NPR Automation Instructions

by Asaf Rokni

The NPR Automation is an executable automation that creates input files for the NPR process as well as the PUP Json file for disabling content.

Inputs:

The automation requires several inputs:

1. A Test Program (TP) on which you wish to perform your actions.
2. A directory with input files.
3. A directory for output files.

The automation is designed to work in several ways:

1. Using the CMD/PowerShell by supplying the 3 arguments in their respective order.
2. Running the .exe file from within a TP. The program will locate the Inputs and Outputs directories which must be provided alongside the .exe file's location. The files should be located 2 directories below the TP itself.
3. From an IDE: the user will be prompted to supply the 3 arguments one by one.

The program works in such a way that if it did not receive the arguments, it will try to locate them automatically (like in the second option) and should that fail as well it will prompt the user for the inputs (like in the third option).

- When providing the output directory link, if the directory is not empty or doesn't exist, the program will alert the user, asking whether to continue with the procedure or not. In the first and second executing options it will create the directory without asking, assuming there is a parent directory.
- After the user approves to continue, a verification process begins.
- If there's anything wrong with the input files, a log file will be created in the output directory providing details on the errors. Otherwise, the procedure will continue to execute the automation and the output files will be placed in the output directory provided.

The input files must adhere to the following rules (an explanation on each can be found below):

1. Rules directory - a directory that contains **only** ".rule" files.
2. CSV file - a file with details on what to capture from the TP.
3. Configuration file - a ".conf" file which holds configurations for the automation.
4. JSON file - Optionally an ab_list file.

Rule files:

1. Must have as part of its name a plist on which it will work.
2. It is important that the naming convention of the plist will contain some form of search or check option (SRH/CHK for example) much like in the test instances themselves.
3. Must have a line that states "::ENABLECONTENT::" only.
4. Every tuple or TID the user wishes to enable (keep) should be typed below the previous line. Each tuple or TID must be in a separated line and with the name of the plist preceding the number itself.

- Rule files can be created by the RuleFilesCreator automation.

CSV file:

1. Must contain 4 columns with the first line always containing in this order:
    a. Test Regex.
    b. Options for flows divided by a pipe character "|". The options must include one for check subflow and search subflow with no regards to the order or appearance (example: CHK|SRH or SRH|CHK|EMAX - here EMAX is an optional subflow).
    c. Power Domain.
    d. Corner.
2. The following lines should be filled according to the header line with a regex to capture test instances, the subflow in which the test instances are supposed to be (only one subflow allowed and it must appear in the header line as well or an error will occur), the power domain which the test instances are reporting to and the corner in which the test instances are located. The corner can remain empty in such cases where the subflows are not a check or search ones.

Configuration file:

1. Every line must be in the same format of "Field: member A, member B, member C…".
2. The mandatory fields are: LocationCodes, Encode, SearchOption, CheckOption, OutputsPathInTP.

3. The optional fields are: ExcludedPlistsRegexes, OtherOption, BaseNumberLength, DontRunChk, IgnorePatternsWithRegexes.

   Further explanation on each field:
   a. LocationCodes - numbers that represent the location codes you wish to work on, separated by commas (example: 6248, 6197).
   b. Encode - letter combinations derived from the BLLEncoder schema, each combination represents a flow to which disable content, separated by commas (example: CH is for CLASSHOT, CC is for CLASSCOLD).
   c. SearchOption - defining the name that represents the search subflow in the TP in the test instances (example: SRH).
   d. CheckOption - defining the name that represents the check subflow in the TP in the test instances (example: CHK).
   e. OutputsPathInTP – a link to where the output files should be created at. This link will be combined with the TP link.
      Example:
      TP link: C:/TP
      OutputsPathInTP: X/Y/Z
      The files will be create at C:/TP/X/Y/Z
      You can place the link with either backslash or forward-slash.
   f. OtherOption - defining the name that represents the other subflows in the TP in the test instances to be searched (example: EMAX, LMAX).
   g. ExcludedPlistsRegexes: a list of the plists to ignore from while executing the automation, meaning they won't be worked on and won't be touched.
   h. BaseNumberLength: the length of the base number when using the json file of the ab_list.
   i. DontRunChk: indicating whether to run the automation on CHK instances as well, can be either True or False.
   j. IgnorePatternsWithRegexes: a list of regexes that the automation will ignore patterns that are caught by them.

Outputs:
Upon finishing the execution, in the output directory 5 files will appear (further explanation below):
1. LogFiles directory.
2. SourceFiles directory.
3. NPRCriteriaFile.csv file.
4. NPRInputFile.csv file.
5. PAS_PTD.pup.json file.
6. FlatFile.csv file.

These outputs can now be used by the test program where needed.

LogFiles:
Inside the directory there are 3 files:
1. BasicStats.csv - this is a table that shows a list of the plists that were being checked, how many patterns they had, how many were disabled, how many not and percentage of disabled.
2. Log.txt - a simple log file detailing all that the automation has done.
3. SRH_CHK_Mapping.csv - a table detailing the test instances caught, their information such as power domain and corner (taken from the input file based on the regex that caught them), their template, scoreboard (if applicable) and patlist.

SourceFiles:
Inside this directory there is a copy of all the input files with the addition of a directory called "Plists" that has a copy of all the plist files that were captured during the process.

NPRCriteriaFile:
This file is a table that has several columns with information.
For each LocationCode value and Encode value combination provided in the configuration file there are 2 lines that create the ModelNames that are being used in the creation of the NPRInputFile file.

NPRInputFile:
A table with 3 columns where for each ModelName from the NPRCriteriaFile there are all the plists that are relevant and if applicable an IP name (such as IP_CPU or IP_PCH).
For a model name that ends with 0 there will be the SRH instances and for the model name that ends with 1 there will be the CHK instances.

PAT_PTD.pup:

A JSON file containing per Encode a list of all the plists that should have patterns (at least one) to disable with said patterns defines in a list per plist.

- The Name and StepName for each process is coming from BLL, therefore if there are any changes to the definitions of the BLL encoder, it should be adjusted in the file called BLLEncoder in the automation's source code.

FlatFile:

This file is a table containing 3 columns of information.

Each row is a pattern (on column C) belonging to a patlist (on column A) and whether this pattern is affected by NPR or PUP. Essentially, it's a list of all the patterns that are going to be executed, a mirror image to the PAT_PTD.pup file.