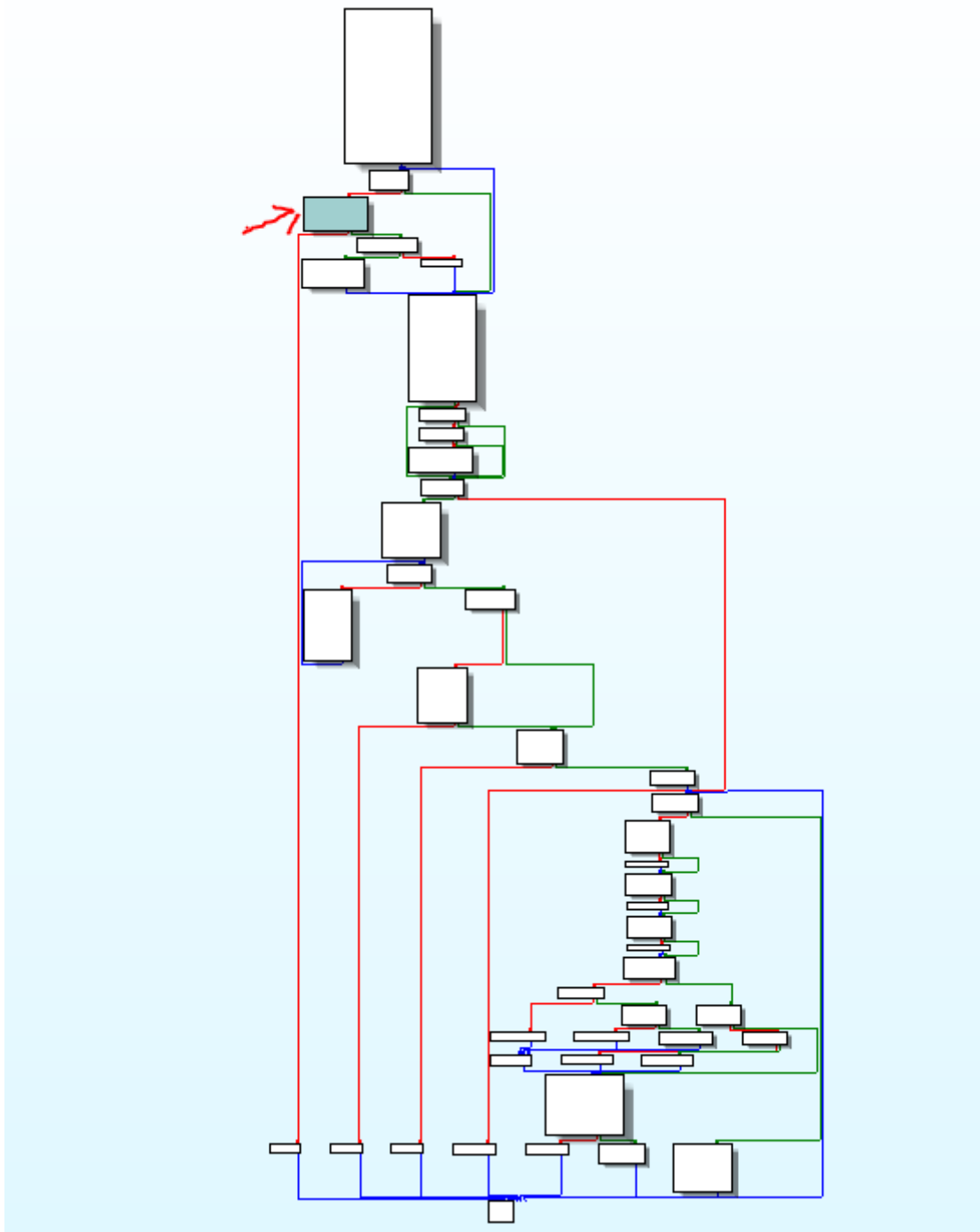
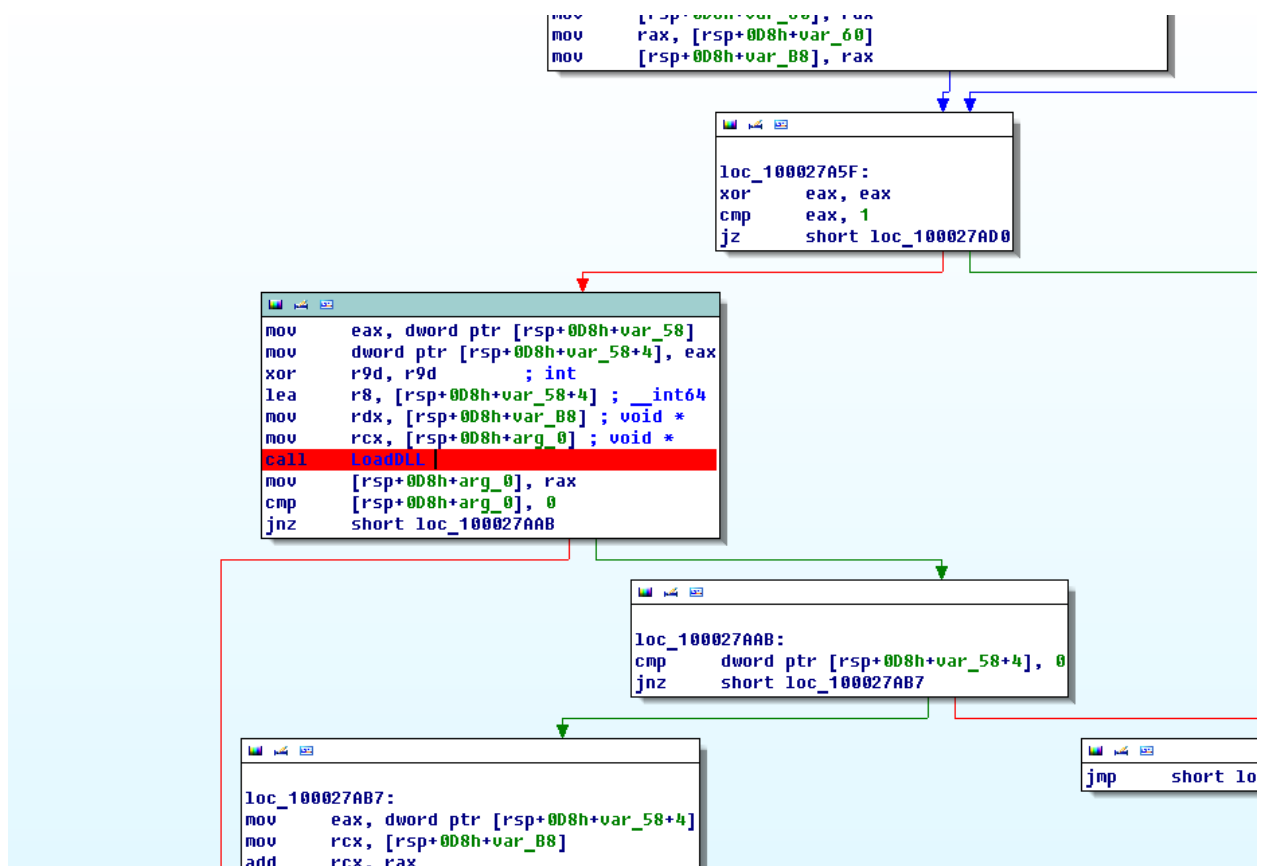


В данном случае интерес представляет функция DllLoader, как до нее добраться написано тут http://web.archive.org/web/20110830080807/http://quequero.org/Armadillo5_x64_Unpacking

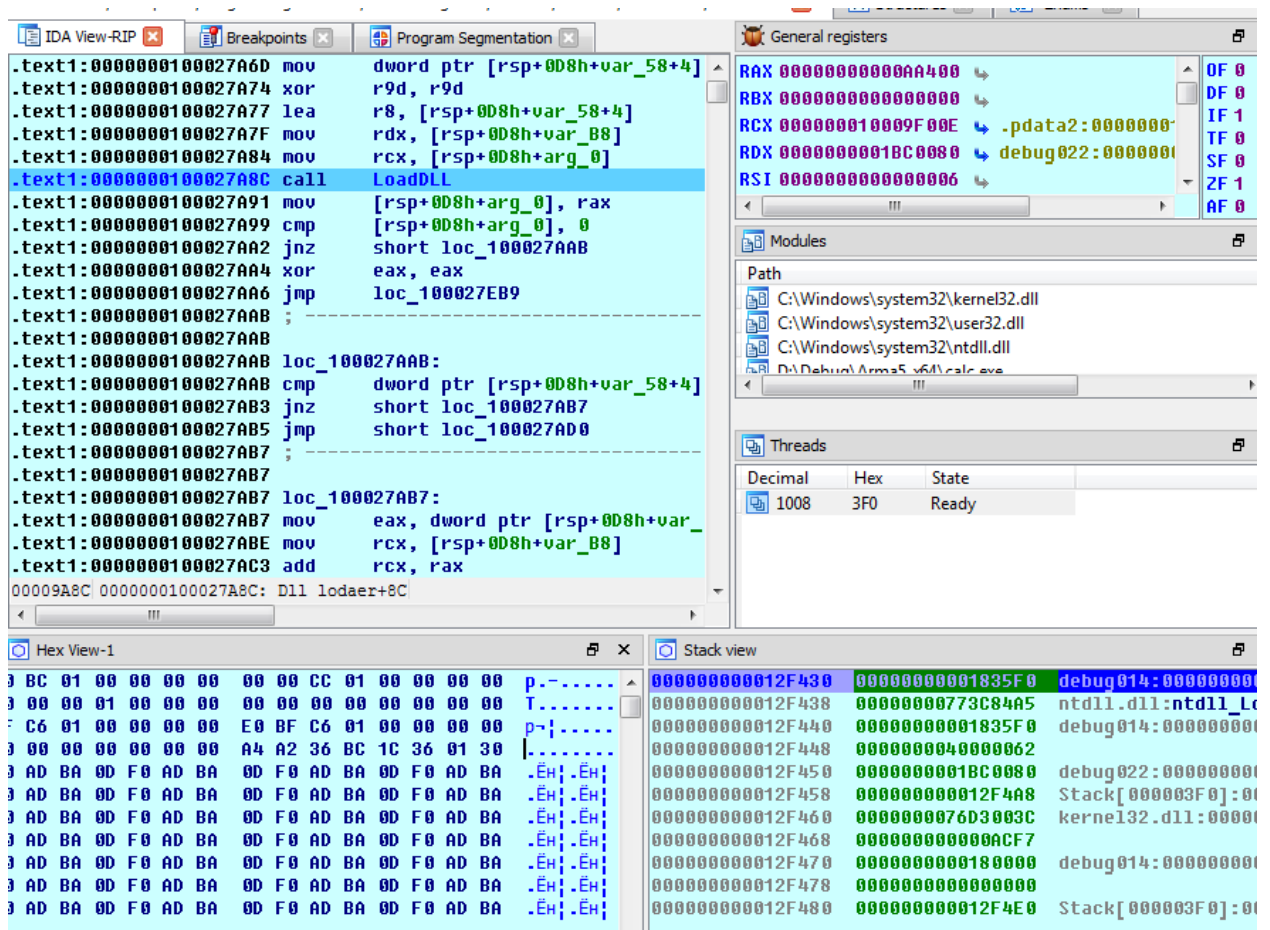
Зайдя в нее видим вот такую картину, нас интересует блок указанный стрелкой



Приблизив видно, что там есть вызов функции, можно назвать ее LoadDll.



Ставим брейкпоинт и запускаем отладчик.



В регистре RDX лежит адрес куда будет распаковываться длл. Естественно сохраняем этот адрес. Жмем F8.

The screenshot shows a debugger interface with three main panels:

- Assembly View:** Displays assembly instructions. The instruction `call LoadDll` at address `00000000100027A8C` is highlighted in red. Other instructions include `mov dword ptr [rsp+0D8h+var_58+4], r9d`, `lea r8, [rsp+0D8h+var_58+4]`, `mov rdx, [rsp+0D8h+var_B8]`, `mov rcx, [rsp+0D8h+arg_0]`, `mov [rsp+0D8h+arg_0], rax`, `cmp [rsp+0D8h+arg_0], 0`, `jnz short loc_100027AAB`, `xor eax, eax`, `jmp loc_100027EB9`, `loc_100027AAB:`, `cmp dword ptr [rsp+0D8h+var_58+4], 0`, `jnz short loc_100027AB7`, `jmp short loc_100027AD0`, `loc_100027AB7:`, `mov eax, dword ptr [rsp+0D8h+var_58+4]`, `mov rcx, [rsp+0D8h+var_B8]`, and `add rcx, rax`.
- Registers View:** Shows the state of registers. RAX is `000000001000E9CD2`, RBX is `0000000000000000`, RCX is `000000000000AA400`, RDX is `0000000000000000`, and RSI is `0000000000000006`.
- Hex View-1:** Displays a memory dump. The address `000000001BC0080` is highlighted in yellow. The dump shows hexadecimal values and their ASCII representation, including the text `is program cannot be run in DOS`.

Теперь в RCX лежит размер длл.

Теперь запускаем Task-Explorerx64 . Выбираем Dump Region, указываем адрес сохраненный ранее и размер из RCX. Сохранить как длл. Все.