After verifying the existence of the vulnerability in section 2.2.1, which determined that distance to EIP at 1056 bytes and that there is 1440 bytes for shellcode. Without this information it is impossible to create a reliable buffer overflow exploit.

In order to gain control of the EIP, the distance to the EIP is filled with characters (in this case 1056 “A”s).

Following the execution of the return in the skin loader, four bytes are popped off the stack leaving the ESP will point to the start of the shellcode as it is located right after the bytes that overwrite the EIP in the skin file/exploit.

However, the exact location of the ESP is unknown so the return address should not be hardcoded into the exploit. To work around this, the EIP is overwritten with a memory address to a “JMP ESP” command that is a fixed address. The `JMP ESP` command tells the assembler to jump to the ESP which is pointing to the shellcode. The address is discovered by running `!mona jm -r esp` in Immunity Debugger (figure 2.3.1a).

* `!mona jm -r esp`

Finally, shellcode to run “calc.exe” was added a *Perl* script (containing the header, pattern of “A”s, EIP/JMP ESP location) that was used to exploit the buffer overflow vulnerability and run “calc.exe”.