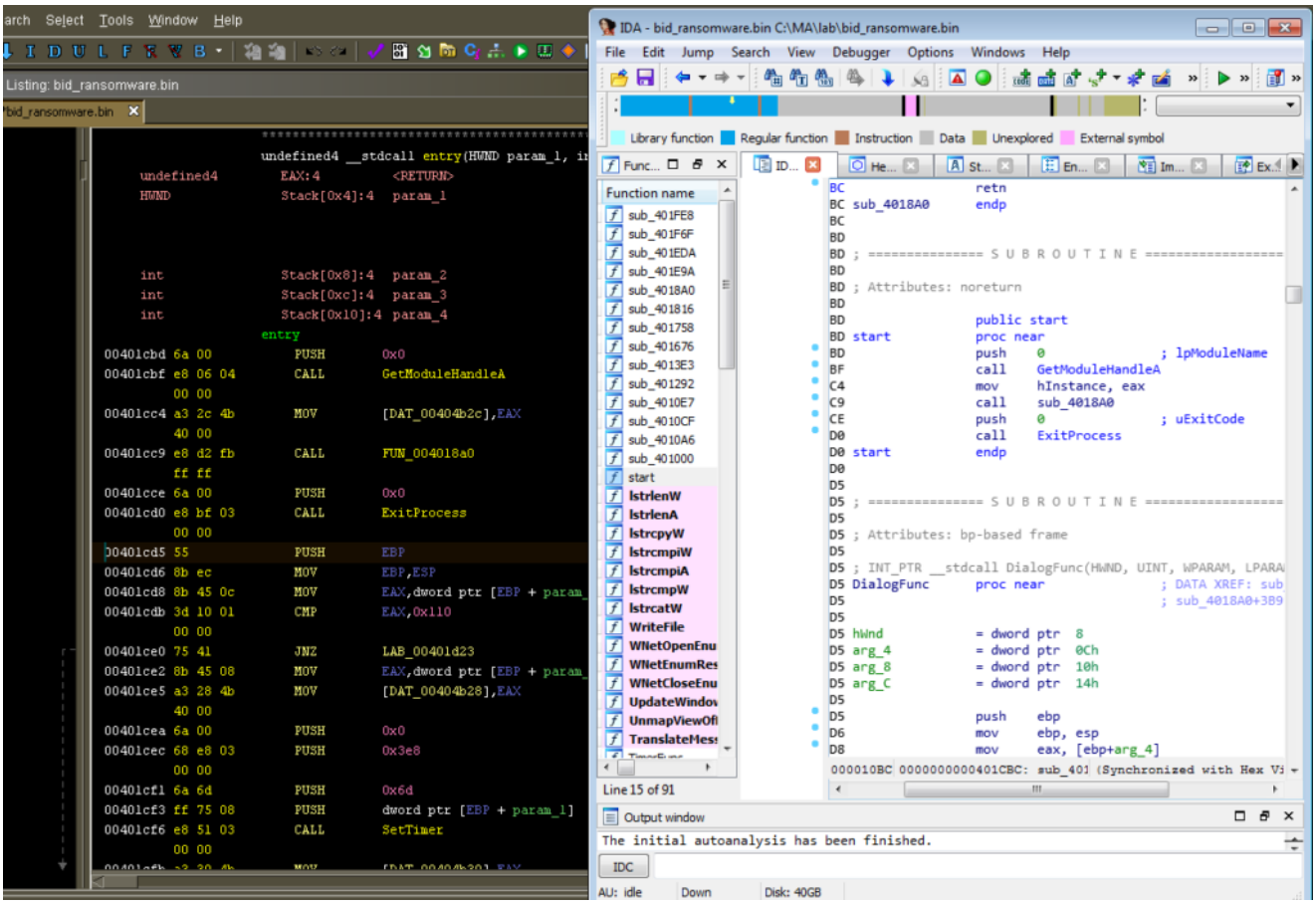
BI_D Ransomware Redux (Now With 100% More Ghidra)

zirconic.net/2019/03/bi_d-ransomware-redux-now-with-100-more-ghidra/

rhyolite March 10, 2019

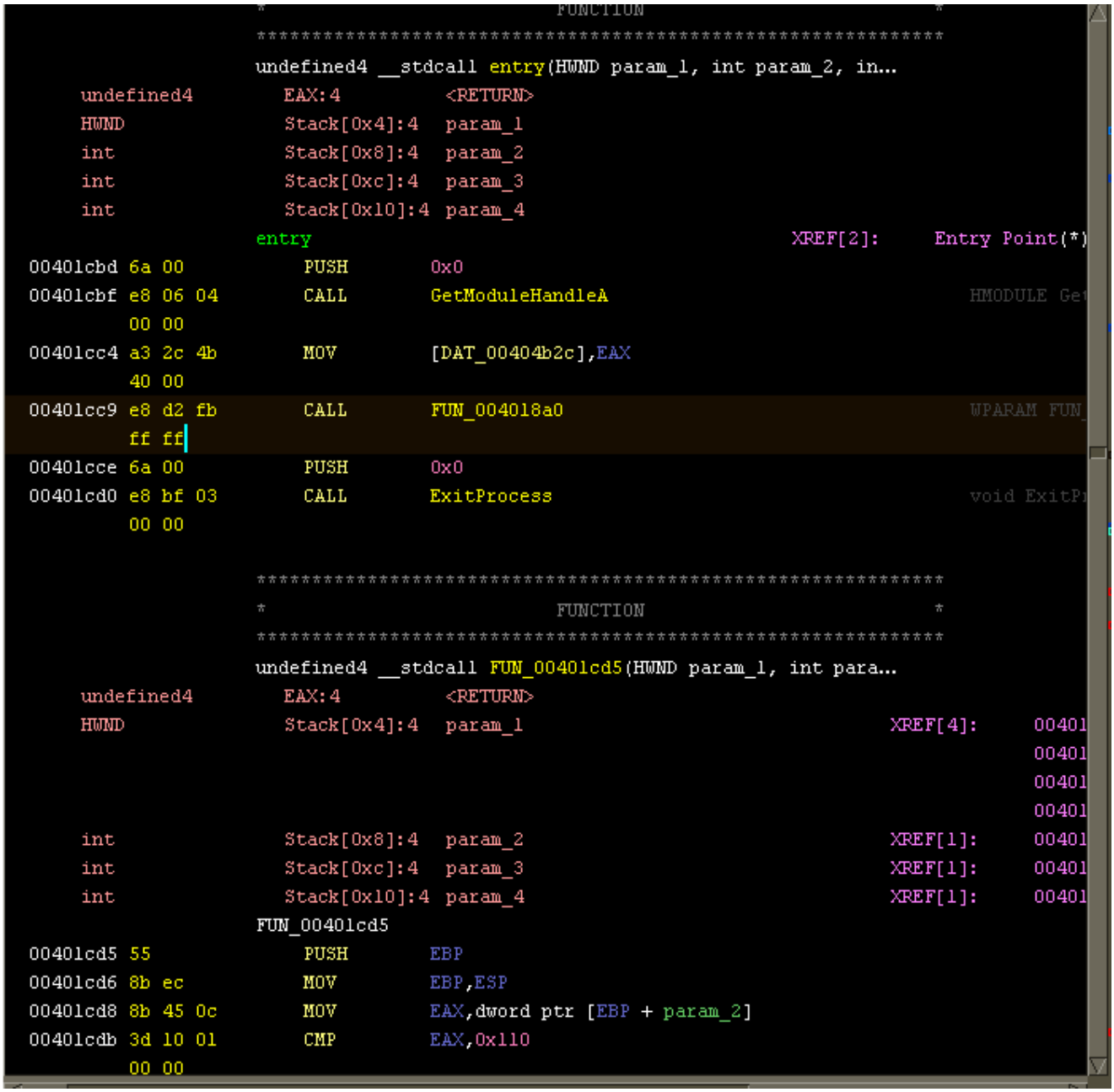
I'm still digging into Ghidra, building off of my last post which was meant to be a kind of "IDA to Ghidra Crossover" guide. For more Ghidra practice, I took a piece of ransomware that I analyzed before (using IDA) and worked on it with Ghidra. Whenever it makes sense I'll do a side-by-side comparison. I'm using Ghidra 9.0 Public and Ida Free 7.0 (both running in a 64-bit VM).

Once I loaded the ransomware, one thing I noticed immediately is that Ghidra didn't catch that there was a new function right after the entry/start function, but IDA did:



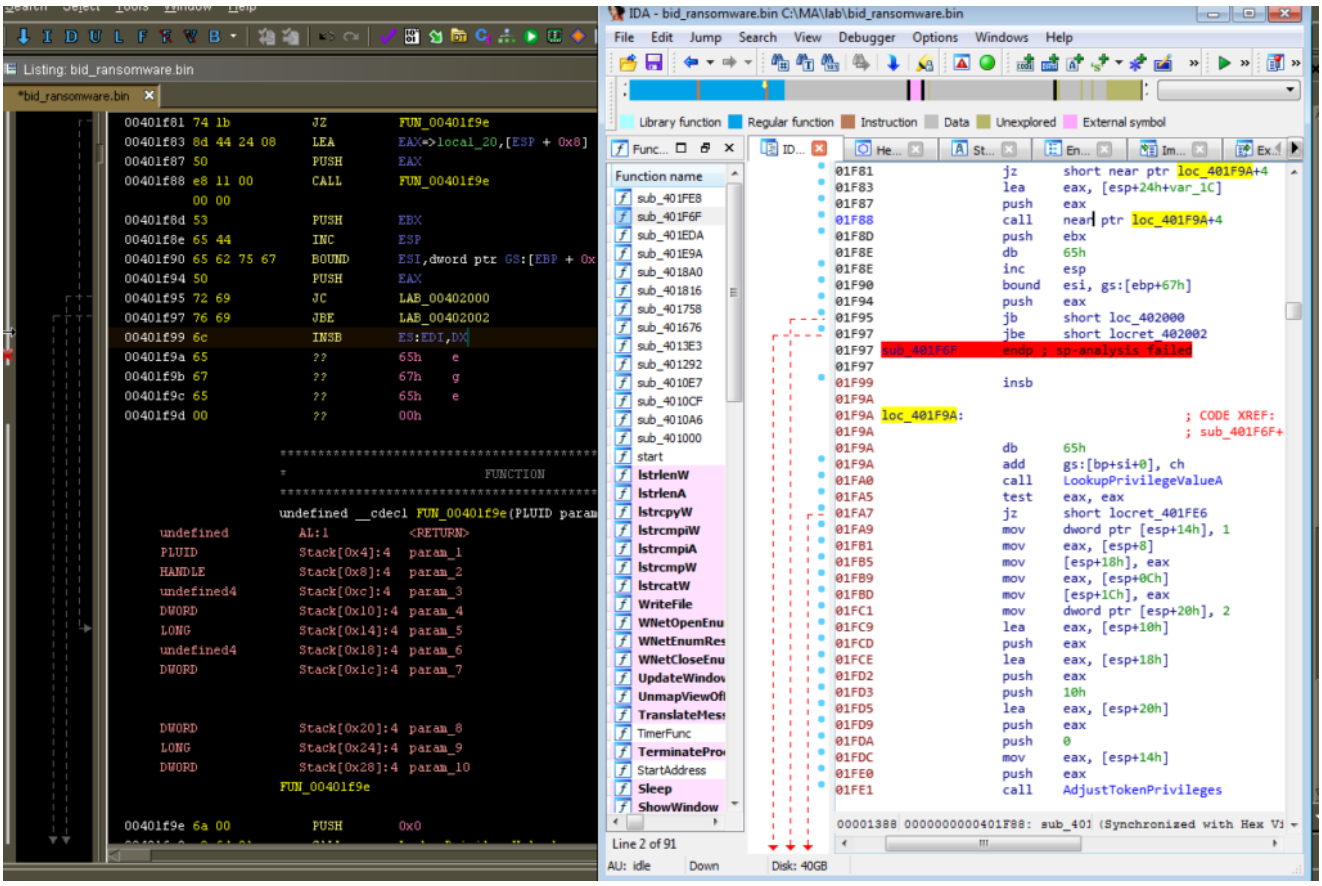
Look at 401CD5…

I'm not sure why this is. One thing I suppose you could do is look for function entry sequences (PUSH EBP; MOV EBP, ESP) and then manually create a function when you find one. In Ghidra, you'd just put the cursor in the spot where you want to create the function, and then hit F:

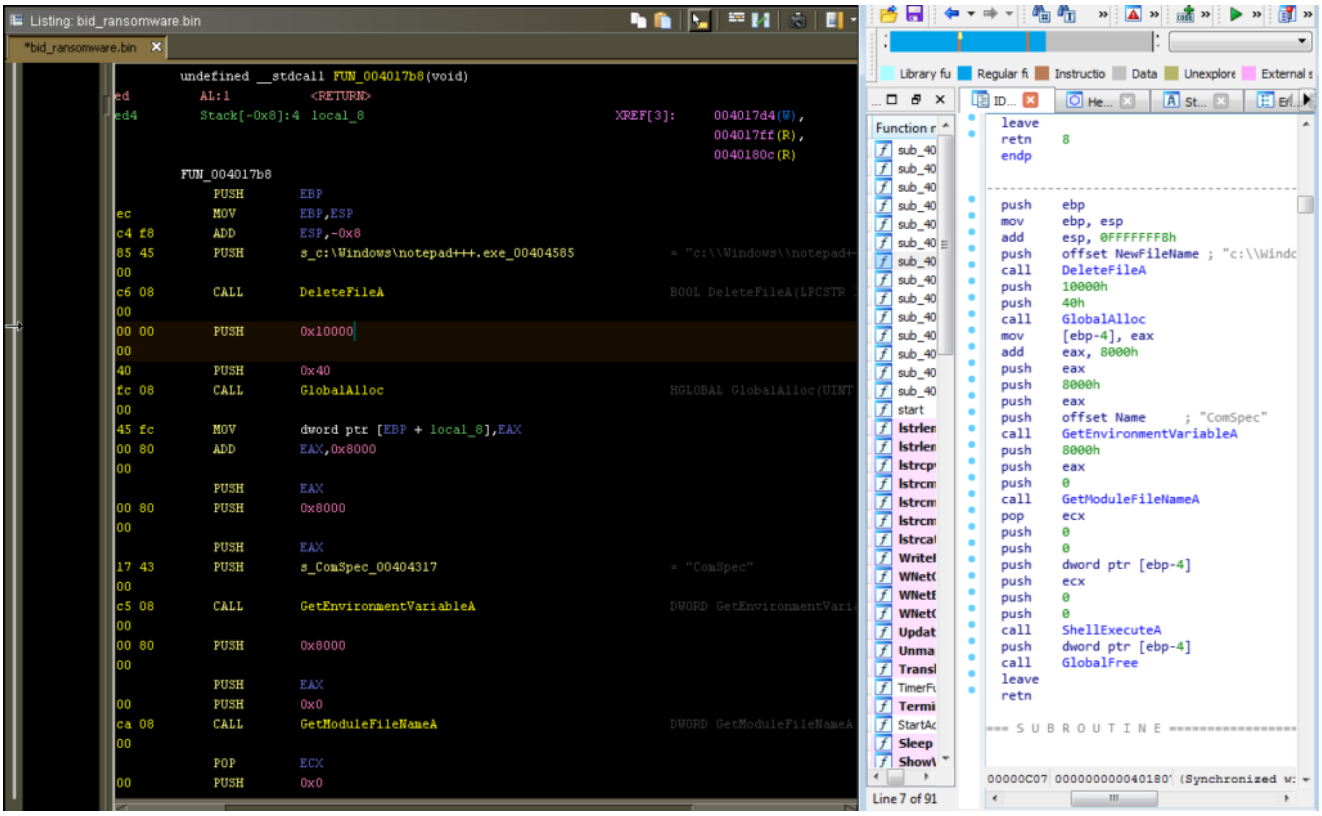


Press F to ~~Pay Respect~~ Create a Function

On the other hand, I remember there was a part of the code that IDA wasn't as successful with. Ghidra did better on this part, at least recognizing that there's a function there while IDA got a bit more confused:

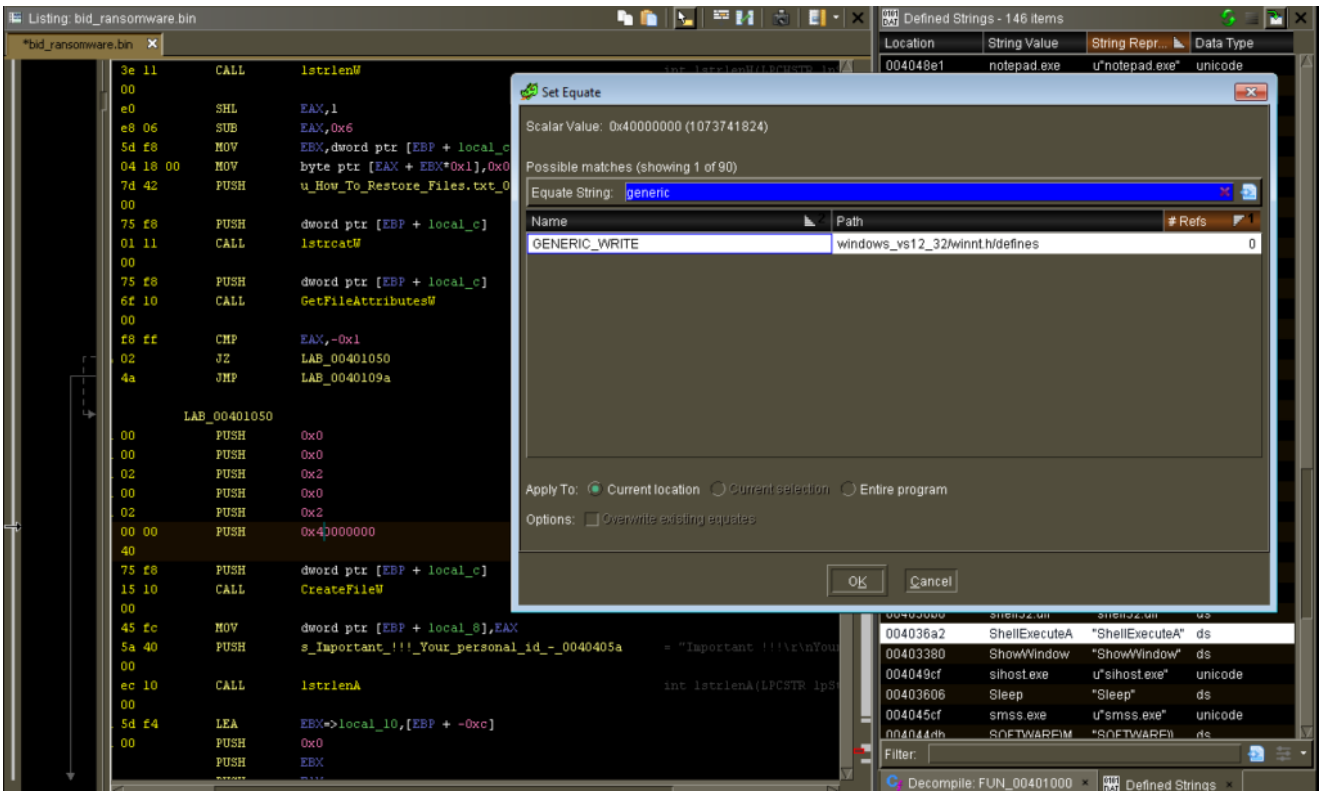


As I was looking at the function at 4017B8, besides noticing that this was another function that IDA didn't recognize, I noticed that Ghidra labels strings in a nice way where the label contains both a reference to the string itself and also the address. IDA will sometimes just give you a very generic name without including the address in the label. You can change the IDA options around strings so that it will not automatically generate a name (and set options like string prefix, etc.) but then you just get something like "asc_401414" which isn't that meaningful either. I'm not sure how IDA generates the names, and the documentation is a bit vague: "If this option is set, IDA will give meaningful names to newly created string literals."



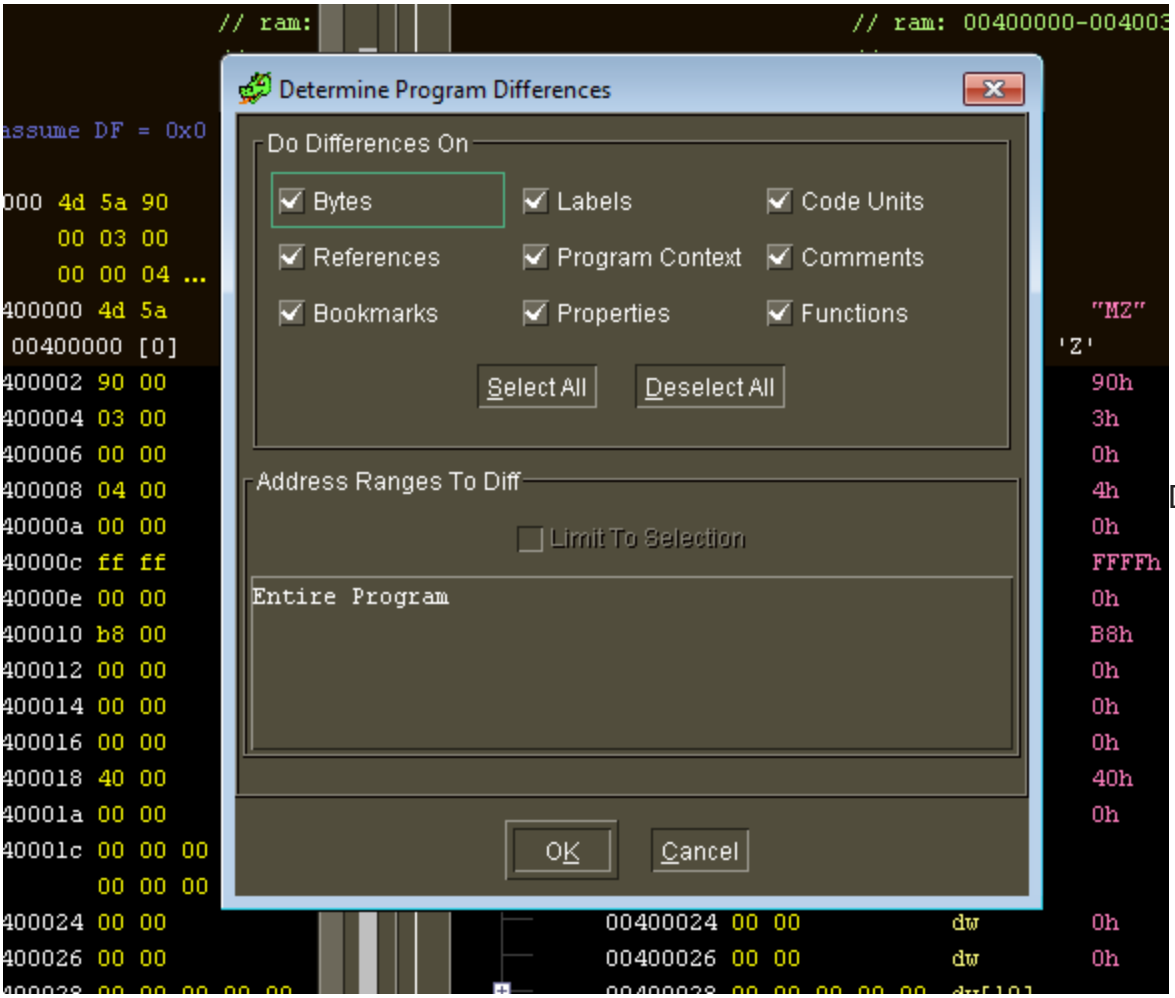
Ghidra and IDA Strings Compared

Changing not very descriptive parameters like 0x40000000 to something like GENERIC_WRITE is easy in both programs. In IDA, it's M to bring up the enumerations, and then you pick one from the list. In Ghidra, it's E to "Set Equate" and then pretty much the same process — look up the value you want to apply there.



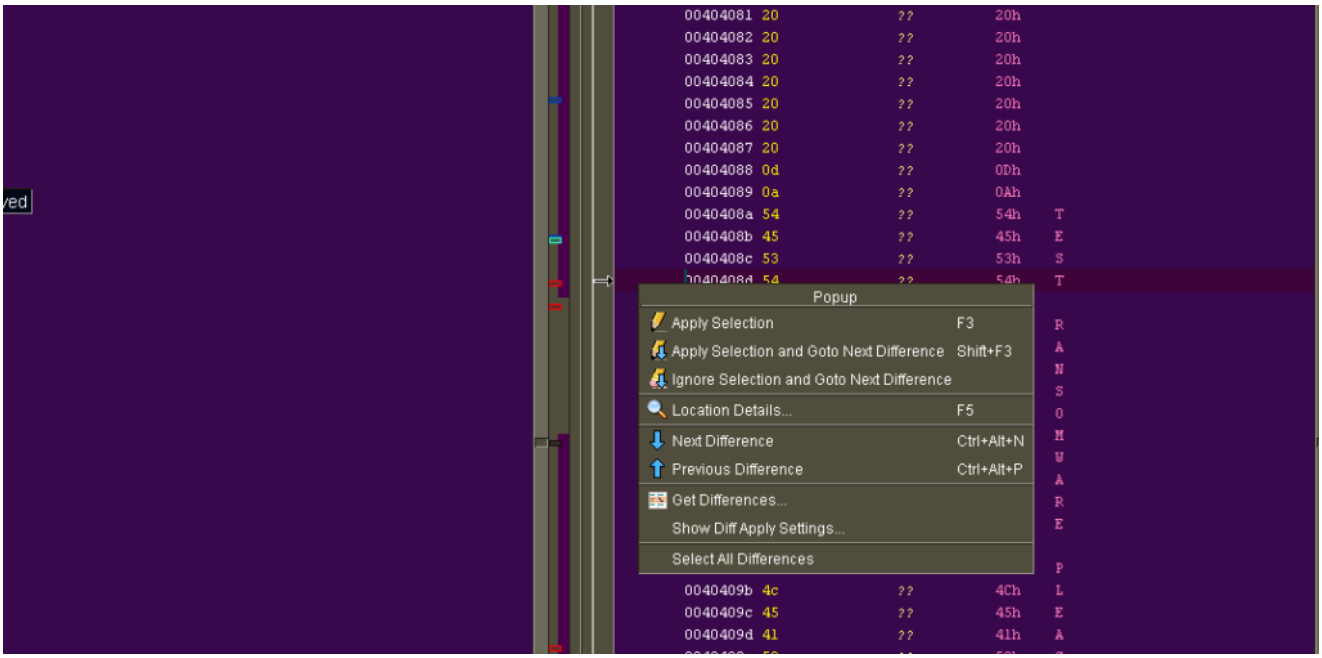
Changing 0x40000000 to GENERIC_WRITE

I decided to make a copy of the sample and changed one of the lines in the ransom note to be “TEST RANSOMWARE PLEASE IGNORE” so I could try out the “Determine Program Differences” window. Seems that you need to import the other file into the current project so you can compare differences between the two programs. There’s a lot of options here that you can use with this tool:



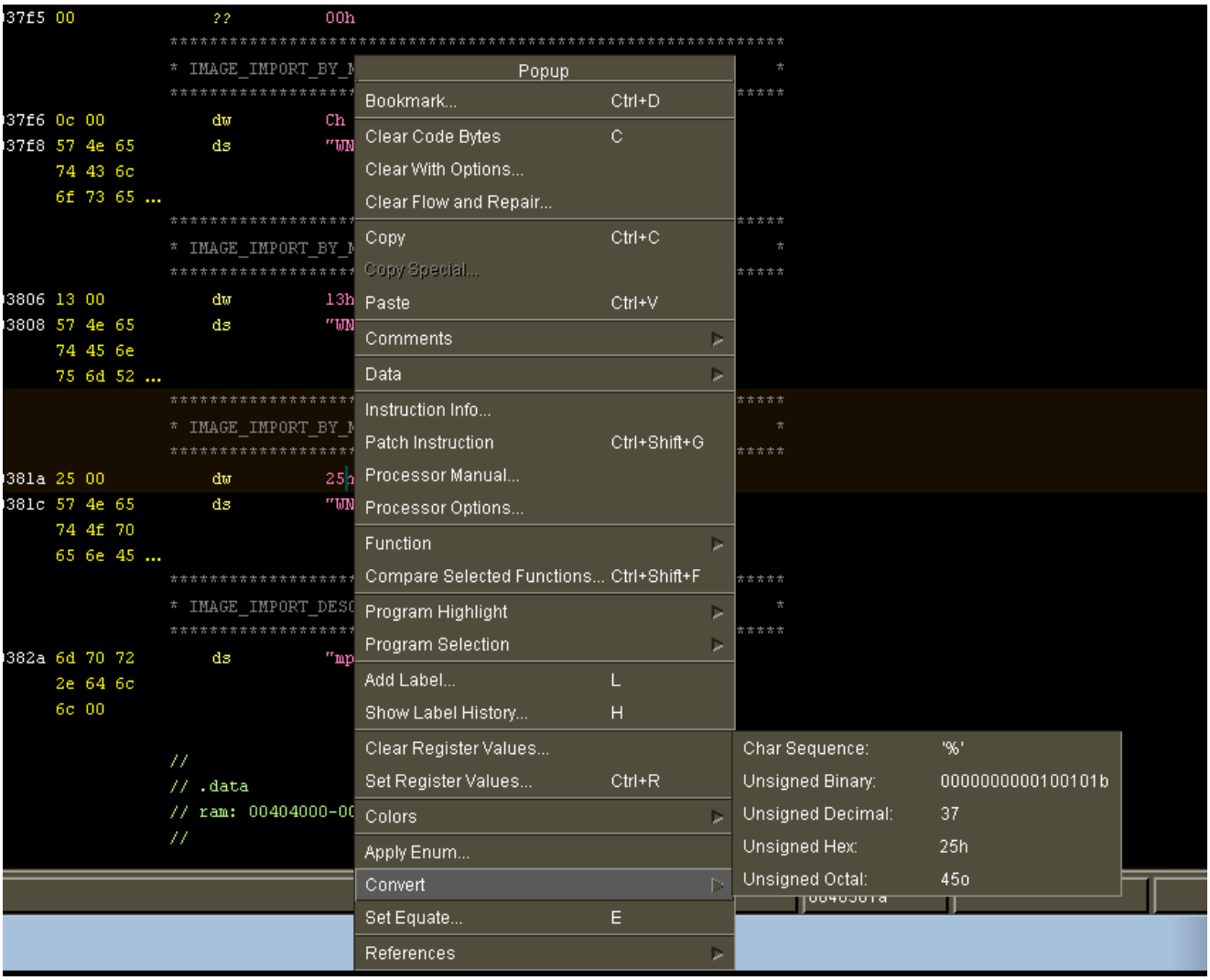
Determine Program Differences Options

Since I just quickly edited it in a text editor it screwed something up because it inserted 0x0D0A in certain places, but even so I can still see how the differences get highlighted, as well as how you can quickly navigate between differences by right-clicking and selecting options from the pop-up menu:



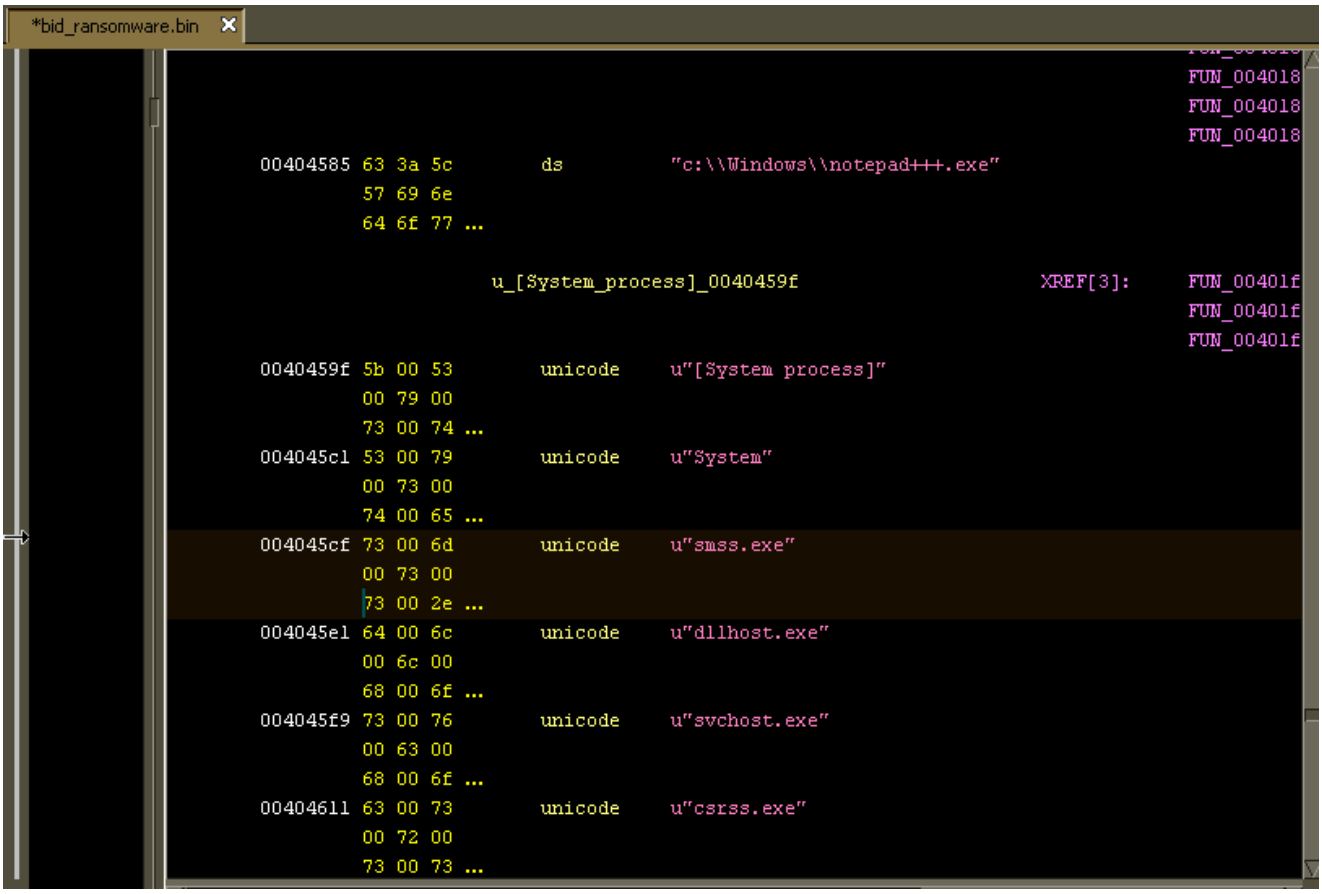
Navigating Around Differences

Like in IDA, you can right click on a value in the program listing and change how it's displayed:



Displaying 0x25 Differently

Also, it's nice to see that Unicode strings are picked up automatically in Ghidra, not just ASCII strings. It's not that big of a deal to tell IDA to treat something as a Unicode string, but having Ghidra automatically do this is one of those little things that I appreciate because it's something that I find tedious (maybe there's a way to make this happen automatically in IDA that I just never learned).



A Mix of ASCII and Unicode Strings in Ghidra's Program Listing

Finally, I like how Ghidra identifies thunk functions:

Insert “Who’da Thunk It?” Joke Here

Going through a sample that I previously analyzed with IDA helped me get more accustomed to Ghidra because I have some idea of how it the final product should look already. The more I use Ghidra the more I like it. I’m still going to keep IDA around — for instance, I tried loading up a really old DOS game executable, and while Ghidra didn’t come up with anything meaningful in the program listing using automated analysis, IDA Free 7.0 at least came up with some results. Time permitting, I’ll try to look at samples in both programs for a while just to see how things differ.