

12 Days of Ghidra

Nathan R

[Twitter](#) | [Mastodon](#)

Day 10 – Function ID

Stripped Binaries

So far all of our binaries have had symbols included

- Function Names

A lot of binaries you see in the wild will be stripped

- No functions names

In Ghidra, all of the functions will just be called FUN followed by random values

This can make RE very time consuming, especially in static binaries

* What are the important functions?

Function ID

Ghidra has a feature to help with this called function ID

- Anyone familiar with IDA/Hex-Rays will know this as FLIRT

A database for function hashes

Ghidra compares the stripped function hash with a known function hash to see if they match

* If they do, this will be marked in the disassembly view

How do I create a .fidb?

Identify what library the function is from

- Versions must match

Compile a static binary with symbols using the target library

Load the known binary into Ghidra

Tools->Function ID->Create empty new fidDB

Tools->Function ID->Populate fidDB from programs

- Make sure the language matches the compiler used on the binary

Using the fidDB

Import a binary that matches the language specified when populating your db

Under the analysis options “Function ID” should appear

If not, make sure the compiler and language are the same

Day 10

More of an exercise than a challenge today on account of the many steps involved

Take today's binary and try and cover information using function id

I will consider this binary done when you can tell me:

1. What library is being used
2. What function is called for printing "hello world"

Suggestions:

- Remember to look at string information
- Google is your friend
- I recommend doing this in a Linux VM
- In order to statically compile a binary you need to use the `--static` flag
 - This looks for `.a` files rather than `.so` files