

Sed

Sed is a powerful stream editor in Unix and Linux systems, perfect for making automated modifications to files. It is widely used in scripts and command-line workflows for its ability to process text line-by-line efficiently.

- Sed operates by reading input line-by-line and applying commands.
- It supports simple text replacements as well as complex transformations.
- Despite its simplicity, sed's documentation often feels overwhelming—don't worry, we'll cover it step by step.

Common Use Cases for Sed

Sed is a versatile tool for many scenarios. Below are some practical applications:

- **Search and Replace:** Quickly locate and replace text in files, such as modifying credentials or configuration settings.
- **Log Sanitization:** Remove or alter entries in logs to obfuscate traces of red team activity.
- **Mass Text Modifications:** Edit multiple files or lines of a file simultaneously.

```
sed 's/oldWord/newWord/flags' < inputFile >
outputFile
```

Command	Explanation
<pre>sed 's/OldWord/NewWord/flags' < InputFile > OutputFile</pre>	To change / replace a word in a file
<pre>sed 's/OldWord/NewWord/;s/OldWord1/NewWord1/' < InputFile > OutputFile</pre>	To change / replace more than one word in a file
<pre>sed 's/OldWord/NewWord/g' < InputFile > OutputFile</pre>	- To change each / all of the OldWord in a file. - g means global and it is a flag
<pre>sed -i 's/OldWord/NewWord/flags' < InputFile > OutputFile</pre>	To save the result of sed in the original file

Command	Explanation
<code>sed '/Word/d' file.txt</code>	To delete a word from a file
<code>sed -i '/Word/d' file.txt</code>	To save the file after deleting a word

Notes

1. `sed` doesn't change the value of the original file.
2. If there is multiple number of `OldWord` the replacement will happen for the first `OldWord` in each file -> when using the normal `sed` without the `/g`.