NFG - Server of NetFlow processing

Contents

Application purpose

NFG is a network server application.

NFG is designed for transmitting NetFlow from emitters (routers, firewalls, other specific devices) into data collectors compatible with Syslog. Splunk is an example of such a collector.

NFG uses memory for storage of intermediate data during processing and does not use the HDD. Therefore, NFG may consume gigabytes of memory in stressful conditions.

Specifications

- Supports OS - Windows Server x86-64, Linux x86-x64
- Receiving information format - NetFlow v5, NetFlow v9.
- Ingress L3 protocols - UDP, TCP.
- Ingress L2 protocols - IPv4, IPv6.
- Sending information format – Syslog.
- Egress L3 protocols – UDP.
- Egress L2 protocols - IPv4, IPv6.

File system

Application uses following directories:

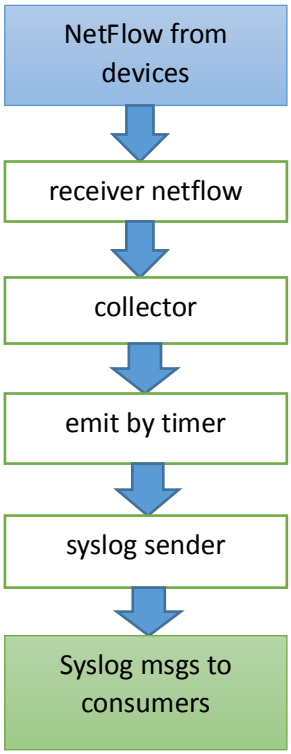
1. Executable files (below – {BIN})
    - Windows: %ProgramFiles%\NFG
    - Linux: /bin
2. Configuration files (below – {CFG})
    - Windows: %AppData%\NFServer\cfg
    - Linux-root: /etc/NFServer
    - Linux-limited user: ~/.NFServer/etc
3. Logs (below – {LOG})
    - Windows: %AppData%\NFServer\log
    - Linux-root: /var/log/NFServer
    - Linux-limited user: ~/.NFServer/log
4. Temporary files (below – {TMP})
    - Windows: %AppData%\NFServer\tmp
    - Linux-root: /tmp/NFServer/
    - Linux-limited user: ~/.NFServer/tmp

Further, all paths in configuration files and application arguments will be considered as relative from these directories.

NFG-file creation

Common description

NFG-file describes the application workflow from NetFlow receiving to Syslog sending



NFG-file is a textual file in UTF8.en-us without BOM encoding. NFG-file is placed in CFG directory.

Commands inside the NFG-file are written by Latin symbols, but the other Unicode letters are allowed in names. Commands are not latter case sensitive, but letter case for names is important. For convenience, all commands will be in lower case.

There are three objects in data processing: receiver, collector and sender. They are described by the following commands:

```
create receiver netflow …;
create collector …;
create sender syslog …;
```

Each command has arguments. Data processing is describing by the command:

```
emit …;
```

Each command with all arguments ends with a semicolon (;).

The file can contain two types of comments:

- Single-line comments in SQL-style, starting from -- (double dash) and ending at the end of the line.
- Multi-line comments in C-style, starting from * (backslash-asterisk) and ending with *\ (asterisk-backslash)

Sender and Receiver

Initially, let's consider the creation of receiver and sender. They are similar, i.e. describe essentially sockets of OS. They have two arguments: name and parameterized list.

```
create receiver netflow <Name> [param-list];

create sender syslog <Name> [param-list];
```

Name is a word without any blanks or special symbols, it has only letters in lower and upper case, underscores (_) and numbers (0-9). Name can begin with a letter only.

Every object should have a unique name, i.e. system should not have two receivers with one name and two senders with one name.

[param-list] is a list of key-values elements, added in brackets and separated by commas. For example,

```
[proto = 0, address="192.168.0.102", port=514]
```

Key is a predefined name. Value may be a number – integer or a float (floating numbers should use grammar of C language) or a string, added in quotes. Moreover, types of values are fixed for the keys. String in the parameterized list cannot contain the following simple symbols: " (quote),] (closing bracket), \ (backslash). These symbols should be screened by backslash: \", \], \\.

Sender and receiver have the following parameterized keys:

Name	Type	Default Value	Description
proto	int	0	Used transport level protocol, 0 – UDP, 1 – TCP
address	string		Remote host address (DNS or IP)
port	int	514 for syslog sender	Port, remote - for sender, local - for receiver

Examples of sender and receiver:

```
create receiver netflow rp1 [ proto = 0, port=9001];

create sender syslog ep1[proto = 0, address="192.168.0.102", port=514];
```

Collector

The next object is NetFlow collector. It can be created by the following command:

```
create collector <Name> (data-list) from <Receivers>;
```

Collector name is similar to the name of sender or receiver. Argument data-list is the list collecting fields in the following format: field_name (space) type, separated by commas.

Field name should be value of Field Type from section 8 «*Field Type Definitions*» IEEE RFC 3954 "Cisco Systems NetFlow Services Export Version 9" pages 18-24. (Available field names list will be extended in future releases.)

Type may be used one of following values:

Type	Size (in bytes)	Description
TINYINT	1	Signed integer value

SMALLINT	2	Signed integer value
INT	4	Signed integer value
BIGINT	8	Signed integer value
CHAR	1	Single character
VARCHAR (N)	N	Multiple character string
FLOAT	4	Floating point value
DOUBLE	8	Floating point value

Receivers in the third argument are names of previously defined receivers, separated by commas.

Collector example:

```
create collector Collector1(
      IN_BYTES int ,
      IN_PKTS int,
      IPV4_SRC_ADDR int,
      L4_SRC_PORT smallint,
      PROTOCOL tinyint
      ) from rp1, rp2;
```

Emit

The 'emit' command controls message emitting. All names in this command should be previously defined. If at time of reading emit arguments, one of them has been undefined, a system error will be generated.

Full description of 'emit' is:

```
emit <field_names>
      as syslog [syslog-params]
      to <senders>
      from <collectors>
      when [timer-params];
```

<field_names> - list of collector names, separated by commas. It is required. Name may be in two variants: short, when there is the field name only (for example: IN_BYTES), and long, when field name is preceded by collector name and separated by comma (for example: Collector1.IN_BYTES). Names should clearly define fields.

as syslog – optional part. It has parametrized list with keys:

Name	**Type**	**Default Value**	**Description**
severity	int	5	Syslog severity
facility	int	1	Syslog emitting facility

to <senders> - sender names, separated by commas. It is a mandatory part.

from <collectors> - collector names, separated by commas. It is a mandatory part.

when [timer-params] – trigger on data processing. It has parametrized list with following keys:

Name	Type	Default Value	Description
duration	int		Period of sending (in seconds)

Emitter example:

```
emit
      IN_BYTES,IPV4_SRC_ADDR, PROTOCOL
      as syslog [severity=5, facility=1 ]
      to ep1
      from Collector1
      when [duration = 10] --sec
      ;
```

Installation

1. Installation on Windows
    - You should run MSI install and choose necessary options.
    - You should create configuration files in {CFG} directory.
2. Installation on Linux
    - You should install RPM or DEB packet.
    - You should create configuration files in {CFG} directory.

Initial configuration

The application can be configured via console (with short name) or via CFG-file (via long name). CFG-file is in CFG directory.

CFG-file format

```
[section]
name = value
```

Command line format

```
NFServer -name=value
```

Parameters

long name	short name	default value	description
General options			
threads	t	16	Initial working thread count

memblock	m	64	Single memory block for stored NetFlow specified in kBytes (increase this parameter in case of huge flows)
	c	srv.cfg	Cfg file name
nfgfile	g	main.nfg	name of NFG file
license	l	license.lic	License filename

Activation

The application should be activated after installation and initial configuration. For the activation you should run the application and type the following command in command line:

```
activate
```

Special file 'activation.txt' will be created in {TMP} directory. You should send this file by email to developers. Developers will send the license key 'license.lic'. You should place 'license.lic' to {CFG} directory.

The activated application is depended on your hardware. If you change hardware, the application may require activation again.

You may check the activation status by the command:

```
license_status
```

It prints the current license and activation status. The application without license cannot process the NetFlow or load NFG-files.

Syslog format

After 'emit' command processing the syslog message according to IEEE RFC-3164 "The Syslog Protocol" standard will be created.

The current application version cannot handle the NetFlow data. The data will send exactly as it was received and will be stored in a collector. In case the data is stored in several collectors, the syslog messages will be sent from each available data field in all possible configurations.

Error handling

The application writes information about its work in log-file NFserver_<datetime>.log, located in {LOG} directory. It has the following format:

```
[date time]<severity>:message
```

Log contains the information about all valuable events, which happen during the application run. It contains all errors founded in configuration files.

In case of successful application start, all created objects will be logged. A partial example of a successful application start could be:

```
[2015-10-23 17:06:58]<normal>:Allocator started
[2015-10-23 17:06:58]<normal>:Compiling NFG file: 'main.nfg'
[2015-10-23 17:06:58]<normal>:Opening NFG file main.nfg
[2015-10-23 17:06:58]<normal>:Parsing succeeded
[2015-10-23 17:06:58]<normal>:Creating collectors.....
```

```
[2015-10-23 17:06:58]<normal>:     Collector 'Collector1' has created
successfully

[2015-10-23 17:06:58]<normal>:All collectors created
```

Control of application

After the successful start the application the console will open. It has the following commands:

add_thread – add one more work thead

quit, exit – close the application

help <args> - help without argument prints list of all available commands, help with argument prints the information about exact command

statistics – prints the current work statistics

Console supports command history, you can choose any previously typed commands by clicking the up and down arrow keys.