex ( descrM, &packetHeader, &packetData );
130                          if ( ( result != -2 ) && ( result != -1 ) )
131                          {
132                                  memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen );
133                                  bytes += packetHeader->caplen;
134                                  memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof( struct
135                                  lengthArray[ counter ] = packetHeader->caplen;
136                          }
137                          else
138                          {
139                                  pcap_close( descrM );
140                                  descrM = pcap_open_offline( fileM, errbuf );
141                                  if ( descrM == NULL )
142                                  {
143                                          printf( "%s\n", errbuf );
144                                          exit( 1 );
145                                  }
146                                  result = pcap_next_ex ( descrM, &packetHeader, &packetData );
147                                  if ( ( result != -2 ) && ( result != -1 ) )
148                                  {
149                                          memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen
150                                          bytes += packetHeader->caplen;
151                                          memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof
```

```
152                                         lengthArray[ counter ] = packetHeader->caplen;
153                                 }
154                                 else
155                                 {
156                                         printf( "Error obtaining packets from the file.\n" );
157                                         exit( 1 );
158                                 }
159                         }
160
161                         if ( ++counter >= totalQuantity ) break;
162                 }
163                 if ( ( quantityN != 0 ) && ( counter < totalQuantity ) ) //read 1 packet from N
164                 {
165                         result = pcap_next_ex ( descrN, &packetHeader, &packetData );
166                         if ( ( result != -2 ) && ( result != -1 ) )
167                         {
168                                 memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen );
169                                 bytes += packetHeader->caplen;
170                                 memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof( struct
171                                 lengthArray[ counter ] = packetHeader->caplen;
172                         }
173                         else
174                         {
175                                 pcap_close( descrN );
176                                 descrN = pcap_open_offline( fileN, errbuf );
177                                 if ( descrN == NULL )
178                                 {
179                                         printf( "%s\n", errbuf );
180                                         exit( 1 );
181                                 }
182                                 result = pcap_next_ex ( descrN, &packetHeader, &packetData );
183                                 if ( ( result != -2 ) && ( result != -1 ) )
184                                 {
185                                         memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen
186                                         bytes += packetHeader->caplen;
187                                         memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof
188                                         lengthArray[ counter ] = packetHeader->caplen;
189                                 }
190                                 else
191                                 {
192                                         printf( "Error obtaining packets from the file.\n" );
193                                         exit( 1 );
194                                 }
195                         }
196                         ++counter;
197                 }
198                 divisionResult=div(counter,divisionResult10th.quot);
199                 if (divisionResult.rem==0)
200                 {
201                         printf("%d%c ",divisionResult.quot*10,'%');
202                         fflush(stdout);
203                 }
204         }
205         printf("\n");
206         pcap_close( descrM );
207         if ( quantityN != 0 )
208                 pcap_close( descrN );
209 }
```

4.3.1.2 void readDumpfileStats (char *const *file*, int *quantity*, int *const *length*)

readDumpfileStats() iterates through the packets in the tcpdump formatted files to determine which sizes will be allocated to the buffers

Definition at line 21 of file dumpread.cpp.

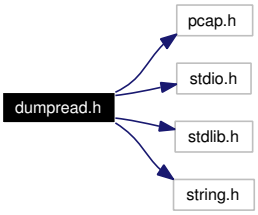
Referenced by main().

```
22 {
23         /*Locals*/
24         pcap_t * descr;
25         char errbuf[ PCAP_ERRBUF_SIZE ];
26         int i;
27         struct pcap_pkthdr *packetHeader;
28         const u_char * packetData; //the value pointed to is constant, not the pointer!
29         int bytes;
30         int result;
31
32         printf("Reading stats from file(s). This may take a while.\n");
33         /*Initialise buffers and variables*/
34         memset( errbuf, 0, PCAP_ERRBUF_SIZE ); //initialise the buffer to all zeros
35         i = 0;
36         bytes = 0;
37         result = 0;
38
39         /*Begin reading the file*/
40         descr = pcap_open_offline( file, errbuf );
41         if ( descr == NULL )
42         {
43                 printf( "%s\n", errbuf );
44                 exit( 1 );
45         }
46
47         for ( i = 0;i < quantity;++i )
48         {
49                 result = pcap_next_ex ( descr, &packetHeader, &packetData );
50                 if ( ( result != -2 ) && ( result != -1 ) )
51                         bytes += packetHeader->caplen;
52                 else
53                 {
54                         pcap_close( descr );
55                         descr = pcap_open_offline( file, errbuf );
56                         if ( descr == NULL )
57                         {
58                                 printf( "%s\n", errbuf );
59                                 exit( 1 );
60                         }
61                         result = pcap_next_ex ( descr, &packetHeader, &packetData );
62                         if ( ( result != -2 ) && ( result != -1 ) )
63                                 bytes += packetHeader->caplen;
64                         else
65                         {
66                                 printf( "Error obtaining packets from the file.\n" );
67                                 exit( 1 );
68                         }
69                 }
70         }
71         ( *length ) = bytes;
72         pcap_close( descr );
73 }
```

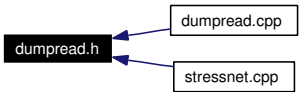
4.4 dumpread.h File Reference

```
#include <pcap.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for dumpread.h:



This graph shows which files directly or indirectly include this file:



Functions

- void readDumpfilePackets (char ∗const fileM, char ∗const fileN, int totalQuantity, int quantityN, int ratio, char ∗const dataBuffer, struct timeval ∗const timingArray, int ∗const lengthArray)

  readDumpfilePackets() iterates through the packets in the tcpdump formatted file to copy the packets which will be sent into the buffers

- void readDumpfileStats (char ∗const file, int quantity, int ∗const length)

  readDumpfileStats() iterates through the packets in the tcpdump formatted files to determine which sizes will be allocated to the buffers

4.4.1 Function Documentation

4.4.1.1 void readDumpfilePackets (char ∗const *fileM*, char ∗const *fileN*, int *totalQuantity*, int *quantityN*, int *ratio*, char ∗const *dataBuffer*, struct timeval ∗const *timingArray*, int ∗const *lengthArray*)

readDumpfilePackets() iterates through the packets in the tcpdump formatted file to copy the packets which will be sent into the buffers

Definition at line 74 of file dumpread.cpp.

Referenced by main().

```
76 {
77         /*Locals*/
78         pcap_t *descrM;
79         pcap_t *descrN;
80         char errbuf[ PCAP_ERRBUF_SIZE ];
81         int i;
82         struct pcap_pkthdr *packetHeader;
83         const u_char *packetData;
84         int bytes;
85         int result;
86         int counter;
87         div_t divisionResult;
88         div_t divisionResult10th;
89         /*Initialise buffers and variables*/
90         memset( errbuf, 0, PCAP_ERRBUF_SIZE ); //initialise the buffer to all zeros
91         i = 0;
```

```
92          bytes = 0;
93          result = 0;
94          counter = 0;
95          descrM = NULL;
96          descrN = NULL;
97
98          /*divide totalQuantity/10 to print dots each time 10% of packets have been processed*/
99          divisionResult10th=div(totalQuantity,10);
100
101          /*Printing a message to indicate the the files are being read*/
102          if(quantityN!=0)
103                  printf("The files %s and %s are being processed,\nthis may take a while.\n",fileM, fil
104          else
105                  printf("The file %s is being processed,\nthis may take a while.\n",fileM);
106
107          /*Begin reading the file*/
108          descrM = pcap_open_offline( fileM, errbuf );
109          if ( descrM == NULL )
110          {
111                  printf( "%s\n", errbuf );
112                  exit( 1 );
113          }
114          if ( quantityN != 0 )
115          {
116                  descrN = pcap_open_offline( fileN, errbuf );
117                  if ( descrN == NULL )
118                  {
119                          printf( "%s\n", errbuf );
120                          exit( 1 );
121                  }
122          }
123          else descrN = NULL;
124
125          while ( counter < totalQuantity )
126          {
127                  for ( i = 0;i < ratio;++i )  //read ratio packets from M
128                  {
129                          result = pcap_next_ex ( descrM, &packetHeader, &packetData );
130                          if ( ( result != -2 ) && ( result != -1 ) )
131                          {
132                                  memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen );
133                                  bytes += packetHeader->caplen;
134                                  memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof( struct
135                                  lengthArray[ counter ] = packetHeader->caplen;
136                          }
137                          else
138                          {
139                                  pcap_close( descrM );
140                                  descrM = pcap_open_offline( fileM, errbuf );
141                                  if ( descrM == NULL )
142                                  {
143                                          printf( "%s\n", errbuf );
144                                          exit( 1 );
145                                  }
146                                  result = pcap_next_ex ( descrM, &packetHeader, &packetData );
147                                  if ( ( result != -2 ) && ( result != -1 ) )
148                                  {
149                                          memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen
150                                          bytes += packetHeader->caplen;
151                                          memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof
152                                          lengthArray[ counter ] = packetHeader->caplen;
153                                  }
154                                  else
155                                  {
156                                          printf( "Error obtaining packets from the file.\n" );
157                                          exit( 1 );
158                                  }
```

```
159                         }
160
161                         if ( ++counter >= totalQuantity ) break;
162                 }
163                 if ( ( quantityN != 0 ) && ( counter < totalQuantity ) ) //read 1 packet from N
164                 {
165                         result = pcap_next_ex ( descrN, &packetHeader, &packetData );
166                         if ( ( result != -2 ) && ( result != -1 ) )
167                         {
168                                 memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen );
169                                 bytes += packetHeader->caplen;
170                                 memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof( struct
171                                 lengthArray[ counter ] = packetHeader->caplen;
172                         }
173                         else
174                         {
175                                 pcap_close( descrN );
176                                 descrN = pcap_open_offline( fileN, errbuf );
177                                 if ( descrN == NULL )
178                                 {
179                                         printf( "%s\n", errbuf );
180                                         exit( 1 );
181                                 }
182                                 result = pcap_next_ex ( descrN, &packetHeader, &packetData );
183                                 if ( ( result != -2 ) && ( result != -1 ) )
184                                 {
185                                         memcpy( &dataBuffer[ bytes ], packetData, packetHeader->caplen
186                                         bytes += packetHeader->caplen;
187                                         memcpy( &timingArray[ counter ], &( packetHeader->ts ), sizeof
188                                         lengthArray[ counter ] = packetHeader->caplen;
189                                 }
190                                 else
191                                 {
192                                         printf( "Error obtaining packets from the file.\n" );
193                                         exit( 1 );
194                                 }
195                         }
196                         ++counter;
197                 }
198                 divisionResult=div(counter,divisionResult10th.quot);
199                 if (divisionResult.rem==0)
200                 {
201                         printf("%d%c ",divisionResult.quot*10,'%');
202                         fflush(stdout);
203                 }
204         }
205         printf("\n");
206         pcap_close( descrM );
207         if ( quantityN != 0 )
208                 pcap_close( descrN );
209 }
```

4.4.1.2 void readDumpfileStats (char *const *file*, int *quantity*, int *const *length*)

readDumpfileStats() iterates through the packets in the tcpdump formatted files to determine which sizes will be allocated to the buffers

Definition at line 21 of file dumpread.cpp.

Referenced by main().

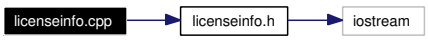
```
22 {
23         /*Locals*/
24         pcap_t * descr;
25         char errbuf[ PCAP_ERRBUF_SIZE ];
```

```
26          int i;
27          struct pcap_pkthdr *packetHeader;
28          const u_char * packetData; //the value pointed to is constant, not the pointer!
29          int bytes;
30          int result;
31
32          printf("Reading stats from file(s). This may take a while.\n");
33          /*Initialise buffers and variables*/
34          memset( errbuf, 0, PCAP_ERRBUF_SIZE ); //initialise the buffer to all zeros
35          i = 0;
36          bytes = 0;
37          result = 0;
38
39          /*Begin reading the file*/
40          descr = pcap_open_offline( file, errbuf );
41          if ( descr == NULL )
42          {
43                  printf( "%s\n", errbuf );
44                  exit( 1 );
45          }
46
47          for ( i = 0;i < quantity;++i )
48          {
49                  result = pcap_next_ex ( descr, &packetHeader, &packetData );
50                  if ( ( result != -2 ) && ( result != -1 ) )
51                          bytes += packetHeader->caplen;
52                  else
53                  {
54                          pcap_close( descr );
55                          descr = pcap_open_offline( file, errbuf );
56                          if ( descr == NULL )
57                          {
58                                  printf( "%s\n", errbuf );
59                                  exit( 1 );
60                          }
61                          result = pcap_next_ex ( descr, &packetHeader, &packetData );
62                          if ( ( result != -2 ) && ( result != -1 ) )
63                                  bytes += packetHeader->caplen;
64                          else
65                          {
66                                  printf( "Error obtaining packets from the file.\n" );
67                                  exit( 1 );
68                          }
69                  }
70          }
71          ( *length ) = bytes;
72          pcap_close( descr );
73  }
```

4.5 licenseinfo.cpp File Reference

`#include "licenseinfo.h"`

Include dependency graph for licenseinfo.cpp:



Functions

- void printGPLText (int year)

  printGPLText() prints on screen that the software is free and subject to GPL version >= 2.

4.5.1 Function Documentation

4.5.1.1 void printGPLText (int *year*)

printGPLText() prints on screen that the software is free and subject to GPL version >= 2.

Parameters:

year integer indicating when the source-code was written.

Definition at line 21 of file licenseinfo.cpp.

Referenced by main().

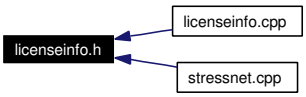
```
22 {
23 using namespace std;
24 cout<<"*******************************************************************************"<<endl;
25 cout<<"*   Copyright (C) "<<year<<" by Yannick Loth                                  *"<<endl;
26 cout<<"*   yannick@loth.be                                                           *"<<endl;
27 cout<<"*                                                                             *"<<endl;
28 cout<<"*   This program is free software; you can redistribute it and/or modify      *"<<endl;
29 cout<<"*   it under the terms of the GNU General Public License as published by      *"<<endl;
30 cout<<"*   the Free Software Foundation; either version 2 of the License, or         *"<<endl;
31 cout<<"*   (at your option) any later version.                                       *"<<endl;
32 cout<<"*                                                                             *"<<endl;
33 cout<<"*   This program is distributed in the hope that it will be useful,           *"<<endl;
34 cout<<"*   but WITHOUT ANY WARRANTY; without even the implied warranty of            *"<<endl;
35 cout<<"*   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the             *"<<endl;
36 cout<<"*   GNU General Public License for more details.                              *"<<endl;
37 cout<<"*                                                                             *"<<endl;
38 cout<<"*   You should have received a copy of the GNU General Public License         *"<<endl;
39 cout<<"*   along with this program; if not, write to the                             *"<<endl;
40 cout<<"*   Free Software Foundation, Inc.,                                           *"<<endl;
41 cout<<"*   59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.                 *"<<endl;
42 cout<<"*******************************************************************************"<<endl;
43 cout<<endl;
44 }
```

4.6 licenseinfo.h File Reference

```
#include <iostream>
```

Include dependency graph for licenseinfo.h:

This graph shows which files directly or indirectly include this file:



Functions

- void printGPLText (int year)

  printGPLText() prints on screen that the software is free and subject to GPL version >= 2.

4.6.1 Function Documentation

4.6.1.1 void printGPLText (int *year*)

printGPLText() prints on screen that the software is free and subject to GPL version >= 2.

Parameters:

year integer indicating when the source-code was written.

Definition at line 21 of file licenseinfo.cpp.

Referenced by main().

```
22 {
23 using namespace std;
24 cout<<"*******************************************************************************"<<endl;
25 cout<<"*   Copyright (C) "<<year<<" by Yannick Loth                                  *"<<endl;
26 cout<<"*   yannick@loth.be                                                           *"<<endl;
27 cout<<"*                                                                             *"<<endl;
28 cout<<"*   This program is free software; you can redistribute it and/or modify      *"<<endl;
29 cout<<"*   it under the terms of the GNU General Public License as published by      *"<<endl;
30 cout<<"*   the Free Software Foundation; either version 2 of the License, or         *"<<endl;
31 cout<<"*   (at your option) any later version.                                       *"<<endl;
32 cout<<"*                                                                             *"<<endl;
33 cout<<"*   This program is distributed in the hope that it will be useful,           *"<<endl;
34 cout<<"*   but WITHOUT ANY WARRANTY; without even the implied warranty of            *"<<endl;
35 cout<<"*   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the             *"<<endl;
36 cout<<"*   GNU General Public License for more details.                              *"<<endl;
37 cout<<"*                                                                             *"<<endl;
38 cout<<"*   You should have received a copy of the GNU General Public License         *"<<endl;
39 cout<<"*   along with this program; if not, write to the                             *"<<endl;
40 cout<<"*   Free Software Foundation, Inc.,                                           *"<<endl;
41 cout<<"*   59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.                 *"<<endl;
42 cout<<"*******************************************************************************"<<endl;
43 cout<<endl;
44 }
```

4.7 memmanagement.c File Reference

```
#include "memmanagement.h"
```

Include dependency graph for memmanagement.c:

Functions

- void * callocBuffer (size_t nbElem, size_t elemSize)

  callocBuffer() trys to allocates a buffer of all zeros

4.7.1 Function Documentation

4.7.1.1 void* callocBuffer (size_t *nbElem*, size_t *elemSize*)

callocBuffer() trys to allocates a buffer of all zeros

Definition at line 21 of file memmanagement.c.

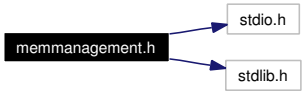
Referenced by main().

```
22 {
23         void *ptr;
24         ptr=calloc(nbElem,elemSize);
25         if(ptr==NULL)
26         {
27                 printf("Memory allocation failed. Try to free more memory or send less\nand/or shorter
28                 exit(1);
29         }
30         return(ptr);
31 }
```

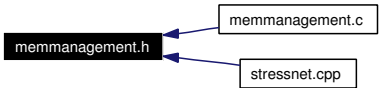
4.8 memmanagement.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
```

Include dependency graph for memmanagement.h:



This graph shows which files directly or indirectly include this file:



Functions

- void * callocBuffer (size_t nbElem, size_t elemSize)
  callocBuffer() trys to allocates a buffer of all zeros

4.8.1 Function Documentation

4.8.1.1 void* callocBuffer (size_t *nbElem*, size_t *elemSize*)

callocBuffer() trys to allocates a buffer of all zeros

Definition at line 21 of file memmanagement.c.

Referenced by main().

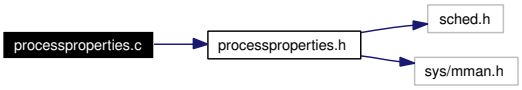
```
22 {
23         void *ptr;
24         ptr=calloc(nbElem,elemSize);
25         if(ptr==NULL)
26         {
```

```
27                 printf("Memory allocation failed. Try to free more memory or send less\nand/or shorter
28                 exit(1);
29         }
30         return(ptr);
31 }
```

4.9 processproperties.c File Reference

```
#include "processproperties.h"
```

Include dependency graph for processproperties.c:



Functions

- void setSchedulingPolicyFIFO (int priority)

  setSchedulingPolicyFIFO() sets the scheduling policy to the linux real time policy and the priority to the given value

4.9.1 Function Documentation

4.9.1.1 void setSchedulingPolicyFIFO (int *priority*)

setSchedulingPolicyFIFO() sets the scheduling policy to the linux real time policy and the priority to the given value

Definition at line 21 of file processproperties.c.

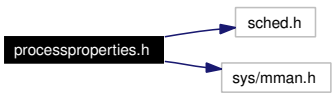
Referenced by main().

```
22 {
23         /* Declarations */
24         int returnvalues;
25         struct sched_param schedul;
26         schedul.sched_priority = priority;
27         returnvalues = sched_setscheduler( 0, SCHED_FIFO, &schedul );
28 }
```

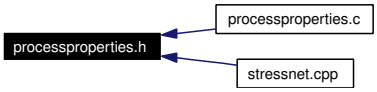
4.10 processproperties.h File Reference

```
#include <sched.h>
#include <sys/mman.h>
```

Include dependency graph for processproperties.h:



This graph shows which files directly or indirectly include this file:



Functions

- void setSchedulingPolicyFIFO (int priority)

  setSchedulingPolicyFIFO() sets the scheduling policy to the linux real time policy and the priority to the given value

4.10.1 Function Documentation

4.10.1.1 void setSchedulingPolicyFIFO (int *priority*)

setSchedulingPolicyFIFO() sets the scheduling policy to the linux real time policy and the priority to the given value

Definition at line 21 of file processproperties.c.

Referenced by main().

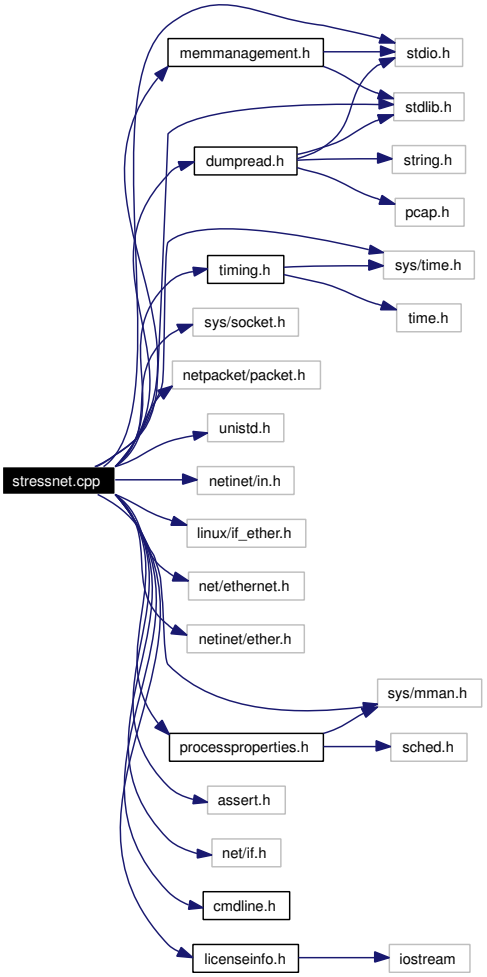
```
22 {
23         /* Declarations */
24         int returnvalues;
25         struct sched_param schedul;
26         schedul.sched_priority = priority;
27         returnvalues = sched_setscheduler( 0, SCHED_FIFO, &schedul );
28 }
```

4.11 stressnet.cpp File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <netpacket/packet.h>
#include <unistd.h>
#include <netinet/in.h>
#include <linux/if_ether.h>
#include <net/ethernet.h>
#include <netinet/ether.h>
#include <sys/mman.h>
#include <assert.h>
#include <net/if.h>
```

```
#include "cmdline.h"
#include "dumpread.h"
#include "licenseinfo.h"
#include "memmanagement.h"
#include "processproperties.h"
#include "timing.h"
```

Include dependency graph for stressnet.cpp:



Functions

- int main (int argc, char ∗argv[ ])

4.11.1 Function Documentation

4.11.1.1 int main (int *argc*, char ∗ *argv*[ ])

Definition at line 46 of file stressnet.cpp.

References gengetopt_args_info::bitrate_arg, calculateOffset(), callocBuffer(), cmdline_parser(), cmdline_parser_configfile(), gengetopt_args_info::configfile_arg, gengetopt_args_info::destMAC_arg, gengetopt_args_info::destMAC_given, gengetopt_args_info::interface_arg, measureGettimeofday(), gengetopt_args_info::packetfileM_arg, gengetopt_args_info::packetfileN_arg, gengetopt_args_info::packetfileN_given, printGPLText(), gengetopt_args_info::priority_arg, gengetopt_args_info::quantity_arg, gengetopt_args_info::ratio_arg, readDumpfilePackets(), readDumpfileStats(), setSchedulingPolicyFIFO(), and timevalAdd().

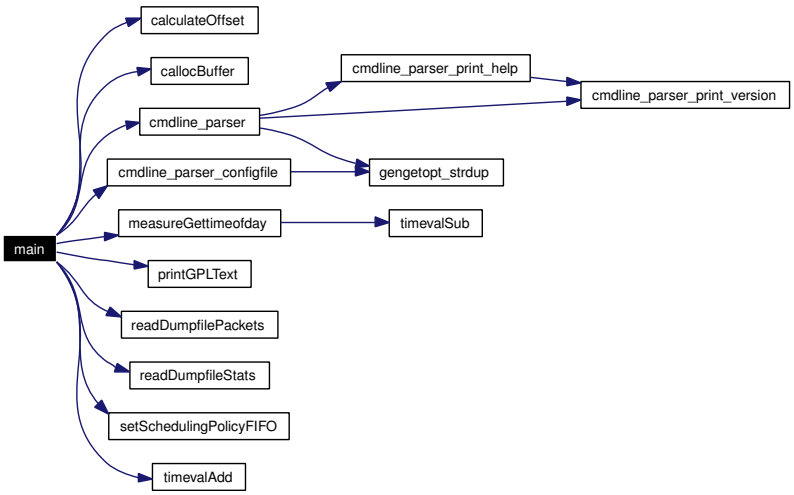
```
47 {
48         int bufferLength;
49         int bytesM;
50         int bytesN;
51         int quantityM;
52         int quantityN;
53         int i;
54         int position;
55         int bindResult;
56         int sendtoResult;
57         int setsockoptResult;
58         struct timeval refTime;
59         struct timeval now;
60         struct timeval startTimeOffset;
61         double dRefTime;
62         double dNow;
63         double gettimeofdayTime;
64         struct ether_header *pEtherHeader;
65         struct ether_addr *pEtherAddr;
66         struct gengetopt_args_info args_info;
67         char **packetPointer;
68         double *dOffsetArray;
69         double *dTimingArray;
70         int *lengthArray;
71         int *positionArray;
72         struct timeval *timingArray;
73         int *sent;
74
75         /*Print GPL info*/
76         printGPLText(2005);
77
78         /* Process command line arguments */
79         //Cmdline parser
80         if ( cmdline_parser ( argc, argv, &args_info ) != 0 )
81                 exit( 1 ) ;
82         //Config file parser
83         if ( args_info.configfile_arg != NULL )
84                 if ( cmdline_parser_configfile ( args_info.configfile_arg, &args_info, 1 ) != 0 )
85                         exit( 1 );
86
87         /*Determine for each file the quantity of packets that'll be sent*/
88         args_info.quantity_arg = args_info.quantity_arg;
89         if ( args_info.packetfileN_given )
90         {
91                 div_t divisionResult;//Integer division faster with div() than with /
92                 divisionResult=div(args_info.quantity_arg,args_info.ratio_arg+1);
93                 quantityM = divisionResult.quot * args_info.ratio_arg + divisionResult.rem;
94                 quantityN = divisionResult.quot;
95         }
96         else
97         {
98                 quantityM = args_info.quantity_arg;
99                 quantityN = 0;
100         }
101
102         /*Initialise the variables*/
103         lengthArray=(int *)callocBuffer(args_info.quantity_arg,sizeof(int));
104         timingArray = (struct timeval *)callocBuffer(args_info.quantity_arg,sizeof(struct timeval)); /
```

```
105         dOffsetArray=(double *)callocBuffer(args_info.quantity_arg,sizeof(double));
106         dTimingArray = (double *)callocBuffer(args_info.quantity_arg, sizeof(double));
107         positionArray = (int *)callocBuffer(args_info.quantity_arg, sizeof(int));
108         packetPointer = (char **)callocBuffer(args_info.quantity_arg, sizeof(char *));
109         sent=(int *)callocBuffer(args_info.quantity_arg,sizeof(int));
110         bufferLength = 0;
111         bytesM = 0;
112         bytesN = 0;
113         dNow=0;
114         gettimeofdayTime = measureGettimeofday()/2;
115
116         /*Determine the length of the data buffer*/
117         readDumpfileStats( args_info.packetfileM_arg, quantityM, &bytesM );
118         if ( args_info.packetfileN_given )
119                 readDumpfileStats( args_info.packetfileN_arg, quantityN, &bytesN );
120         bufferLength = bytesM + bytesN;
121         printf( "Buffer length: %d\n", bufferLength );
122         char * const dataBuffer = new char[ bufferLength ];
123
124         /*Now put the packets in the buffer*/
125         readDumpfilePackets( args_info.packetfileM_arg, args_info.packetfileN_arg, args_info.quantity_
126         printf( "Number of packets to send: %d\n", args_info.quantity_arg );
127         //Calculate positions of each packet in data-buffer
128         packetPointer[ 0 ] = dataBuffer;
129         for ( i = 1;i < args_info.quantity_arg;++i )
130         {
131                 positionArray[ i ] = positionArray[ i - 1 ] + lengthArray[ i - 1 ];
132                 packetPointer[ i ] = dataBuffer + positionArray[ i ];
133         }
134
135         /*Check wether the addresses in positionArray are within dataBuffer*/
136         for ( i = 0;i < args_info.quantity_arg;++i )
137         {
138                 if (packetPointer[ i ] < dataBuffer)
139                 {
140                         printf("Packet %d address before buffer!\n",i);
141                         exit(1);
142                 }
143                 else if (packetPointer [ i ] > (dataBuffer + bufferLength - lengthArray[args_info.quan
144                 {
145                         printf("Packet %d address after buffer!\n",i);
146                         exit(1);
147                 }
148         }
149
150         //Modify ethernet addresses to unique address if any given as argument
151         if(args_info.destMAC_given!=0)
152                 for ( i = 0;i < args_info.quantity_arg;++i )
153                 {
154                         pEtherHeader=(ether_header *)packetPointer[i];
155                         pEtherAddr=ether_aton(args_info.destMAC_arg);
156                         memcpy(pEtherHeader->ether_dhost,pEtherAddr->ether_addr_octet,ETH_ALEN);
157                 }
158         free(positionArray);//Array not needed any more
159         //Calculate timings
160         calculateOffset( lengthArray, timingArray, args_info.quantity_arg, args_info.bitrate_arg);
161         for ( i = 0 ; i < args_info.quantity_arg ; ++i )
162         {
163                 dOffsetArray[i]=timingArray[i].tv_sec+timingArray[i].tv_usec*1e-6;
164         }
165         //On modern hardware, 1s should be enough to do all calculations before send
166         startTimeOffset.tv_sec = 1;
167         startTimeOffset.tv_usec = 0;
168         gettimeofday( &refTime, NULL );
169         timevalAdd( &refTime, &startTimeOffset );
170         dRefTime = refTime.tv_sec + refTime.tv_usec * 1e-6;
171         //Calculate sending times
```

```
172         printf("\tCalculating sending times...\n");
173         for ( i = 0;i < args_info.quantity_arg;++i )
174         {
175                 dRefTime += timingArray[ i ].tv_sec + timingArray[ i ].tv_usec * 1e-6;
176                 dTimingArray[ i ] = dRefTime;
177         }
178         /*Lock buffers in physical memory (=disable swapping of these buffers)*/
179         mlock( dataBuffer, bufferLength );
180         mlock( dTimingArray, args_info.quantity_arg * sizeof( double ) );
181         mlock( lengthArray, args_info.quantity_arg * sizeof( int ) );
182         mlock( packetPointer, args_info.quantity_arg * sizeof( char * ) );
183         mlock( sent, args_info.quantity_arg * sizeof( int ) );
184         /*Create the socket and send packets*/
185         printf("\tCreating the socket...\n");
186         int socketFileDescriptor;
187         struct sockaddr to;
188         int tolen;
189         //Create the SOCK_PACKET socket:
190         socketFileDescriptor = socket( PF_PACKET, SOCK_RAW, htons( ETH_P_ALL ) );
191         assert( socketFileDescriptor >= 0 );
192         printf("\tSocket successfully created...\n");
193         //Set socket options:
194         int optval;
195         optval=(int)2.1*bufferLength; //set enough kernel buffer space
196         setsockoptResult=setsockopt(socketFileDescriptor, SOL_SOCKET, SO_SNDBUF, &optval, sizeof(int))
197         //Bind the socket to the desired interface(necessary to know where to send the packets):
198         struct sockaddr_ll sSockAddr;
199         unsigned int sockaddr_llSize;
200         sockaddr_llSize=sizeof(struct sockaddr_ll);
201         memset( &sSockAddr, '\0', sizeof( struct sockaddr_ll ) );
202     sSockAddr.sll_family = AF_PACKET;
203         sSockAddr.sll_protocol = htons(ETH_P_ALL);
204         sSockAddr.sll_ifindex = if_nametoindex(args_info.interface_arg);
205         bindResult = bind( socketFileDescriptor, (struct sockaddr *) &sSockAddr, sizeof( sSockAddr ) )
206         assert( bindResult >= 0 );
207         printf("\tSocket successfully bound to device %s...\n",args_info.interface_arg);
208         //Prepares for sendto()
209         memset( &to , '\0', sizeof( to ) );
210         to.sa_family = AF_INET;
211         strcpy( to.sa_data, args_info.interface_arg );
212         tolen = sizeof( to );
213         position = 0;
214         //Change scheduling policy and priority
215         if (args_info.priority_arg>0)
216         {
217                 printf("\tChanging scheduling policy (to FIFO) and priority...\n");
218                 setSchedulingPolicyFIFO( args_info.priority_arg );
219         }
220         int count;
221         count=0;
222         int errorOccured;
223         errorOccured=0;
224         double dTimePreviousPacket;
225         dTimePreviousPacket = 0.0;
226         printf("\tSending packets...\n\n");
227         if(args_info.bitrate_arg!=0)//0 for maximum speed (no busy-waiting)
228         {
229                 for ( i = 0;i < args_info.quantity_arg;++i )
230                 {
231                         /*Absolute precalculated timing*/
232                         for(;dNow < ( dTimingArray[ i ] - gettimeofdayTime);)
233                         {
234                                 gettimeofday( &now, NULL );
235                                 dNow = now.tv_sec + now.tv_usec * 1e-6;
236                         }
237
238                         sendtoResult = write( socketFileDescriptor,  packetPointer[ i ], lengthArray[
```

```
239                         if (sendtoResult>=0)
240                         {
241                                 ++count;
242                                 ++sent[i];
243                         }
244                         else if (errorOccured==0) //only prints the error if the previous packet was s
245                         {
246                                 errorOccured=1;
247                                 perror("sendto() return-value < 0 ");
248                                 printf("\tIf error 'No buffer space available',\n\ttry to put a unique
249                         }
250                 }
251         }
252         else //0 for maximum speed (no busy-waiting)
253         {
254                 for ( i = 0;i < args_info.quantity_arg;++i )
255                 {
256                         //No waiting...
257                         sendtoResult = write( socketFileDescriptor,  packetPointer[ i ], lengthArray[
258                         if (sendtoResult>=0)
259                         {
260                                 ++count;
261                                 ++sent[i];
262                         }
263                         else if (errorOccured==0) //only prints the error if the previous packet was s
264                         {
265                                 errorOccured=1;
266                                 perror("sendto() return-value < 0 ");
267                                 printf("\tIf error 'No buffer space available',\n\ttry to put a unique
268                         }
269                 }
270         }
271         printf("\nNumber of packets sent: %d\n",count);
272         int bytesSent;
273         bytesSent=0;
274         for(i=0;i<args_info.quantity_arg;++i)
275         {
276                 if (sent[i]!=0)
277                         bytesSent+=lengthArray[i];
278         }
279         printf("Number of bytes sent: %d\n",bytesSent);
280         /*Set scheduling priority to 0*/
281         setSchedulingPolicyFIFO( 0 );
282         /*Unlock buffers from memory*/
283         munlock( dataBuffer, bufferLength );
284         munlock( dTimingArray, args_info.quantity_arg * sizeof( double ) );
285         munlock( lengthArray, args_info.quantity_arg * sizeof( int ) );
286         munlock( packetPointer, args_info.quantity_arg * sizeof( char * ) );
287         munlock( sent, args_info.quantity_arg * sizeof( int ) );
288
289         /*Close everything opened and free memory*/
290         free(lengthArray);
291         free(dTimingArray);
292         free(dataBuffer);
293         free(packetPointer);
294         free(timingArray);
295         free(dOffsetArray);
296         close( socketFileDescriptor );
297         printf("%s exited.\n",argv[0]);
298         printf("*************************************\n");
299         return EXIT_SUCCESS;
300 }
```

Here is the call graph for this function:

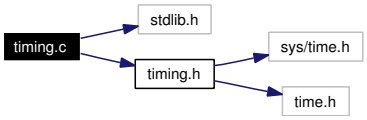


4.12 timing.c File Reference

```
#include <stdlib.h>
```

```
#include "timing.h"
```

Include dependency graph for timing.c:



Defines

- #define COUNT 1000000

Functions

- void calculateOffset (int *pLengthArray, struct timeval *tArray, int tArrayLength, int bitrate)

  calcultateOffset() calculates the time interval between the sending time of the previous packet and the sending time of the current packet

- double measureGettimeofday ()

  measureGettimeofday() measures the time needed for one execution of the waiting loop

- void timevalAdd (struct timeval *to, struct timeval *val)

  timevalAdd() adds two struct timeval

- void timevalSub (struct timeval *to, struct timeval *const val)

  timevalSub() substracts two struct timeval

4.12.1 Define Documentation

4.12.1.1 #define COUNT 1000000

Definition at line 22 of file timing.c.

Referenced by measureGettimeofday().

4.12.2 Function Documentation

4.12.2.1 void calculateOffset (int * *pLengthArray*, struct timeval * *tArray*, int *tArrayLength*, int *bitrate*)

calcultateOffset() calculates the time interval between the sending time of the previous packet and the sending time of the current packet

Definition at line 69 of file timing.c.

Referenced by main().

```
70 {
71         double brate;
72         double delta;
73         double intermediate;
74         int i;
75         int offset; //offset in microseconds, relative to the previous packet
76         brate = ( double ) bitrate;
77         intermediate=8*1e3;
78         delta = intermediate/brate;//intermediate to avoid loss of precision due to numerical division
79
80         /*Note that we've calculated all constants for the loop out of the loop to gain processing time
81         div_t divstruct;
82         for ( i = 0;i < tArrayLength;++i )
83         {
84                 offset = ( int ) ( ( pLengthArray[ i ] + 4 ) * delta ); //+4 because of FCS/CRC appende
85                 if ( offset >= 1000000 )
86                 {
87                         divstruct=div(offset,1000000);
88                         tArray[ i ].tv_sec = divstruct.quot; //transform the offset in (sec + usec) ins
89                         tArray[ i ].tv_usec = divstruct.rem/*offset % 1000000*/;
90                 }
91                 else
92                 {
93                         tArray[ i ].tv_sec = 0;
94                         tArray[ i ].tv_usec = offset;
95                 }
96         }
97 }
```

4.12.2.2 double measureGettimeofday ()

measureGettimeofday() measures the time needed for one execution of the waiting loop

Definition at line 23 of file timing.c.

References COUNT, and timevalSub().

Referenced by main().

```
24 {
25         int i;
26         struct timeval tv1;
```

```
27          struct timeval tv2;
28          gettimeofday( &tv1, NULL );
29          for ( i = 0;i < COUNT;++i )
30          {
31                  gettimeofday( &tv2, NULL );
32          }
33          timevalSub( &tv2, &tv1 );
34          return ( ( double ) ( tv2.tv_sec + tv2.tv_usec * 1e-6 ) / COUNT );//division after other operat
35  }
```

Here is the call graph for this function:

4.12.2.3 void timevalAdd (struct timeval * *to*, struct timeval * *val*)

timevalAdd() adds two struct timeval

Definition at line 52 of file timing.c.

Referenced by main().

```
53  {
54          to->tv_sec += val->tv_sec;
55          to->tv_usec += val->tv_usec;
56
57          //  timevalfix
58          if ( to->tv_usec < 0 )
59          {
60                  to->tv_sec--;
61                  to->tv_usec += 1000000;
62          }
63          if ( to->tv_usec >= 1000000 )
64          {
65                  to->tv_sec++;
66                  to->tv_usec -= 1000000;
67          }
68  }
```

4.12.2.4 void timevalSub (struct timeval * *to*, struct timeval *const *val*)

timevalSub() substracts two struct timeval

Definition at line 36 of file timing.c.

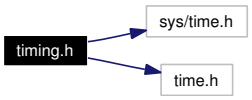
Referenced by measureGettimeofday().

```
37  {
38          to->tv_sec -= val->tv_sec;
39          to->tv_usec -= val->tv_usec;
40          //timevalfix
41          if ( to->tv_usec < 0 )
42          {
43                  to->tv_sec--;
44                  to->tv_usec += 1000000;
45          }
46          if ( to->tv_usec >= 1000000 )
47          {
48                  to->tv_sec++;
49                  to->tv_usec -= 1000000;
50          }
51  }
```

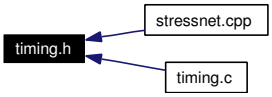
4.13 timing.h File Reference

```
#include <sys/time.h>
#include <time.h>
```

Include dependency graph for timing.h:



This graph shows which files directly or indirectly include this file:



Functions

- void calculateOffset (int *pLengthArray, struct timeval *tArray, int tArrayLength, int bitrate)

  calcultateOffset() calculates the time interval between the sending time of the previous packet and the sending time of the current packet

- double measureGettimeofday ()

  measureGettimeofday() measures the time needed for one execution of the waiting loop

- void timevalAdd (struct timeval *to, struct timeval *val)

  timevalAdd() adds two struct timeval

- void timevalSub (struct timeval *to, struct timeval *val)

  timevalSub() substracts two struct timeval

4.13.1 Function Documentation

4.13.1.1 void calculateOffset (int * *pLengthArray*, struct timeval * *tArray*, int *tArrayLength*, int *bitrate*)

calcultateOffset() calculates the time interval between the sending time of the previous packet and the sending time of the current packet

Definition at line 69 of file timing.c.

Referenced by main().

```
70 {
71         double brate;
72         double delta;
73         double intermediate;
74         int i;
75         int offset; //offset in microseconds, relative to the previous packet
76         brate = ( double ) bitrate;
```

```
77          intermediate=8*1e3;
78          delta = intermediate/brate;//intermediate to avoid loss of precision due to numerical division
79
80          /*Note that we've calculated all constants for the loop out of the loop to gain processing time
81          div_t divstruct;
82          for ( i = 0;i < tArrayLength;++i )
83          {
84                  offset = ( int ) ( ( pLengthArray[ i ] + 4 ) * delta ); //+4 because of FCS/CRC appende
85                  if ( offset >= 1000000 )
86                  {
87                          divstruct=div(offset,1000000);
88                          tArray[ i ].tv_sec = divstruct.quot; //transform the offset in (sec + usec) ins
89                          tArray[ i ].tv_usec = divstruct.rem/*offset % 1000000*/;
90                  }
91                  else
92                  {
93                          tArray[ i ].tv_sec = 0;
94                          tArray[ i ].tv_usec = offset;
95                  }
96          }
97  }
```

4.13.1.2 double measureGettimeofday ()

measureGettimeofday() measures the time needed for one execution of the waiting loop

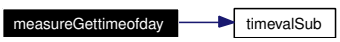
Definition at line 23 of file timing.c.

References COUNT, and timevalSub().

Referenced by main().

```
24  {
25          int i;
26          struct timeval tv1;
27          struct timeval tv2;
28          gettimeofday( &tv1, NULL );
29          for ( i = 0;i < COUNT;++i )
30          {
31                  gettimeofday( &tv2, NULL );
32          }
33          timevalSub( &tv2, &tv1 );
34          return ( ( double ) ( tv2.tv_sec + tv2.tv_usec * 1e-6 ) / COUNT );//division after other operat
35  }
```

Here is the call graph for this function:



4.13.1.3 void timevalAdd (struct timeval * *to*, struct timeval * *val*)

timevalAdd() adds two struct timeval

Definition at line 52 of file timing.c.

Referenced by main().

```
53  {
54          to->tv_sec += val->tv_sec;
55          to->tv_usec += val->tv_usec;
```

```
56
57          //  timevalfix
58          if ( to->tv_usec < 0 )
59          {
60                  to->tv_sec--;
61                  to->tv_usec += 1000000;
62          }
63          if ( to->tv_usec >= 1000000 )
64          {
65                  to->tv_sec++;
66                  to->tv_usec -= 1000000;
67          }
68  }
```

4.13.1.4 void timevalSub (struct timeval * *to*, struct timeval * *val*)

timevalSub() substracts two struct timeval

Definition at line 36 of file timing.c.

Referenced by measureGettimeofday().

```
37  {
38          to->tv_sec -= val->tv_sec;
39          to->tv_usec -= val->tv_usec;
40          //timevalfix
41          if ( to->tv_usec < 0 )
42          {
43                  to->tv_sec--;
44                  to->tv_usec += 1000000;
45          }
46          if ( to->tv_usec >= 1000000 )
47          {
48                  to->tv_sec++;
49                  to->tv_usec -= 1000000;
50          }
51  }
```

Index