Deliverable: Documentation

Link to README_Cont

The information in this README is stated here as well.
Link to README:
https://github.com/bferro373/ECE49595-OSS/blob/dev/README_Cont.md

Developer Experience Report

When contributing to Cutter, while the documentation is very thorough, there is some room left to be desired. There were some questions about the specifics of the app and there wasn't a true guide to the different files within cutter and what they do. Cutter was absent of a system diagram, but that is alright. The documentation was enough to understand how the system works.

We found it hard to get started due to the sheer size of the code base with how many files there were. Again, no guide to those files made us spend a lot of time exploring what these files were and which ones were important to our solution.

The build documentation for cutter was amazing from a linux and mac perspective, however it is a little outdated as some of the dependencies required to build cutter are no longer in use, requiring us to find their updated forms. From a windows perspective, the build is incredibly complicated and in depth, requiring a few extra steps but this is well documented on Cutter.

The contribution guidelines and how to format a PR request was also very well documented and easy to understand, allowing for a seamless transition between coding our solution and getting feedback. Though it took the developers quite a while to get the feedback to us at points. Which slowed down the development cycle overall.

Overall, while the documentation of Cutter can be lacking in some places and is outdated in others, the parts that truly matter, the contribution guidelines and PR request formatting, was communicated very well. The instructions for how to use and develop this tool are somewhat similar, though the development talks more in depth about formatting.

Issue Documentation

Issue 1: Allow adding flags from hexdump

Link to Issue: https://github.com/rizinorg/cutter/issues/2932

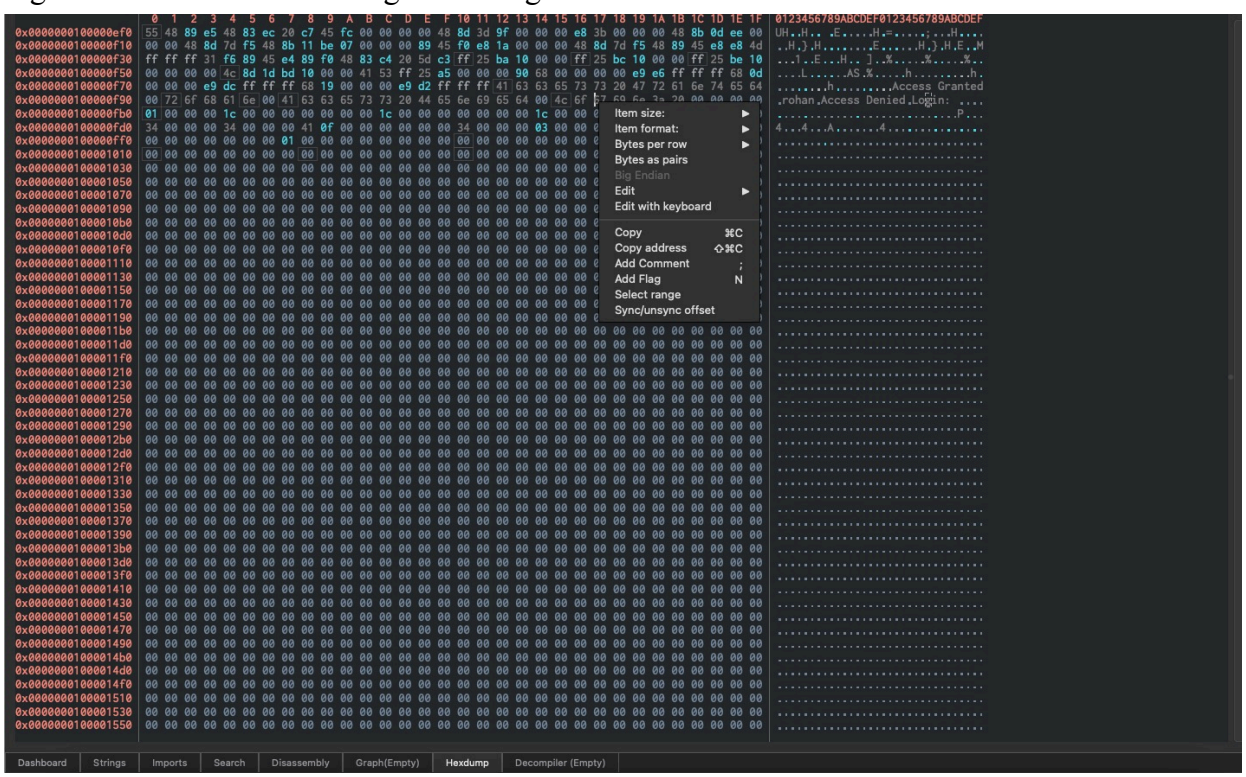
For this issue, we were tasked with adding a feature in the hexdump view that allows flags to be added into the hexdump view.

This is done by right clicking or pressing the keyboard shortcut N on the line you wish to add a flag to, entering a valid flag name, then confirming that action.

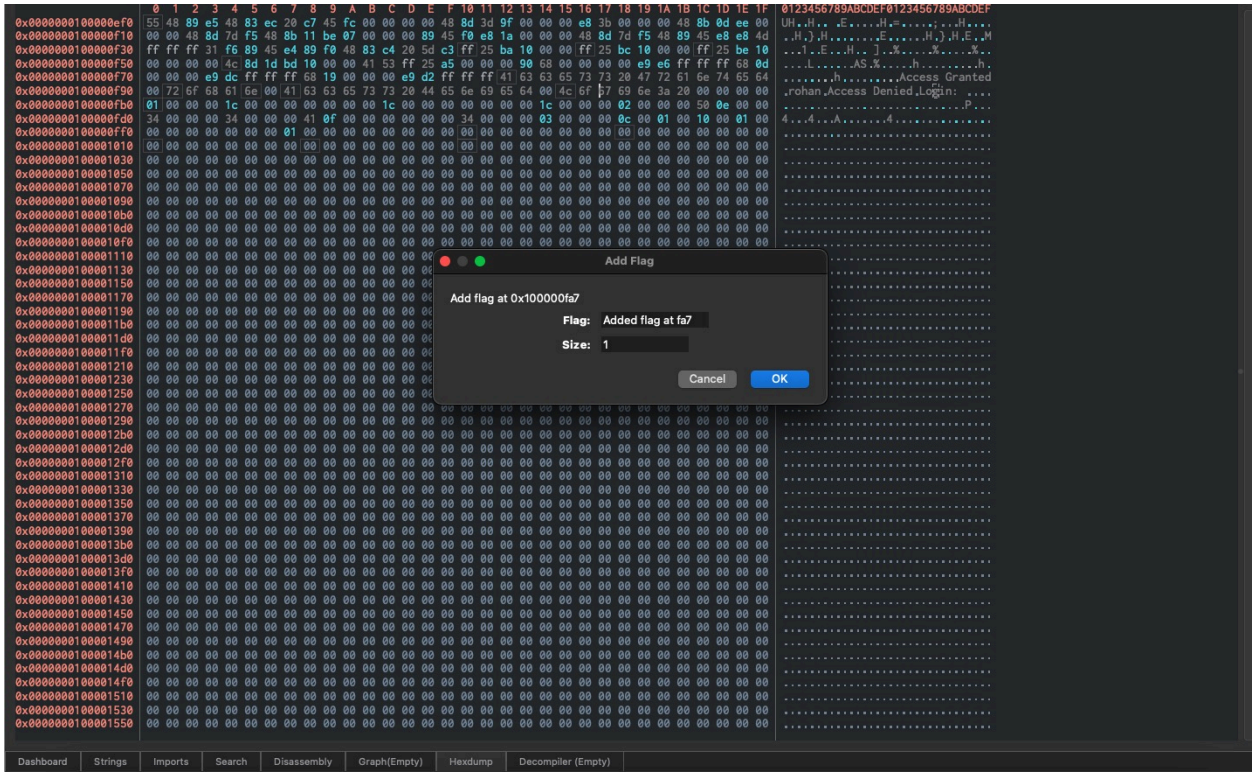
The flags added into the hexdump view are synchronized with the disassembly view.

Screenshots for issue 1:

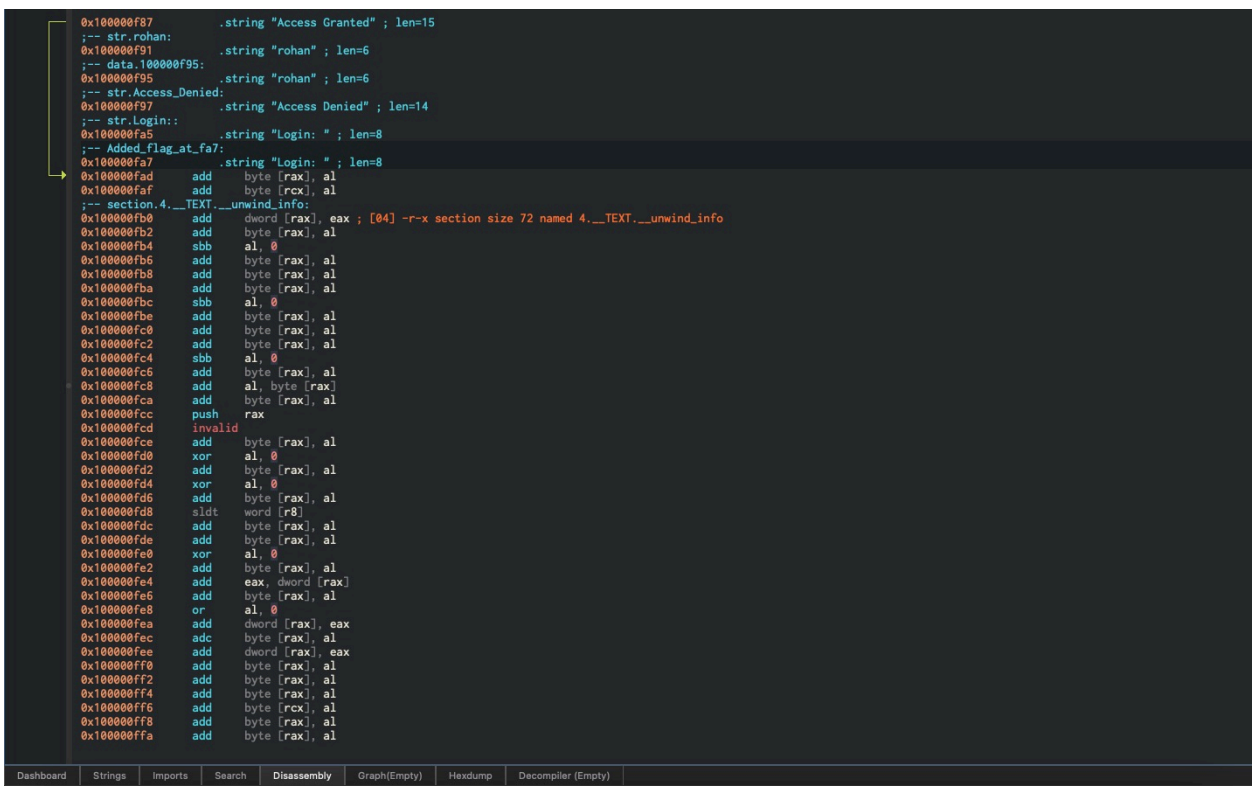
Right Click shortcut for adding a new flag:



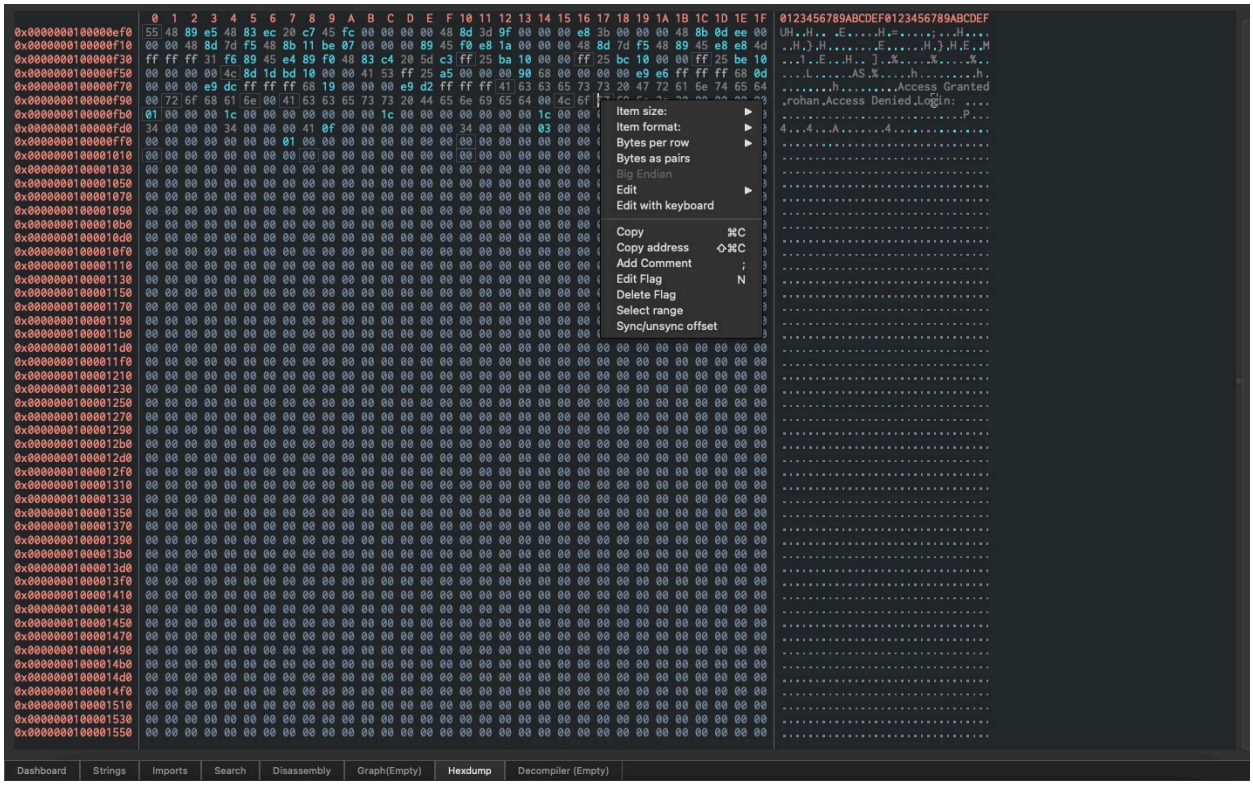
Menu for adding a flag



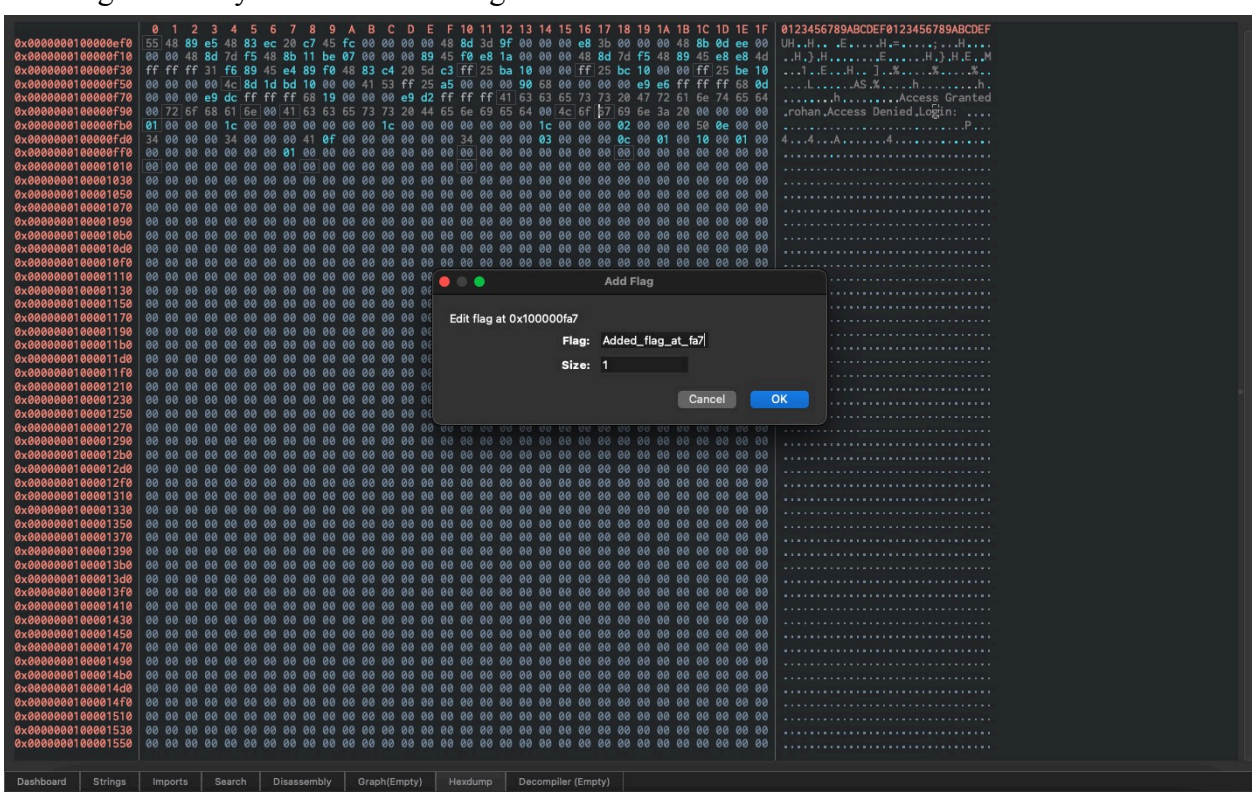
Synchronization in the disassembly view



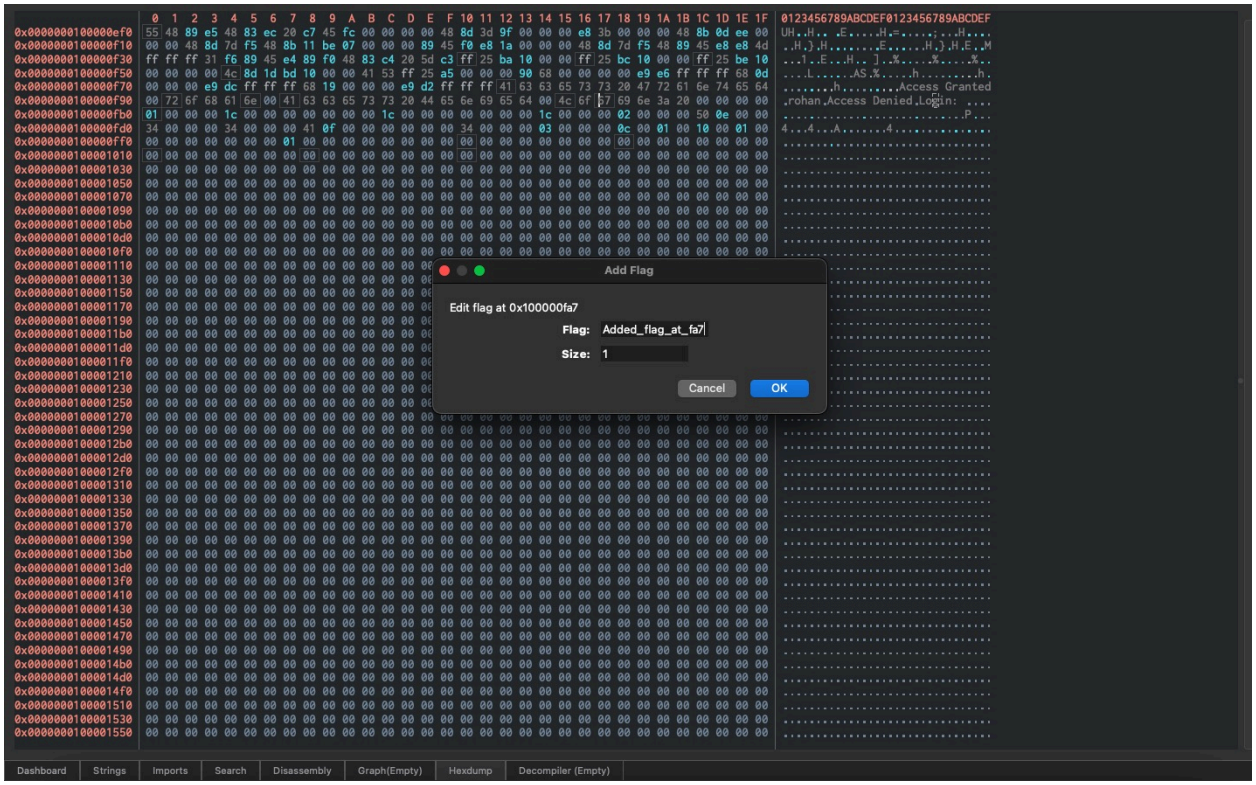
Showing the new flag in the hexdump view



Showing the ability to edit the new flag



Showing the edit menu for an existing flag in the hexdump view



Issue 2: Multiline Operations / Multiline Selection

Link to issue two: https://github.com/rizinorg/cutter/issues/2601

For this issue, we were tasked with adding the ability to select multiple lines of code at a time and let the user perform operations across these selected lines.

You can select multiple lines by pressing shift while you left click each line you want to perform operations on. The operations menu can then be opened by right clicking and selecting the operation.

Some of the new operations that will be added as we develop are the ability to define lines as code if they are currently defined as data, and define lines as data if they are currently defined as code.

Screenshot for issue two:

Multiple lines being selected

```
0x00000350    invalid
0x00000354    ldr     d23, 0xc9788
0x00000358    invalid
0x0000035c    strh    w27, [sp, w12, sxtw 1]
0x00000360    invalid
0x00000364    ldr     s20, 0xffffffffffffc25a8
0x00000368    invalid
0x0000036c    invalid
0x00000370    invalid
0x00000374    ldxp    w29, w15, [x6]
0x00000376    ldff1h  { z29.s }, p3/z, [x10, z12.s, uxtw]
0x0000037c    invalid
0x00000380    invalid
0x00000384    invalid
0x00000388    stp     x9, x2, [x27, 0x60]!
0x0000038c    smov    x12, v29.b[8]
0x00000390    ldr     d10, 0xb26e8
0x00000394    invalid
0x00000398    invalid
0x0000039c    adr     x2r, 0xffffffffffff64d9b
0x000003a0    invalid
0x000003a4    invalid
0x000003a8    bic     p3.b, p10/z, p11.b, p4.b
0x000003ac    ldnrh   { z9.s }, p4/z, [x10, 0x28]
0x000003b0    invalid
0x000003b4    prfm    pldl1strm, 0x45e54
0x000003b8    invalid
0x000003bc    invalid
0x000003c0    adds    x6, x2, 0x3f1
0x000003c4    mov     z26.b, p2/m, z8ff.b,w12, 3
0x000003c8    stp     d13, d0, [x2, -0x70]!
0x000003cc    tbz     w18, 0x11, 0x34cc
0x000003d0    invalid
0x000003d4    st4b    { z17.s }, p6, [z2.s, x12]
0x000003d8    invalid
0x000003dc    invalid
0x000003e0    strh    w2, [x13, 0x18a]
0x000003e4    stlur   q13, [x16, -0x56]
0x000003e8    invalid
0x000003ec    invalid
0x000003f0    invalid
0x000003f4    invalid
0x000003f8    invalid
0x000003fc    sth     { z22.s }, p4, [x25, z27.s, sxtw 1]
0x00000400    invalid
0x00000404    invalid
0x00000408    invalid
0x0000040c    cbnz    x2r, 0xe4978
0x00000410    invalid
0x00000414    b       0xffffffffffd6e5d38
0x00000418    invalid
0x0000041c    ldr     q19, [x18, x27]
0x00000420    invalid
0x00000424    invalid
0x00000428    invalid
0x0000042c    attr    x4, [x4, -0x59]
0x00000430    invalid
0x00000434    fmla    z11, z16, v13.s[3]
0x00000438    facgt   v12.2d, v14.2d, v7.2d
0x0000043c    subs    x27, x10, x26, asr 52
0x00000440    orr     z1, z8, z3, asr 56
0x00000444    invalid
0x00000448    stp     x8, x6, [x11, 0x188]
0x0000044c    invalid
0x00000450    invalid
0x00000454    fnsub   h6, h7, h27, h5
0x00000458    invalid
0x0000045c    invalid
```