|  |
| --- |
| Title : MPE/MDI/Parsing Case Work Training: Using the Log Analyzer Tool for SCMPE Analysis. |
| URL Name |
| Identification: This article is written as part of a group of Knowledge Articles titled “Parsing Cases General Training ”. In a few of those articles there is reference to the Log Analyzer Tool and this article is designed to show you how to leverage this tool in respect to parsing.  Basic Information: Log Analyzer is by navigating to the following address. Note. You must be on the VPN in order to obtain access.  <http://10.5.0.233/logAnalyzer.php>  If successful, you should be looking at something like Screenshot 1  **Screenshot 1**  Text  Description automatically generated  Your first step on this screen will be to upload your scmpe.log that you should have collected from the customer. Notice in Screenshot 1 *if your file name is anything other than scmpe.log as pictured below*: **it won’t work**. At times with downloading and such the file becomes scmpe(1) or scmpe\_insertname. Anything other than **scmpe.log** will appear to load and then do nothing. See in screenshot 2 where this has been corrected. This will process and you’ll see it snap into place upon clicking “upload log”.  **Screenshot 2**  Graphical user interface, text  Description automatically generated  Once you have your results loaded they should looking something like Screenshot 3. Following that you will see screenshot 4 which will be the basis for the rest of this knowledge Article.  **Screenshot 3**  A picture containing text  Description automatically generated  **Screenshot 4**  **Table  Description automatically generated**  This is the final output that will be helpful to attach to cases using the Log Analyzer tool. You can see that the Log Analyzer breaks apart the 2052 warnings in the scmpe.log into separate sections and quantifies them. Seeing the scmpe.log in this format can often be just as (if not more) informative as reading the log itself. You can see that the columns divide it out into:  **MsgSourceID TimeOuts –** This should align with Log Source ID on the Log Sources Tab in Client Console.  **MPERuleRegexIDTime-outs** – This will align with a specific MPERuleRegex found in MPE Rule Builder by clicking on the yellow folder. Every policy is made up of MPE Rules which each have MPERuleRegexID.  **Base MPERuleID Time-outs –** Many times you will see these align with the MPERuleRegexID and the biggest difference is that an MPERuleID “can” be a sub rule.  **MsgSourceTypeID Time-outs –** This links to a table in the EMDB on SQL where every type of log that LogRhythm can parse is listed. If you can’t identify the type of message it is then you can reference this table and at least see what the machine took it to be.  **What to do with this information?**  The most important thing to gather from this screen is a numeric of how many timeouts are occuring and where. Start by looking at the **MPERuleRegexIDTime-outs** column. In screenshot 4 you can see that the top **MPERuleRegexIDTime-outs** are:   |  |  |  |  | | --- | --- | --- | --- | | **Count** | **MPERuleRegexIDTime-outs** | **Plain Text Name** | **Policy** | | 506 | 1000478 | DHCP Message | Syslog - InfoBlox | | 411 | 1011797 | User Privileges Logs | Syslog- VMWare | | 400 | 1005006 | SSH Connection | Various Syslog | | 249 | 1006108 | Query Statement | Syslog - InfoBlox | | 246 | 1009928 | General VMWare | Syslog- VMWare |   You can use this information combined with the KA - Parsing Improvement Timeouts & Rule Performance Improvement to analyze the LPS\_Detail.log. In order to align MPEERuleRegexID with actual rule names you can use the MPE Rule Builder or the queries built below:  The below query was constructed by a few support engineers working together and has a two primary functions: identifying MPERuleID OR identifying MPERuleRegexID. See below how to use:   1. Open SQL Management Studio 2. Select “New Query” (don’t worry about selecting a DB..it’s pre-populated in the query) 3. Paste the below query 4. Where you see “SET” on lines 5/6. Modify your MPERule ID or MPERuleRegexID 5. Uncomment the specific function you are using to lock in the variable by deleting the - - 6. Note: It will probably work in strange ways and give weird info if you use both variables at the same time but it’s easier to just use one of them.   USE LogRhythmEMDB  GO  DECLARE @MPERuleID INT, @MPERuleRegexID INT  --SET @MPERuleID = 1055609  --SET @MPERuleRegexID = 1000478  SELECT mper.MPERuleRegexID, mper.MPERuleID, mper.FullName AS "MPE Rule Name", mst.FullName AS "Log Source Type"  FROM dbo.MPERule mper  FULL JOIN dbo.MPERuleToMsgSourceType mpertmst on mper.MPERuleRegexID = mpertmst.MPERuleRegexID  FULL JOIN dbo.MsgSourceType mst ON mpertmst.MsgSourceTypeID = mst.MsgSourceTypeID  WHERE mper.MPERuleRegexID = @MPERuleRegexID OR MPERuleID = @MPERuleID  ORDER BY [Log Source Type] |
| Remediation Process   1. Open LogAnalyzer <http://10.5.0.233/logAnalyzer.php> 2. Upload scmpe.log (ensuring file name is correct) 3. Interpret results. Use SQL queries to align MPERuleRegexID, MPERuleID, and CommonEventID with information found in logs and Rule Builder. |
| Root Cause |
| Additonal Resources |
| Tags |