|  |
| --- |
| Title : Parsing Defects/Parsing Improvements: Where To Start When You Don’t Know |
| URL Name |
| **PART II of VI - Parsing Defects.**  Parsing Defects will make up a part of your normal queue as a LogRhythm Technical Support Engineer. There is often confusion as to the difference between defect and improvement. In an effort to standardize those definitions you can ask yourself the following: ***“do I have one or more field parsed into an INCORRECT field”?*** If the answer is “yes” you have a defect. If the answer is “no” you are more than likely looking at an improvement. Further explanation on why that is the case is offered in the “additional resources” section below.  An additional note is that you will see more parsing improvements than defects in your role as a Technical Support Engineer. When you do see a defect in the wild it will look something like the example provided in Screenshot 4. You might notice the log sample in Screenshot 4 from the Knowledge Article – Parsing Cases General Training. I used this log and intentionally broke the parsing policy to simulate what it would look like to have a defective parsing rule out in the wild.  **In screenshot 4 you can notice that I switched *dname and session* in the regex**. This is a good point to re-highlight that the regex is not magic. It is simply a grammar structure that states “the information for [*dname]* will be here” and “the information for [*session]* will be there”. The system parses and tabulates the information at those locations into those categories. When I switched them, you can see that I was able to parse [glsp01.southcentralbank.biz] into the session category and [0] into the [Dname] category. It could also be one field parsed into the wrong field rather than two. The point is that the parsed information is clearly incorrect and ***defective.*** Revert to those words ***incorrect/defective*** when deciding between parsing improvement or parsing defect and you’ll be fine.  **Screenshot 4**  Graphical user interface, text  Description automatically generated  When you see this in the wild; your steps will be to access the directions on Confluence. Pro-Tip: bookmark the page but access Confluence each time. **DO NOT screenshot those instructions.** They change more frequently than the forecast in Colorado. Reading from Confluence will ensure you are doing it correctly. Below I have laid out step by step how I would work this case if it were in my queue.   1. Inform customer I am working on the case and ensure that I have LLX, CSV, SCMPE, and LPS Detail. 2. Open Rule Builder in Super-User Mode (ask a teammate about Super-User if confused). 3. Run the LLX against the entire policy to identify which rule in the policy these logs align with if that information is not already known. 4. Run the LLX against that specific rule. Analyze the results in your rule builder against those in the customer’s CSV. **(\*CSVs will show how a log parsed on the customer side…LLX is a secure transmission of log raw data that can be parsed and re-parsed).** 5. In this case I would see glsp01.southcentralbank.biz parsed to “session” on my rule builder as well as on the CSV. This would conclude as a “parsing defect”.   The next step will be to complete the template provided on Confluence for Parsing Defect. The template at the time of authorship on 24 Jan is shown below. You will notice that the template covers classification and common event. You can add what you need to. The template shows in purple and response are in green. **REMEMBER** to always check Confluence to make sure no changes have been made to this template in an elevation analysis:  **Parsing Defect - Misclassified/Wrong Common Event : It seems that a user is being parsed into the session category incorrectly.**  Existing customer or proof of concept? Existing Customer  Complete log source type name: MS Windows Event Logging XML – System  Is the customer on the latest KB? If not, why not? Yes, Version 7.1.630  Does the logging format adhere to our standards (e.g., the correct IIS fields are enabled, etc.)? Yes  Which base-rule is affected by the wrong classification? Catch All: Level 2  MPERuleRegexID: 1008164  Actual classification: Classification is fine….user being parsed to session.  Actual common event: Common event is fine…user being parsed to session.  Desired classification: parse this user correctly to Dname.  Desired common event: parse this user correctly to Dname.  **\*You can and I normally add some prose at the bottom such as:** Hey MDI Team. I can see on this case that the dName is Session seemed to be mixed up. I looked at the regex and my best thought is that the regex in fact looks defective because it is showing that the computer should parse into session which doesn’t look correct. I tested the logs with Rule Builder on my end and they match the customer’s CSV. Elevating to you for further troubleshooting.  At this point you would complete the process by informing the customer and elevating the case as the instruction on Confluence indicate. |
| Remediation Process |
| Root Cause |
| Additonal Resources  This is a great resource for 20 most common Regex Patterns to learn more:  <https://regexland.com/most-common-regular-expressions/>  Another great tool for work and for learning about Regex is Regex101. It’s interactive:  <https://regex101.com/>  **Rationale for Defining Defect over Improvement:**  The rationale for this narrow definition is because it needs to be sorted out if a parsing rule is defective or if the regex wasn’t originally written to parse that field. A great analogy for this is if you buy a Prius because it gets great gas mileage in the city. If you try to drive said Prius up a mountain pass in a snowstorm it might not work. That doesn’t make the Prius defective, it simply means the Prius wasn’t built for driving up mountain passes in snowstorms.  This applies to many customer cases in that our regex policies are developed by teams of technical engineers who collaborate with sales engineers who collaborate with marketing experts. The idea is that out of the box it is designed to meet the vast majority of use-cases that is predicted to be used by our customer base. Does it occasionally happen that things are misidentified? Yes, that is why this process exists. Does that make the policy defective? No.  It should further be noted on this topic that this is why it is referenced above to use the Confluence page for up-to-date information on MDI templates. The logistic challenge to find, understand, and improve regex patterns is a massive undertaking and you will occassionally run into customers who do not have the understanding of how massive that undertaking is. Technical Support Engineers do not build customer regex as it is typically out of our skill set. We can elevate to MDI after identifying the issue OR help facilitate communication with the account team to obtain Professional Services. |
| Tags |