sflogcollect工具包，用于采集排障所需日志，具体使用方法请参考下文。

命令行参数

```
usage: sflogcollect [-c <config>] [-o <output.bz>]

These are common sflogcollect commands used in various situations :

start to collect edr logs:
  -c <config>    [optional] specialize a configuration file for collectting
logs, default is config.ini.
    --config <config>           same with -c.
  -o <output.zst>[optional] specialize the path of outputing file, default is
YYYYMMDD-mmHHss.bz.
    --output <output.zst>       same with -o.
  -h                       printf help string for sflogcollect.
    --help                      same with -h.
examples:
sflogcollect -c config.ini -o edr_20240716-111120.zst
```

```
-c 同--config，用于指定配置文件，配置文件内容编写，请参考“配置文件”章节
-o 同--output，用于指定最终的输出文件，输出文件格式为zst格式，使用WinRAR或7Z解压缩
-h 同--help，用于查看帮助
```

配置文件

配置文件示例:

```
[file]
bpmt = %allusersprofile%\SF\EDR\bpmt
logs = %allusersprofile%\SF\EDR\log
wer = %allusersprofile%\Microsoft\Windows\WER
winevt = %systemroot%\System32\winevt
var = $(reg:edrpath)var
config = $(reg:edrpath)config
xs_agent\var = $(reg:edrpath)xs_agent\var
xs_agent\config = $(reg:edrpath)xs_agent\config

[proc]
systeminfo.txt = systeminfo.exe
fltmc.txt = fltmc.exe

[extfiles]
dumps = %.dmp,524288000

[regmap]
edrpath = HKLM,SOFTWARE\SF\EDR\AGENT,InstallLocation
```

当前配置文件，支持file、proc、extfiles、plugin、regmap五个节，下面一一说明。

1. regmap节

```
[regmap]
edrpath = HKLM,SOFTWARE\SF\EDR\AGENT,InstallLocation
```

`regmap`节，通常用于从注册表中加载某个字段的值，要求值类型必须是字符串格式。加载后的值，在其他节中，可以以`$(reg:xxxxx)`形式引用。

2. file节

```
[file]
bpmt = %allusersprofile%\SF\EDR\bpmt
logs = %allusersprofile%\SF\EDR\log
wer = %allusersprofile%\Microsoft\Windows\WER
winevt = %systemroot%\System32\winevt
var = $(reg:edrpath)var
config = $(reg:edrpath)config
xs_agent\var = $(reg:edrpath)xs_agent\var
xs_agent\config = $(reg:edrpath)xs_agent\config
```

`file`节，通常用于采集路径固定的文件或文件夹，文件或文件夹路径支持环境变量扩展，也可以引用regmap节中定义的值。

`file`节中的每一行，分别为一个键值对，键标识了采集的文件在压缩包中的路径，值标识了文件的原始路径。

例如:

```
bpmt = %allusersprofile%\SF\EDR\bpmt
```

表示采集`%allusersprofile%\SF\EDR\bpmt`路径对应的文件或文件夹，将这些文件打包到压缩包中的`bpmt`路径。

实际效果如下:

3. proc节

```
[proc]
systeminfo.txt = systeminfo.exe
fltmc.txt = fltmc.exe
```

`proc`节，用于执行某个进程，并将进程的标准输出中的内容打包到压缩包中的指定文件中。

`proc`节中的每一行，分别为一个键值对，键标识了采集的文件在压缩包中的路径，值标识了进程启动路径及参数，路径也支持`regmap`及环境变量扩展。

4. extfiles节

```
[extfiles]
dumps = %.dmp,524288000
```

`extfiles`节，如果存在，无论有没有配置键值对，都会检索系统盘中的所有文件，并记录到数据库中。数据库会压缩到最终的压缩包中，名称固定为`all_disk_files.sdb`，查看该数据库文件内容，请以`sqlite3`数据库格式打开。

`extfiles`节中的键值对，键标识了压缩包中的目录，值可以为以下两种形式：

1). `like`子句中的模糊匹配串

2). `like`子句中的模糊匹配串 + `,` + 最大文件大小

第一种形式，将会忽略大小限制，直接检索`系统盘`下所有文件名能与`like`子句中模糊串匹配的文件，压缩到键标识的压缩包目录中，为避免重复，文件将会被重命名为`原始文件名` + `编号`的形式。

第二种形式，设置了最大文件大小，单位为`byte`，超过该大小的文件，将会被忽略，不打包到最终的压缩文件中。

5. plugin节

`plugin`节为用户自身编写的`dll`，该`dll`文件必须有`collect`导出函数，该导出函数签名如下：

```
/* collect_fn_t plugin对外sflogcollect进程导出的collect函数签名格式，导出的collect函数将会被                    sflogcollect进程调用，用户可以在该函数中，写入该插件需要采集的日志内容
 * stream:       hplugin句柄，即插件dll的模块句柄.
 * buf:          缓冲区，插件中写入该缓冲区，即可直接写入压缩包中.
 * size:         缓冲区长度，插件务必保证不会写入超出该大小的数据到缓冲区中.
 * return   int32_t:     返回写入长度，返回0表示流读取结束.
 */
typedef int32_t (*collect_fn_t)(void *stream, void *buf, int32_t size);

// collect函数必须符合collect_fn_t的签名形式，否则可能会导致采集进程崩溃。
```

注意: 当前版本，所有用`,`分隔的场景下，`,`前后都不能有空格或其他空白字符。