NFSTRACE. Release notes.

Introduction

The nfstrace tool performs live Ethernet 1Gbps - 10Gbps packet's capturing and helps to determine NFS procedures in raw network traffic. It performs packet's capturing, filtration, dumping traces, compression, statistical analysis, visualization and provides API for custom analysis modules.

The nfstrace supports following protocols: Ethernet – IPv4 | IPv6 – UDP | TCP – SunRPC/NFSv3.

NFSv4 protocol will be added later.

Definitions

SUS – Single UNIX Specification

POSIX – Portable Operating System Interface for Unix

GPL – General Public License

Wireshark – Enterprise quality tool-set for network traffic analysis.

BPF – Berkeley Packet Filter

LSF – Linux Socket Filtering, is derived from the BPF

NIC – Network Interface Card

NFS – Network File System Protocol

RPC/NFSv3 headers – Parts of RPC messages, which are useful in analysis of NFS operations

Payload – User's data transferred by NFS protocol. It is useless in analysis

wsize,rsize – options of NFS client connection to a NFS server

DPI – Deep Packet Inspection, nfstrace performs DPI of raw network traffic

gnuplot – CLI tool that can generate two- and three-dimensional plots of data, and data fits.

Portability

Target OS

The software has been developed and run as command line utility on GNU/Linux (Fedora Core 19, OpenSUSE 13.1, Ubuntu 13.10) and FreeBSD (FreeBSD 8.4, FreeBSD 10.0). The implementation written on C++11 programming language. It uses standard POSIX interfaces and following libraries: libpthread, libpcap, libstdc++.

```
$ ldd nfstrace
        linux-vdso.so.1 =>  (0x00007fff1daf7000)
        libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f2ff342c000)
        libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f2ff320f000)
        libpcap.so.0.8 => /usr/lib/x86_64-linux-gnu/libpcap.so.0.8 (0x00007f2ff2fd3000)
        libstdc++.so.6 => /usr/lib/x86_64-linux-gnu/libstdc++.so.6 (0x00007f2ff2ccf000)
        libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x00007f2ff2ab9000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f2ff26f0000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f2ff3645000)
        libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007f2ff23ec000)
```

The software does not use any external libraries due to portability reasons and licensing limitations. Utility does not use any Wireshark libraries due to the code of it libraries (protocols' dissectors) have published under GPLv2 license.

Hardware Interfaces

The nfstrace captures raw packets from an Ethernet interface via libpcap interface of Linux (LSF) or FreeBSD (BPF) implementations. Current implementation makes assumption, that libpcap delivers correct TCP and UDP packets. Assembling Ethernet frames to IP packets and IP packets defragmentation performs in the Kernel. The software have tested on direct linked workstations with 1Gbps integrated NICs (Ethernet 1000baseT/Full). The nfstrace demonstrates perfect performance even in live capturing/filtration/analysis mode.

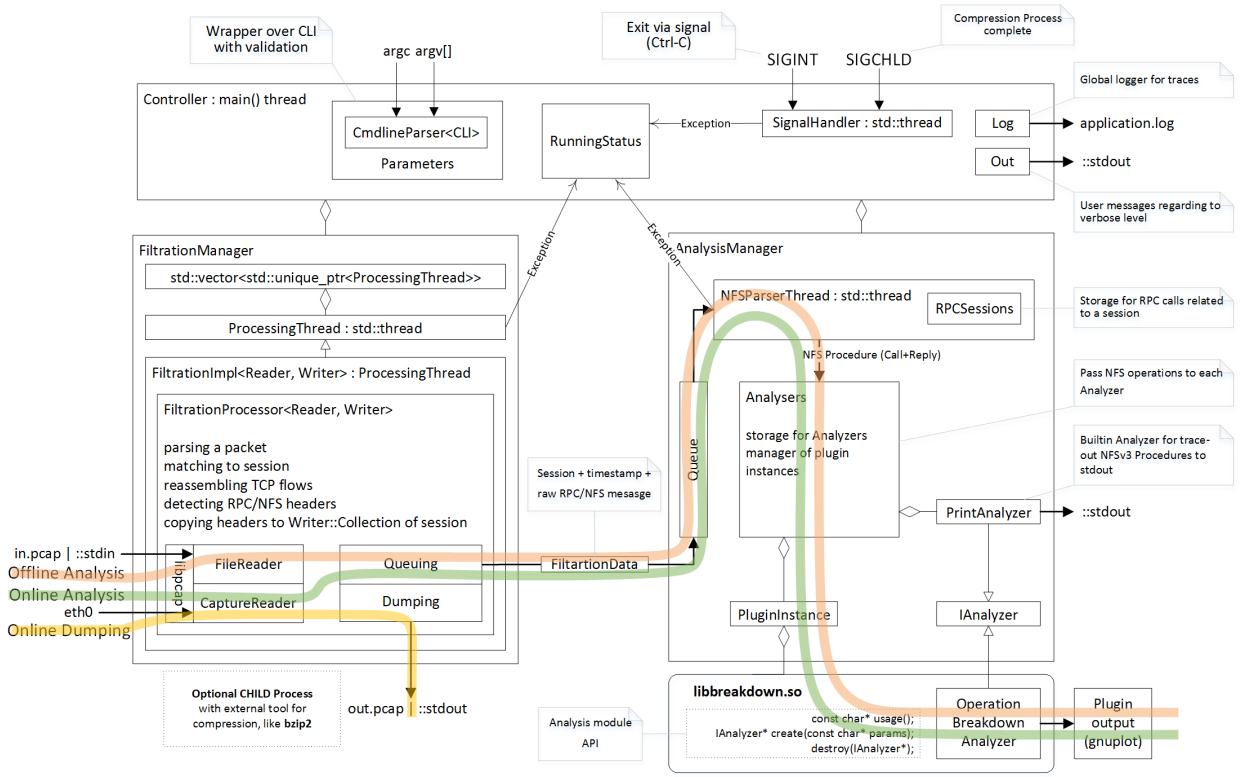
Implementation Details

Command-line interface (CLI)

The nfstrace has following command-line interface:

```
$ ./nfstrace -a ./analyzers/libbreakdown.so -a ./analyzers/libofws.so -a ./analyzers/libofdws.so -h
nfstrace 0.2-alpha (Release)
built on Linux-3.11.0-12-generic
by C++ compiler GNU 4.8.1
Usage: ./nfstrace [OPTIONS]...
  -m, --mode=live|dump|stat  (default:live) set runing mode
  -i, --interface=INTERFACE  (default:) listen interface, it is required for live and dump modes
  -f, --filtration=BPF       (default:port 2049) a packet filtration in libpcap BPF syntax
  -s, --snaplen=0..65535     (default:65535) max length of raw captured packet. May be used ONLY FOR UDP
  -t, --timeout=Milliseconds (default:100) set the read timeout that will be used on a capture
  -b, --bsize=MBytes         (default:20) size of capturing kernel buffer
  -p, --promisc              (default:true) put the interface into promiscuous mode
  -d, --direction=in|out|inout  (default:inout) set the direction for which packets will be captured
  -a, --analysis=PATH#opt1,opt2=val,...(default:) specify path to analysis module and set desired options
  -I, --ifile=PATH           (default:INTERFACE-BPF.pcap) input file for stat mode, the '-' means stdin
  -O, --ofile=PATH           (default:INTERFACE-BPF.pcap) output file for dump mode, the '-' means stdout
  -C, --command="shell command"  (default:) execute command for each dumped file
  -D, --dump-size=MBytes     (default:0) size of dumping file portion, 0 = no limit
  -L, --list                 (default:false) list all available network interfaces
  -M, --msg-header=1..4000   (default:512) RPC message will be truncated to this limit in bytes
  -Q, --qcapacity=1..65535   (default:4096) initial queue capacity of RPC messages
  -T, --trace                (default:false) print collected NFSv3 procedures, true if no modules added
  -Z, --droproot=username (default:) drops root privileges, after opening the capture device, but before reading
from it
  -v, --verbose=0|1|2   (default:1) level of print out additional information
  -h, --help            (default:false) print help message and usage for modules, then exit
Usage of ./analyzers/libbreakdown.so:
ACC - for accurate evaluation, MEM - for memory efficient evaluation (default). Options cannot be combined
Usage of ./analyzers/libofws.so:
Arguments aren't supported. TODO: add description of OFWS Analyser
Usage of ./analyzers/libofdws.so:
bu_size - for specifying amount of buckets. Range: 1..32767 [16 by default]
bl_size - for specifying block size [KB]. Range: 1..31 [8 by default]
```

This example shows the output of --help (-h) operation. It shows command line interface description of the nfstrace and description of pluggable analysis modules passed by –analysis (-a) parameters.



Picture 1 - General schema

Running modes

The nfstrace tool may be run in 3 different modes:

- on-line analysis (--mode=live): perform online capturing, filtration and live analysis detected NFS procedures by a pluggable analysis modules or print-out them to console (-T option) (see green line on Picture 1).
- on-line dumping (--mode=dump): perform online traffic capturing, filtration and dumping packets to the specified output file (see yellow line on Picture 1).
- off-line analysis (--mode=stat): perform offline filtration any .pcap file with captured traces and perform analysis by pluggable modules or print out found NFSv3 procedures to console (-T option)(see orange line on Picture 1).

Packets Filtration by BPF

The nfstrace captures network traffic from NIC by libpcap. Libpcap provides portable interface to native Kernel API for capturing network traffic. All platforms implement low-level filtration of packets in Kernel-space by BPF. Default BPF filter in nfstrace is ‘port 2049’. It means that each packet, which delivered to user-space from the Kernel, satisfies following conditions: it has IPv4 or IPv6 header and it has TCP or UDP header with source or destination port number equals 2049(default NFS port). This filter is ‘heavy’ and IPv6 support is experimental. So, for faster filtration of IPv4-only traffic, use following BPF filter: ‘ip and port 2049’ (via -f option).

Payload Filtration

Let see to following case: writing 1Gb file to a NFS storage:

This action will be performed by some LOOKUP, READDIRPLUS, COMMIT, GETATTR, etc. and some (thousands?) WRITE procedures of NFS protocol. The nfstrace must catch all of them. How many RPC/NFS procedures will be perform? It depends on wsize. Wsize is an option of NFS client, defines the number of bytes that NFS uses when writing files to an NFS server by one WRITE procedure. When wsize is 512 Kbytes a NFS client should (in general) perform 1Gb/512Kb WRITE procedures. So, for writing 1Gb file to a NFS storage, a NFS client (with wsize=512k) should perform 2000 WRITE procedures + some other procedures like GETATTR, etc.

Each NFS procedure consists by two RPC messages: Call – request from client to server and Reply – reply from server with result of requested procedure. Both RPC messages may contain data useful for analysis. Both RPC messages may contain thousands of payload bytes useless for analysis. The nfstrace must capture headers of calls and replies, and then match pairs of them to complete NFS procedures.

The '--snaplen' option set-up the number of bytes of a packet for uprising from the Kernel to user-space. In case of TCP transport layer this options is useless because TCP connection is a bidirectional stream of data (instead of UDP that is form of interchange up to 64k datagrams). In case of NFS over TCP, the nfstrace must capture whole packets and copy them to user-space from the Kernel for DPI and performing NFS statistics analysis.

Finally, the nfstrace must filtrate whole NFS traffic passed from the Kernel to user-space and detects RPC/NFSv3 message headers (up to 4Kbytes) within gigabytes of network traffic.

Detected headers are copied into internal buffers (or dumped to a .pcap file) for statistics analysis.

The key principle of Filtration in the nfstrace is **"Discard payload ASAP"**.
The Filtration module works in separate thread that captures packets from network interface by libpcap. It matches a packet to a related session (TCP or UDP) and performs reassembling TCP flow from TCP segment of a packet. After that, a part of packet will be pass to RPCFiltrator.

There are two RPCFiltrator in one TCP session. Both of them known state of current RPC message in related TCP flow. They can detect RPC messages and perform action on a packet – discard it or collect it for analysis.

The '-b' option of nfstrace set-up size of capturing buffer in the Kernel of operating system in megabytes. This option is very critical for capturing performance.
The wsize and rsize of NFS connections are meaningful for filtration and analysis performance too.

Dump File Format

The nfstrace uses libpcap file format for input and output files. Filtered packets are saved in .pcap format too. Any external tool (like Wireshark) may be used to inspect filtered traces.

Filtrated headers of RPC messages are passed from Filtration to Analysis modules in raw data buffers via Queue of Analysis module.

Compression

The compression and decompression performs by external tools. It is very similar to tcpdump -z option. For example:

```
$ sudo ./nfstrace --mode=dump -i eth0 -f "ip and port 2049" -O dump.pcap --dump-size=10 -C "bzip2 -f -9"
Capture from interface: eth0
  BPF filter  : ip and port 2049
  snapshot len: 65535 bytes
  read timeout: 100 ms
  buffer size : 20971520 bytes
  promiscuous mode: on
  capture traffic : inout
Dump packets to file: dump.pcap
  file rotation size: 10485760 bytes
  file rotation command: [bzip2 -f -9]
Processing packets. Press CTRL-C to quit and view results.
```

This command run nfstrace in dumping mode (--mode=dump). Capturing from network interface requires super-user privileges, so – sudo. In this mode, the application performs live capturing packets from network interface eth0 (-i eth0) and filtration NFS procedures (calls + replies). Then it dumps filtered packets that contains RPC/NFSv3 headers (or parts of these headers). Filtration in the Kernel performs by BPF filter (-f "ip and port 2049"). It means that we are interested in any TCP or UDP packets send to or from port 2049(default port for NFS servers) over IPv4. This command specifies the output file dump.pcap (-O dump.pcap). In case of the output file will be huge (tons of Mb) it will be divided to parts by 10Mb (--dump-size=10).

When the part's dumping is compete, the dumping will be continued to another file (dump.pcap-1, dump.pcap-2, … dump.pcap-N) and nfstrace spawns child process (by fork() & exec() calls) that executes command (passed by -C "bzip2 -f -9") "bzip2 -f -9 dump.pcap" over each dumped part.

After dumping interruption via Control-C(SIGINT) the application closes with flushing captured data and we get a set of compressed parts of filtered traffic:

```
$ ls -la
total 1780
drwxrwxr-x  6 nst  nst    4096 Apr  3 13:46 .
drwxrwxr-x 10 nst  nst    4096 Mar 27 13:49 ..
-rw-r--r--  1 root root  79711 Apr  3 13:46 dump.pcap-10.bz2
-rw-r--r--  1 root root 128455 Apr  3 13:45 dump.pcap-1.bz2
-rw-r--r--  1 root root 129402 Apr  3 13:45 dump.pcap-2.bz2
-rw-r--r--  1 root root 132811 Apr  3 13:45 dump.pcap-3.bz2
-rw-r--r--  1 root root 127140 Apr  3 13:45 dump.pcap-4.bz2
-rw-r--r--  1 root root 130819 Apr  3 13:45 dump.pcap-5.bz2
-rw-r--r--  1 root root 135586 Apr  3 13:45 dump.pcap-6.bz2
-rw-r--r--  1 root root 130894 Apr  3 13:45 dump.pcap-7.bz2
-rw-r--r--  1 root root 129277 Apr  3 13:45 dump.pcap-8.bz2
-rw-r--r--  1 root root 129772 Apr  3 13:46 dump.pcap-9.bz2
-rw-r--r--  1 root root 127589 Apr  3 13:45 dump.pcap.bz2
```

These compressed parts have filtered data and may be join to one .pcap file by:

```
$ ls dump.pcap*.bz2 | sort -n -t - -k 2 | xargs bzcat > dump.pcap
```

The ls and sort commands sort file names of parts right order and a pipe passes them to bzcat tool to decompressing parts to one huge file dump.pcap.
dump.pcap file may be open in any external tool that know .pcap format, f.e. – Wireshark. Only the dump.pcap.bz2 has .pcap file header, other parts have only data and they are not be recognized as .pcap file by external tool.
The compressed parts of dump can be decompressed by bzcat tool to stdin of nfsrace for offline analysis by an analyzer module or prints-out to console:

```
$ ls dump.pcap*.bz2 | sort -n -t - -k 2 | xargs bzcat | ./nfstrace --mode=stat -I - -T
```

This command runs bzcat tool, which decompress parts, merges them to one stream and put in to a pipe. Nfstrace runs in statistic mode (--mode=stat) for reading data from that pipe via stdin (-I -) and prints (by -T) all detected NFS procedures (call+reply) to console:

```
…
10.6.137.109:876 --> 10.6.137.113:2049 [UDP] WRITE
      CALL  [ file: 10007010...9335c1f4 offset: 2024476672 count: 32768 stable: UNSTABLE ]
      REPLY [ status: OK ]
10.6.137.109:876 --> 10.6.137.113:2049 [UDP] COMMIT
      CALL  [ file: 10007010...9335c1f4 offset: 0 count: 0 ]
      REPLY [ status: OK ]
...
```

The --verbose=2 option enables whole output:

```
...
10.6.137.109:876 --> 10.6.137.113:2049 [UDP] XID: 3875993544 RPC version: 2 RPC program: 100003 version: 3
WRITE
      CALL  [ file: 10007010ad108000000000002b960d115ac704d6c92360d50abcf7f8921080009335c1f4 offset:
2024476672 count: 32768 stable: UNSTABLE data: skipped on filtration ]
      REPLY [ status: OK file_wcc:  before:  void  after:  attributes:  type: REG mode: OWNER_READ OWNER_WRITE
GROUP_READ GROUP_WRITE OTHER_READ  nlink: 1 uid: 1000 gid: 1000 size: 2097152000 used: 2097156096 rdev:
specdata1: 0 specdata2: 0 fsid: 16302950409496320814 fileid: 524585 atime: seconds: 1395737557 nseconds:
372964699 mtime: seconds: 1396521963 nseconds: 558634635 ctime: seconds: 1396521963 nseconds:
558634635 count: 32768 committed: UNSTABLE verf: 53314027000b8fb4 ]
10.6.137.109:876 --> 10.6.137.113:2049 [UDP] XID: 3892770760 RPC version: 2 RPC program: 100003 version: 3
COMMIT
      CALL  [ file: 10007010ad108000000000002b960d115ac704d6c92360d50abcf7f8921080009335c1f4 offset: 0
count: 0 ]
      REPLY [ status: OK file_wcc:  before:  void  after:  attributes:  type: REG mode: OWNER_READ OWNER_WRITE
GROUP_READ GROUP_WRITE OTHER_READ  nlink: 1 uid: 1000 gid: 1000 size: 2097152000 used: 2097156096 rdev:
specdata1: 0 specdata2: 0 fsid: 16302950409496320814 fileid: 524585 atime: seconds: 1395737557 nseconds:
372964699 mtime: seconds: 1396521963 nseconds: 558634635 ctime: seconds: 1396521963 nseconds:
558634635 verf: 53314027000b8fb4 ]
...
```

Pluggable Analysis Modules

The API of nfstrace provides C++ headers with definition of IAnalyzer interface and couple of related types and NFS data structures. An IAnalyzer interface is a set of NFSv3 handlers that will be called by Analysis module for each NFSv3 procedure.
A pluggable analysis module should be a dynamic linked shared object that exports following C functions:

```
const char* usage (); // return description of expected opts for create(opts)
IAnalyzer*  create (const char* opts); // create and return an instance of an Analyzer
void        destroy (IAnalyzer* instance); // destroy created instance of an Analyzer
```

The usage() function must return C-string with description of module and required parameters for creation an instance of analyzer, this string will be shown in output of "--help" option.

The IAnalyzer* create(const char* opts) must create and return an instance of analyzer accordingly to passed options.

The void destroy(IAnalyzer* instance) must destroy previously created analyzer instance and performs required clean-up (for instance - close connection to a database).

All existing analyzers are implemented as pluggable analysis modules and may be attached to nfstrace via -a option.

Analysis Toolset

Operation Breakdown Analyzer (OB)

The OB implemented as pluggable analysis module libbreakdown.so.

The NFS procedure latency have calculate as difference between timestamp of come-in last TCP or UDP packet of header of RPC/NFS reply-message and timestamp of come-in last TCP or UDP packet of header of related NFS call-message.

Standard Deviation of latency may be calculate by two algorithms. Two-pass algorithm generates correct standard deviation value but requires a lot of memory for storage all latencies till final computation. One-pass algorithm is memory-efficient but it can accumulate computation error in case of large number of small latencies. OB implements both algorithms. They may be chosen by passing parameter while attaching OB analyzer to nfstrace.

```
$ ./nfstrace -a ./analyzers/libbreakdown.so -h
nfstrace 0.2-alpha (Release)
built on Linux-3.11.0-12-generic
by C++ compiler GNU 4.8.1
Usage: ./nfstrace [OPTIONS]...
...
Usage of ./analyzers/libbreakdown.so:
ACC - for accurate evaluation, MEM - for memory efficient evaluation(default). Options cannot be combined
```

MEM means Knuth's one-pass algorithm. ACC means stable two-pass algorithm. MEM is default.

Let see example of OB usage. bzcat tool extracts collected trace from compressed archive to pipe. This archive contains traces of packets of two NFS sessions over TCP protocol. Each session mounts a NFS folder (there are two auxiliary session with NULL NFS procedures) and writes 10Mb of data to it.

The mount to 10.6.137.24 NFS server has wsize=512Kb and we can see 20 WRITE procedures for it. The mount to 10.6.136.214 NFS server has wsize=32Kb and we can see 320 WRITE procedures for it.

```
$ bzcat ../traces/2sessions-tcp-wsize32k-tcp-wsize512k.pcap.bz2 | ./nfstrace -m stat -l - -a analyzers/libbreakdown.so#ACC
Loading module: 'analyzers/libbreakdown.so' with args: [ACC]
Processing packets. Press CTRL-C to quit and view results.
Detect session 10.6.136.107:9316 --> 10.6.136.214:2049 [TCP]
Detect session 10.6.136.107:9318 --> 10.6.136.214:2049 [TCP]
Detect session 10.6.136.107:9320 --> 10.6.137.24:2049 [TCP]
Detect session 10.6.136.107:9322 --> 10.6.137.24:2049 [TCP]
###  Breakdown analyzer  ###
Total calls: 413. Per operation:
NULL          4   0.97%
GETATTR       7   1.69%
SETATTR       2   0.48%
LOOKUP       17   4.12%
ACCESS       15   3.63%
```

READLINK   0   0.00%
READ   5   1.21%
WRITE   340 82.32%
CREATE   2   0.48%
MKDIR   2   0.48%
SYMLINK   0   0.00%
MKNOD   0   0.00%
REMOVE   2   0.48%
RMDIR   2   0.48%
RENAME   0   0.00%
LINK   0   0.00%
READDIR   0   0.00%
READDIRPLUS   7   1.69%
FSSTAT   0   0.00%
FSINFO   4   0.97%
PATHCONF   2   0.48%
COMMIT   2   0.48%
Per connection info:
Session: 10.6.136.107:9322 --> 10.6.137.24:2049 [TCP]
Total: 46. Per operation:
NULL   Count: 1 (2.17%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
GETATTR   Count: 4 (8.70%) Min: 0.000 Max: 0.001 Avg: 0.000 StDev: 0.00006949
SETATTR   Count: 1 (2.17%) Min: 0.013 Max: 0.013 Avg: 0.013 StDev: 0.00000000
LOOKUP   Count: 5 (10.87%) Min: 0.000 Max: 0.001 Avg: 0.001 StDev: 0.00033471
ACCESS   Count: 4 (8.70%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00051993
READLINK   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READ   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
WRITE   Count: 20 (43.48%) Min: 0.052 Max: 0.673 Avg: 0.226 StDev: 0.18726920
CREATE   Count: 1 (2.17%) Min: 0.018 Max: 0.018 Avg: 0.018 StDev: 0.00000000
MKDIR   Count: 1 (2.17%) Min: 0.047 Max: 0.047 Avg: 0.047 StDev: 0.00000000
SYMLINK   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKNOD   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
REMOVE   Count: 1 (2.17%) Min: 0.019 Max: 0.019 Avg: 0.019 StDev: 0.00000000
RMDIR   Count: 1 (2.17%) Min: 0.013 Max: 0.013 Avg: 0.013 StDev: 0.00000000
RENAME   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LINK   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIR   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIRPLUS   Count: 3 (6.52%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00004571
FSSTAT   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSINFO   Count: 2 (4.35%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00001626
PATHCONF   Count: 1 (2.17%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
COMMIT   Count: 1 (2.17%) Min: 0.148 Max: 0.148 Avg: 0.148 StDev: 0.00000000
Session: 10.6.136.107:9320 --> 10.6.137.24:2049 [TCP]
Total: 1. Per operation:
NULL   Count: 1 (100.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
GETATTR   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
SETATTR   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LOOKUP   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
ACCESS   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READLINK   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READ   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
WRITE   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
CREATE   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKDIR   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
SYMLINK   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKNOD   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
REMOVE   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
RMDIR   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
RENAME   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LINK   Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000

```
READDIR        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIRPLUS    Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSSTAT         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSINFO         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
PATHCONF       Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
COMMIT         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
Session: 10.6.136.107:9318 --> 10.6.136.214:2049 [TCP]
Total: 365. Per operation:
NULL           Count:    1 (  0.27%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
GETATTR        Count:    3 (  0.82%) Min: 0.001 Max: 0.004 Avg: 0.002 StDev: 0.00186981
SETATTR        Count:    1 (  0.27%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
LOOKUP         Count:   12 (  3.29%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00003838
ACCESS         Count:   11 (  3.01%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00026510
READLINK       Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READ           Count:    5 (  1.37%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00023933
WRITE          Count:  320 ( 87.67%) Min: 0.004 Max: 0.019 Avg: 0.008 StDev: 0.00281979
CREATE         Count:    1 (  0.27%) Min: 0.002 Max: 0.002 Avg: 0.002 StDev: 0.00000000
MKDIR          Count:    1 (  0.27%) Min: 0.002 Max: 0.002 Avg: 0.002 StDev: 0.00000000
SYMLINK        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKNOD          Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
REMOVE         Count:    1 (  0.27%) Min: 0.016 Max: 0.016 Avg: 0.016 StDev: 0.00000000
RMDIR          Count:    1 (  0.27%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
RENAME         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LINK           Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIR        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIRPLUS    Count:    4 (  1.10%) Min: 0.001 Max: 0.005 Avg: 0.002 StDev: 0.00190528
FSSTAT         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSINFO         Count:    2 (  0.55%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00001697
PATHCONF       Count:    1 (  0.27%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
COMMIT         Count:    1 (  0.27%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
Session: 10.6.136.107:9316 --> 10.6.136.214:2049 [TCP]
Total: 1. Per operation:
NULL           Count:    1 (100.00%) Min: 0.001 Max: 0.001 Avg: 0.001 StDev: 0.00000000
GETATTR        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
SETATTR        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LOOKUP         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
ACCESS         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READLINK       Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READ           Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
WRITE          Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
CREATE         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKDIR          Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
SYMLINK        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKNOD          Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
REMOVE         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
RMDIR          Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
RENAME         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LINK           Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIR        Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIRPLUS    Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSSTAT         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSINFO         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
PATHCONF       Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
COMMIT         Count:    0 (  0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000

./nfstrace: Filtration is done
```

The OB produces .dat files for each detected NFS session:

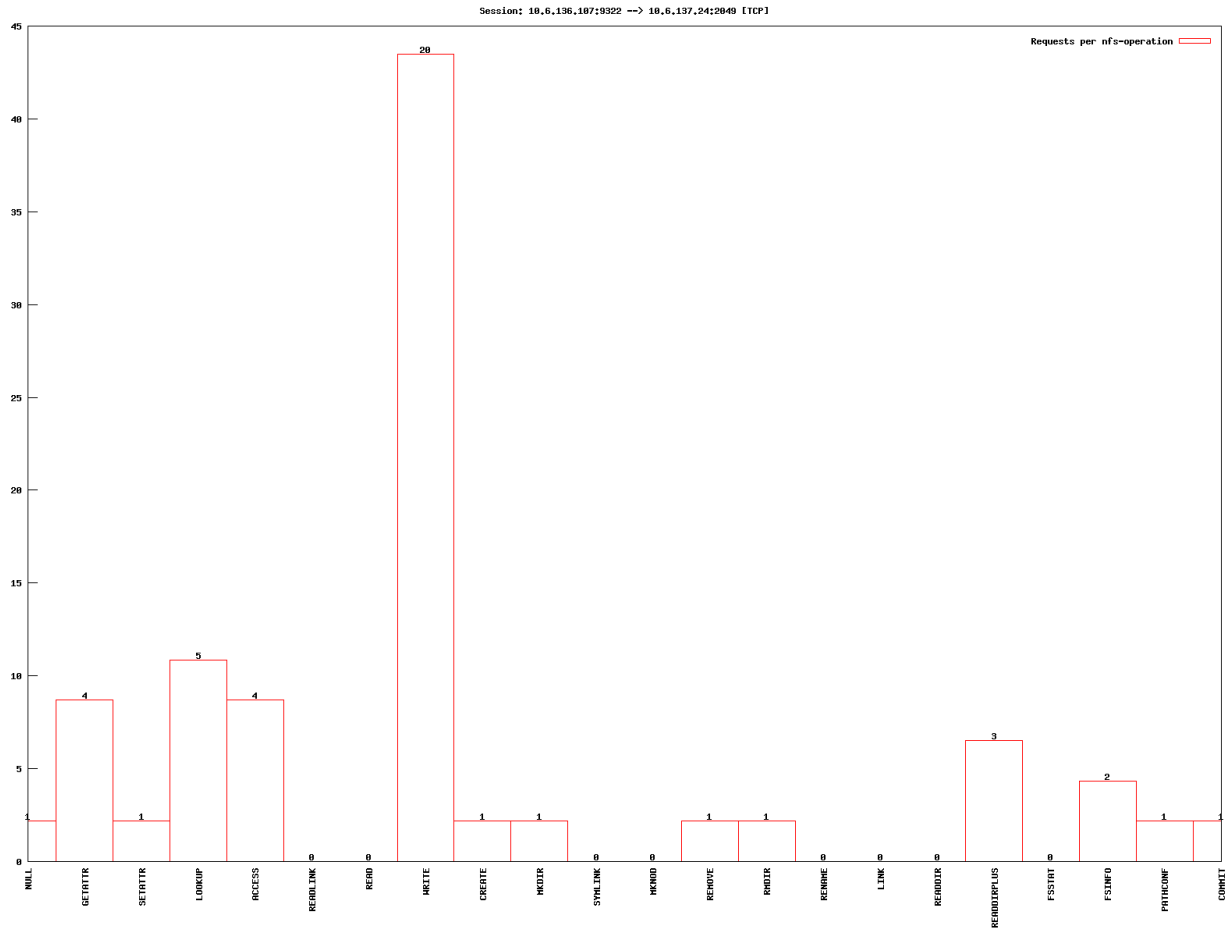
```
$ ls -la
total 109632
drwxrwxr-x  6 nst  nst      4096 Apr  3 14:08 .
drwxrwxr-x 10 nst  nst      4096 Mar 27 13:49 ..
drwxrwxr-x  3 nst  nst      4096 Apr  2 13:25 analyzers
-rw-rw-r--  1 nst  nst       499 Apr  3 14:53 breakdown_10.6.136.107:9316 --> 10.6.136.214:2049 [TCP].dat
-rw-rw-r--  1 nst  nst       961 Apr  3 14:53 breakdown_10.6.136.107:9318 --> 10.6.136.214:2049 [TCP].dat
-rw-rw-r--  1 nst  nst       498 Apr  3 14:53 breakdown_10.6.136.107:9320 --> 10.6.137.24:2049 [TCP].dat
-rw-rw-r--  1 nst  nst       910 Apr  3 14:53 breakdown_10.6.136.107:9322 --> 10.6.137.24:2049 [TCP].dat
```

OB has two optional modules for demonstration visualization via gnuplot:

- breakdown.plt – gnuplot script for a histogram generation;
- nst.sh – “glue” for passing a .dat file to breakdown.plt and invocation gnuplot tool.

The example of invocation:

```
$ ../analyzers/nst.sh -a ../analyzers/breakdown.plt -d . -p 'breakdown_10.6.136.107:9322*.dat' -v
gnuplot -e "i_file='./breakdown_10.6.136.107:9322 --> 10.6.137.24:2049
[TCP].dat';o_file='./breakdown_10.6.136.107:9322 --> 10.6.137.24:2049 [TCP].dat.png'" ../analyzers/breakdown.plt
```



Picture 2 - breakdown_10.6.136.107:9322 -->10.6.137.24:2049 [TCP].dat.png

Overall File Working Set Analyzer (OFWS)

The OFWS implemented as pluggable analysis module libofws.so. The example of invocation:

```
$ bzcat ../traces/2sessions-tcp-wsize32k-tcp-wsize512k.pcap.bz2 | ./nfstrace -m stat -I - -a analyzers/libofws.so -v 2
Loading module: 'analyzers/libofws.so' with args: []
Processing packets. Press CTRL-C to quit and view results.
Detect session 10.6.136.107:9316 --> 10.6.136.214:2049 [TCP]
Detect session 10.6.136.107:9318 --> 10.6.136.214:2049 [TCP]
Detect session 10.6.136.107:9320 --> 10.6.137.24:2049 [TCP]
Detect session 10.6.136.107:9322 --> 10.6.137.24:2049 [TCP]
###  Overall File Working Set (OFWS) analyzer  ###
Total number of files accessed: 16
FileHandle,NFS
Ops,NULL,GETATTR,SETATTR,LOOKUP,ACCESS,READLINK,READ,WRITE,CREATE,MKDIR,SYMLINK,MKNOD,REMOVE,RM
DIR,RENAME,LINK,READDIR,READDIRPLUS,FSSTAT,FSINFO,PATHCONF,COMMIT
9725bb51046621880c00000026ea01005f7f0538000000000000000,2,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c00000031ea010025f77e6f0000000000000000,2,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c0000007bea010008f6d8030000000000000000,3,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c00000089ea0100ca9725330000000000000000,3,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c00000062ea010071c35a5c0000000000000000,3,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c0000006dea0100e4410e0e0000000000000000,3,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c00000099ea01008823c23d0000000000000000,3,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
9725bb51046621880c0000003cea01008a7e4071000000000000000000,4,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
9725bb51046621880c00000065260100807 2c4330000000000000000,4,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0
9725bb51046621880c000000a68c020078286c3e0000000000000000,5,0,1,0,1,1,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0
01000701471b060000000000065d92d0770abb44000000000000000000ed1b06008981bb17,6,0,1,0,1,2,0,0,0,0,0,0,0,
0,1,0,0,0,1,0,0,0,0
01000700471b060000000000065d92d0770abb440000000000000000,6,0,2,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0
9725bb51046621880c000000d88c020071e9b85c0000000000000000,7,0,1,1,0,1,0,0,0,0,1,0,0,1,0,0,0,0,2,0,0,0,0
01000701471b060000000000065d92d0770abb44000000000000000000f1021e0084cf361c,7,0,1,1,0,1,0,0,0,0,1,0,0,1
,0,0,0,0,2,0,0,0,0
01000701471b060000000000065d92d0770abb44000000000000000000f2021e0085cf361c,22,0,0,0,0,0,0,0,20,1,0,0,
0,0,0,0,0,0,0,0,0,0,1
9725bb51046621880c000000d18d0200cdc19e1e0000000000000000,322,0,0,0,0,0,0,0,320,1,0,0,0,0,0,0,0,0,0,0,
0,1

./nfstrace: Filtration is done
```

File Handle (FH) in NFS may have various length for items (files/directories). There are two kind of FH 28 and 36 bytes length in this example.

Filtration Performance Testing Results

The nfstrace tool can be used to perform live traffic capturing, NFS procedures filtration and statistics analysis by pluggable analysis modules. It can be ran by following command:

```
$ sudo ./nfstrace --mode=live --interface=eth0 -b 80 -a ./analyzers/libbreakdown.so -a ./analyzers/libofws.so -a
./analyzers/libofdws.so
```

The live traffic capturing from network interface requires super-user privileges, so use `sudo` but don't forget to drop root privileges using `--droproot=username` The nfstrace runs in line mode (`--mode=live`) to capture packets from network interface eth0 (`--interface=eth0`) and sets capturing buffer in the Kernel to 80Mb (`-b 80`). Finally three analyzers are attached (`-a ./analyzers/libbreakdown.so -a ./analyzers/libofws.so -a ./analyzers/libofdws.so`).

Test-environment is a two development workstations with 1G Ethernet NICs connected by 1G

network switch. Client workstation mounts NFS shared folder with wsize=32k on TCP protocol.

```
$ mount
epbyminw1962t6:/home/nst/nfs-share on /mnt/nfs-ubuntu13 type nfs
(rw,relatime,vers=3,rsize=32768,wsize=32768,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,mount
addr=10.6.137.113,mountvers=3,mountport=42985,mountproto=tcp,local_lock=none,addr=10.6.137.113)
```

Machine 2-client (10.6.137.109) will write 16Gb data to Machine 1-server (10.6.137.113) via dd command (dd if=/dev/zero of=/mnt/nfs-ubuntu13/zeros bs=1M count=16000) to a NFS shared folder.
Machine 2-client generates NFS traffic to Machine 1-server.
Machine 1-server runs nfstrace and nfsstat -Z tool for verification. The log of it has written to nfstrace.log file.

Due to wsize=32Kb and invocation of dd if=/dev/zero of=/mnt/nfs-ubuntu13/zeros bs=1M count=16000 we expect 512000 WRITE procedures.

```
> dd if=/dev/zero of=/mnt/nfs-ubuntu13/zeros bs=1M count=16000
16000+0 records in
16000+0 records out
16777216000 bytes (17 GB) copied, 218.17 s, 76.9 MB/s
```

Results from nfsstat -Z on the NFS server:

```
$ nfsstat -Z
Collecting statistics; press CTRL-C to view results from interval (i.e., from pause to CTRL-C).
^CSignal 2 received; displaying (only) statistics gathered over the last 4 minutes, 8 seconds:

Server rpc stats:
calls      badcalls   badclnt    badauth    xdrcall
512962     0          0          0          0

Server nfs v3:
null         getattr      setattr      lookup       access       readlink
0         0% 1         0% 1         0% 0         0% 2         0% 0         0%
read         write        create       mkdir        symlink      mknod
0         0% 512913    99% 0        0% 0         0% 0         0% 0         0%
remove       rmdir        rename       link         readdir      readdirplus
0         0% 0         0% 0         0% 0         0% 0         0% 0         0%
fsstat       fsinfo       pathconf     commit
0         0% 0         0% 0         0% 45        0%
```

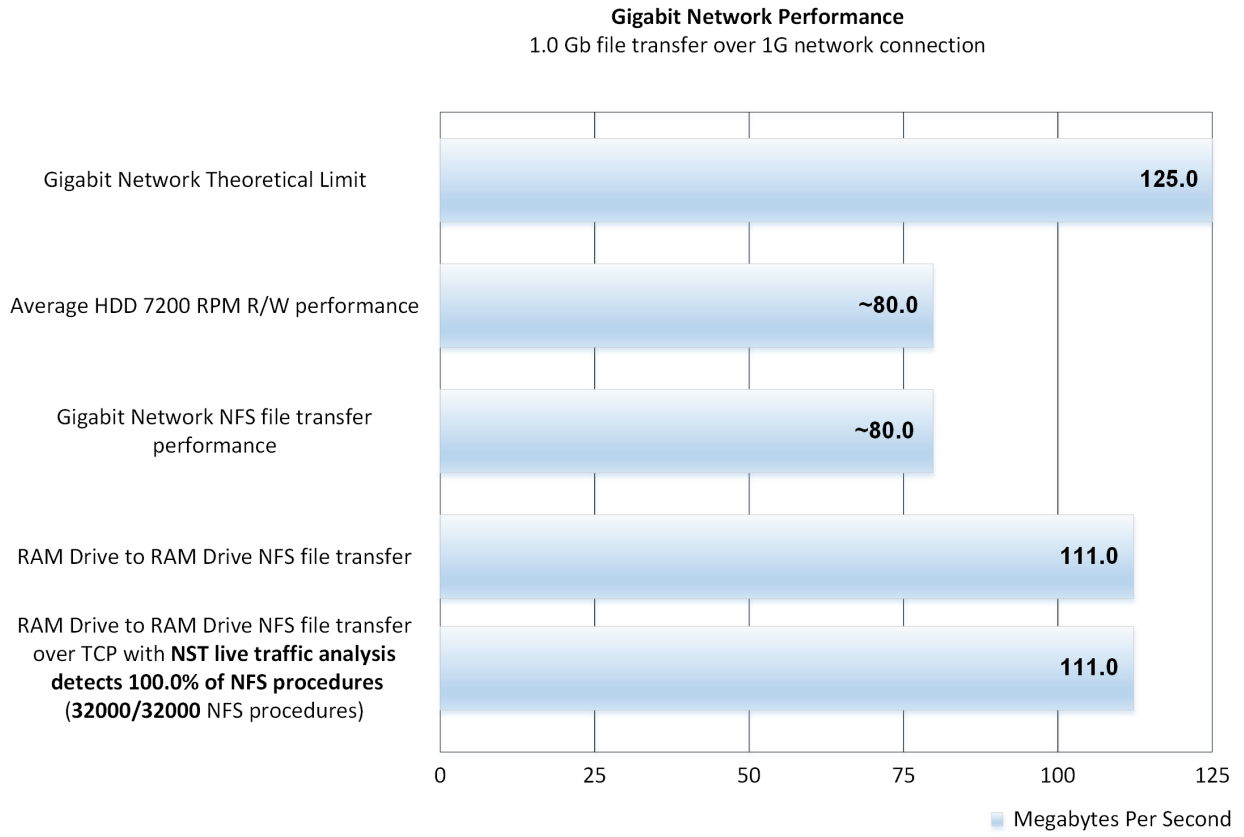
After exit by Control-C from nfstrace we have following results:

```
$ sudo ./nfstrace --mode=live --interface=eth0 -b 80 -a ./analyzers/libbreakdown.so -a ./analyzers/libofws.so -a
./analyzers/libofdws.so
Loading module: './analyzers/libbreakdown.so' with args: []
Loading module: './analyzers/libofws.so' with args: []
Loading module: './analyzers/libofdws.so' with args: []
Capture from interface: eth0
  BPF filter  : ip and port 2049
  snapshot len: 65535 bytes
  read timeout: 100 ms
  buffer size : 83886080 bytes
  promiscuous mode: on
```

capture traffic : inout
Processing packets. Press CTRL-C to quit and view results.
Detect session 10.6.137.109:743 --> 10.6.137.113:2049 [TCP]
^C### Breakdown analyzer ###
Total calls: 512962. Per operation:
NULL 0 0.00%
GETATTR 1 0.00%
SETATTR 1 0.00%
LOOKUP 0 0.00%
ACCESS 2 0.00%
READLINK 0 0.00%
READ 0 0.00%
WRITE 512913 99.99%
CREATE 0 0.00%
MKDIR 0 0.00%
SYMLINK 0 0.00%
MKNOD 0 0.00%
REMOVE 0 0.00%
RMDIR 0 0.00%
RENAME 0 0.00%
LINK 0 0.00%
READDIR 0 0.00%
READDIRPLUS 0 0.00%
FSSTAT 0 0.00%
FSINFO 0 0.00%
PATHCONF 0 0.00%
COMMIT 45 0.01%
Per connection info:
Session: 10.6.137.109:743 --> 10.6.137.113:2049 [TCP]
Total: 512962. Per operation:
NULL Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
GETATTR Count: 1 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
SETATTR Count: 1 (0.00%) Min: 0.441 Max: 0.441 Avg: 0.441 StDev: 0.00000000
LOOKUP Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
ACCESS Count: 2 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00003818
READLINK Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READ Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
WRITE Count:**512913** (99.99%) Min: 0.000 Max: 0.924 Avg: 0.004 StDev: 0.04471008
CREATE Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKDIR Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
SYMLINK Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
MKNOD Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
REMOVE Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
RMDIR Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
RENAME Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
LINK Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIR Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
READDIRPLUS Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSSTAT Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
FSINFO Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
PATHCONF Count: 0 (0.00%) Min: 0.000 Max: 0.000 Avg: 0.000 StDev: 0.00000000
COMMIT Count: 45 (0.01%) Min: 0.000 Max: 2.765 Avg: 1.271 StDev: 1.21419871
Overall File Working Set (OFWS) analyzer
Total number of files accessed: 2
FileHandle,NFS
Ops,NULL,GETATTR,SETATTR,LOOKUP,ACCESS,READLINK,READ,WRITE,CREATE,MKDIR,SYMLINK,MKNOD,REMOVE,RM
DIR,RENAME,LINK,READDIR,READDIRPLUS,FSSTAT,FSINFO,PATHCONF,COMMIT
01000700da01080000000000b269d011a57c406d9c32065da0cb7f8f,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
01000701da01080000000000b269d011a57c406d9c32065da0cb7f8f2901080039531c4f,512961,0,1,1,0,1,0,0,5129
13,0,0,0,0,0,0,0,0,0,0,0,0,0,45

OFDWS Analyzer
Read total: 0 Write total: 16777216000
File ranked:
01000701da01080000000000b269d011a57c406d9c32065da0cb7f8f2901080039531c4f 0 16777216000
Once accessed: 65.67%
Statistic from interface: eth0
packets received by filtration: 2659749
packets dropped by kernel : 0
packets dropped by interface : 0
./nfstrace: Interrupt

We got 512913 WRITE procedures and 49 additional procedures on 76.9 Mb/s data transfer speed.



Picture 3 – nfstrace performance on Gigabit Ethernet

On Picture 3 you can see filtration performance of nfstrace tool during write 1G data from ram to ram drive over 1Gbps Ethernet.