**Debugging in Linux**

**PEDA** stands for **Python Exploit Development Assistance**, which can make GDB easier to use

and look nicer.

We need to download it first:

**$ git clone https://github.com/longld/peda.git ~/peda**

Then, copy that file to gdbinit inside your home directory:

**$ echo "source ~/peda/peda.py" >> ~/.gdbinit**

Then, start GDB:

**$ gdb**

**$ nasm -felf64 hello.nasm -o hello.o**

**$ ld hello.o -o hello**

Now run ./hello with GDB as follows:

**$ gdb ./hello**

We are going to set the disassembling mode to Intel:

**set disassembly-flavor Intel**

Then, we are going to set a breakpoint where we want to start our debugging step by step

because we are going to track all instructions, so let's put our breakpoint at \_start:

**break \_start**

Now, let's continue our work by typing stepi, or you can just use s, and this will begin to

*The stepi command will step into instructions such as call, which will cause the*

*flow of debugging to be switched inside that call, whereas the s command or step*

*will not do this, and will just get the return values from the call instruction by*

*stepping into the call instruction.*

Also, as you can see, there is 1 inside the RDI register and the next instruction is movabs rsi,0x6000d8. Let's try to see what is inside memory address 0x6000d8 using xprint 0x6000d8

****