Table of Contents

Receiving IPFIX data by ipfixreceiver

Introduction

The utility is designed to receive data stream from devices using the IPFIX protocol and save the data as a file for subsequent processing by other means.

Installation and Update

CentOS6

1. add the 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. instal the ipfixreceiver:

```
yum install -y ipfixreceiver
```

3. check for the changes in the configuration files so they to be consistent with ipfixreceiver current version, see the "Important changes" section.

CentOS7

1. add the 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 the epel repository

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

3. install the forencis repository:

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

4. install the ipfixreceiver:

```
yum -y install libfixbuf --disablerepo=forensics
yum -y install  netsa-python netsa_silk
```

```
yum -y install ipfixreceiver --disablerepo=forensics
```

5. check for the changes in the configuration files so they to be consistent with ipfixreceiver current version, see the "Important changes" section.

Important changes in version 1.0.3 relative to 1.0.2 one

1. the configuration file has been changed with respect to IP address translation, starting from the 1.0.3 version you should specify `decodeipv4`, decodeipv6 in the export model, for example:

```
  source_ip4, ''decodeipv4''
```

```
  destination_ip4, decodeipv4
```

2. the process of information saving has been allocated to a separate process; remember that when dealing with a large number of sessions (> 25k sessions per second) the process will completely load 2 processor cores. In order to check that the process has time to process the entire data stream the following messages are added in the DEBUG mode:
(a)cnt=NNNNN - the buffer has been sent with the given number
(b)cnt=YYYYY - the buffer with the given number is saved.
3. a new `buffer_size` parameter is added; it specifies the size of the i/o buffer between the process of receiving and writing to a file, it is used in the [dump] section, the default value of the parameter is 100000 records (it is focused on 20 Gbit traffic or 25 000 sessions per second). If the number of sessions per second is considerably less than the mentioned value, then you should to change this parameter proportionally.

The files supplied with the ipfixreceiver

1. configuration examples:

```
/etc/dpiui/ipfixreceiver.conf is the clickstream configuration sample
(http requests)
/etc/dpiui/ipfixreceiverflow.conf is the sample configuration for
information on sessions (netflow counterpart)
/etc/dpiui/ipfixreceiversip.conf is the sample configuration for
information on sip connections
```

2. program files are located under the:

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

   directory

3. auxiliary files:

```
/etc/dpiui/port_proto.txt contains the information on the translation
of protocol identifier to its string representation,
it is used by the utility to get the protocol text-based name by its
identifier
```

4. links to the executables:

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

Additional OS settings

1. configure iptables to accept external data
   For ipfixreceiver to work properly, you should open the ports that will also be used in the [connect] section of the configuration. For example, you use the TCP protocol, port 1500 and IP=212.12.11.10

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

   To receive an IPFIX stream, you should have the following rule in the /etc/sysconfig/iptables:

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

   Do not forget that after the adding rule to iptables a restart is required:

   ```
   service iptables restart
   ```

2. configure log rotation
   An example of rotation for the /var/log/dpiuiflow.log log file: you should create within the /etc/logrotate.d/ directory the flowlog file of the following content:

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

   Please pay attention that the copytruncate method is used, otherwise the file will be recreated and writing the log by the process will stop.
   Respectively, in the ipfixreceiver configuration file, you have the following settings in the [handler_ipfixreceiverlogger] section:

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

3. Configure the deleting of old files. For example, deleting old archives (more than 31 days) containing sessions records packed with gzip:

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

Change the line according to your requirements and add to the /var/spool/cron/root file.

Ipfixreceiver startup options

The ipfixreceiver utility has the following startup options:

```
usage: ipfixreceiver start|stop|restart|status|-v [-f <config file>]
где
  start   - start as a service
  stop    - service stop
  state   - get the service state
  restart - service restart
  -v      - show version info
  -f <config file> - specify the configuration file for the service to start

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

Configuration

The default configuration file is /etc/dpiui/ipfixreceiver.conf.
:!:More information on configuring logging can be found here: Logging

Service sections

1. loggers - specifies the log identifiers used
2. handlers - specifies the handlers used to save the log
3. formatters - specifies the formats used for the log

logger_root

1. level - specifies the logging level (upper level)
   Possible values are:

```
CRITICAL  - only critical errors, minimum message level
ERROR     - including errors
```

```
WARNING   - including warnings
INFO      - including information
DEBUG     - including debug messages
NOTSET    - all, the maximum level of messages (including all of the
above)
```

Example:

```
level=DEBUG
```

2. handlers - message handlers used
   Example:

```
handlers=ipfixreceiverlogger
```

handler_ipfixreceiverlogger

1. class - handler class
   Example:

```
class=FileHandler
```

2. level - message level

```
level=DEBUG
```

3. formatter - message format name

```
formatter=ipfixreceiverlogger
```

4. args - handler parameters

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

formatter_ipfixreceiverlogger

1. format - message format description
   Example:

```
format=%(asctime)s - %(name)s - %(levelname)s - %(message)s
here
%(name)s      - log name
%(levelname)s - message level ('DEBUG', 'INFO', 'WARNING', 'ERROR',
'CRITICAL').
%(asctime)s   - date, the default format is “2003-07-08 16:49:45,896”
(the comma field corresponds to milliseconds).
%(message)s   - message
```

2. datefmt - date format description
   Example:

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

connect

1. protocol - protocl (tcp or udp).

```
protocol=udp
```

2. host - server IP or its name.

```
host=localhost
```

3. port - port number.

```
port=9996
```

dump

1. rotate_minutes is the period in minutes, after which the temporary file in dumpfiledir/<port>.url.dump will be moved to the archive (mv) and a new temporary file will be created.

```
rotate_minutes=10
```

1. processcmd is the command that will be launched at the end of the file rotation, the file name parameter with the path to it.

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

2. dumpfiledir is a directory to store the files with data received.

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

3. buffer_size is the size of the i/o buffer between the process of receiving and writing to a file, it is used in the [dump] section, the default value of the parameter is 100000 records (it is focused on 20 Gbit traffic or 25 000 sessions per second). If the number of sessions per second is considerably less than the mentioned value, then you should to change this parameter proportionally.

InfoModel

The block specifies the data received via the IPFIX protocol.

1. InfoElements - parameter describing the information model elements for IPFIX

```
InfoElements =  octetDeltaCount,       0,    1, UINT64, True
                packetDeltaCount,      0,    2, UINT64, True
                protocolIdentifier,    0,    3, UINT8
                session_id,        43823, 2000, UINT64, True
here,
  session_id - is the name of the field from the IPFIX description, for
more detail see corresponding sections
  43823  - unique organization number (enterprise number)
  1      - unique field number
  UINT64 - field type
  True   - use reverse byte order (endian). Possible values are: 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

The field names and their description can be accessed from the following links:

1. Netflow export template using the IPFIX format
2. Clickstream and SIP export templates
3. AAA export template using the IPFIX format

Additional information:
Information Model for IP Flow Information Export

ExportModel

specifies the model parameters used for export, is reserved for future use.

1. Mode - the type of export used

```
Mode = File
```

ExportModelFile

Description of the File export model.

1. Delimiter (\t - tabulation, more examples - |,;)

```
Delimiter = \t
```

2. ExportElements - description of the fields that will be saved to the 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
where the fields in each row are the following:
  name - the field name from the information model [InfoModel] (login,
session_id and etc.)
  handler - field processing procedure before output
              seconds       - field in seconds, format is expected
              milliseconds  - field in milliseconds, microseconds,
nanoseconds format is expected
              decodehost    - recode from punycode to UTF-8
              decodepath    - recode from urlencoding to UTF-8
              decodereferer - recode from (punycode,urlencoding) to
UTF-8
              decodeproto   - recode the protocol identifier to the
string
  format - format description for seconds, milliseconds.
           Example: %%Y-%%m-%%d %%H:%%M:%%S.%%f+0300
           Result: 2016-05-25 13:13:35.621000+0300
```

Creation of systemd ipfixreceiver service in Centos7

Step-by-step creation of service in Centos 7, here the service name is **ipfix1** , its configuration is in the **/etc/dpiui/ipfixreceiver.conf** file, listening port is **1500**.

Create the /etc/systemd/system/ipfix1.service file as follows:

```
[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
```

Issue the following commands:

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

Check whether the service is running:

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

 **Do not forget to check the service status after rebooting!**

Troubleshooting

1. how to get utility version?
   You should use the following commands:

   ```
   ipfixreceiver -v
   ```

   ```
   yum info ipfixreceiver
   ```

2. Is it allowed to send IPFIX flows from different DPI to one port?
   Yes, it is. The only thing is that they can not be distinguished in the recorded flow.
3. How can I understand that the utility works properly?
   a) check that the port specified in the configuration file is listened on by the utility, for example 1500:

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

b) check the log for errors
c) check that the writing to the temporary file occurs, for example for port 9996 (directory for dump files: /var/dump/dpiui/ipfixurl):

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

4. everything is checked, but the messages are not received?
   a) it seems you have forgotten to open port in iptables
   b) it seems you have initialized ipfixreceiver with the wrong server IP.
5. a huge number of sessions (more than 2 million sessions/min) is going from the DPI, it can be seen with the DEBUG mode on that the buffer exchange counter does not have time to write before receiving the next block of records, what can be done in this case?
   a) remove the date-to-line conversion, this will reduce the time needed for processing and in addition you will receive reduction the size of resulting file
   b) remove the decodeipv4 conversion; it will not affect significantly, but you can get the higher speed of writing the file
   c) configure the `buffer_size` when number of sessions per second is more than 30k along with the following item d)
   d) increase the processor frequency and RAM