Table of Contents

Description of NAT statistics

The output of statistics in fastdpi_stat.log is set by dbg_log_mask.

0x40000 - output of NAT initialization details (profiles, public addresses, etc.) to the alert log
0x100000 - output of statistics by blocks of public addresses (if 0x2000000 is set)
0x2000000 - NAT statistics output.

Example:
dbg_log_mask=0x2000000

Common for all profiles

```
[STAT    ][2021/12/22-17:33:17:513859] NAT statistics : itrnsl=0, iprof=2,
profile 'cgnat', nttype=0, ref_cnt=1, cidr=94.140.198.84/30
[STAT    ][2021/12/22-17:33:17:513869] NAT statistics : itrnsl=1, iprof=3,
profile 'nat1_1', nttype=1, ref_cnt=1, cidr=16.35.121.0/24

    itrnsl  - internal index of the private<-->public proile converter.
    iprof   - internal index of the profile
    profile - profile name
    nttype  - profile type (0 - cgnat, 1 - 1:1)
    ref_cnt - counter of references of converter usage by profiles
    (Profiles can use one set of CIDR, but different number of restrictions
per session)
    cidr    - list of profile's public addresses CIDR
```

CG-NAT Profile

```
[STAT    ][2021/12/22-17:33:19:252622] NAT statistics : itrnsl=0, iprof=2,
profile 'cgnat', nttype=0, ref_cnt=1, cidr=94.140.198.84/30
        k=0, itrnsld=0, cidr=94.140.198.84/30
                total  TCP : 30/20/0/7/17/ 0/0/0 50/20/0/50/0 5516/8/121
                actual TCP : 0/0/0/0/0 0/0/0 0/0/0/0/0 0/0/0
                total  UDP : 13/4/0/13/1/ 0/0/0 17/4/11/17/0 28/1/3
                actual UDP : 0/0/0/0/0 0/0/0 0/0/0/0/0 0/0/0
                total GRE : 0/0
```

Information on the data converter

```
        k=0, itrnsld=0, cidr=94.140.198.84/30
    k       - itrnsld number
    itrnsld - internal converter data index - the one serving the CIDR
    cidr    - exact cidr
    total   - total statistics
    actual  - статистика изменеий счетчиков за период вывода статистики
(delta_alarm parameter, 15 seconds by default)

    total  TCP : 30/20/0/7/17 0/0/0 50/20/0/50/0 5516/8/121
```

Four groups группы:
1 group -- operations with public address ports

```
30/20/0/7/17:

30 - allocation of a new public port
20 - public port reuse
0  - errors of allocating a new public port
7  - performs decrement of number of subscriber sessions on flow releasing
17 - performs decrement of the number of subscriber sessions on public port
reuse
```

2 group -- general statistics

```
0/0/0:

0 - calculated CRC by IP when accessed to allocate a public address. Should
be == 0
0 - excessive number of sessions for subscribers
0 - different public addresses in flow and converters - Should be == 0
```

3 group -- cache statistics *private -→ public*

```
50/20/0/50/0:

50 - added records to cache
20 - deleted records from cache
0  - found records in cache when new public port is allocated
50 - no public address found for private
0  - errors of adding to cache
```

4 group -- conversion statistics *public -→ private* (*inet-→subs*)

```
5516/8/121:

5516 - successful translation public --> private
8    - port is not in the allocated range of public ports
121  - translation public --> private was not found
```

For TCP/UDP and total/actual statistics are the same.

For GRE – it is GRE by default (when session is not found in PPTP). Only one such session can be created per public address.

```
total GRE : 0/0
0 - address used
0 - number of attempts to create sessions on an already allocated public
address
```

1:1 Profile

```
[STAT    ][2021/12/22-17:17:28:749622] NAT statistics : itrnsl=1, iprof=3,
profile 'nat1_1', nttype=1, ref_cnt=1, cidr=16.35.121.0/24
        k=0, itrnsld=1, cidr=16.35.121.0/24
                total 256/256/0/0/0/0 0/0
```

Group 2 statistics:
1 group - 256/256/0/0/0/0
2 group - 0/0

Example:
dbg_log_mask=0x2100000

Output of public addresses statistics

```
[STAT    ][2021/12/22-21:14:48:385991] NAT statistics : itrnsl=0, iprof=2,
profile 'cgnat', nttype=0, ref_cnt=1, cidr=94.140.198.84/30
        k=0, itrnsld=0, cidr=94.140.198.84/30
                total  TCP : 26/4/0/4/2/ 0/0/0 30/4/0/30/0 3045/1/36
                actual TCP : 0/0/0/0/0 0/0/0 0/0/0/0/0 0/0/0
                        TCP whiteblck ip_mask=0x0, nwhaddr=2
                                whip=94.140.198.84 : sb=64, lsb=64, nb=1008,
whpa=64512, whpb=0, whpf=64512, awhb=4, fwhb=1004, puwhb=0.40%
                                        thr=0, ublock=1, uport=0
                                        thr=1, ublock=1, uport=0
                                        thr=2, ublock=1, uport=0
                                        thr=3, ublock=1, uport=0
                                whip=94.140.198.86 : sb=64, lsb=64, nb=1008,
whpa=64512, whpb=26, whpf=64486, awhb=4, fwhb=1004, puwhb=0.40%
                                        thr=0, ublock=1, uport=0
                                        thr=1, ublock=1, uport=0
                                        thr=2, ublock=1, uport=13
                                        thr=3, ublock=1, uport=13
```

```
TCP whiteblck ip_mask=0x0, nwhaddr=2 :

    ip_mask - addresses mask
    nwhaddr - the number of public addresses that are under the mask
```

```
whip=94.140.198.84 : sb=64 ( 64 ), nb=1008, whpa=64512, whpb=0, whpf=64512,
awhb=4, fwhb=1004, puwhb=0.40%

```

```
    whip=94.140.198.84 - public address
    sb=64       - port block size
    lsb=64      - size of the last block
    nb=1008     - number of port blocks
    whpa=64512  - total ports
    whpb=0      - ports occupied
    whpf=64512  - free ports
    awhb=4      - blocks issued
    fwhb=1004   - free blocks
    puwhb=0.40% - percentage of blocks occupied
```

Added in version 12.1.0

```
whp_salfs   - how many ports are in the 'short' queue
whp_lalfs   - how many ports are in the 'long' queue
whp_ruse    - how many ports can be reused
whp_ruse_salfs  - how many ports can be reused from the 'short' queue
whp_ruse_lalfs  - how many ports can be reused from the 'long' queue
whp_dthr    - how many ports were created in one worker thread but used in
another thread
whp_dthr_salfs  - how many ports were created in one worker thread but used
in another from the 'short' queue
whp_dthr_lalfs  - how many ports were created in one worker thread but used
in another from the 'long' queue
```

Within the public address you can see the distribution of captured ports/blocks by worker threads

```
thr=0, ublock=1, uport=0
thr=0 - worker thread number
ublock=1 - public port blocks used
uport=0 - public port blocks used
```

```
thr_salfs   - how many ports are in the 'short' queue
thr_lalfs   - how many ports are in the 'long' queue
thr_ruse    - how many ports can be reused
thr_ruse_salfs  - how many ports can be reused from the 'short' queue
thr_ruse_lalfs  - how many ports can be reused from the 'long' queue
thr_dthr    - how many ports were created in one worker thread but used in
another thread
thr_dthr_salfs  - how many ports were created in one worker thread but used
in another from the 'short' queue
thr_dthr_lalfs  - how many ports were created in one worker thread but used
in another from 'long' queue
```

The output format is the same.

```
fdpi_ctrl list status --service 11 --ip 192.168.4.20
Autodetected fastdpi params : dev='em1', port=29001
connecting 94.140.198.68:29001 ...
```

```
================================
192.168.4.20    crcip=0xd649d853        nttype=0        profile='cgnat'
itrnsl=0 itrnsld=0       whiteip=94.140.198.86   sess_tcp=127
sess_udp=108   indmtd=4
```

Output:

```
192.168.4.20        - private IP
crcip=0xd649d853    - CRC of the private IP
nttype=0            - NAT type: 0 - cgnat, 1 - 1:1
profile='cgnat'   - profile name
itrnsl=0        - internal index of the converter private<-->public profile.
itrnsld=0           - internal converter data index
whiteip=94.140.198.86   - public address
sess_tcp=127        - number of TCP sessions
sess_udp=108        - number of UDP sessions
indmtd=4                - internal index of subscriber data (subscriber
metadata)

fdpi_ctrl list status --service 11 --ip 192.168.4.20 --outformat=json
fdpi_ctrl list status --service 11 --ip 192.168.4.20 --outformat=json  | jq
.

fdpi_ctrl list all status --service 11
fdpi_ctrl list all status --service 11 --outformat=json
```

The format is the same.

Key to command to view NAT profile statistics via fdpi_ctrl

Command:

```
fdpi_ctrl list status --service 11 --profile.name cgnat
```

Output:

```
nttype=0        profile='test_nat_cgnat'        itrnsl=0        nitrnsld=1
        itrnsld=0       cidr=94.140.198.84/30
                proto=TCP       ip_mask=0x0     nwhaddr=2
                        proto=TCP       ip_mask=0x0     whip=94.140.198.84
sb=64   lsb=64  nb=1008 whpa=64512      whpb=0  whpf=64512      awhb=4
fwhb=1004       puwhb=0.40%     whp_salfs=0     whp_lalfs=0     whp_ruse=0
whp_ruse_salfs=0        whp_ruse_lalfs=0        whp_dthr=0
whp_dthr_salfs=0        whp_dthr_lalfs=0
                                nthr=0  ublock=1        uport=0 thr_salfs=0
thr_lalfs=0     thr_ruse=0      thr_ruse_salfs=0        thr_ruse_lalfs=0
thr_dthr=0      thr_dthr_salfs=0        thr_dthr_lalfs=0
                                nthr=1  ublock=1        uport=0 thr_salfs=0
thr_lalfs=0     thr_ruse=0      thr_ruse_salfs=0        thr_ruse_lalfs=0
```

```
thr_dthr=0      thr_dthr_salfs=0        thr_dthr_lalfs=0
```

Key:

```
nttype         - profile type (0 - cgnat, 1 - 1:1)
profile        - profile name
itrnsl         - internal index of private<-->public profile converter
nitrnsld       - number of profile converter data (number of CIDR)
itrnsld        - internal converter data index - the one serving the CIDR
cidr           - exact CIDR
proto          - TCP/UDP
ip_mask        - addresses mask
nwhaddr        - the number of public addresses which fall under the mask or
CRC (depends on rx_dispatcher parameter)
whip           - public address
sb             - size of the block of ports to be allocated
lsb            - size of the last block
nb             - number of prot blocks
whpa           - ports in total
whpb           - ports occupied
whpf           - free ports
awhb           - blocks issued
fwhb           - free blocks
puwhb          - percentage of blocks occupied
whp_salfs      - is in the 'short' queue
whp_lalfs      - in the 'long' queue
whp_ruse       - can be used
whp_ruse_salfs - can be used in 'short' queue
whp_ruse_lalfs - can be used in 'long' queue
whp_dthr       - number of elements ithr_owner != ithr by queue
whp_dthr_salfs - number of elements ithr_owner != ithr by 'short' queue
whp_dthr_lalfs - number of elements ithr_owner != ithr by 'long' queue
```

`rx_dispatcher` parameter description at the link

Key to command to view the dump NAT profile statistics

Command:

```
fdpi_cli -r 127.0.0.1:29001 nat dump whaddr queue test_nat_cgnat
```

Output:

```
profile='test_nat_cgnat' itrnsl=0
        cidr='94.140.198.84/30' itrnsld=0
                whip=94.140.198.86
                        proto=TCP
```

```
                                entryp :
                                        ithr=0, ihead=0, itail=0
                                        ithr=1, ihead=0, itail=0
                                        ithr=2, ihead=133, itail=265
                                        ithr=3, ihead=193, itail=327
                                data :
                                        sind=129, inext=257, iprev=258,
whport=1152, graddr=192.168.4.20:60637 tml='2023/03/06 16:28:09,
-00:00:10.657 (7472516905147512 ticks)', lifetime=120, canreuse=0, ialf=1,
imtd=516, iown=2, ilst=2, subproto=0, decr_sess=0, ind_gcache_slice=1,
igcache=40
                                        sind=130, inext=151, iprev=148,
whport=1153, graddr=192.168.4.20:52553 tml='2023/03/06 16:27:50,
-00:00:29.455 (7472459405058624 ticks)', lifetime=30, canreuse=0, ialf=0,
imtd=516, iown=2, ilst=2, subproto=0, decr_sess=0, ind_gcache_slice=1,
igcache=1
```

Key:

```
profile        - profile name
itrnsl         - internal index of the private<-->public profile converter

cidr           - exact CIDR
itrnsld        - internal converter data index - the one serving the CIDR

whip           - public address
proto          - TCP/UDP
```

The implementation of the used ports queue for public addresses uses a single array - let's call it WHP, of size 0xffff. It is used to build a list of used ports for the worker thread. Index 0 is used as a stub (empty).

Thread queues cannot be output as a list, because records in the queue are moved in the process, which may cause the output to loop. Therefore, the WHP array is output 'as is' for occupied entries.

```
entryp : sets entry points to the list of public ports of the worker thread
    ithr   - worker thread number
    ihead  - top of the list
    itail  - the last element of the list
```

```
data : WHP white port array data (only occupied entries are output)

    sind         - record index
    inext            - next record index
    iprev            - previous record index
    whport           - public port
    graddr           - private address, which the public address is assigned
to
    tml              - time of the last record access
    lifetime         - timeout, time in seconds, the record lifetime
(depends on the parameters for short/long queue)
```

```
    canreuse         - sign that the record can be used again
    ialf             - number of the processing queue :
                            en_nalfs_shrt = 0,  # queue with a short
lifetime
                            en_nalfs_long = 1,  # long queue
    indmtd           - internal index of the subscriber's data (subscriber's
metadata)
    iown             - owner thread which has created the record
    ilst             - number of the thread which has last accessed the
record
    subproto         - the protocol the record was allocated for from UDP
                        typedef enum en_nat_borrw_udp: u_int8_t
                        {
                            ennatborwu_ORG     = 0, # UDP/TCP
                            ennatborwu_DFLTGRE = 1, # общий GRE
                            ennatborwu_MAX     = 2, # ICMP
                        } en_nat_borrw_udp_t;
    decr_sess        - a mark that the port usage counter on the private
address has decremented
    ind_gcache_slice - index of the 'private --> public' transcoding cache-
slice
    igcache          - index in the corresponding cache-slice recoding
'private --> public'
```