Table of Contents

Administration issues

How do I know the current release (CCC)?

```
fastdpi -re
```

How do I know the current version?

```
fastdpi -ve
```

How do I downgrade to the previous version?

```
Example of rollback from 2.7 version to 2.6:
yum downgrade fastdpi-2.6
```

In the log I found an error "error loading DSCP settings, res=-4"

The error is displayed because there is no dscp on the standalone systems. You can ignore it.

Not all the commands are always processed, the following error appears: Can't connect to 127.0.0.1:29000, errcode=99 : Cannot assign requested address Autodetected fastdpi params : dev='lo', port=29000 connecting 127.0.0.1:29000 ... I suspect that our way of loading the subscribers to the SSG is not quite good for it (we load each subscriber separately, which leads to >50000 commands during initialization, which we do once a day)

fdpi_ctrl uses a common linux stack to connect the dpi, so the tuning recommendations are similar to those for web servers (like nginx) under high load

The settings are similar to those for nginx, which recommend to put in the /etc/sysctl.conf file (in order to keep them after reboot)

```
# The OS network stack optimization
```

```
net.core.netdev_max_backlog=10000
net.core.somaxconn=262144
net.ipv4.tcp_syncookies=1
net.ipv4.tcp_max_syn_backlog = 262144
net.ipv4.tcp_max_tw_buckets = 720000
net.ipv4.tcp_tw_recycle = 1
net.ipv4.tcp_timestamps = 1
net.ipv4.tcp_tw_reuse = 1
net.ipv4.tcp_fin_timeout = 30
net.ipv4.tcp_keepalive_time = 1800
net.ipv4.tcp_keepalive_probes = 7
net.ipv4.tcp_keepalive_intvl = 30
net.core.wmem_max = 33554432
net.core.rmem_max = 33554432
net.core.rmem_default = 8388608
net.core.wmem_default = 4194394
net.ipv4.tcp_rmem = 4096 8388608 16777216
net.ipv4.tcp_wmem = 4096 4194394 16777216
```

for a 1Gbit interface:

```
 net.core.netdev_max_backlog=10000
```

for a 10Gbit interface:

```
 net.core.netdev_max_backlog=30000
```

To avoid having to reboot, you can change them on the fly by using the command:
sysctl -w
e.g. sysctl -w net.ipv4.tcp_tw_reuse=1

This should solve the problem

For CentOS 7.*
Example:

```
# The OS network stack optimization
net.core.netdev_max_backlog=65536
net.core.optmem_max=25165824
net.core.somaxconn=1024
net.ipv4.tcp_max_orphans = 60000
net.ipv4.tcp_no_metrics_save = 1
net.ipv4.tcp_window_scaling = 1
net.ipv4.tcp_timestamps = 1
net.ipv4.tcp_sack = 1
net.ipv4.tcp_syncookies=1
net.ipv4.tcp_max_syn_backlog = 262144
net.ipv4.tcp_max_tw_buckets = 720000
net.ipv4.tcp_tw_recycle = 1
net.ipv4.tcp_timestamps = 1
net.ipv4.tcp_tw_reuse = 1
```

```
net.ipv4.tcp_fin_timeout = 30
net.ipv4.tcp_keepalive_time = 1800
net.ipv4.tcp_keepalive_probes = 7
net.ipv4.tcp_keepalive_intvl = 30
net.core.wmem_max = 33554432
net.core.rmem_max = 33554432
net.core.rmem_default = 8388608
net.core.wmem_default = 4194394
net.ipv4.tcp_rmem = 4096 8388608 16777216
net.ipv4.tcp_wmem = 4096 4194394 16777216
```

Update command:

```
sysctl –system
```

ДMore information on CentOS7

Scripts for migration from SCE SM to Stingray DB, description inside

How to check the load by cores and why they are loaded unevenly

To view the CPU load by cores in the top utility, press 1 To view the load by DPI task, run the command

```
ps -p `pidof fastdpi` H -o %cpu,lwp,pri,psr,comm
```

Output example:

```
 %CPU   LWP PRI PSR COMMAND
0.0  23141  41   0 fastdpi_main
0.0  23146  41   0 fastdpi_dl
0.3  23147  41   0 fastdpi_ctrl
35.8 23148  41   0 fastdpi_ajb
32.7 23152  41   1 fastdpi_rx_1
34.1 23165  41   2 fastdpi_wrk0
34.1 23170  41   3 fastdpi_wrk1
```

In DPI, COMMAND tasks are functionally separated by PSR cores so as not to interfere with each other:
- The wrk threads analyze data in network packets
- The rx thread is responsible for the transit of data between network ports
- Other threads perform application and auxiliary tasks (netflow generation, control command reception, list loading, pcap writing, etc.) and can cause CPU peak loads, so they are moved to a separate core.

We got an error in fastdpi_alert.log, how to solve the issue? [CRITICAL][2017/10/06-16:36:44:616019][0x7fdb297ac700] metadata_storage : Can't allocate memory [repeat 1], cntr=188889, allocated=188889

In DPI, everything is preallocated, by default to a given number of subscribers. This is regulated by the parameter in the configuration, mem_ip_metadata_recs. **For example** to increase up to 500000 subscribers, chanfge the configuration /etc/dpi/fastdpi.conf:

```
mem_ip_metadata_recs=500000
You will need to restart:
service fastdpi restart
```

Which files do you recommend to archive?

```
cp /etc/pf_ring/ /BACKUPDIR/pf_ring
cp /etc/dpi /BACKUPDIR/etc/
mdb_copy /var/db/dpi /BACKUPDIR/db/
(you can make a backup from mdb_copy with fastdpi running)
```

ipmi takes up 100% of cpu, degrades the DPI performance

```
echo 100 > /sys/module/ipmi_si/parameters/kipmid_max_busy_us
This command can be added to /etc/rc.local to make sure that the setting is
not lost upon server restart
```

Error in the alert log: [ERROR] bpm : thread #1 - does not change self-monitoring counters, and the DPI restarted, and created a core file (or went into bypass)

The DPI performs self-diagnostics during operation, and if one worker thread froze and can no longer process traffic, then DPI detects this condition and restarts with the generation of a core-file on the Abort signal

Important: trace and dbg settings in fastdpi.conf are intended for troubleshooting and debugging, not for continuous operation. E.g:
if disk recording is blocked by other process (e.g. by rotation of logs which usually happens between 3 and 4 am), then when tracing is on, it may cause blocking working thread to write to diagnostic (slave) log and put dpi into bypass or restart, so remember to turn off these settings after diagnostic is done.

The problem occurs only on some servers and if your server is among them, we recommend changing the default disk scheduler to deadline:

```
echo deadline > /sys/block/sda/queue/scheduler
echo deadline > /sys/block/sdb/queue/scheduler
```

Why does process memory consumption grow during work?

The DPI allocates memory statically: at process startup and when some service profiles (such as NAT, blacklists and whitelists) are created. During operation no additional memory is allocated. So why does memory consumption grow?
The Linux operating system distinguishes between resident memory (denoted in the top as RES) and virtual (denoted in the top as VIRT) process memory. The peculiarity is that as long as the memory is not initialized (actually initialized by zero), it is not written by linux to the resident memory and is moved there as it is initialized.
By setting mem_preset=1 in /etc/dpi/fastdpi.conf you can make the DPI initialize all (or nearly all) of the allocated memory so the resident part won't grow in size as it runs. This option slows down startup and is good when physical RAM is enough, so it is better just to consider this factor and watch for virtual memory consumption (VIRT) and resident memory consumption (RES) separately.

One of the SSGs has a lot of zombie processes named wd_*. Only a restart can help?

```
166206 ?        Z      0:00  \_ [wd_fastdpi.sh] <defunct>
166219 ?        Z      0:00  \_ [wd_fastpcrf.sh] <defunct>
```

To restart watchdog is enough

```
service watchdog restart
```

Issues with protocol or signature detection

To resolve the issue with protocol or signature detection, you have to run three tests with each of the devices from the list:

- a personal computer,
- an iOS based smartphone,
- an Android OS based smartphone.

The following actions will help to remove the redundant traffic.

- When performing a test on a PC, it is recommended to run it in the browser in the “Incognito/Private Window” mode.
- When performing a test on a smartphone, you need to turn on the “Energy Saving” mode on it.

Test Performance:

1. Check if the /etc/dpi/fastdpi.conf file includes the following parameters:

   ```
   trace_ip="subscriber's ip"
   ajb_save_ip="subscriber's ip"
   plc_trace_ip="subscriber's ip"
   ```

   If any of these parameters is enabled, then comment it out and run `service fastdpi reload`.

2. Run the command:

   ```
   find /var/log/dpi -type f -name "fastdpi_slave_*.log" -exec sh -c 'cat /dev/null > {}'  \;
   ```

   This command should remove data from fastdpi_slave_*.log files.

3. Clear all files from /var/dump/dpi/.
4. Open the /etc/dpi/fastdpi.conf file in a text editor. Add parameters to the file:

   ```
   trace_ip="subscriber's ip"
   ajb_save_ip="subscriber's ip"
   plc_trace_ip="subscriber's ip" – For this parameter to operate, a policing profile should be enabled for the test subscriber.
   ```

5. Prepare the launch of the test subscriber and the devices in order to generate problematic traffic.
6. Run `service fastdpi reload`.
7. Start generating traffic. Record traffic for 1 minute.
8. Open the fastdpi.conf file. Comment out the parameters:

   ```
   trace_ip="subscriber's ip"
   ajb_save_ip="subscriber's ip"
   plc_trace_ip="subscriber's ip"
   ```

9. Run `service fastdpi reload`.
10. Run the following commands and forward the outputs into files:

    ```
    "fastdpi -ve"
    "dscp2lst /etc/dpi/protocols.dscp"
    "fdpi_ctrl list --policing --ip “subscriber ip"
    "dscp2as /etc/dpi/asnum.dscp".
    ```

11. Prepare an archive with the files from step 10, as well as with the fastdpi.conf file. From /var/log/dpi: fastdpi_stat.log, fastdpi_slave_*.log. From /var/dump/dpi udp_*.pcap.
12. Repeat the required number of tests with different devices. It is also important for us to understand which types of devices you have used for the tests — please specify this information either in the name of the archive or in the archive itself in the readme.txt file.
13. Attach the archives to the ticket. If the files turned out to be too large, then please upload them to any cloud file sharing service and provide us with a link.