Содержание

IPFIX receiver utility

Introduction

IPFIX receiver is used for receiving an IPFIX (Netflow 10) stream from DPI devices and store the stream to a local file. The stored file can be processed as a text file any unix utilities.

Installation and upgrade

CentOS6

1. use VAS Experts repository

```
rpm --import http://vasexperts.ru/centos/RPM-GPG-KEY-vasexperts.ru
rpm -Uvh
http://vasexperts.ru/centos/6/x86_64/vasexperts-repo-1-0.noarch.rpm
```

2. install ipfixreceiver:

```
yum install -y ipfixreceiver
```

3. check changes in configuration files for installed version look at part "Important changes in version ..."

CentOS7

1. use VAS Experts repository

```
rpm --import http://vasexperts.ru/centos/RPM-GPG-KEY-vasexperts.ru
rpm -Uvh
http://vasexperts.ru/centos/6/x86_64/vasexperts-repo-1-0.noarch.rpm
```

2. add the epel repository

```
yum -y install epel-release
```

3. add the forensics repository

```
rpm --import https://forensics.cert.org/forensics.asc
rpm -Uvh
https://forensics.cert.org/cert-forensics-tools-release-el7.rpm
```

4. install ipfixreceiver:

```
yum install -y ipfixreceiver
```

5. check changes in configuration files for installed version look at part "Important changes in

version ..."

Important changes in version 1.0.3 vs 1.0.2

1. chenged configuration file in part of IP address transformation, since 1.0.3 version use decodeipv4, decodeipv6 in Export model to exoprt IP in readable mode. Example:

```
  source_ip4, decodeipv4
```

```
  destination_ip4, decodeipv4
```

2. saving data now in separate process, important if DPI has more than 25 000 session per second, it can load upto 2 proccesor cores. In DEBUG loging added check records to controll save processing
   (a)cnt=NNNNN - send NNNNN buffer
   (b)cnt=YYYYY - saved YYYYY buffer.
3. buffer_size parameter added - size of buffer to interchange between receiver and saver processes, use it in [dump] section, default value - 100000 records (for 20Gbe or 25 000 seesion per second). If the buffer size is not reached then 30sec timeout is used to push buffer into saver process.

Supplied files

1. configuration expamples:

```
/etc/dpiui/ipfixreceiver.conf - clickstream example (http/https
clickstream)
/etc/dpiui/ipfixreceiverflow.conf - session example (netflow 10/IPFIX
full session export)
/etc/dpiui/ipfixreceiversip.conf - meta information (sip connections)
example
```

2. programm files directory:

```
/usr/local/lib/ipfixreceiver.d/
```

3. additional files:

```
/etc/dpiui/port_proto.txt - for translation protocol number to text
protocol name
```

4. link to executable:

```
/usr/local/bin/ipfixreceiver -> link to
/usr/local/lib/ipfixreceiver.d/ipfixreceiver
```

Additional OS settings

1. set iptables for receive external data
   Ipfixreceiver is requred to open ports that will be used to receive IPFIX streams (in configuration see section [connect])
   For instance you are using TCP protocol, 1500 port and IP=212.12.11.10

   ```
   [connect]
   protocol=tcp
   host=212.12.11.10
   port=1500
   ```

   For ipfixreceiver working in /etc/sysconfig/iptables you have to insert the next rule:

   ```
   -A INPUT -p tcp -m state --state NEW -m tcp --dport 1500 -j ACCEPT
   ```

   Do not forget that after changes iptables r - restart service is requried:

   ```
   service iptables restart
   ```

2. configure logrotate
   Example for logrotate file /var/log/dpiuiflow.log, create in /etc/logrotate.d/ the file "flowlog" with the next content

   ```
   /var/log/dpiui*.log {
       rotate 5
       missingok
       notifempty
       compress
       size 10M
       daily
       copytruncate
       nocreate
       postrotate
       endscript
   }
   ```

   Using copytruncate is requred, otherwise the log file will be recreated and log write to the file in stopped.
   According to ipfixreceiver configuration in section [handler_ipfixreceiverlogger] is setted the next:

   ```
   args=('/var/log/dpiuiflow.log', 'a+')
   ```

3. configure remove old files. Example, removing old archive files (more then 31 days) with session records in gzip:

   ```
   15 4 * * * /bin/find /var/dump/dpiui/ -name url_\*.dump.gz -cmin +44640
   -delete > /dev/null 2>&1
   ```

Change it to meet you requirements and put into the file /var/spool/cron/root.

Execution parameters

ipfixreceiver utility has next parameters:

```
usage: ipfixreceiver start|stop|restart|status|-v [-f <config file>]
где
  start   - start as service
  stop    - stop service
  state   - get state of service
  restart - restart service
  -v      - version info
  -f <config file> - config file name (required)

Example:
  ipfixreceiver start -f /etc/dpiui/ipfixreceiverflow.conf
```

Configuration

By default config file /etc/dpiui/ipfixreceiver.conf is used .
ℹ More information about config parameter you can find by link Logging

Logging sections

1. loggers - define logging identifiers
2. handlers - define used logging workers
3. formatters - define used logging formats

logger_root

1. level - log level
   Values:

   ```
   CRITICAL  - critical errors only, minimum log level
   ERROR     - errors included
   WARNING   - warnings included
   INFO      - information included
   DEBUG     - debug messages included
   NOTSET    - all, maximum log level
   ```

   Example:

```
level=DEBUG
```

2. handlers - used message handlers
   Example:

```
handlers=ipfixreceiverlogger
```

handler_ipfixreceiverlogger

1. class - class of the message handler
   Example:

```
class=FileHandler
```

2. level - log level

```
level=DEBUG
```

3. formatter - name of formatter that is used

```
formatter=ipfixreceiverlogger
```

4. args - handlers' parameters

```
args=('/var/log/dpiuiflow.log', 'a+')
```

formatter_ipfixreceiverlogger

1. format - log message format description
   Example:

```
format=%(asctime)s - %(name)s - %(levelname)s - %(message)s
here:
%(name)s      - logger name
%(levelname)s - level ('DEBUG', 'INFO', 'WARNING', 'ERROR',
'CRITICAL').
%(asctime)s   - daye, default - “2003-07-08 16:49:45,896” (with
milliseconds after comma).
%(message)s   - message
```

2. datefmt - date format
   Example:

```
datefmt='%m-%d %H:%M'
```

connect

1. protocol - protocol(tcp or udp).

```
protocol=udp
```

2. host - IP or server name or 0.0.0.0 (to receive from all devices).

```
host=localhost
```

3. port - port number.

```
port=9996
```

dump

1. rotate_minutes - rotation period in minuties, after it temp file will be moved to dumpfiledir/<port>.url.dump and new tempfile will be created.

```
rotate_minutes=10
```

2. processcmd - command that will be executed to process new data file after rotation, parameter is full file name.

```
processcmd=gzip %%s
```

3. dumpfiledir - directory where received data files will be stored.

```
dumpfiledir=/var/dump/dpiui/ipfixflow/
```

4. buffer_size - size of buffer to interchange between receiver and saver processes, use it in [dump] section, default value - 100000 records (for 20Gbe or 25 000 seesion per second). If the buffer size is not reached then 30sec timeout is used to push buffer into saver process.

InfoModel

the section describes IPFIX receiveng template.

1. InfoElements - parameter with description of information model elements for IPFIX template

```
InfoElements =  octetDeltaCount,       0,    1, UINT64, True
                packetDeltaCount,      0,    2, UINT64, True
                protocolIdentifier,    0,    3, UINT8
                session_id,        43823, 2000, UINT64, True
here,
  session_id - field name according to IPFIX description table (see
according sections)
  43823  - enterprise number or general IPFIX protocol number
  1      - unique field name
  UINT64 - field type
  True   - endian (True or empty).
```

Field types:

Type	Length	Type IPFIX
OCTET_ARRAY	VARLEN	octetArray
UINT8	1	unsigned8
UINT16	2	unsigned16
UINT32	4	unsigned32
UINT64	8	unsigned64
INT8	1	signed8
INT16	2	signed16
INT32	4	signed32
INT64	8	signed64
FLOAT32	4	float32
FLOAT64	8	float64
BOOL	1	boolean
MAC_ADDR	6	macAddress
STRING	VARLEN	string
SECONDS	4	dateTimeSeconds
MILLISECONDS	8	dateTimeMilliseconds
MICROSECONDS	8	dateTimeMicroseconds
NANOSECONDS	8	dateTimeNanoseconds
IP4ADDR	4	ipv4Address
IP6ADDR	16	ipv6Address

Field names and their description:

1. Flow export template in IPFIX
2. Meta information export template
3. AAA template export using IPFIX

additional information:
Information Model for IP Flow Information Export

ExportModel

defines the export model parameters, reserved for future use.

1. Mode - type used export (File only)

```
Mode = File
```

ExportModelFile

defines the File export model.

1. Delimiter field delimiter (\t - TAB, examples - |,;)

```
Delimiter = \t
```

2. ExportElements - fields description that will be saved to file.

```
ExportElements = timestamp, seconds, %%Y-%%m-%%d %%H:%%M:%%S.000+03
                 login
                 source_ip4
                 destination_ip4
                 host, decodehost
                 path, decodepath
                 referal, decodereferer
                 session_id
here:
  name - field name from information model described before [InfoModel]
(login, session_id и т.п.)
  worker - internal transformation routune
               seconds       - field in seconds, format requred
               milliseconds  - field in milliseconds, microsecconds,
nanoseconds, format requred
               decodehost    - decoder from punycode to UTF-8
               decodepath    - decoder from urlencoding to UTF-8
               decodereferer - decoder from (punycode,urlencoding) to
UTF-8
               decodeproto   - decoder from protocol number to protocol
port
               decodeipv4    - decoder from decimal number to IP
address string
  format - format for seconds, milliseconds.
           Example: %%Y-%%m-%%d %%H:%%M:%%S.%%f+0300
           Result: 2016-05-25 13:13:35.621000+0300
```

Centos7 service for ipfixreceiver

Centos7 service creation step by step, service name **ipfix1**, config file name **/etc/dpiui/ipfixreceiver.conf**, used port **1500**.
Create the file /etc/systemd/system/ipfix1.service with:

```
[Unit]
Description=ipfix test restart
After=network.target
After=syslog.target

[Service]
Type=forking
PIDFile=/tmp/ipfixreceiver.1500.pid
ExecStart=/usr/local/bin/ipfixreceiver start -f
/etc/dpiui/ipfixreceiver.conf
ExecStop=/usr/local/bin/ipfixreceiver stop -f /etc/dpiui/ipfixreceiver.conf
ExecReload=/usr/local/bin/ipfixreceiver restart -f
```

```
/etc/dpiui/ipfixreceiver.conf
Restart=always
RestartSec=10s

[Install]
WantedBy=multi-user.target
```

Execute to register and run service:

```
systemctl enable ipfix1.service
systemctl start ipfix1.service
systemctl daemon-reload
```

Check status:

```
systemctl status ipfix1.service -l
```

ℹ check service start after reboot

Troubleshooting

1. how can I get version?
   Use:

   ```
   ipfixreceiver -v
   ```

   ```
   yum info ipfixreceiver
   ```

2. can I send IPFIX streams fom differ DPI deveces to one port?
   Yes, for UDP. Ipfixreceiver will write it to the same output files.
3. How can I learn that utility is working?
   a) check port is lissening, example 1500:

   ```
   netstat -nlp | grep 1500
   ```

   b) check utility log for errors
   c) chech that data is writing to temporary file, example for 9996 port (dump directory - /var/dump/dpiui/ipfixurl):

   ```
   tail -f /var/dump/dpiui/ipfixurl/9996.url.dump
   ```

4. all checked, but no data is received?
   a) check iptables rules.
   b) check ipfixreceiver configuration for IP server address.
5. DPI sends more then 2 millions session per second, in DEBUG level I see that counter in saver is slower then counter in receiver. How can I tune performance ?
   a) remove decoder date to string, it'll lower processing time and reduce file size
   b) remove decodeipv4, it'll also a little lower processing
   c) check buffer_size for more 30 000 session per sec

d) upgrade processor for more frequency