

Harnessing Large Language Models for

Enhanced Malware Reverse Engineering

## Introduction

Who am I?

## Jeremy (Dyngnosis) Richards

Started reversing microsoft patches for a company called nCircle in the mid 2000s, came up through BBS and IRC scene before that.

20 years of reversing/vuln research / exploit dev

5 years firmware / embedded research

5 years malware research (mobile / desktop)

Lead Malware Reverse Engineer at Arctic Wolf Labs.

**●** Intersection of AI/ML and Security **●** 



#### dyngnosis @dyngnosis

machine learning, reversing engineering, malware, vulns, exploits, hardware, firmware, League of Legends.

# Agenda

| 01 | Introduce Problem & Goals |
|----|---------------------------|
| 02 | The Project               |
| 03 | Results                   |

04

Future work & Q/A

# **Introduction: The Problem**

The Need for Automation in Reverse Engineering

### Reverse engineering...

- Is labor-intensive and time-consuming task.
- requires a deep understanding of the programming languages, libraries, and frameworks used in the software
- requires the ability to recognize patterns and structures within the code.
- analyzing malicious code is a specialized task (obfuscation, anti-analysis, etc.)

## **Rise of the LLMs**

Large Language Models have showcased their prowess in processing and generating text that mirrors humanlike understanding, but can it understand decompiled code?

# Large Language models

Why Are They Good for Explaining Decompiled Code?

### **Contextual Understanding:**

- Models recognize patterns in decompiled code.
- . Translate machine-level operations to high-level concepts.

## Natural Language Bridging:

- Converts complex code into simple, layman-understandable terms.
- Explains functions, variables, and logic flow.

## **Assists in Reverse Engineering**

- Provides insights into code's purpose and behavior.
- Helps developers understand legacy or unfamiliar codebases



## **Introduction: Goals**

Why are we here?

### **Project Objectives**

The aim of this project is to enhance the reverse engineering experience, and improve the velocity and quality of analyst output by providing tools to automate the boring and illuminate the interesting..

### **Presentation Objectives**

Encourage others to adopt the usage of Lupine and contribute the model

Show my method, mistakes, and successes when fine tuning a model

Provide the background necessary to start your own project focused on fine-tuning a local model to increase performance on a custom tasks

# **Project Objective**

**Project Goals** 

Fine-tune a local model to improve performance on the following three reverse engineering tasks:

- 1. Explain <code> step by step
- 2. Provide a summary for the following <code>
- 3. Supply a descriptive new name that describes the following <code>

Providing automation around these high level reversing tasks allows an analyst to identify interesting functions by name and summary.

# Introduction: Reverse Engineering

Reversing an unknown binary

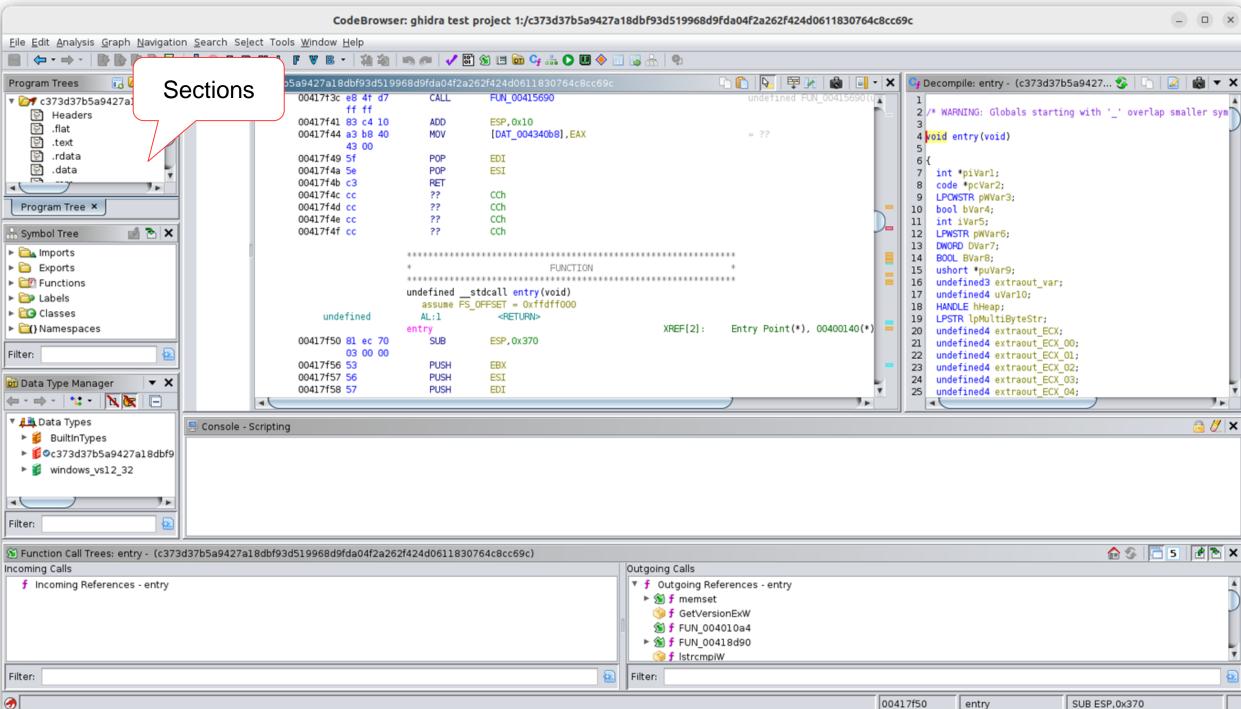
When approaching the task of reverse engineering a binary, the analyst usually gets a little context but assuming there is none we:

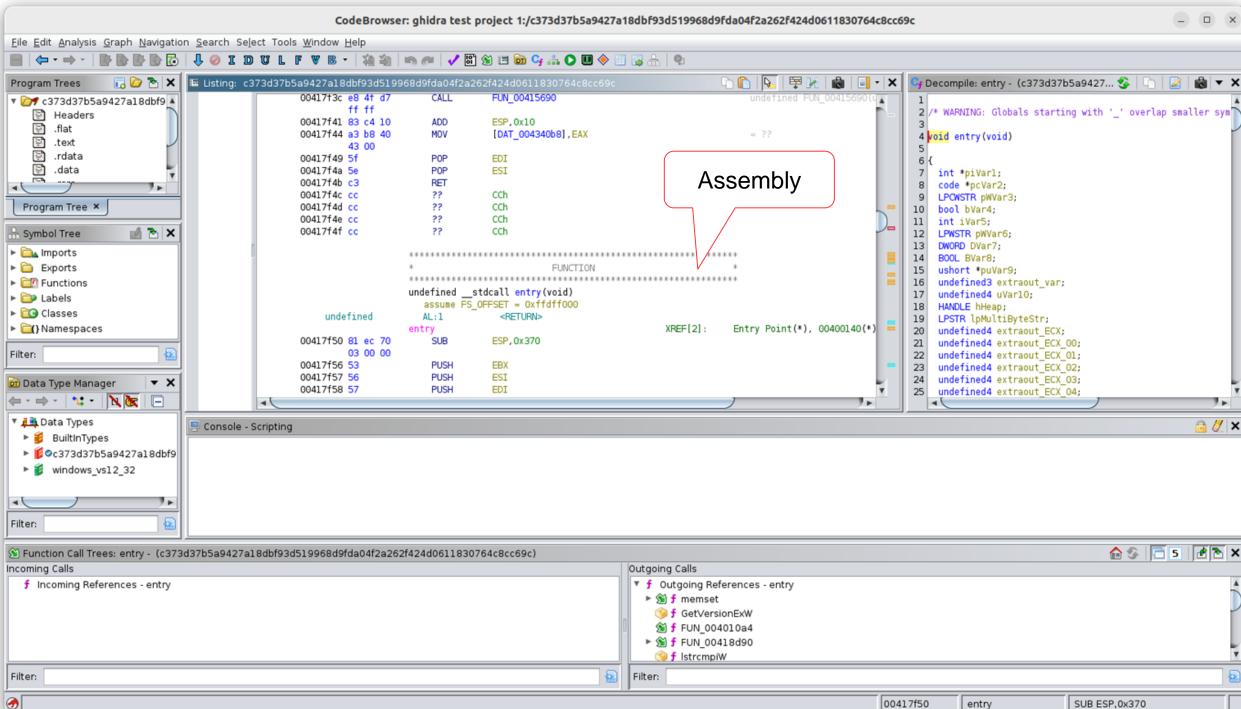
- 1. Identify the file type & architecture (**PE32**, ELF, MACHO/x86, x64, arm)
- 2. PE Specific file parsing (headers, resources, imports, exports)
- 3. xref interesting API calls to find interesting functions
- 4. xref interesting strings to find interesting functions
- 5. reverse / trace execution from entry point to find interesting functions

| THE LEADER IN SECURITY O | PERATIONS |
|--------------------------|-----------|
|--------------------------|-----------|

|        | € 64 bit          |   |                              |                        |            |                             |                      |               |                            |          |
|--------|-------------------|---|------------------------------|------------------------|------------|-----------------------------|----------------------|---------------|----------------------------|----------|
|        | 0                 | 1   | 2                            | 3                      | 4          | 5                           | 6                    | 7             |                            |          |
|        | Signatu           | re 0x5A4D   |                              |                        |            |                             |                      |               |                            |          |
|        |                   |   |                              |                        |            |                             |                      |               |                            |          |
|        |                   |   |                              |                        |            |                             |                      |               |                            |          |
| 0x0000 | Si                | gnature   | 0x5045000                    | 0                      | Mac        | nine                        | #Number0             | fSections     | 1†                         |          |
| 0x0008 |                   | TimeDat   | teStamp                      |                        | Po         | interToSy<br>(depre         | mbolTabl             | e             | COFF<br>Header             |          |
| 0x0010 | # N               | umber0fS<br>(depr   | ymbolTabl                    | e                      | SizeOfOpt: | ionalHeader                 | Characte             | eristics      |                            |          |
| 0×0018 | Ма                | gic   | MajorLinker<br>Version       | MinorLinker<br>Version |            | Size0<br>(sum of al         | fCode<br>1 sections) |               | 1                          | t        |
| 0x0020 | Si                | zeOfIniti   | alizedDat                    | ta                     | Siz        | eOfUninit                   | ializedD             | ata           | Standard<br>COFF<br>Fields |          |
| 0x0028 | A                 |   | ntryPoin <sup>.</sup><br>va) | t                      |            | Base0                       |                      |               |                            |          |
| 0x0030 |                   |   | fData<br>va)                 |                        |            | Image                       | Base                 |               | †                          |          |
| 0x0038 |                   | SectionA  | -                            |                        |            | FileAli                     | -                    |               |                            |          |
| 0x0040 | Major0p<br>System | erating<br>/ersion  | Minor0p<br>System            | erating<br>/ersion     | Ver        | Inage<br>sion               | Ver                  | Image<br>sion |                            |          |
| 0x0048 | MajorSu<br>Ver:   | bsystem<br>sion   | MinorSu<br>Vers              | bsystem<br>sion        |            | Win32Vers<br>(zeros         | filled)              |               | Windows                    |          |
| 0x0050 |                   | SizeOf  | fImage                       |                        |            | SizeOf                      | leaders              |               | Specific                   |          |
| 0x0058 |                   | Chec<br>(images not   | kSum<br>checked)             |                        | Subsy      | ystem                       | DllCharac            | teristics     |                            |          |
| 0x0060 | 5                 | SizeOfSta   | ckReserve                    |                        |            | SizeOfSta                   | ackCommit            | :             |                            |          |
| 0x0068 |                   | SizeOfHea   |                              |                        |            | SizeOfHe                    | apCommit             |               |                            |          |
| 0x0070 |                   |   | filled)                      |                        | #          | NumberOfR                   | vaAndSiz             | es            | l 🖡                        |          |
|        |                   |   | VA)                          |                        |            | SizeOfExp                   | ortTable             |               | 1                          |          |
|        |                   |   | VA)                          |                        |            | SizeOfImp                   | ortTable             | 1             |                            | Optional |
|        |                   |   | VA)                          |                        | s          | ize0fResc                   | ourceTabl            | e             |                            | Header   |
|        |                   |   | VA)                          |                        | Si         | zeOfExce                    | ptionTabl            | Le            |                            |          |
|        |                   |   | VA)                          |                        | Siz        | eOfCerti                    | ficateTab            | ole           |                            |          |
|        | В                 | (R  | ationTable<br>va)            | e                      | Size       | OfBaseRel                   | ocationT             | able          |                            |          |
|        |                   | (R  | oug<br>va)                   |                        |            | Size0f                      | Debug                |               |                            |          |
|        |                   |   | VA)                          |                        | Siz        | eOfArchi                    | tectureDa            | ata           | Data                       |          |
|        |                   |   | VA)                          |                        | 00         | 00                          | 00                   | 00            | Directories                |          |
|        |                   |   | VA)                          |                        |            | SizeOfT                     | LSTable              |               |                            |          |
|        |                   |   | VA)                          |                        | Si         | zeOfLoadC                   | onfigTab             | le            |                            |          |
|        |                   | (R  | Emport<br>va)                |                        |            | SizeOfBou                   | -                    |               |                            |          |
|        |                   | ImportAddressTable<br>(RVA)<br>DelayImportDescriptor<br>(RVA) |                              |                        |            | SizeOfImportAddressTable    |                      |               |                            |          |
|        | De                |   |                              |                        |            | SizeOfDelayImportDescriptor |                      |               |                            |          |
|        |                   |   | meHeader<br>va)              |                        | Siz        | eOfCLRRu                    | ntimeHead            | ler           |                            |          |
|        | 00                | 00  | 00                           | 00                     | 00         | 00                          | 00                   | 00            | •                          | ł        |
|        |                   |   | 10 9 10 1                    | Na                     | me         | Virtual                     | Address              |               | Section                    |          |
|        |                   | Virtua  |                              |                        |            | (R                          | VA)                  |               | Table                      |          |
|        |                   | SizeOfRawData   |                              |                        |            | PointerT                    |                      |               |                            |          |
|        |                   |   | elocation                    |                        | Po         | ointerToL.                  |                      | rs            |                            |          |
|        | NumberOfRe        | locations   | NumberOfLi                   | nenumbers              |            | Characte                    | eristics             |               | ↓                          |          |

Information Classification: General

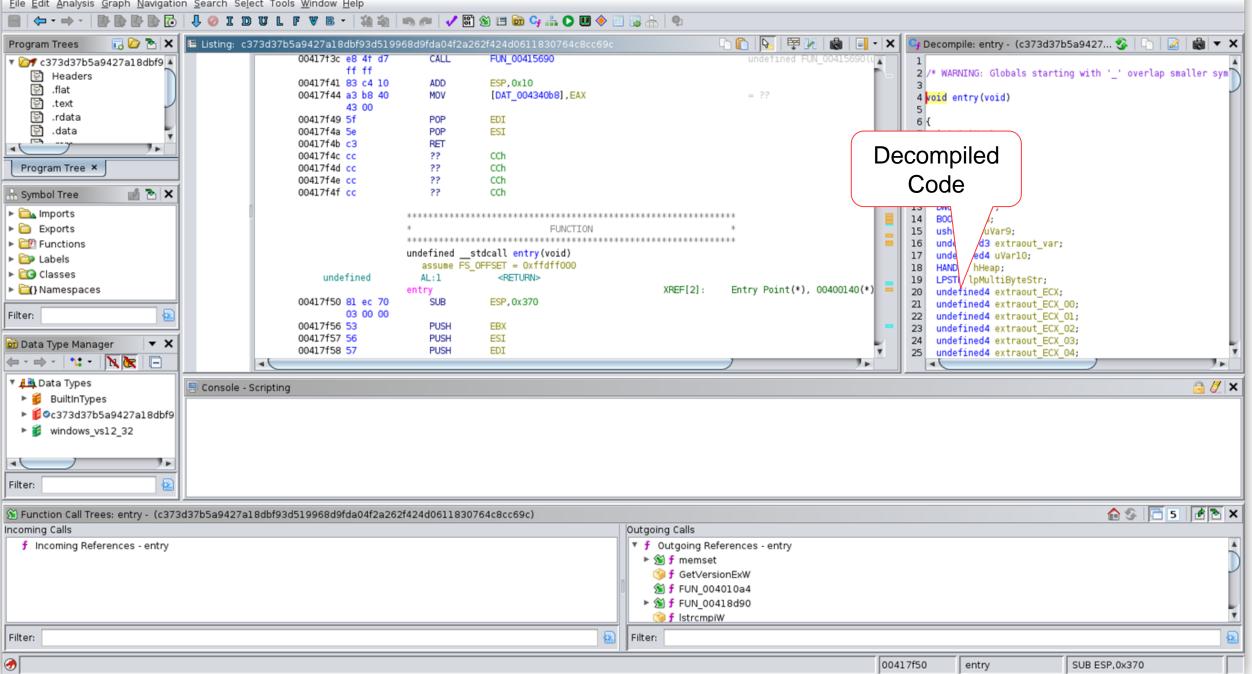




1

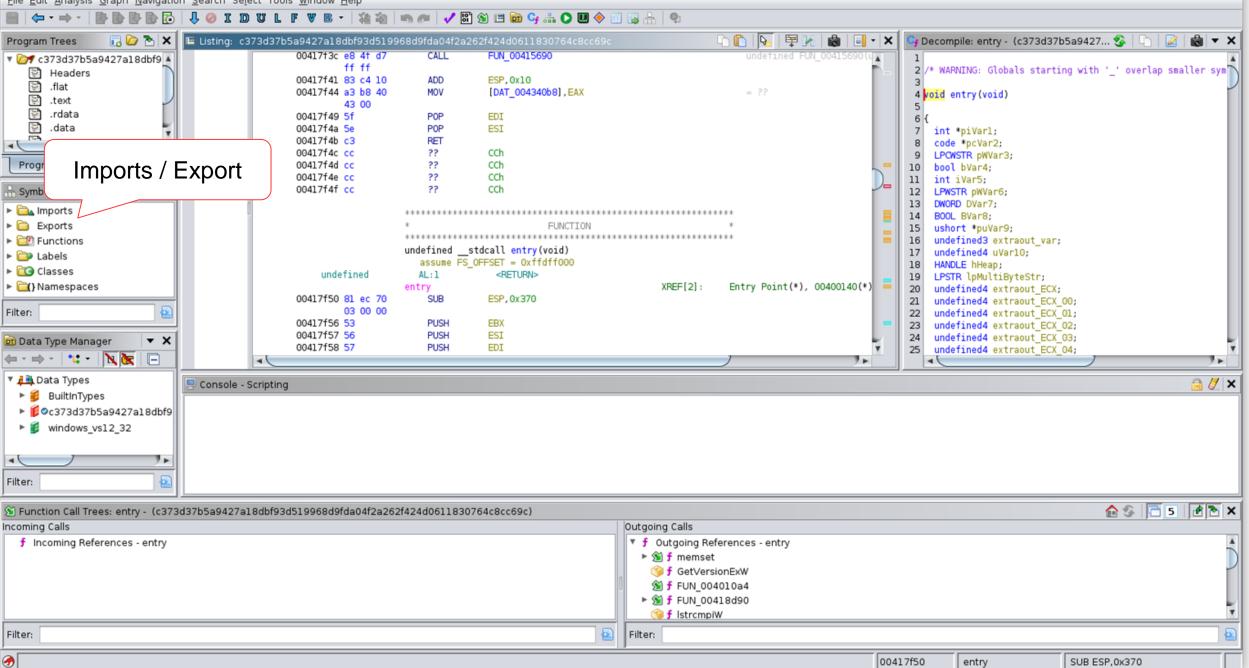
SUB ESP,0x370

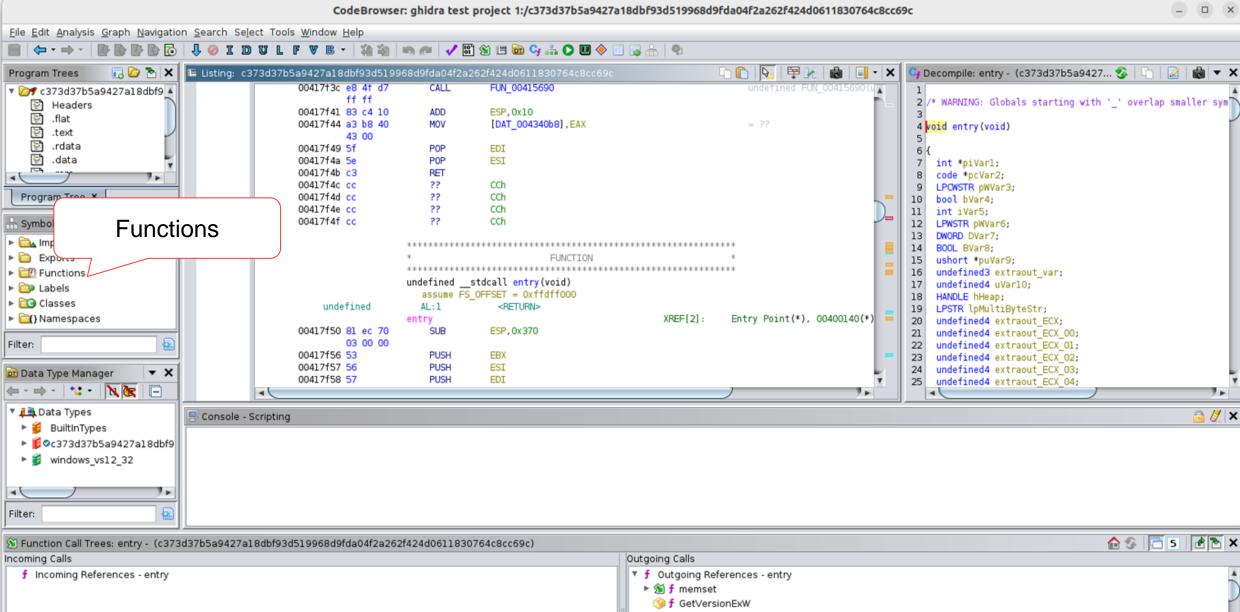
entry



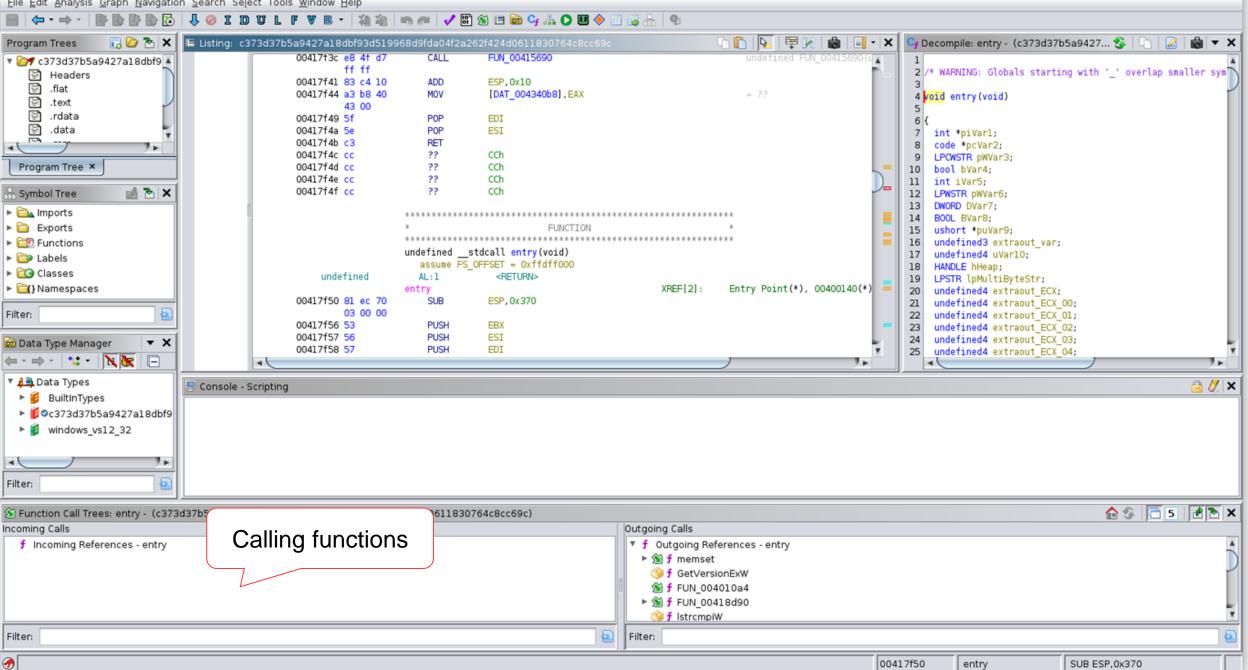


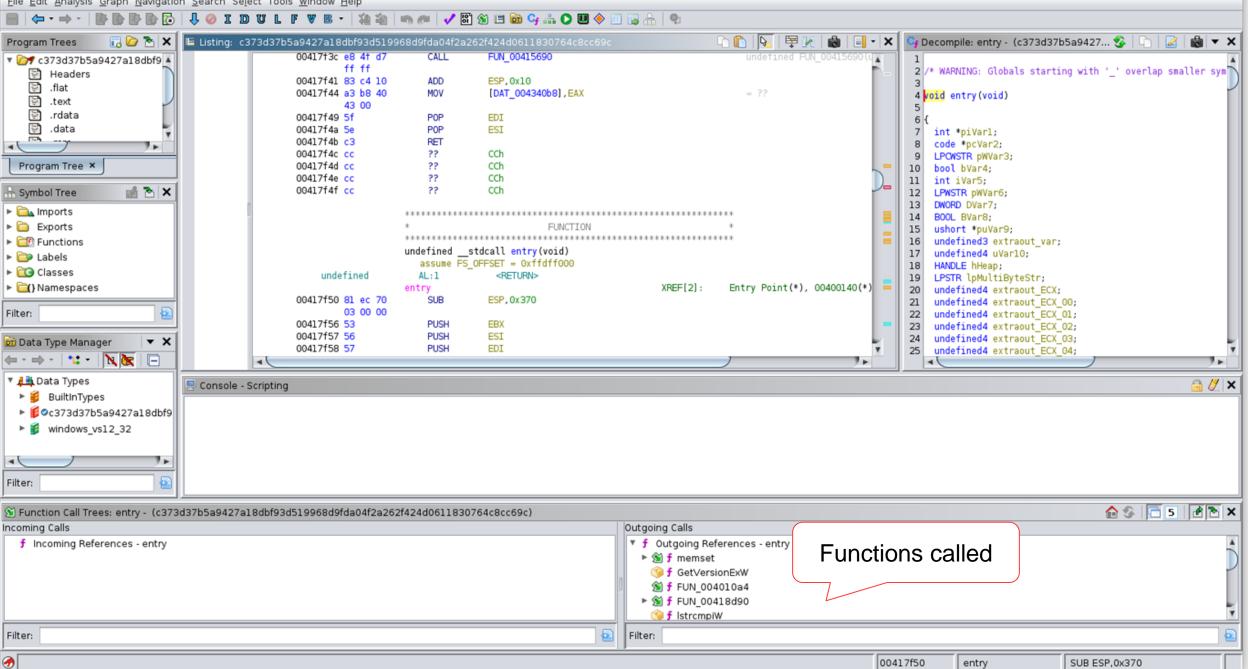
\_ \_ ×





| f Incoming References - entry | 0 | <ul> <li>f Outgoing References - entry</li> <li>▶ M f memset</li> <li>④ f GetVersionExW</li> <li>M f FUN_004010a4</li> <li>► M f FUN_00410a4</li> </ul> |          |       |                | Ď |
|-------------------------------|---|---|----------|-------|----------------|---|
| Filter:                       |   | ▶ 🖄 ₱ FUN_00418d90<br>🎯 ₱ IstrcmpiW<br>ilter:   |          |       |                |   |
|                               |   |   | 00417f50 | entry | SUB ESP. 0x370 | - |





### 

|   | J L F W B | 9 ~   4        | i i i i i i i i i i i i i i i i i i i  |   |                                   |  |
|---|-----------|----------------|--|---|-----------------------------------|--|
| Program Trees 🗊 🖄 🗙   | _         |                | Decompile: entry - (4c8f8899d02737d9c1c00f8848f73298a                        | 2749ff7a1a75a0ca2acd68117d2b515)  |                                   |  |
| Kosf8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd     Kosf8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd     Kosfata     Kosfata     Kosfata |           | 2              | /* WARNING: Function:SEH_prolog4_GS<br>/* WARNING: Globals starting with '_' |   | address */                        |  |
| ···)당 .rsrc<br>···)당 .rsic<br>···)당 .reloc  |           | 5              | 4  | References to RegOpenKeyExA - 7 locations [<br>Edit Help                | Xref RegOpenKey fror<br>functions |  |
| Debug Data  |           | 8<br>9<br>10   | void *pvVar2;  | References to RegOpenKeyExA - 7 locations                               | Label                             | Code Unit Context  |
| Program Tree X  | 1         | 11 12          | <pre>code *pcVar4;<br/>void *pvVar5;</pre>                                   | 00402009 FUN_00401fe5<br>004022a5 FUN_00402269                          |                                   | CALL dword p COMPUTED_CALL<br>CALL dword p COMPUTED_CALL                               |
| Symbol Tree 📝 🏝 🗙   | -         | 13<br>14<br>15 | BOOL BVar7;<br>uint uVar8;   | 004025bb FUN_0040257e<br>00402608 FUN_0040257e<br>00402841 FUN_0040278b | $\bigvee$                         | CALL dword p COMPUTED_CALL<br>CALL dword p COMPUTED_CALL<br>CALL dword p COMPUTED_CALL |
| CABINET.DLL<br>COMCTL32.DLL<br>GDI32.DLL  |           | 16<br>17<br>18 | undefined *extraout_ECX;   | 004069cb FUN_0040591e<br>0040a008                                       | PTR_Reg.                          | CALL dword p COMPUTED_CALL<br>addr ADVAPI3 DATA  |
| KERNEL32.DLL<br>MSVCRT.DLL<br>USER32.DLL  |           | 19<br>20       | undefined4 extraout_EDX;<br>undefined4 extraout_EDX_00;                      | Filter:   |                                   | <u>图 节 ·</u>   |
| ia-<br>→<br>→<br>→<br>→<br>→<br>→<br>→<br>→<br>→<br>→   |           | 21<br>22<br>23 | undefined8 uVarll;<br>undefined4 uVarl2;                                     |   |                                   |  |
|   |           | 24<br>25<br>26 | undefined *puVar14;  |   |                                   |  |
| È- ͡͡͡͡ FUN_0040<br>È- ͡͡͡ FUN_00401<br>È- ͡͡ FUN_00401645  |           | 27<br>28<br>29 | uint local_28;   |   |                                   |  |
|   | →         | 30<br>31       | undefined4 uStack_c;<br>undefined4 local_8;                                  |   |                                   |  |
| B·· f FUN_00401795<br>B·· f FUN_004017c0<br>B·· f FUN_0040187a  |           | 32<br>33<br>34 | <pre>FUN_004072d8();<br/>uStack_c = 0x406beb;</pre>                          |   |                                   |  |
| ⊕··· f FUN_004019c0     ⊕··· f FUN_00401a69     ⊕··· f FUN_00401a62     ⊕··· f FUN_00401a62     ⊕··· f FUN_00401a9a                                 |           | 35<br>36<br>37 | <pre>local_8 = 0;<br/>GetStartupInfoA(&amp;local_70);</pre>                  |   |                                   |  |
|   |           | 38<br>39<br>40 | local_8 = 1;   |   |                                   |  |
| PUN_00402 ii fun_00402033   |           | 41<br>42<br>43 | LOCK ();   |   |                                   |  |
|   |           | 44             | if (DAT_004088ac != (void *)0x0) (   |   |                                   |  |
|   |           |                |  |   |                                   |  |

| Decompile: entry [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519968d9fda   | Decompile: entry [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519968d9fda04f2a262f4   | Decompile: entry [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c] 📒 😐 🗴 |
|--|--|---|
| Eile Edit Navigation Search Select Help  | Eile Edit Navigation Search Select Help  | Elle Edit Navigation Search Select Help   |
| ■  | ■ (⇔・⇒・   ■ ● ● ● □   ↓ ⊘ IDULFV   • 0 @   |   |
| C; Decompile: entry - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c)   | Cy Decompile: entry - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c)   | 😋 Decompile: entry- (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c) 😵 🕞 📓 💌                                |
| 1<br>2 /# WARNING, Clabele starting with ' ' evenlap smaller symbols at the same address */  |  | 191DAT_00434224 = (code *)FUN_00415690(0x2a5ebfb7, &DAT_0042f418, 0xc);<br>192 (* DAT_00434224)(puVar17);                     |
| <pre>2 /* WARNING: Globals starting with '_' overlap smaller symbols at the same address */<br/>3</pre>  |  | <pre>192 (*_DAI_00434224)(puval/);<br/>193 BVAR8 = ShellExecuteExW((SHELLEXECUTEINFOW *)auStack_36c);</pre>                   |
| 4 void entry(void)   | 90 uVar16 = FUN 004010a4(extraout ECX 03,(int)((ulonglong)uVar16 >> 0x20),0x42f110,0xd,  | 194 if (BVar8 != 0) { 195 /* WARNING: Subroutine does not return */   |
| 6 {  | $\frac{1}{2} = \frac{1}{2} \left( \frac{1}{2} - \frac{1}{2} \right) \left( \frac{1}{2}$ | 196 ExitProcess(0x65);  |
| 7 int *piVarl;<br>8 code *pcVar2;  | 93 for $(uVar]4 = DAT 00436920; 9 < uVar]4; uVar]4 = uVar]4 % 10) {$   | 197 }<br>198 }  |
| 9 LPCWSTR pWVar3;  | 94 uVarl2 = uVarl4 / 10;<br>95 }   | 199 }   |
| 10 bool bVar4;<br>11 int iVar5;  | 96 DAT_00436920 = UVAT14;  | 200 DAT_00436928 = (LPCNSTR)FUN_00415610(0x80);<br>201 DAT_0042fef4 = (code *)FUN_00415690(0xaff20ccb, &DAT_0042f0fc,9);      |
|  | 98 EIN 00418d90(extraout ECX 05 uVar14)  | 202 (*DAT_0042fef4) (auStack_104, &DAT_00425170);   |
| 13 DWORD DVar7;<br>14 BOOL BVar8;  | 99 DAT 00436580 - DAT 00425510   | 203 puVar9 = FUN_0041c220();<br>204 uVar12 = 0;   |
| 15 ushort *puVar9;   |  | 205 do {<br>206 wsprintfW(aWStack 334,L"%x",(uint)puVar9[uVar12]);  |
| 17 undefined4 uVarl0;  | 102 DAT_00436292 = 0;  | 207 uVarl2 = uVarl2 + 1;  |
| 18 HANDLE hHeap;<br>19 LPSTR lpMultiByteStr;   | <pre>104 uVar16 = FUN_004010a4(extraout_ECX_06,extraout_EDX_00,0x42f124,0xd, (undefined *)&amp;DAT_0042f47c,3);</pre>  | 208 } / while (uVarl2 < 5);<br>209 if (5 < DAT 004340e) {   |
| 20 undefined4 extraout_ECX;<br>21 undefined4 extraout_ECX;   | $\frac{105}{106} = \frac{100}{100} = $   | 210 FUN_0041e130();   |
| 21 undefined4 extraout_ECX_00;<br>22 undefined4 extraout_ECX_01;   | <pre>107 uVar16 = FUN_004010a4(extraout_ECX_08, (int)((ulonglong)uVar16 &gt;&gt; 0x20),0x42f42c,0xc,</pre>   | 211 }<br>212 pWWar3 = DAT 00436928;   |
| <pre>22 undefined4 extraout_ECX_01;<br/>23 undefined4 extraout_ECX_02;</pre>   | [100 EIN 004010a4(extraout ECX 09 (int)((u)anglong)(xar)6 >> 0x20) 0x42f024 0xb  | 213 _DAT_004342a8 = (code *)FUN_00415690(0x76a4fa0b,&DAT_0042f0fc,0xb);   |
|  | 110 (undefined *)&DAT 0042f47c 3)  | 214 1 Var5 = (* DAT_004342a8)(0x1f0001,0,pWar3);<br>215 pWar3 = DAT 0043928;<br>215 pWar3 = DAT 0043928;                      |
| 26 undefined4 extraout_ECX_05;   | 112 _DAT_00436f80 = 0;   | 216 if (iVar5 != 0) {   |
| 28 undefined4 extraout_ECX_0/;   | 113 DAT_0047518 = 0  | 217 /* WARNING: Subroutine does not return */<br>218 ExitProcess(0);  |
| 29 undefined4 extraout_ECX_08;   | 115 DAT_0042f514 = 0;  | 219 }<br>220DAT_00434240 = (code *)FUN_00415690(0xed3e7aaa,&DAT_0042f0fc,0xd);  |
| 30 undefined4 extraout_ECX_09;<br>31 undefined4 extraout_ECX_10;   | The DAT_DOMODOW (astronut ECX 1) extronut EDX 01 0x425440 0xc (undefined *)SDAT 0042547c 3);   | 221 (*_DAT_00434240)(0,0,pWVar3);   |
| 32 Underlined4 extraout Ecx 11;  | 118 FUN_00418d90(extraout_ECX_12,0xf);   | 222 FUN_00402030();<br>223 FUN_00415690(0xa38e7de7,6DAT_0042f0fc,0x1b);   |
| <pre>34 undefined4 extraout_ECX_13;</pre>  | ris reveloper(exclade(   | 224 FUN_00415690(0x97622898,6DAT_0042f0fc,10);  |
| 35 int iVaril;   | 121 DAT_0042fee0 = FUN_00415690(0x2ba92ec1,&DAT_0042f14c,0xc);   | 225 bVar4 = FUN_004192e0();<br>226 iVar5 = CONCAT31(extraout var,bVar4);  |
| 3/ Undefined4 extraout_ECX_IS;   | 122 uVar12 = 0;<br>123 uVar14 = uVar12:  | 227 if (iVar5 == 0) {   |
| 38 Undefined4 extraout_ECX_16;   | 124 do J   | 228 return:<br>229 }  |
| 40 undefined4 extraout_ECX_18;   | 126 if (uvar12 < 5, uvar12 + uvar12 + 1) { 126 if (uvar14 + uvar12) {  | 230 _DAT_0042f880 = (code *)FUN_00415690(0x48b64e68,&DAT_0042f0fc,0x12);  |
| 41     undefined4     extraout_ECX_19;       42     undefined4     extraout_EDX;   | 127 austack soc = (underined (4)/oko;  | 231 uVarl0 = (*_DAT_0042f880)();<br>232 _DAT_004342b0 = (code *)FUN_00415690(0xf0df0lcc,&DAT_0042f0fc,0x11);                  |
| <pre>43 uint uVarl2;<br/>44 undefined4 extraout_EDX_00;</pre>  | 129 }  | 233 (* DAT_004342b0)(uVar10.0x80);<br>234 PTR DAT_0042f038 = (undefined *)FUN_00415650(0x100);                                |
|  | 130 UV412 = UV414 + 1;<br>131 UV412 = UV414 + 1;   | 235 vsprintfW((LPWSTR)PTR_DAT_0042f038,L".%s.%s%s",DAT_00436928,L"anobtanium@tutanota.com",L".pandora"                        |
|  | <pre>132 } while (uVar12 &lt; 5);</pre>  | 236 );<br>237 iVarl1 = FUN_00419b09();  |
|  | 133 piveri = {(Int +)(*(Int +)((Int))rocessenvironmentBlock + 0xc) + 0xi0);<br>134 piveris = niveri  | 238 if (iVarl1 != 0) {  |
| 49 int *plVarl5;<br>50 undefined8 uVarl6;  | 135 do {   | 229 FUL_004198a0(6DAT_0042f160,0x2b5);<br>240 }   |
| SI SIZE I dWBytes;   | <pre>137 if ((piVar15[6] != 0) &amp;&amp; (iVar5 = lstrcmpiW((LPCWSTR)piVar15[0xc],L"kernel32.dll"), iVar5 == 0))</pre>  | 241 iVarll = lstrlenW((LPOWSTR)PTR_DAT_0042f038);<br>242 DAT 00436930 = iVarll * 2 + 2;                                       |
| 52 Underined *puvari/;<br>53 undefined *puStack_370;   |  | 2442 UAT_004305330 = IV#11 * 2 + 2;<br>23 DAT_0043200 = FUN 0041569(0)ka74699e,&DAT_0042f0fc.0xc);                            |
| 54 undefined auStack_36c [4];<br>55 undefined auStack_368 [16];  | 140 _DAT_00436f0c = (code *)FUN_00415700(piVar15[6],0x1f3ffddb,0xd,0x7d10e76b);  | 244 DAT_0043692c = (LPCSTR)FUN_00415650(0x1fa0);<br>245 FUN_00415610(0x1000);   |
| 56 undefined auStack_358 [16];   | 142 DAT 00434254 = (code *)EUN 00415700(i)Var5 0x4dd730h7 0xf 0x36ca1f6d)  | 246 dwBytes = 0x100;  |
| 57 void *pvStack_348;<br>58 WCHAR aWStack_334 [8];   | 143 DAT_0042f550 = (*_DAT_00434254)();   | 247 DVar7 = 8;<br>248 hHeap = GetProcessHeap();   |
| 59 undefined auStack 324 [20]:   |  | 249 lpMultiByteStr = (LPSTR)HeapAlloc(hHeap.DVar7,dvBytes);   |
| 60 WCHAR aWStack_310 [2];<br>61 undefined auStack_30c [2];   | 146  | 250 if (IVar5 == 1) {<br>251 FUN_0041bc60();  |
| 62 undefined auStack_30a [518];<br>63 undefined auStack_104 [248];   | 147 1var5 = 1var5 + 1;<br>148 } while (iVar5 != 0);  | Z52         FUM_0041be00(extraout_ECX_15,6DAT_00436720);           253         pcVarl3 = "<br>vbr>";                          |
| C4 undefined 4 under http://www.carefulline.com  | <pre>149 piVarl = *(int **)(*(int *)((int)ProcessEnvironmentBlock + 0xc) + 0x10);<br/>150 piVarl5 = piVarl;</pre>  | <pre>233 ptvaris = <up></up></pre> /254 uVario = extraout_ECX_16;<br>255 }  |
|  |  | 255 }<br>256 else {   |
| 67 DAT 004340e0 = 0xllc;   | <pre>prvoids = (and )/prvoids<br/>153 if (piver15(6) != 0) &amp;&amp; (iVar5 = lstrcmpiW((LPOWSTR)piVar15[0xc],L"kernel32.dll"), iVar5 == 0))</pre>  | 257 uVarl0 = extraout_ECX_14;   |
| <pre>68 GetVersionExW((LPOSVERSIONINFOW)&amp;DAT_004340e0);<br/>69 uVar16 = FUN_004010a4(extraout_ECX,extraout_EDX,0x42f0fc,0xd,(undefined *)&amp;DAT_0042f47c,3);</pre> | 154 break  | 258 if (iVar5 != 3) goto LAB_004186f9;<br>259 DVar7 = 0;  |
| 70 uVar16 = FUN_004010a4(extraout_ECX_00,(int)((ulonglong)uVar16 >> 0x20),0x42f14c,9,  | <pre>156 _DAT_00436f0c = (code *)FUN_00415700(piVar15[6],0x1f3ffddb,0xd,0x7d10e76b);</pre>   | 260 WideCharToMultiByte(0,0,DAT_00436928,-1,lpMultiByteStr,0x80,(LPCSTR)0x0,(LPB00L)0x0);                                     |
| 71 (undefined *)&DAT_0042f47c,3);<br>72 _DAT_00436f80 = DAT_0042f510;  | 157 iVar5 = (* DAT 00436f0c)(&DAT 0042f14c)  | 261 FUN_0041bc60();<br>262 FUN_0041be00(extraout_ECX_17,lpMultiByteStr);  |
| 72 DAT 00425510 - DAT 00425514   | <pre>159 pW/sr6 = GetCommandLinew();</pre>   | 263 pcVarl3 = "<br>br>";<br>264 uVarl0 = extraout_ECX_18;   |
| 75 DAT $0042f514 = 0$ :  | Lot  | 265 }   |
|  | 162 if ((((5 < _DAT_004340e4) && ((int)puStack_370 < 3)) && (iVar5 = FUN_00417540(), iVar5 != 0)) &&   | 266 FUN_0041be00(uVar10,pcVar13);<br>267 uVar10 = extraout_ECX_19;  |
| 78 (undefined *)&DAT_0042f47c,3);  | 164 DAT 00434214 = (code *)FUN 00415690 (0x2a13bb3, 6DAT_0042f418, 0xe);   | 268 LAB_004186f9:   |
| 79 DAT 00436180 = DAT 00421510:  | 165 (*DAT_00434214)(0x80000001]  "Software\\Microsoft\\Windows\\CurrentVersion\\Bun"_0_0xf003f   | 269 FUN_0041c0b0(uVar10,uStack_c,DVar7);<br>270 FUN_00417d90();   |
| 81 _DA1_0042T518 = 0;  | 107 DVar7 = GetModuleFileNamew((HMODULE)0x0,aWStack 310,0x104);  | 271 DAT_00436934 = lstrlenA(DAT_0043692c);  |
| <pre>82 DAT_0042f514 = 0;<br/>83 uVar16 = FUN_004010a4(extraout_ECX_02,(int)((ulonglong)uVar16 &gt;&gt; 0x20),0x42f418,0xd,</pre>  | 168 if (DVar7 == 0) {<br>169 FUN_0041b980(0);  | 272 FUN_0041ea50();<br>273 pcVar2 = (code *)swi(3);   |
| 84 (undefined *)&DAT_0042f47c,3);<br>85 _DAT_00436f80 = DAT_0042f510;  |  | 274 (*pcVar2)();<br>275 return;   |
|  | 172 iVar5 - 0  | 276 }   |
| 87 _DAT_0042f518 = 0;  | 173 iVarll = 0;  | 277   |
|  |  |   |

## Introduction:

Potential of LLMs in Reverse Engineering Malware

### Solution:

We hypothesize that with appropriate base model, fine-tuning, and a quality dataset, LLMs can significantly contribute to malware reverse engineering. They might automate the recognition of malicious code patterns, extraction of embedded constants, identification of malware signatures, and even suggest potential detection measures. This accelerates the malware analysis process and lowers the entry barrier for individuals venturing into this critical cybersecurity domain.

Today, I'll unfold my journey of constructing a dataset and a model, developing a plugin, and a plugin server, and amalgamating these elements to probe the potential of Large Language Models in the realm of malware reverse engineering.

## Literature Review (tools)

<u>G-3PO:</u> A Protocol Droid for Ghidra: An Al assistant developed by Olivia Lucca Fraser at Tenable for analysing and annotating decompiled code in Ghidra, which queries OpenAl and/or Anthropic's language models. See this writeup on the <u>Tenable tech blog</u> for details.

<u>Gepetto</u>: An IDA Pro plugin that queries GPT models for explanatory comments and meaningful variable names (like G-3PO for IDA Pro). Developed by Ivan Kwiatkowski.

<u>ai for Pwndbg</u>: Your trusty AI debugging sidekick, developed by Olivia Lucca Fraser at Tenable as a Pwndbg command.

ai for GEF: Same as above, but implemented as a GEF command. Developed by Olivia Lucca Fraser at Tenable.

<u>GPT-WPRE:</u> Whole-program Reverse Engineering with GPT-3. This is a little toy prototype of a tool that attempts to summarize a whole binary using GPT-3 (specifically the text-davinci-003 model), based on decompiled code provided by Ghidra. Developed by Brendan Dolan-Gavitt.

**IATelligence**: IATelligence is a Python script that extracts the Import Address Table (IAT) from a PE file and uses OpenAI's GPT-3 model to provide details about each Windows API imported by the file. The script also searches for related MITRE ATT&CK techniques and explains how the API could potentially be used by attackers. Developed by Thomas Roccia.

## Literature Review (models)

<u>Code Llama</u>: Code Llama is a family of large language models for code based on Llama 2 providing state-of-the-art performance among open models, infilling capabilities, support for large input contexts, and zero-shot instruction following ability for programming tasks. (7B, 13, 34B)

Starcoder: The StarCoder models are 15.5B parameter models trained on 80+ programming languages

<u>CodeAlpaca</u>: The Code Alpaca models are fine-tuned from a 7B and 13B LLaMA model on 20K instruction-following data

<u>replit-code-v1.5</u>: Replit Code v1.5 is a 3.3B parameter Causal Language Model focused on Code Completion.

# **Literature Review (papers)**

LmPa: Improving Decompilation by Synergy of Large Language Model and Program Analysis

DIRTY Usenix 2022 Conference Video: Augmenting Decompiler Output with Learned Variable Names and Types Code Paper PDF Demo

```
int find_unused_picture(int a1, int a2, int a3) {
    int i, j, v1;
    if (a3) {
        for (i = <Num>;; ++i) {
        if (i > \langle Num \rangle)
            goto LABEL_13;
        if (!*(*(<Num> * i + a2) + <Num>))
            break;
        }
        v1 = i;
    } else {
        for (j = <Num>;; ++j) {
        if (j > <Num>) {
        LABEL 13:
            av_log(a1, <Num>, <Str>);
            abort();
        if (pic_is_unused(<Num> * j + a2))
            break;
        }
        v1 = j;
    return v1;
```

| ID         | Developer                     | DIRTY               |
|------------|-------------------------------|---------------------|
| <b>a</b> 1 | AVCodecContext_0 *avctx       | MpegEncContext_0 *s |
| <b>a</b> 2 | <pre>Picture_0 *picture</pre> | Picture_0 *pic      |
| <b>a</b> 3 | int shared                    | int shared          |
| <b>v1</b>  | int result                    | int result          |

# Large Language models

How do they work?

1. **Input Embeddings**: Text sequences are converted into vectors. These vectors hold the meaning and context of words.

2. Attention Mechanism: Determines which parts of the input are relevant.

Weights importance of words based on context.

Allows for capturing long-range dependencies in text.

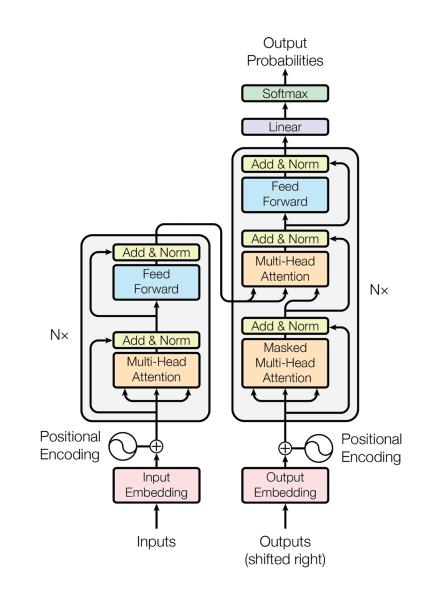
3. Encoder-Decoder Stacks: Processes and transforms input embeddings.

Multiple layers of encoders capture the context.

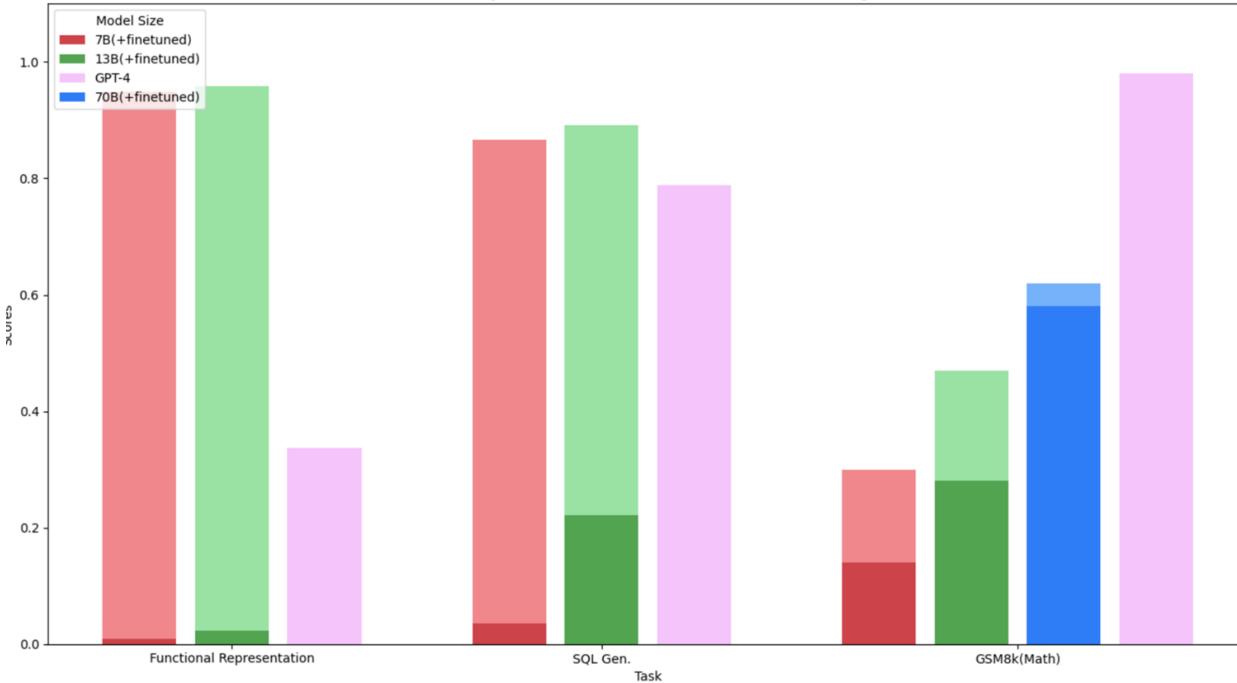
Decoders generate the output, be it text translation, summarization, or other tasks.

- 4. Positional Encoding: Ensures sequence order is maintained.
- Adds information about the position of each word in the sequence.

Transformers United 2023: Introduction to Transformers w/ Andrej Karpathy Neural Networks: Zero to Hero



Scores by task and model size before and after fine-tuning



How it's done

(1) Generate a dataset of decompiled functions paired with the answers to the questions we will ask.

- (2) Develop a set of prompts (questions) whose answers generate the data we require.
- (3) Select and fine tune an existing model.
- (4) Deploy webserver that loads the model and provides inference.
- (5) Provide IDE integration to send & receive code comments and function names for analysis & update IDE
- (6) Provide feedback mechanism for Analysts to make suggestions
- (7) Retrain / Test / Announce / Release

(1) Generate a dataset of decompiled functions that includes the following:

- 0. decompiled code
- 1. descriptive name
- 2. short summary
- 3. step-by-step description.

Develop a set of prompts

This shows an example function that builds a training example out of the dataset. The **input** is the decompiled code and the **output** is the function name being learned.

\* Context should be treated as a budget like in LmPa

```
def tokenize(prompt):
    result = tokenizer(
        prompt,
        truncation=True,
        max_length=4096,
       padding=False,
        return_tensors=None,
   # "self-supervised learning" means the labels are also the inputs:
   result["labels"] = result["input_ids"].copy()
   return result
def generate_and_tokenize_prompt(data_point):
   full_prompt =f"""You are an advanced malware reverse engineer capable \
of understanding decompiled C code and identifying malicious \
functionality please create a ### New Function Name from the \
decompiled code in the ### Context.
You must output a descriptive ### Function name for the decompiled code
provided in ### Context.
### Context:
{data_point["input"]}
```

```
### New Function Name:
  {data_point["output"]}
```

2

5

6

8 9

10

11

.2

13

14 15

16

17

18

19

20

21

22

23 24

26

28 29 30

```
Information Classification. General All rights reserved. Public
```

return tokenize(full\_prompt)

Generate a dataset

A synthetic malware dataset is generated with ttp\_code\_finder.py

- (1) recursively generate instruction prompts to request the implementation of a specific TTP using a specific API call
- (2) compile response and execute response
- (3) keep successful binaries
- (4) analyze / decompile the binaries

With this workflow we have the function description/summary/function name (instruction prompt) AND the decompiled code for the interesting function.

\*NOTE: Prompt template for ttp\_code\_finder.py to include print/debug statements and lots of comments when generating source of truth.

\*NOTE: Remove print statements before training or you end up relying on them instead of APIs & Arguments

### def get\_ttp\_code\_for\_api(api, action):

# Step 1: send the conversation and available functions to GPT

```
ip_address = "172.105.16.201"
port = "9000"
domain_name = "apt.watch"
```

sp\_content = "You are a C programming assistant designed to create working C code that "
sp\_content +="produces specific funtionality using a specific windows API. You must always "
sp\_content +="test your code with a compiler and it must compile with x86\_64-w64-mingw32-gcc. "
sp\_content +="Comment the code and print debug messages to help the user understand how "
sp\_content +="the code works. Try to make the code as unique and different as possible. "
sp\_content +="When generating code that demonstrates connection to a server use the "
sp\_content +=f"IP Address: {ip\_address} and Port: {port}. When contacting a domain name use "
sp\_content +=f"Domain name: {domain\_name}. When sending data to a server send \*.log files. "

#### messages=[

{"role": "system", "content": f"{sp\_content}"},
{"role": "user", "content": f"Please provide example code for how an attacker might abuse the api \"{api}\" \
 to provide \"{action}\" functionality. and test it with a compiler. You will need to pass compiler flags like \
 -lpsapi -lwininet -lws2\_32 in some cases"}

#### functions = [

```
"name": "compile_ttp_code",
"description": "This function will test to make sure code provided compiles.",
"parameters": {
   "type": "object",
    "properties": {
       "code": {
            "type": "string",
            "description": "C source code to compile",
        },
        "action": {
           "type": "string",
            "description": "functionmality provided",
        },
        "api": {
            "type": "string",
            "description": "the Function name being abused",
        },
        "compiler_options": {
            "type": "string",
            "description": "arguments to pass the compiler",
    },
```

enumeration = ["CreateToolhelp32Snapshot","EnumDeviceDrivers","EnumProcesses","EnumProce injection = ["CreateFileMappingA","CreateProcessA","CreateRemoteThread","CreateRemoteThread","CreateRemoteThread evasion = ["CreateFileMappingA","DeleteFileA","GetModuleHandleA","GetProcAddress","Load spying = ["AttachThreadInput","CallNextHookEx","GetAsyncKeyState","GetClipboardData","Get internet = ["WinExec","FtpPutFileA","HttpOpenRequestA","HttpSendRequestA","HttpSendRequestA","HttpSendRequestA","GetLogica anti\_debugging = ["CreateToolhelp32Snapshot","GetLogicalProcessorInformation","GetLogica ransomware =["CryptAcquireContextA","EncryptFileA","CryptEncrypt","CryptDecrypt","CryptO helper = ["ConnectNamedPipe","CopyFileA","CreateFileA","CreateMutexA","CreateMutexExA","

#### messages=[

{"role": "system", "content": f"{sp\_content}"},

{"role": "user", "content": f"Please provide example code for how an attacker might abuse the api \"{api}\" \
 to provide \"{action}\" functionality. and test it with a compiler. You will need to pass compiler flags like \
 -lpsapi -lwininet -lws2\_32 in some cases"}

e\_Connect\_2b94c14445d6434861a07cfba... e\_Connect\_f1b2d3c712c4bcb32e3081211... e\_ControlService\_69217efc11564d5a4e48... e\_CreateFile2\_953cb2b364e8721e673821... e\_CreateMutexA\_45dc0150489d4ea27559... e\_CreateToolhelp32Snapshot\_b7b61530f... e\_FindResourceA\_6c0dcf4aa0c1d86d2f2e... e\_FindResourceExA\_cc6209a64876bd405... e\_Gethostbyname\_08b1f5f3a563741edb6... e\_GetLogicalDrives\_aacce21b9411541ef9... e\_GetProcessHeap\_de2b06d945ba9f6bec... e\_GetTempPathA\_5ec7f371d7766fc271ef... e\_GetTickCount\_dbcfead40b372cc548bd9... e\_HttpSendRequestExA\_1a92a5cca95982... e\_Inet\_addr\_3afa27f9480dc13d773b80ffe... e\_InternetOpenA\_6facdb9a898e2c2de858... e\_InternetOpenUrIA\_7a3874a3fa1616034f... e\_InternetReadFileExA\_698449738c542c7... e\_lsDebuggerPresent\_1c17d09acb028176... e\_LoadLibraryExA\_3c02259a9fb5b38fa3b...

e IstrcatA 96c4e57d0f1dd53585c06c51c0...

abuse\_Connect\_4a736b99562a4543476f39301... abuse\_Connect\_f2926b9053c7374f4bd062fb56... abuse\_CopyFileExA\_81ae8edc599fe3fd7a62b8... abuse\_CreateFileA\_24c73653a547191f26248a4... abuse\_CreateMutexExA\_29ad0fd133eb2d7d6d... abuse\_DeviceloControl\_b0911d0caa283a1bf3f... abuse\_FindResourceA\_7ad3d16f66fa5e6d78d7... abuse\_GetDesktopWindow\_186fd76dea42835... abuse\_Gethostbyname\_b5b8ec7670d5a774f4a... abuse\_GetModuleBaseNameA\_b1223443bfb8... abuse\_GetTempFileNameA\_0a637ebff756b61... abuse GetTempPathA 28ccd015af36b94b6796... abuse\_GetTickCount\_e7ccf8d324ce0e9f3563e0... abuse\_HttpSendRequestExA\_6d2d115f6744e1... abuse\_Inet\_addr\_9a47bb22dfe1ccae8ba90577... abuse\_InternetOpenA\_54fe06fe8adcdc3fcf960... abuse\_InternetOpenUrIA\_8063131149e9c7647... abuse\_InternetReadFileExA\_e776a10490f1335... abuse\_Listen\_2b0fefe8a5a456d70a773be12d1... abuse\_LockResource\_5f09752376ae5c140396a... abuse IstrcatA abc05c345b85536f3bcfd53096...

abuse\_Connect\_5fe816f4721b7c820decf22159... abuse\_Connect\_fe408739100bd5a33f66df69f9... abuse\_CopyFileExA\_99efe425cb40c5b4ea821b... abuse\_CreateFileA\_b01e51e463e30f7bee4674... abuse\_CreatePipe\_5ba5a9983ae339d9b28c39... abuse\_FindClose\_4969a78b7fcef0675cb53a94f... abuse\_FindResourceA\_783e79c97db579be8ed... abuse\_GetDriveTypeA\_abd75edc10fd2fbb31d... abuse\_Gethostname\_2c6e7e89606a54358890b... Abuse\_GetModuleFileNameA\_3ee3b2327cf93... abuse\_GetTempFileNameA\_218b640825d401e... abuse\_GetTempPathA\_200fe38580990c891474... abuse\_HttpAddRequestHeaders\_68e55dd25fa... abuse\_HttpSendRequestExA\_6d199a44f7016c... abuse\_Inet\_addr\_12e01483b427356d222a4fd9... Abuse\_InternetOpenA\_824d51a3cde035699cb... abuse\_InternetOpenUrIA\_cb22410c7482c4aa6... abuse\_InternetSetOptionA\_8a8cdf8aa3bac59e... abuse\_Listen\_65ac8b64edfbb0780f3e90b1ce5... abuse\_LockResource\_7d065117ebbc42250861... abuse\_lstrcatA\_d9c2f0fb4ef1bd3c5989a791c8...

abuse\_Connect\_c313a0b26c8e14e5b25775560... abuse\_ConnectNamedPipe\_02906d69cb98c3f... abuse\_CopyFileExA\_f3bcdef4641152201a29d4... Abuse\_CreateFileA\_e18d5e040076391c49d79c... abuse\_CreatePipe\_5baf19f261a80cb635afc087... abuse\_FindNextUrlCacheEntryA\_4b35e49eb6d... abuse\_FindResourceExA\_796bfe4b681b6a61b... abuse\_Gethostbyname\_4e561917e8994fa2bd0... abuse\_GetLogicalDrives\_1ddbb3272f5021b7e... abuse\_GetProcAddress\_85f6ce89f070a1e0974f... abuse\_GetTempFileNameA\_d51f243a61ccb41... abuse GetThreadId 62261d1723041d683e372... abuse HttpOpenRequestA 6d4a0c3a7836e2d... abuse\_HttpSendRequestExA\_cfb0a6d0a59e99... abuse\_Inet\_addr\_d3a3c60d846ee55ed66accf6... abuse\_InternetOpenA\_eb53aeec87bac0ce4259... abuse\_InternetReadFileExA\_0708ea4eda66d6e... abuse\_InternetSetOptionA\_42e0ee06c2e6bec0... abuse\_Listen\_a3830a2a8d8dc84211478fb9acc... abuse\_LockResource\_141e237c4f305d82f09e4... abuse\_MoveFileA\_44ae480f6fe2610c470816af...

**Generate a dataset: Malware Processing Pipeline** 

Synthetic data is a great start, but need to get more real world malware samples in front of the model for training. We need to be able to recognize anti-dissassembly / anti-debugging malware code.

We leverage an existing set of tools to generate candidate functions for analysis and store them for review.

### AiAnalyzer.py Workflow



#### 1. PEfile and Detect It Easy

The Python library `pefile` and the CLI tool `Detect It Easy` (DIE) are potent utilities for analyzing PE (Portable Executable) files. `pefile` facilitates the inspection of a PE file's structure and attributes, including headers, sections, imports, exports, and resources, aiding in the understanding of its behavior. Additionally, `Detect It Easy` offers extended functionality, including the identification of file types, entropy analysis, and detection of specific packers.

### 2. **YARA**

YARA is known for its pattern matching capabilities, making it a valuable tool for identifying and classifying malware based on textual or binary patterns. It can be easily extended

### 3. **Capa**

Capa is utilized for capability hunting which is crucial for understanding what a particular piece of code or software is capable of doing.

### 4. Radare2

This is a comprehensive framework for analyzing binaries, dealing with the analysis of imports, exports, strings and decompilation, which are critical for reverse engineering purposes.

### 5. Langchain

Interaction with LLM, generate requests and store responses

### 6. Ghidra llm.py

The output from AiAnalyzer is then loaded by a plugin in Ghidra, which is a software reverse engineering (SRE) framework. This plugin is designed to process and display the intelligence extracted by the LLM.



# Execute CAPA

**Finding interesting functions** 

We execute CAPA to detect interesting regions of code and end up with a bunch of memory addresses (offsets).

0

.2

6

18 19 20

26

## <u>Github</u>

| ıle:  |
|---|
| meta:   |
| name: log keystrokes                                |
| namespace: collection/keylog                        |
| authors:  |
| – moritz.raabe@mandiant.com                         |
| scope: function                                     |
| att&ck:   |
| – Collection::Input Capture::Keylogging [T1056.001] |
| examples:   |
| – C91887D861D9BD4A5872249B641BC9F9:0x4015FD         |
| features:   |
| - or:   |
| - and:  |
| – api: SetWindowsHookEx                             |
| - api: GetKeyState                                  |
| - and:  |
| – api: RegisterHotKey                               |
| - api: user32.keybd_event                           |
| - api: UnregisterHotKey                             |
| - and:  |
| – api: CallNextHookEx                               |
| – api: user32.GetKeyNameText                        |
| – api: user32.GetAsyncKeyState                      |
| – api: user32.GetForgroundWindow                    |
| - api: user32.AttachThreadInput                     |
| – api: user32.MapVirtualKey                         |



# **Execute CAPA**

**Finding interesting functions** 

We execute CAPA to detect interesting regions of code and end up with a bunch of memory addresses (offsets).

## <u>Github</u>

| rule:   |    |
|---|----|
| meta:   |    |
| name: capture public ip                             |    |
| namespace: collection/network                       |    |
| authors:  |    |
| - "@_re_fox"  |    |
| scope: function                                     |    |
| att&ck:   |    |
| - Discoverv::System Network Configuration Discoverv | [T |

- Discovery::System Network Configuration Discovery [T1016]
examples:

- 84f1b049fa8962b215a77f51af6714b3:0x100061e5
features:

- and:

2

4

10 11

12

13

14

15

16

17

18

19

20 21

22 23

24

25 26

27

28

- api: InternetOpen
- api: InternetOpenUrl
- api: InternetReadFile
- or:
  - substring: "bot.whatismyipaddress.com"
  - substring: "ipinfo.io/ip"
  - substring: "checkip.dyndns.org"
  - substring: "ifconfig.me"
  - substring: "ipecho.net/plain"
  - substring: "api.ipify.org"
  - substring: "checkip.amazonaws.com"
  - substring: "icanhazip.com"
  - substring: "wtfismyip.com/text"
  - substring: "api.myip.com"
- substring: "ip-api.com/line"



# **Execute CAPA**

**Finding interesting functions** 

We execute CAPA to detect interesting regions of code and end up with a list of memory addresses (offsets).

### <u>Github</u>

#### import capa

rules = capa.main.get\_rules([f"{self.caparulepath}"])

#### try:

extractor = capa.main.get\_extractor(file\_path, format\_, \_os, backend, sigpaths) capabilities, \_ = capa.main.find\_capabilities(rules, extractor) output\_data = {} for rule\_name, matches in capabilities.items(): for match in matches: memory\_address = match[0] if str(memory\_address) not in output\_data: output\_data[str(memory\_address)] = [] output\_data[str(memory\_address)].append(rule\_name) self.analysis\_results["capa"] = output\_data # Store analysis results in capa key **#TODO:** Implement better error handling except capa.exceptions.UnsupportedFormatError: **#TODO** This causes a parsing bug down the line when a dict is expected but a str is found. self.analysis\_results["capa"] = f"Unsupported format: {file\_path}" print("Unsupported format: ", file\_path) except ValueError: **#TODO** This causes a parsing bug down the line when a dict is expected but a str is found. self.analysis\_results["capa"] = f"ValueError: {file\_path}" print("ValueError: ", file\_path)

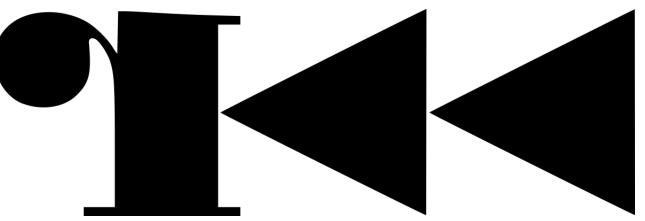


# **Decompile Function**

**Decompiling interesting functions** 

With the interesting function offsets above we use radare2 to decompile the function and store the results in the analysis\_results dict.

Website Github



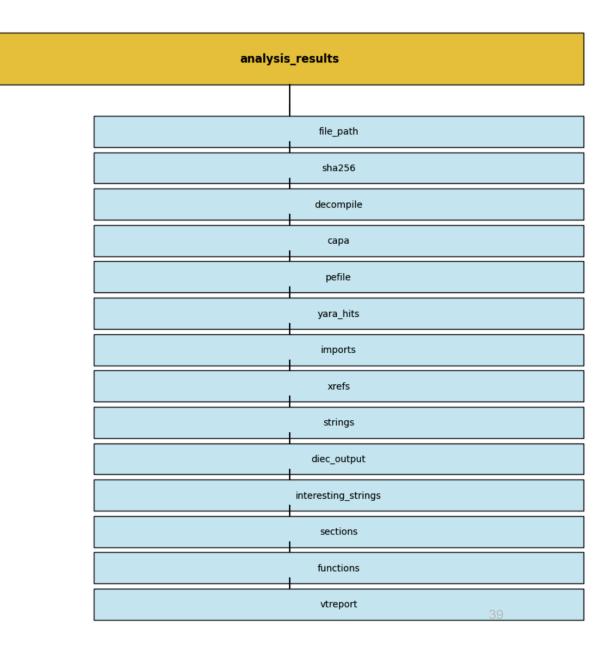


## Analysis Results Data

**Decompiling interesting functions** 

`analysis\_results` is a dictionary of dictionaries containing the results of the entire analysis pipeline.

We're interested in the capa dictionary which store the function offsets for matched rules and the functions dictionary that contains the decompiled code.



1/1 - + 0 9

| 1. сривски – –       |            |
|----------------------|------------|
| Android Flanct       | cetav      |
| Android, Greywolf    | Colib      |
| Android .Hunningbad  | Colle      |
| Android.Etausinc     | Colle      |
| Android Medusa       | Confi      |
| Android.Octo         | Conti      |
| Android Priphone     | Cores      |
| Android RATHILad     | Crate      |
| Android Runnus       | Grent      |
| Android . Sharkbot   | Crust      |
| Android Teabut       | Cryat      |
| Android, Kevier      | Crypt      |
| Andromeda            | Crypt      |
| AnglerEX             | Crypt      |
| Aria-Body            | Cryxe      |
| Arkeistesler         | CLIM       |
| AsyecRAT             | CTB-L      |
| AtomSile             | Oub-all    |
| AveRarta             | Curat      |
| áveo                 | CVE-2      |
| AvosLockerBansonware | CVE-2      |
| Azorult              | CVE-2      |
| Elter28              |            |
| Babededa             | CVE-1      |
| Babuk.               | CVE-2      |
| RandarChorRansonware | OVE-3      |
| Bankshot             | CVE-1      |
| Banload              | Cycle      |
| Bartalex             | Die is all |
| Basbanke             | Sario      |
| Bashlite             | Darith     |
| Batloader            | Da nico    |
| Bazarbeckdoor        | Derks      |
| BazarLoader          | Darkt      |
| Bedep.               | Derict     |
| BHUNTStealer         | Denky      |
| EtanLizeApesonware   | Daser      |
| BUCHAT               | DORAT      |
| #Lierro              | Ocede      |
| ElackSestalansorware | Deard      |
| al action to         | Other Md   |

Courbs. FritzFrog riteale Generedon GaneoverP27 Gasettrah ression Gauss GenshinDrive GlobelImposter Gluptebs etress GOODOSING GoSearch. Ler GRELIETE Grandoretro answeightere GPL PENDESE 1-10271 Gwiststecker HARREST 1-5662 HolkBack -1199 HevenaCrypt -22954 HunkEyeKeyLogge HeaderTip Lafanserware HelloKitts Hornet1chipe Hivelandoneare stits. CE RAE MOTOPOLEEADT Houdtet Hypersing Ratsonware Bancomiane InedId somular e #LackCatRansorma decompress.py Instantionite Innortal Stealer Demonia Elackintus monthit' index.html Dexpla SLevol Remonant e DiscordRAT **DiscordTokenStealers** InnApent Distruck Diva DeSpyTrejan 3sff 3Lanna bridex-Halders 3rtp8ot locksindex.html OTPacker 35ocket Dyre Echelenitealer

FLaundGrace

**Fornbook** 

Latata Greater Racharbance Lanbert Nenhalsha Hane-core Necro LenonDuck NerbianEAT LightningFrancwark Lilitheassonware Lineratio NetFilter Linux-Spike Netwalker LocksttRassonmere NetHUPERAT LockerGoga Networn LockyBansonware Neutrinets Agrbet Log43Halware Lohibet ALELOVEPOS Lokillocker Rensoner o nede-Lpc-Protestware Nekoyawaitansenwara NuclearEX Lorenzikansonware Rec05-AdLoad Nenando systendernentse Rec05.AppleJeus Bar05.Calisto **OldGreelin** Calimer Spenbot Mac05.Coldrest Mac05.Convutter CaysRansonware Crist: Rac05.Dek Ortetalegeer Mac05.Evilguest HacOS Kellanger MacOS.EltH MacOS.LeoShu Cutoria Parichan Fandadaaker Rac05 PLFF11 Fanderskansonware MacOS, Shlayer Parellestat Rac05 Tarmac Passey RECOS-RESSET Rac05.XLoader Rac05.Zurs Pegasus PhobosRanserware RegulberRansonar PickandPlacetal Nandrake Rangusaka PLatinumGroup Mas Ransomere Pany Peseldoe HedussLocker HekotioBenker FowerSteger MenentoRanconwar Recurtalitealer

Rapperbot THEFT RetDispesser TypeHash RawPos USeateAT Reaven usbeatprit RedAlertkensorware Usberry Vagen HORT LODGER REPLOY VerwillenStrike V Lden Sk Leine Roaminglantis Wirlock RonberrElk Hel Utess Herdlod Rotalastro Rozena REPOS. whiteRabbitRansonware Windwaler Sansantassoniar MILLINGS. Hinstoader **UpBruteflot** Scheron SFileRansonware Shellbot ShellCrew xenomorph XFilesStealer Shinel orker Second Second **EPertkat** Skloper XTrenellat Sealer and or SnakeKeyLogger VanLuowangKanzonware Soctholish vorekey SolarwindsBreech YoungLatus SpiderpigHAT SabNet Stantinko ZeusAction Stealstt stoader Stegolowder Strifelater Zunanick ZuoRAT STRRAT authell. SendownER SVCReady "predator the Thief"



(base) gputigou:-5 2 gputtiger -/tode/Alteratyree v

BlackholeEE

BlackMetter

Lacknoon

LackRock

Lacksowl

LackTech

intenaco.

listerLooder

lin name: process\_function\_18003753 lln name: update\_liskei\_list\_1880384d lln name: check\_and\_increment\_process\_id\_10003659 Lin name: process control 190030cb Lin name: find\_aligned\_eddress\_1800399c In name: process\_paran\_1\_10001005 lin name: process\_paran\_1\_10001061 lin name: process\_event\_logging\_18001422 IIn name: process\_event\_iogging\_imeDia/ IIn name: process\_logging\_18001883 IIn name: process\_saram 1 10001801 IIn name: check\_process\_slatus\_18001507 IIn name: find\_common\_elements\_18001304 In name: find\_composes\_bit\_log00502 In name: coy\_and\_process\_bit\_10001602 In name: taltigize\_critical\_section\_10001609 In name: resize\_buffer\_10001603 lin name: cloanup\_process\_resources\_10001d2c lin name: register\_event\_10001d0; lin name: process\_data\_18082027 lin name: process loop 1808255d lin name: process coop immersio lin name: process\_window\_hook\_IM0030825 lin name: process\_loop\_10002075 lin name: process\_data\_10002937 Lin name: namage\_hotkeys\_18002970 [In name: close\_handles\_and\_reset\_variables\_100029d1

2 coufficour -/code/AiAnalyzer

3 gpuBguu -food/Alvaless burnepiling. . detHid+ Decempiling. . detHid+ Decempiling. . detHid+ Decempiling. . detHid: Decempiling. . decempiling. . detHid: Decempiling. . decempiling. . decempiling. . decempiling. . decempili

n x 7.goopgne-/coce/AlAndyne ~ 1LP name: execute\_function\_chain\_004056e3 Lin mar: wreis, intrib.com, or wrose Lin mar: gr.ndbl.endls.do. Lin mar: gr.ndbl.endls.md.st.posters.ge000710 Lin mar: gr.ndbl.e.hardte\_md\_st\_posters.ge000710 Lin mar: execute\_pices.jesg.function\_d000000 Lin mar: execute\_ficat\_st\_outers.ge000700 Lin mar: execute\_ficat\_st\_outers.ge0007010 Lin mar: execute\_ficat\_st\_outers.ge0007010 Its name: execute function chain obidides lin nume: execute\_finction\_chain\_db0m0de3 lin nume: certe\_finction\_chain\_db0m0de3 lin nume: free\_main\_db0m0de\_set\_pointers\_0000710 lin nume: free\_main\_db0m0de\_set\_pointers\_0000710 lin nume: certe\_finct\_bardle\_and\_set\_pointers\_0000710 lin nume: certe\_finct\_bardle\_and\_set\_pointers\_0000710 lin nume: centert\_finct\_to\_integer\_0000750 lin nume: contert\_finct\_to\_integer\_0000750 lin nume: contert\_finct\_to\_intege

Lin Ann: free.norv, 3043703 Lin Ann: free.norv, 3043703 Lin Anne: convert\_float\_to\_introjer\_0040750 Lin Anne: det\_bad\_instruction\_dsta\_00425009 Lin Anne: get\_process\_type\_0040720 lls name: free\_memory\_0043/9980 lls name: handle\_bad\_lsstruction\_data\_00425569 Un nanat get process\_type\_00407c20 11m name: handle bad instruction data 00423869 lin nene: get\_process\_type\_09407c20 \_in nene: get\_process\_type\_09407c20

LLn name: execute\_process\_with\_parans\_Bededca8 LLn name: execute\_process\_Bedebc4e

Lin name: Finces, and execute, Bededdee Lin name: process\_artical\_section\_Bededdee Lin name: process\_critical\_section\_BededDea2 Lin name: process\_critical\_section\_BededDea1 Lin name: process\_critical\_section\_BededDea1

lin name: check\_and\_call\_function\_dedetc71 lin name: process\_function\_dede3680

ilm name: process\_critical\_section\_0ede032b
ilm name: process\_critical\_section\_0ede0585

lln name: validate and process cata Bede7181

tin name: delete and cleanup process dedeo#92 lin name: center\_ang\_croces\_uth\_proces\_uth\_caran\_dede7Sab Lin name: initialize\_proces\_uth\_caran\_dede7Sab Lin name: initialize\_proces\_dede7f4c \_lin name: process\_data\_eadeaebs

lln name: copy\_vartants\_bedecd44 lln name: cleanup\_process\_data\_0ede0bf2

Lin name: handle window message Dede1367 Lin name: calculate offset bedes/be lln name: free\_memory\_0edeS2c4 Lin name: free\_and\_reset\_paran\_1\_BedesSac Lin name: process\_cleanup\_Dede5c25

D × E: gputligpiz -/code/WAnalyzer ~

lin name: process\_data\_ondebios lin name: process\_data\_0cdebbd4

lin name: check and acquire lock Sediaffa lin name: update\_process\_status\_56d3977c lin name: update\_paran\_2\_pointer\_56d39268 11m name; ranove element from list sedjusdu lin name: execute\_process\_56d394f8 Lin name: upcate\_process\_status\_56d39b14

D X Staputhcow -/code/AlAnalyzer -

m a 4 gru@gat -/code/AlAnsiyce ~

lin name: execute\_process\_sodc4f47

lin name: initialize process\_S6d37750 lin name: process\_param\_S6d380fc

lin name: process function 560381a8 lin name: get\_cpuid\_info\_5503868c lin name: initialize\_paran\_1\_56038448

tin name: execute\_process\_56d3852c Tin name: initialize\_memory\_55d38628

lin name: process\_paran\_2\_56d3e948 lin name: initialize process pointer\_56d38728 lin name: check and execute callback\_Std1899c lin name: find\_and\_call\_function\_S6d38b38

lin name: call\_function\_with\_paran\_\$6d30c20 lin name: execute\_function\_\$6d18c78 lin name: process\_execution\_S6d30cd0

lin name: check and execute task \$6630650 lin name: get\_logical\_processor\_information\_56038054

lin name: reset\_and\_call\_another\_function\_Sod30f00 lin name: execute process \$6430f70

lin name: leave\_critical\_section\_100dSedt
lin name: initialize\_process\_100e2de7 lin name: call\_LCMapStringtk\_100e2e01
lin name: convert\_locale\_name\_to\_lcid\_t00e2e18 lin name: convert\_locate\_name\_up\_ctig\_toperio lin name: eater\_critical\_section\_i00e4a0f lin name: leave\_critical\_section\_i00e4a0f lin name: calculate\_polymontal\_approximation\_i00e6ade lin name: convert\_bc\_float10\_i00e7ab0 Lin name: process\_status\_update\_100e001e lin name: process\_data\_indexc5a lin name: call\_fcn\_with\_reversed\_params\_100eao7e Lin name: call\_ccm\_wich\_reveryse\_params\_topeaare Lin name: call\_cro\_100c1066\_10002c80 Lin name: wait\_for\_slople.object\_and\_decrement\_counter\_t003bbd0 Lin name: wait\_and\_close\_handle\_10049e00 Lin name: hitisitze\_process\_10001473 tin name: calculate\_weights\_10004ea0 lln name: validate\_string\_10017090 lln name: compare\_memory\_regions\_100170a0 lln name: get\_unique\_value\_isozoobo lln name: find\_index\_of\_value\_is\_array\_100307f0 lin name: find first process with flag 10036696 tin name: find process by param 10035cfs lin name: get value from array 1001af00 lin name: execute\_process\_with\_collback\_10055c10 lin name: terminate\_and\_exit\_process\_1001f010

0/638

| 7: gpu@gpu: ~/code/AiAnalyzer   ~  | × | 4: gpu@gpu: ~/code/AiAnalyzer 🗸  |
|--|---|--|
| <pre>llm name: execute_function_chain_004066e0<br/>llm name: allocate_memory_004067e0<br/>llm name: get_module_handle_and_set_pointers_00406710<br/>llm name: get_module_handle_and_set_pointers_00406710<br/>llm name: execute_function_chain_004066e0<br/>llm name: execute_function_chain_004066e0<br/>llm name: execute_function_chain_004066e0<br/>llm name: convert_float_to_integer_00407610<br/>llm name: get_module_handle_and_set_pointers_00406710<br/>llm name: free_memory_00407990<br/>llm name: free_memory_00407990<br/>llm name: free_memory_00407990<br/>llm name: free_memory_00407990<br/>llm name: convert_float_to_integer_00407610<br/>llm name: free_memory_00407990<br/>llm name: convert_float_to_integer_00407610<br/>llm name: free_memory_00407990<br/>llm name: convert_float_to_integer_00407610<br/>llm name: free_memory_00407990<br/>llm name: convert_float_to_integer_00407610<br/>llm name: handle_bad_instruction_data_00425069<br/>llm name: free_memory_0040720<br/>llm name: free_memory_0040720<br/>llm name: free_memory_0040720<br/>llm name: handle_bad_instruction_data_00425069<br/>llm name: get_process_type_00407c20<br/>llm name: get_process_type_00407c20</pre> |   | <pre>llm name: execute_process_56dc4f47<br/>llm name: initialize_process_56d37f50<br/>llm name: process_param_56d380fc<br/>llm name: process_function_56d381a8<br/>llm name: get_cpuid_info_56d3840c<br/>llm name: initialize_param_1_56d38448<br/>llm name: initialize_param_1_56d38448<br/>llm name: execute_process_56d3852c<br/>llm name: initialize_memory_56d38628<br/>llm name: process_param_2_56d3e948<br/>llm name: initialize_process_pointer_56d3872<br/>llm name: check_and_execute_callback_56d3890<br/>llm name: find_and_call_function_56d38c38<br/>llm name: call_function_with_param_56d38c20<br/>llm name: check_and_execute_task_56d38c48<br/>llm name: process_execution_56d38c78<br/>llm name: process_execution_56d38c48<br/>llm name: check_and_execute_task_56d38d50<br/>llm name: reset_and_call_another_function_56<br/>llm name: execute_process_56d38f78<br/>llm name: update_process_status_56d3927c<br/>llm name: update_param_2_pointer_56d39368<br/>llm name: remove_element_from_list_56d393d8<br/>llm name: update_process_status_56d394f8<br/>llm name: update_process_status_56d394f8</pre> |
| 8: gpu@gpu: ~/code/AiAnalyzer      ~   | × | 5: gpu@gpu: ~/code/AiAnalyzer ~  |
| <pre>llm name: process_data_0ede0b05<br/>llm name: process_data_0ede0b04<br/>llm name: execute_process_with_params_0ede0ca0<br/>llm name: execute_process_oede004e<br/>llm name: process_and_execute_0ede0eaa<br/>llm name: process_critical_section_0ede0eaa<br/>llm name: process_critical_section_0ede1011<br/>llm name: process_function_0ede120<br/>llm name: process_data_0ede1320<br/>llm name: check_and_call_function_0ede1c71<br/>llm name: process_function_0ede3680<br/>llm name: claculate_offset_0ede47be<br/>llm name: free_nemory_0ede52c4<br/>llm name: free_nemory_0ede52c4<br/>llm name: process_critical_section_0ede612b<br/>llm name: process_critical_section_0ede612b<br/>llm name: process_critical_section_0ede6585<br/>llm name: cleanup_process_data_0ede6ff2<br/>llm name: cleanup_process_data_0ede6ff2<br/>llm name: validate_and_process_0ede7101<br/>llm name: initialize_process_0ede7f4e<br/>llm name: initialize_process_0ede7f4e<br/>llm name: process_data_0ede8ed5</pre>  |   | <pre>llm name: leave_critical_section_100dSed1<br/>llm name: initialize_process_100e2de7<br/>llm name: call_LCMapStringEx_100e2e01<br/>llm name: convert_locale_name_to_lcid_100e2e01<br/>llm name: enter_critical_section_100e4a8f<br/>llm name: leave_critical_section_100e4ad7<br/>llm name: calculate_polynomial_approximation<br/>llm name: convert_to_float10_100e70ab<br/>llm name: process_status_update_100e961e<br/>llm name: process_data_100e965a<br/>llm name: call_fcn_with_reversed_params_100e0<br/>llm name: call_fcn_100d4a66_100d2e0b<br/>llm name: wait_for_single_object_and_decremed<br/>llm name: wait_and_close_handle_10049e00<br/>llm name: initialize_process_10001df3<br/>llm name: validate_string_10017090<br/>llm name: get_unique_value_100200b0<br/>llm name: find_index_of_value_in_array_10036<br/>llm name: find_first_process_with_flag_10036<br/>llm name: get_value_from_array_1005af00<br/>llm name: execute_process_with_callback_1009<br/>llm name: terminate_and_exit_process_1001f43</pre>   |

728 90c n\_56d38d94 56d38f08

e2e1b lon\_100e6ade 00eaa7e ement\_counter\_1003bbd0 0307f0 036690 llm name: execute\_process\_with\_callback\_10055e10
llm name: terminate\_and\_exit\_process\_1001f410

#### **Review Dataset Item**



#### **Unreviewed Items Remaining: 1**

#### Suggested name:

LLM Suggested name:

all\_retrieveOperatingSystemAndComputerInfo

#### LLM Summary:

Retrieves the operating system name, computer name, and a constant string

#### Modify Short Summary:

Retrieves the operating system name, computer name, and a constant string

#### Modified by step Summary:

- 1. Opens the registry key 'SOFTWARE\Microsoft\Windows NT\CurrentVersion' using RegOpenKeyExW function.
- 2. Queries the value of 'ProductName' from the opened registry key using RegQueryValueExW function.
- 3. Closes the opened registry key using RegCloseKey function.
- 4. If the return value of RegCloseKey is not 0, sets the 'local\_210' variable to '???' using lstrcpyW function.
- 5. Retrieves the fully qualified domain name of the computer using GetComputerNameExW function.
- 6. Formats the retrieved operating system name and computer name into a JSON string and appends it to 'DAT\_0043c8e4' using lstrcatW function.
- 7. Appends a constant string 'venus' to 'DAT\_0043c8e4' using lstrcatW function.

1. Opens the registry key 'SOFTWARE\Microsoft\Windows NT\CurrentVersion' using RegOpenKeyExW function.

2. Queries the value of 'ProductName' from the opened registry key using RegQueryValueExW function.

- 3. Closes the opened registry key using RegCloseKey function.
- 4. If the return value of RegCloseKey is not 0, sets the 'local\_210' variable to '???' using lstrcpyW function.
- Retrieves the fully qualified domain name of the computer using GetComputerNameExW function.

6. Formats the retrieved operating system name and computer name into a JSON string and appends it to 'DAT 0043c8e4' using lstrcatW function.

7. Appends a constant string 'venus' to 'DAT\_0043c8e4' using lstrcatW function.

### Save Changes

### Code:

#### 17 <sub>14</sub>

This code retrieves the operating system name, computer name, and a constant string. The code performs the following steps:

1. Opens the registry key 'SOFTWARE\Microsoft\Windows NT\CurrentVersion' using RegOpenKeyExW function.

 Queries the value of 'ProductName' from the opened registry key using RegQueryValueExW function.

3. Closes the opened registry key using RegCloseKey function.

 If the return value of RegCloseKey is not 0, sets the 'local\_210' variable to '???' using lstrcpyW function.

 Retrieves the fully qualified domain name of the computer using GetComputerNameExW function.
 Formats the retrieved operating system name and computer name into a JSON string and appends it to 'DAT\_00043c8e4' using lstrcatW function.

7. Appends a constant string 'venus' to 'DAT\_0043c8e4' using lstrcatW function. \*/

#### void retrieveOperatingSystemAndComputerInfo(void)

#### {

LSTATUS LVar1; WCHAR local\_210 [256]; DWORD local\_10; DWORD local\_c; HKEY local\_8;

local\_8 = (HKEY)0x0;

#### local\_c = 0x100;

RegOpenKeyExH((HKEY)0x80000002,L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion",0,1,&local\_8); RegQueryValueExH(local\_8,L"ProductName",(LPDWORD)0x0,(LPDWORD)0x0,(LPBYTE)local\_210,&local\_c); LVar1 = RegCloseKey(local\_8); if (LVar1 != 0) { lstrcpyH(local\_210,L"???");

#### local 10 = 0x100;

wsprintfW(DAT\_0043ab08,L"\"OS\": \"%s\",",local\_210); lstrcatW(DAT\_0043c8e4,DAT\_0043ab08); GetComputerNameExW(ComputerNamePhysicalDnsFullyQualified,local\_210,&local\_10); wsprintfW(DAT\_0043ab08,L"\"CompName\": \"%s\",",local\_210); lstrcatW(DAT\_0043c8e4,DAT\_0043ab08); wsprintfW(DAT\_0043ab08,L"\"ext\": \"%s\"",L"venus"); lstrcatW(DAT\_0043c8e4,DAT\_0043ab08);

```
Information Classification: Ceneral All rights reserved. Public
```

```
A
```

```
int __fastcall FUN_00402e90(int param_1,uint param_2,uint param_3,uint param_4)
                     int iVar1;
                     int iVar2;
                     uint uVar3;
                     int iVar4;
                     uint uVar5;
                     uint local 8;
                                                                                                                                         FUN_00402e90 -> findMatchingStringIndex
                     uVar5 = 0;
                                                                                                                                          param_1 -> pImportDesc
                     local_8 = 0xffffffff;
                                                                                                                                          param_2 -> searchedHash
                     iVar4 = *(int *)(*(int *)(param_1 + 0x3c) + 0x78 + param_1);
                                                                                                                                          param 3 -> checksum
                     iVar1 = *(int *)(iVar4 + 0x24 + param_1);
                                                                                                                                          param_4 -> funcRva
Update Rating
                     iVar4 = iVar4 + param_1;
                                                                                                                                          iVar1 -> nameTableRva
                     iVar2 = *(int *)(iVar4 + 0x20);
                                                                                                                                          iVar2 -> addrTableRva
                     if (*(int *)(iVar4 + 0x18) != 1) {
                                                                                                                                          uVar3 -> currCheckSum
                       do {
                                                                                                                                          iVar4 -> pThunk
                         uVar3 = FUN_00417be0((int *)(*(int *)(iVar2 + param_1 + uVar5 * 4) + param_1),param_3,param_4)
                                                                                                                                          uVar5 -> i
                                                                                                                                          local_8 -> matchingIndex
                         if (param_2 == uVar3) {
                           local_8 = (uint)*(ushort *)(iVar1 + param_1 + uVar5 * 2);
                           break;
                         uVar5 = uVar5 + 1;
                       } while (uVar5 < *(int *)(iVar4 + 0x18) - 1U);</pre>
                     return *(int *)(*(int *)(iVar4 + 0x1c) + local_8 * 4 + param_1) + param_1;
```

# LLMs in the Lab

LLM empowered Malware Analysis.

# Create a training dataset that will work with hugging face transformers

```
import os
import json
import click
from datasets import Dataset
import pandas as pd
from multiprocessing import Pool, cpu_count
def process_file(json_file):
    data = []
    with open(json_file, 'r') as f:
           data_dict = json.load(f)
           for function in data_dict['decompile']:
                    code = data_dict['decompile'][function]['decompiled_code']
                   # Remove lines that start with '//'
                   code = "\n".join(line for line in code.split('\n') if not line.strip().startswith('//'))
                   llm_name = data_dict['decompile'][function]['llm_name']
                   llm_name = llm_name.rpartition('_')[0]
                   text = "<s>" # Begin of Sentence token
                   text += "You are a malware analyst. You have been given the following decompiled code:\n"
                   text += code
                    text += "\n\nYour task is to determine what the code does and provide a descriptive name for the function.\n'
                    text += "Please enter your answer below:\n"
                   text += "Answer:" + llm_name + "</s>" # End of Sentence token
                   data.append({"text": text})
               except KevError as e:
                   print(f'Error processing function in JSON file: {json_file} Error: {e}')
       except Exception as e:
           print(f'Error loading JSON file: {json_file} Error: {e}')
    return data
@click.command()
@click.option('---dir', required=True, help='The directory to search for JSON files.')
def create_dataset(dir):
```

```
Recursively walk through directory and find JSON files.
Create a HuggingFace Dataset from the data and split into train and validation sets.
```

json\_files = [os.path.join(root, file) for root, \_, files in os.walk(dir) for file in files if file.endswith('.json')]

```
# Create a pool of workers and distribute the file processing among them
with Pool(cpu_count()) as p:
    data_list = p.map(process_file, json_files)
```

# Flatten the list of lists into a single list
data = [item for sublist in data\_list for item in sublist]

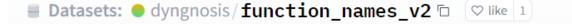
dataset = Dataset.from\_pandas(pd.DataFrame(data))

# split into train and validation sets
datasets = dataset.train\_test\_split(test\_size=0.2)

# save datasets to disk
datasets['train'].save\_to\_disk('train\_dataset-large-v2')
datasets['test'].save\_to\_disk('validation\_dataset-large-v2')

print('Saved training set to disk.')
print('Saved validation set to disk.')

if \_\_name\_\_ == "\_\_main\_\_":
 create\_dataset()



→ Files and versions Community Dataset card Dataset Viewer 🖸 Auto-converted to Parquet 📣 API 🛛 🖽 Go to dataset viewer Split train (60.5k rows)  $\sim$ Q Search this dataset instruction input output string · *classes* string · lengths string · lengths 1 value 42 9.67k 3 561 Analyze the supplied decompiled code and produce a void fcn.140030b80(ulong param\_1, ulong param\_2, process\_with\_params descriptive name for the function. The function ... ulong param\_3) { ulong uVar1; uVar1 =... Analyze the supplied decompiled code and produce a void \_\_thiscall fcn.005ac32c(uint \*param\_1, uint call function with parameter descriptive name for the function. The function... param\_2, uint \*param\_3, uint param\_4, uint... Analyze the supplied decompiled code and produce a uchar \_\_fastcall fcn.0049b228(uint param\_1, find first negative index descriptive name for the function. The function... int32\_t param\_2, int32\_t param\_3, int32\_t param\_4... Analyze the supplied decompiled code and produce a int32\_t \_\_thiscall fcn.0048264a(int32\_t param\_1, process\_status\_update uint param\_2) { int32\_t iVar1; fcn.0047ae5b();... descriptive name for the function. The function... Analyze the supplied decompiled code and produce a void fcn.1000c7ad(void) { int32 t in EAX; set\_process\_address descriptive name for the function. The function... fcn.1000cb20(); \*(in EAX + 0x54) = 0x100518a4;... Analyze the supplied decompiled code and produce a uint \_\_fastcall fcn.004cd798(uint param\_1, char 1.1.1 < Previous 1 2 3 ... 605 Next >

Information Classification: General



# LLMs in the Lab

LLM empowered Malware Analysis.

# fine tuned StarCode->StarDecoder (It sucked)

| <pre>train_data.start_iteration = 0</pre>  |
|--|
| <pre>print("Starting main loop")</pre>   |
| <pre>training_args = TrainingArguments(<br/>output_dir=args.output_dir,<br/>dataloader_drop_last=True,<br/>evaluation_strategy="steps",<br/>max_steps=args.max_steps,<br/>eval_steps=args.eval_freq,<br/>save_steps=args.save_freq,<br/>logging_steps=args.log_freq,<br/>per_device_train_batch_size=args.batch_size,<br/>per_device_eval_batch_size=args.batch_size,<br/>learning_rate=args.learning_rate,<br/>lr_scheduler_type=args.lr_scheduler_type,<br/>warmup_steps=args.num_warmup_steps,<br/>gradient_accumulation_steps=args.gradient_accumulation_ste<br/>gradient_checkpointing=not args.no_gradient_checkpointing,<br/>fp16=not args.no_fp16,<br/>bf16=args.bf16,<br/>weight_decay=args.weight_decay,</pre> |
| <pre>run_name="2600-StarCoder-finetuned",     report_to="wandb",</pre>   |
| <pre>run_name="2600-StarCoder-finetuned", report_to="wandb",</pre>   |
| <pre>ddp_find_unused_parameters=False, optim="paged_adamw_8bit"</pre>  |
|  |

| 79 | <b>binaryninja</b> Update READ | ME.md                           | 0a62ac1 on Jun 2 | 🕑 76 comm  |
|----|--------------------------------|---------------------------------|------------------|------------|
|    | chat                           | Add example to fine-tune StarCo | oder for chat-b  | 4 months a |
|    | finetune                       | enabled qlora / 4 bit training  |                  | 4 months a |
| Ľ  | .gitignore                     | Add example to fine-tune StarCo | oder for chat-b  | 4 months a |
| Ľ  | LICENSE                        | Create LICENSE                  |                  | 5 months a |
| Ľ  | README.md                      | Update README.md                |                  | 4 months a |
| Ľ  | requirements.txt               | Update requirements.txt         |                  | 5 months a |
|    |                                |                                 |                  |            |

### 2600-StarCoder QLoRA 4bit ∉

I followed this guide Making LLMs even more accessible with bitsandbytes, 4-bit quantization and QLoRA by huggingface and implement the code in this repo to load the model in 4bit int and train using the methods outlined in the paper.

To make this work you're going to need the latese accelerate, transformers, and bitsandbytes libs.

pip install -q -U bitsandbytes

README.md

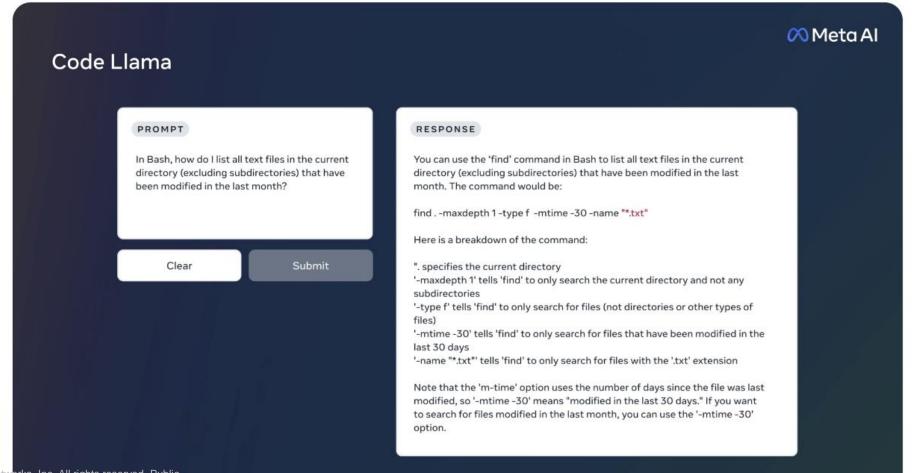
Q

pip install -q -U git+https://github.com/huggingface/transformers.git
pip install -q -U git+https://github.com/huggingface/peft.git
pip install -q -U git+https://github.com/huggingface/accelerate.git



## Introducing Code Llama, a state-of-the-art large language model for coding

August 24, 2023



# Takeaways

- Code Llama is a state-of-the-art LLM capable of generating code, and natural language about code, from both code and natural language prompts.
- Code Llama is free for research and commercial use.
- Code Llama is built on top of Llama 2 and is available in three models:
  - Code Llama, the foundational code model;
  - Codel Llama Python specialized for Python;
  - and Code Llama Instruct, which is fine-tuned for understanding natural language instructions.
- In our own benchmark testing, Code Llama outperformed state-of-the-art publicly available LLMs on code tasks

## Finetune codellama for SQL Task

### A complete guide to fine-tuning Code Llama

In this guide I show you how to fine-tune Code Llama to become a beast of an SQL developer. For coding tasks, you can generally get much better performance out of Code Llama than Llama 2, especially when you specialise the model on a particular task:

- I use the <u>b-mc2/sql-create-context</u> which is a bunch of text queries and their corresponding SQL queries
- A Lora approach, quantizing the base model to int 8, freezing its weights and only training an adapter
- Much of the code is borrowed from <u>alpaca-lora</u>, but I refactored it quite a bit for this

I used an A100 GPU machine with Python 3.10 and cuda 11.8 to run this notebook. It took about an hour to run. (I also tested that this code works on Colab Pro.)

\*This the corresponding notebook.

### <code>eval\_prompt = """You</code> are a powerful text-to-SQL model. Your job is to a

- You must output the SQL query that answers the question. ### Input:
- Which Class has a Frequency MHz larger than 91.5, and a City of licens  $\!$

#### ### Context:

CREATE TABLE table\_name\_12 (class VARCHAR, frequency\_mhz VARCHAR, city\_

### ### Response:

.....

model\_input = tokenizer(eval\_prompt, return\_tensors="pt").to("cuda")

### model.eval() with torch.no\_grad():

print(tokenizer.decode(model.generate(\*\*model\_input, max\_new\_tokens

And the model outputs:

SELECT class FROM table\_name\_12 WHERE frequency\_mhz > 91.5 AND city\_of\_

### https://ragntune.com/blog/guide-fine-tuning-code-llama

**Fine-tuning Notes** 

## Data parallel vs model parallel

## **Context size limits based on memory**

## Batch size / gradient / accumulation steps.

## Local training (Speed and context suffers)

# **Coreweave training (\$\$\$)**

Fine-tuning: Review

Define the set of tasks

**Collect the data (answers)** 

- Leverage existing tools to automate the generation of your dataset
- . Use larger language models (eg: GPT4) to generate training data
- . Get creative (ttp\_code\_finder.py)

Construct the prompt

• Read the paper! Implement system / init prompts to match training, add special tokens for end of sequence.instruction, and input

Tokenization

• Ensure prompt + code is under the max len. Don't poison your training with truncated samples.

Fine-tuning: PEFT/QLoRA

PEFT (Parameter-Efficient Fine-Tuning) and QLoRA are mechanisms aimed at training large language models (LLMs) on modest hardware.

1. PEFT:

PEFT is a subset of fine-tuning that emphasizes parameter efficiency. Unlike traditional fine-tuning where all coefficients of the model are altered, PEFT selectively tunes a subset of them, which significantly reduces the computational and memory requirements, making it a desirable approach for training large models like Falcon 7B where efficiency is crucial [1].

PEFT methods optimize the adaptation of LLMs to specific tasks, enhancing their performance and speed without demanding extensive GPU power and memory resources [2].

Some notable methods under PEFT include LoRA and QLoRA, which are specifically designed for finetuning LLMs in a parameter-efficient manner [3].

<sup>[1]</sup> https://www.analyticsvidhya.com/blog/2023/10/IIm-fine-tuning-with-peft-techniques/#:~:text=,7B%2C%20where%20efficiency%20is%20crucial

<sup>[2]</sup> https://www.analyticsvidhya.com/blog/2023/08/lora-and-qlora/#:~:text=Overview%20As%20we%20delve%20deeper,LLMs%29%20to%20specific%20tasks

<sup>[3]</sup> https://abvijaykumar.medium.com/fine-tuning-IIm-parameter-efficient-fine-tuning-peft-lora-qlora-part-1-571a472612c4#:~:text=,We%20will%20understnad%20how

Fine-tuning: PEFT/QLoRA

QloRA [1] is a method derived by combining Quantization, a weight reduction technique, with LoRA, a parameter-efficient fine-tuning technique. This combination facilitates the fine-tuning of large models with very resource-efficient utilization4.

Specifically, QLoRA enables the fine-tuning of 7-billion-parameter models such as Lllama-2 7B or Bloom 7B on a 16GB GPU. The process involves adding quantization to LoRA, which allows for the training of significantly larger and more powerful models while maintaining resource efficiency [2].

An example highlighted is the fine-tuning of a 7-billion-parameter Bloom model on a T4 16GB GPU in Google Colab, which was made possible through the application of QLoRA4.

[2] https://towardsai.net/p/machine-learning/qlora-training-a-large-language-model-on-a-16gb-gpu

<sup>[1]</sup> https://www.arxiv-vanity.com/papers/2305.14314/

AiAnalyzer.py >> output x 5000

```
dataset2 = load dataset("dyngnosis/function names v2", split="train")
35
36
     (variable) split data: Any | DatasetDict
# Split the dataset into training and evaluation sets
37
      split data = dataset2.train test split(test size=0.01)
38
      train dataset = split data["train"]
39
      eval dataset = split data["test"]
40
41
      base model = "codellama/CodeLlama-34b-hf"
42
      model = AutoModelForCausalLM.from pretrained(
43
44
          base model,
          load in 8bit=True,
45
          torch dtype=torch.float16,
46
          device map="auto",
47
48
      tokenizer = AutoTokenizer.from pretrained("codellama/CodeLlama-34b-hf")
49
```

### Setup prompt template. Tokenize and prep model for 8bit training

### LORA Config

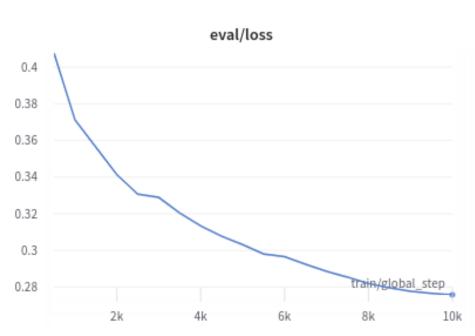
| 105 | <pre>config = LoraConfig(</pre>                  |
|-----|--|
| 106 | r=16,  |
| 107 | lora_alpha=16,                                   |
| 108 | <pre>target_modules=[</pre>                      |
| 109 | "q_proj",  |
| 110 | "k_proj",  |
| 111 | "v_proj",  |
| 112 | "o_proj",  |
| 113 |  |
| 114 | lora_dropout=0.05,                               |
| 115 | <pre>bias="none",</pre>                          |
| 116 | <pre>task_type="CAUSAL_LM",</pre>                |
| 117 | )  |
| 118 | <pre>model = get_peft_model(model, config)</pre> |

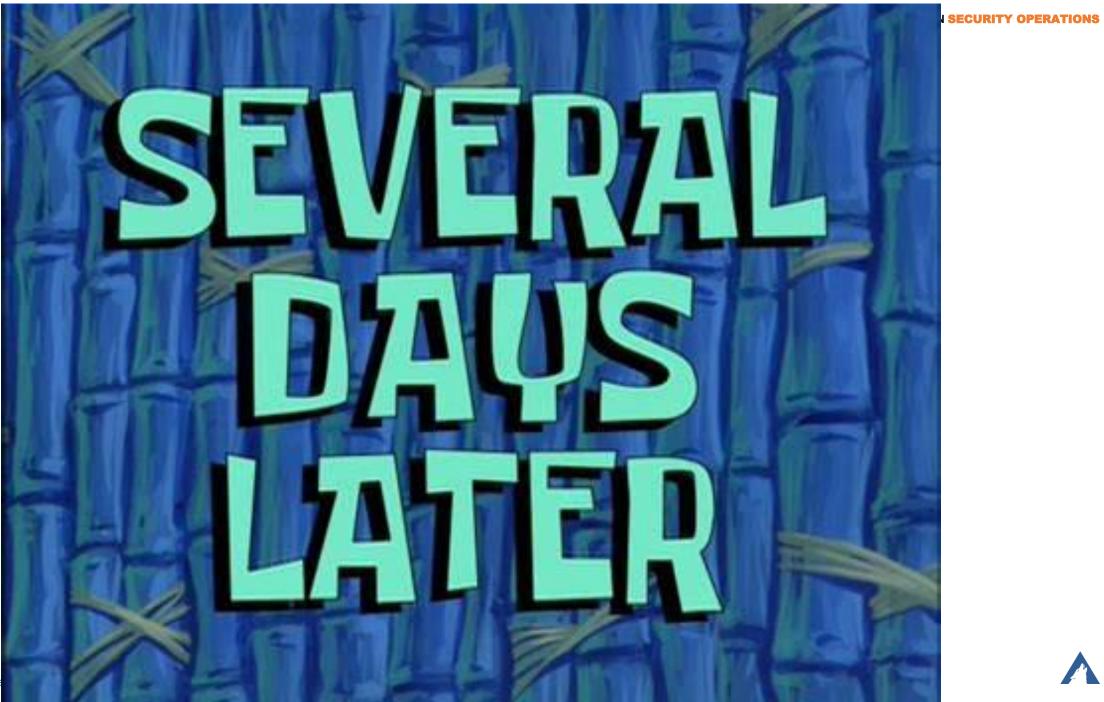
### **Setup training arguments**

| 135 | if t                           | orch.cuda.device_count() > 1:   |  |  |  |
|-----|--------------------------------|---|--|--|--|
| 136 |                                | # keeps Trainer from trying its own DataParallelism when more than 1 gpu is available                       |  |  |  |
| 137 | model.is_parallelizable = True |   |  |  |  |
| 138 |                                | <pre>model.model_parallel = True</pre>  |  |  |  |
| 139 |                                |   |  |  |  |
| 140 | batc                           | h_size = 4  |  |  |  |
| 141 | per_                           | device_train_batch_size = 1   |  |  |  |
| 142 | grad                           | ient_accumulation_steps = batch_size // per_device_train_batch_size   |  |  |  |
| 143 | outp                           | ut_dir = "decode-llama-4096-34b"  |  |  |  |
| 144 |                                |   |  |  |  |
| 145 | trai                           | ning_args = TrainingArguments(  |  |  |  |
| 146 |                                | <pre>per_device_train_batch_size=per_device_train_batch_size,</pre>   |  |  |  |
| 147 |                                | gradient_accumulation_steps=gradient_accumulation_steps,  |  |  |  |
| 148 |                                | #this was defaulting to 8 and causing OOM errors  |  |  |  |
| 149 |                                | <pre>per_device_eval_batch_size=per_device_train_batch_size,</pre>  |  |  |  |
| 150 |                                | warmup_steps=50,  |  |  |  |
| 151 |                                | <pre>max_steps=10000,</pre>   |  |  |  |
| 152 |                                | learning_rate=3e-4,   |  |  |  |
| 153 |                                | fp16=True,  |  |  |  |
| 154 |                                | logging_steps=10,   |  |  |  |
| 155 |                                | optim="adamw_torch",  |  |  |  |
| 156 |                                | <pre>evaluation_strategy="steps", # if val_set_size &gt; 0 else "no",</pre>                                 |  |  |  |
| 157 |                                | <pre>save_strategy="steps",</pre>   |  |  |  |
| 158 |                                | #eval_steps=100,  |  |  |  |
| 159 |                                | #set to 10 for debugging  |  |  |  |
| 160 |                                | eval_steps=500,   |  |  |  |
| 161 |                                | save_steps=500,   |  |  |  |
| 162 |                                | output_dir=output_dir,  |  |  |  |
| 163 |                                | <pre># save_total_limit=3,</pre>  |  |  |  |
| 164 |                                | <pre>load_best_model_at_end=False,</pre>  |  |  |  |
| 165 |                                | # ddp_find_unused_parameters=False if ddp else None,  |  |  |  |
| 166 |                                | <pre>group_by_length=True, # group sequences of roughly the same length together to speed up training</pre> |  |  |  |
| 167 |                                | <pre>report_to="wandb", # if use_wandb else "none",</pre>   |  |  |  |
| 168 |                                | <b>run_name=f"decodellama-{datetime.now().strftime('%Y-%m-%d-%H-%M')}",</b> # if use_wandb else None,       |  |  |  |
| 169 |                                |   |  |  |  |
| 170 |                                |   |  |  |  |

## GPUs go Brrrr....

| Every 2.0s: nvidia-smi<br>Fri Jun 2 16:57:47 2023 |                                 |  |                            |            |                   |                                |  |   |
|---|---------------------------------|--|----------------------------|------------|-------------------|--------------------------------|--|---|
| NVID  | IA-SMI                          | 515.6                                  | 5.01 Driv                  | ver Ve     | rsion:            | 515.65.01                      | CUDA Versio  | on: 11.7                                |
| GPU<br>Fan  | Name<br>Temp                    | Perf                                   | Persistence<br>Pwr:Usage/( |            | sus-Id            | Disp.A<br>Memory-Usage         | Volatile<br>GPU-Util                                   | Uncorr. ECC  <br>Compute M.  <br>MIG M. |
| =====<br>0<br>100%                                | ======<br>NVIDI4<br>86C         | ======<br>A GeFo<br>P0                 | rce Off<br>202W / 350      | •          |                   | 9:0B:00.0 On<br>iB / 24576MiB  | · · ·  | ======================================  |
| 1<br>71%  | NVIDIA<br>58C                   | A GeFo<br>P2                           | rce Off<br>165W / 390      |            |                   | 0:0C:00.0 Off<br>iB / 24576MiB | •  | N/A  <br>Default  <br>N/A               |
| Proc  | esses:<br>GI<br>ID              | CI<br>ID                               | PID                        | туре       | Ргосе             | ess name                       |  | GPU Memory                              |
|   |                                 |  |                            |            |                   |                                |  | Usage  <br>=======                      |
| =====<br>0<br>0                                   | =======<br>N/A<br>N/A           | ======<br>N/A<br>N/A                   | 4335<br>4476<br>9099       | <br>G<br>G | /usr/             | /lib/xorg/Xor<br>/bin/gnome-sh | g<br>nell  | ======================================  |
| =====<br>0  | N/A<br>N/A<br>N/A<br>N/A<br>N/A | N/A<br>N/A<br>N/A<br>N/A<br>N/A<br>N/A |                            |            | /usr/<br>Re<br>13 | /lib/xorg/Xor                  | g<br>ell<br>ePerProcess<br>/bin/python<br>08370,131072 | ======================================  |





# GPUs go Brrrrr....

#### ### Function Summary:

This function appears to be a part of a malware that is designed to perform a service-based attack. The function ta kes three parameters: a handle to a service, a pointer to a wide-character string, and an integer. The function fir st opens a service manager and then opens a service using the handle and the wide-character string. It then checks if the service is running and if it is not, it changes the service configuration and starts the service. The functi on then queries the service status and waits for the service to start. If the service does not start within a certa in time limit, the function calls another function that is likely to be a part of the malware.

### New Function Name:

ServiceStart

[INST]<<SYS>> You are an advanced malware reverse engineer capable of understanding decompiled C code and identify

You must output a descriptive ### Function Summary that describes the following decompiled code followed by a descr iptive ### New Function Name

Do not provide any extra information in the ### New Function Name. Only provide the name of the function. Do not in clude any extra information.[/INST]

### Code:

void \_\_fastcall fcn.004029c9(uint param\_1, uint param\_2, int32\_t param\_3)

```
int32_t iVar1;
int32_t iVar2;
int32_t iVar3;
int32_t iVar4;
bool bVar5;
uchar auStack_30 [4];
int32_t iStack 2c;
uint32_t uStack_18;
uchar auStack_c [4];
uint32_t uStack_8;
```

fcn.00408842();

return:

ing malicious functionality<</SYS>>

```
uStack 8 = *0x423008 ^ &stack0xfffffff;
bVar5 = false;
iVar1 = (* sym.imp.ADVAPI32.dll OpenSCManagerW)(0, 0, 0xf003f);
if (iVar1 != 0) {
   iVar2 = (* sym.imp.ADVAPI32.dll OpenServiceW)(iVar1, param 2, 0x2c);
   if (iVar2 != 0) {
       if (param_3 != -1) {
            iVar3 = (* sym.imp.ADVAPI32.dll ChangeServiceConfigW)
                              (iVar2, 0xffffffff, param 3, 0xffffffff, 0, 0, 0, 0, 0, 0, 0);
            bVar5 = iVar3 != 0;
       iVar3 = (* sym.imp.ADVAPI32.dll QueryServiceStatusEx)(iVar2, 0, auStack 30, 0x24, auStack c);
       if (((iVar3 != 0) && (iStack 2c != 1)) &&
           (iVar3 = (* sym.imp.ADVAPI32.dll ControlService)(iVar2, 1, auStack 30), iVar3 != 0)) {
           iVar3 = (* sym.imp.KERNEL32.dll GetTickCount)();
           if (15000 < uStack 18) {
                uStack 18 = 15000;
            if (iStack 2c != 1) {
                while( true ) {
                    (* sym.imp.KERNEL32.dll Sleep)(uStack 18);
                    iVar4 = (* sym.imp.ADVAPI32.dll QueryServiceStatusEx)(iVar2, 0, auStack 30, 0x24, auStack c
                    if (iVar4 == 0) break;
                    if (iStack 2c == 1) {
                       bVar5 = true;
                       break;
                   iVar4 = (* sym.imp.KERNEL32.dll GetTickCount)();
                   if ((15000 < iVar4 - iVar3) || (iStack 2c == 1)) break;
            if (!bVar5) {
                fcn.00402821();
        (* sym.imp.ADVAPI32.dll CloseServiceHandle)(iVar2);
    (*_sym.imp.ADVAPI32.dll_CloseServiceHandle)(iVar1);
```

### ### Function Summary:

This function appears to be a part of a malware that is designed to perform a service-based attack. The function ta kes three parameters: a handle to a service, a pointer to a wide-character string, and an integer. The function fir st opens a service manager and then opens a service using the handle and the wide-character string. It then checks if the service is running and if it is not, it changes the service configuration and starts the service. The function then queries the service status and waits for the service to start. If the service does not start within a certa in time limit, the function calls another function that is likely to be a part of the malware.

### New Function Name:

ServiceStart

## GPUs go Brrrrr....



#### (iVar6 != 0x6f) iVarll = iVar4; if (iVar6 == θ) { for (iVar6 = \*(int \*)(iVar11 + 0x10); iVar6 != 0; iVar6 = \*(int \*)(iVar6 + 8)) { iVar8 = Ordinal\_12(\*(undefined4 \*)(\*(int \*)(iVar6 + 0xc) + 4)); if (iVar8 != 0) { FUN 004202d0(iVar8); for (iVar6 = \*(int \*)(iVar11 + 0x14); iVar6 != 0; iVar6 = \*(int \*)(iVar6 + 8)) { iVar8 = Ordinal\_12(\*(undefined4 \*)(\*(int \*)(iVar6 + 0xc) + 4)); if (iVar8 != 0) { FUN 004202d0(iVar8): piVar2 = (int \*)(iVar11 + 8); iVar11 = \*piVar2; } while (\*piVar2 != 0); FUN\_00415630(iVar4); break: FUN\_00415630(iVar4); uVar12 = uVar12 + 1; } while (uVar12 < 3); iVar4 = 0; if (0 < DAT 004346f0) { do { NetworkShareEnumerator((char \*)(&DAT\_00434700)[iVar4]) iVar4 = iVar4 + 1; } while (iVar4 < DAT\_004346f0);</pre> bVar3 = false: do { Sleep(240000); if ((5 < \_DAT\_004340e4) & (iVar4 = 0, 0 < DAT\_004346f0)) {</pre> NetworkShareEnumerator((char \*)(&DAT 00434700)[iVar4]); iVar4 = iVar4 + 1; } while (iVar4 < DAT 004346f0);</pre> iVar4 = DAT\_004342ec; piVar2 = \*(int \*\*)(\*(int \*)((int)ProcessEnvironmentBlock + 0xc) + 0x10); piVar13 = piVar2; do { piVar13 = (int \*)\*piVar13; if ((piVarl3[6] != 0) δδ (iVar6 = lstrcmpiW((LPCWSTR)piVarl3[θxc],L"kernel32.dll"), iVar6 == θ)) break; } while (piVar2 != piVar13); DAT 00436f0c = (code \*)HashTableLookup(piVar13[6],0x1f3ffddb,0xd,0x7d10e76b); iVar6 = (\*\_DAT\_00436f0c)(&DAT\_0042f0fc); DAT 004340c4 = (code \*)HashTableLookup(iVar6,0xadc9faf4,0x17,0x7e990fb0); (\* DAT 004340c4)(iVar4.SDAT 004342f0.1,0xffffffff); DAT 004342ec = 0;if (!bVar3) { bVar3 = true; iVar4 = FUN\_00419b00(); uVar9 = extraout\_ECX; uVar10 = extraout EDX: if (iVar4 != 0) { ExecuteCmdShell(&DAT 0042f040,0x91); uVar9 = extraout\_ECX\_00; uVar10 = extraout EDX 00; , FUN 00419bc0(uVar9,uVar10); while( true );

## GPUs go Brrrrr....

### Function Summary:

This function is a malicious function that is designed to perform a variety of malicious activities, including:

- Loading and executing a kernel32.dll function
- Enumerating logical drives and creating threads to scan for specific files
- Enumerating network resources and creating threads to scan for specific files.
- Enumerating network shares and creating threads to scan for specific files
- Creating a new thread to perform a malicious activity

### New Function Name:

MaliciousFunction

|   | Decompile: entry [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519968d9fda                     | Decompile: entry [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519968d9fda04f2a262f4   | Decompile: entry [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c] 😑 😐 🗴 |
|---|--|--|---|
|   | Eile Edit Navigation Search Select Help  | Eile Edit Navigation Search Select Help  |   |
|   | ■ ( ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●  | ■ (  |   |
|   | C, Decompile: entry - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c)                     |  |   |
|   | 1<br>2//# WARNING: Globale starting with ' ' ovarian smaller symbols at the same address #/                  |  |   |
|   | 3  |  | 193 BVar8 = ShellExecuteExW((SHELLEXECUTEINFOW *)auStack_36c);  |
|   | 4 void entry(void)   | 90 uVar16 = FUN 004010a4(extraout ECX 03,(int)((ulonglong)uVar16 >> 0x20),0x42f110,0xd,  |   |
| S           |  | $\frac{1}{2} = \frac{1}{2} $ | 196 ExitProcess (0x65);   |
|   | 7 int *piVarl;<br>8 code *ncVar2   | 93 for $(y y ^2) = DAT 00436920; 9 < y y ^2 = y y ^2 = y y ^2$   |   |
| No.         No. <td>9 LPOWSTR pWVar3;</td> <td>94 uVarl2 = uVarl4 / 10;<br/>95 }</td> <td>199 }</td>  | 9 LPOWSTR pWVar3;  | 94 uVarl2 = uVarl4 / 10;<br>95 }   | 199 }   |
| No.         No. <td>10 bool bVar4;<br/>11 int iVar5;</td> <td>96 DAT_00436920 = uVarl4;</td> <td></td>  | 10 bool bVar4;<br>11 int iVar5;  | 96 DAT_00436920 = uVarl4;  |   |
|   |  | 98 EIN 00418d90(extraout ECX 05 uVar14)  | 202 (*DAT_0042fef4) (auStack_104, &DAT_00425170);   |
| Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm         Bit Norm           Bit Norm         Bit Norm         Bit Norm         Bit Norm   |  | 99 _DAT_00436f80 = DAT_0042f510;   |   |
| Set Number         Set Num   | 15 ushort *puVar9;   |  | 205 do {  |
|   | 17 undefine data vite 17.0   | 102 DA1_00421314 = 0;  |   |
|   | 18 HANDLE hHeap;   | <pre>104 uVar16 = FUN_004010a4(extraout_ECX_06,extraout_EDX_00,0x42f124,0xd, (undefined *)&amp;DAT_0042f47c,3);</pre>  | 208 } while (uVarl2 < 5);   |
|   | 20 undefined4 extraout_ECX;  | 105 Uvario = Pow_00401044(extraout_extraout_f(uting tong uvario >> 0x20), 0x421010, 0xc,   | 210 FUN_0041e130();   |
|   | 21 undefined4 extraout_ECX_00;<br>22 undefined4 extraout_ECX_00;   | <pre>uVar16 = FUN_004010a4(extraout_ECX_08,(int)((ulonglong)uVar16 &gt;&gt; 0x20),0x42f42c,0xc,</pre>  | 211 }   |
|   | 23 undefined4 extraout_ECX_02;   | $\frac{100}{100} = \frac{100}{100} $   | 213 _DAT_004342a8 = (code *)FUN_00415690(0x76a4fa0b,&DAT_0042f0fc,0xb);   |
|   | 24 undefined4 extraout_ECX_03;   | 110 (undefined *)&DAT_0042f47c,3);   | 214 iVar5 = (*_DAT_004342a8)(0x1f0001,0,pWVar3);  |
|   | <pre>26 undefined4 extraout_ECX_05;</pre>  | 111 PM_00436450 = 0.   | 216 if (iVar5 != 0) {   |
|   | 2/ undefined4 extraout_ECX_06;   | 113 DAT 0042f510 = 0;  |   |
|   | 29 undefined4 extraout_ECX_08;   | $\frac{114}{115} \text{ DAT } 00421516 = 0;$   | 219 }   |
|   | 30 undefined4 extraout_ECX_09;   | 116 DAT_00436920 = 0;  | 220DAT_00434240 = (code *)FUN_00415690(0xed3e7aaa,50AT_0042f0fc,0xd);   |
|   | 32 undefined4 extraout_ECX_11;   | III FON_0040164(extraout_ECX_12, extraout_EDX_01, 0.421440, 0.42, (underlined *) adv1_00421472, 3);  | 222 FUN_00402030();   |
|   | 33 undefined4 extraout ECX 12:   | 119 FUN_004010a4(extraout_ECX_13,extraout_EDX_02,0x42f0e8,0xc,(undefined *)&DAT_0042f47c,3);   |   |
|   | 35 int iVaril;   | 120 pdstake_5/0 = DM_004214/559 (0x2ba92ec1,&DAT_0042f14c,0xc);  | 225 bVar4 = FUN_004192e0();   |
|   | 36 undefined4 extraout_ECX_14;<br>37 undefined4 extraout ECX 15;   | 122 UVar12 = 0;  |   |
|   | 38 undefined4 extraout_ECX_16;   | 124 do {   | 228 return;   |
| a)         a)<  | 40 undefined4 extraout_ECX_18;   |  |   |
|   | 41 undefined4 extraout_ECX_19;   | 127 auStack 36c = (undefined [4])0x0;  |   |
| Bit         Bit <td>43 uint uVarl2;</td> <td></td> <td>233 (*_DAT_004342b0)(uVarl0,0x80);</td>  | 43 uint uVarl2;  |  | 233 (*_DAT_004342b0)(uVarl0,0x80);  |
| 0           | 44 underined4 extraout EDX 00;   | 130 uVarl2 = uVarl4 + 1;   |   |
|   | <pre>46 undefined4 extraout_EDX_02;</pre>  | 131 Uvall4 = Uvall2;   | 236 );  |
| B)         B)<  | 4/ char *pcVarI3;  | <pre>133 piVar1 = *(int **)(*(int *)((int)ProcessEnvironmentBlock + 0xc) + 0x10);</pre>  |   |
| and methods         bits/s (unt "privates         bits/s (unt "privates         bits/s (unt "privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates           bits/s (unt privates         bits/s (unt privates         bits/s (unt privates         bits/s (unt privates <td><pre>49 int *piVar15;</pre></td> <td>195 do {</td> <td>239 FUN_004198a0(&amp;DAT_0042f160,0x2b5);</td>  | <pre>49 int *piVar15;</pre>  | 195 do {   | 239 FUN_004198a0(&DAT_0042f160,0x2b5);  |
| Bit Market 2010         If Win = (-Control (  | 50 undefined8 uVar16;<br>51 SIZE T dwBytes;  | 136 piVar15 = (int *)*piVar15;   |   |
| Bit Market 2010         If Win = (-Control (  | 52 undefined *puVar17;   | 138 break  | 242 DAT_00436930 = iVarll * 2 + 2;  |
| Bit Market 2010         If Win = (-Control (  | 53 undefined *pustack_3/0;<br>54 undefined auStack_36c [4];  | 139 / White (piver: := piver:);<br>140 / DRT 00436for = (cod ))[UN 00415700(n)(on15[6] 0v1f3ffddb 0vd 0v7d10n76b);   | 244 DAT 0043692c = (LPCSTR)FUN 00415650(0x1fa0);  |
| 50         100         1000729         10000729         1000729         100072  | 55 undefined auStack_368 [16];   | 141 IVar5 = (*_DAT_00436f0c)(&DAT_0042f0fc);   | 245 FUN_00415610(0x1000);<br>246 USENE - 02100  |
| Note action (34) [31]         144         19/1 =  | 57 void *pvStack 348;  | 142 DAT_0042550 = (* DAT_00434254 )):  | 247 DVar7 = 8;  |
| 60         Model method:         100 <t< td=""><td>58 WCHAR aWStack_334 [8];<br/>59 undefined auStack_324 [20]:</td><td>144 iVar5 = 5;</td><td></td></t<>   | 58 WCHAR aWStack_334 [8];<br>59 undefined auStack_324 [20]:  | 144 iVar5 = 5;   |   |
| 0.         undefined usitad_30: [2];<br>undefined usitad_30: [3];<br>undefined usitad_30: [3];<br>u | 60 WCHAR aWStack_310 [2];  | 146 puStack 370 = (undefined *)((uint)puStack 370 ^ 0x4a108947);   | 250 if (iVar5 == 1) {   |
| Bill and state 2 [06] (add state 2 [1])         Bill and state 2 [06] (add s  | 61 undefined auStack_30c [2];  | 147 iVar5 = iVar5 + -1;  |   |
| 66         messet(DAT_0043404, 0, 0118);         72         0 <sup>1</sup> µVr13 = (11 *)*pivr15];         72         74   | 63 undefined auStack_104 [248];  | 140 p wilter (14/15/1-0/)  | <pre>253 pcVarl3 = "<br/>";</pre>   |
| 66         messet(DAT_0043404, 0, 0118);         72         0 <sup>1</sup> µVr13 = (11 *)*pivr15];         72         74   |  | 150 piVarl5 = piVarl;<br>151 do {  | zore Warto = extraout_ct.to;<br>25 }  |
| 13         if (pixels)         13   | bb memset(&DAI_004340e4,0,0X118);  | 112 piVar15 = (int *)*piVar15;   | 256 else {  |
| 69         War16 = FLW_004004(ettraout_ECX, outfout_OUX, OUX, Outfout_OUX, OUX, OUX, OUX, OUX, OUX, OUX, OUX,  | 68 GetVersionExW((LPOSVERSIONINFOW)&DAT 004340e0);   | <pre>153 if ((piVar15[6] != 0) &amp;&amp; (iVar5 = lstrcmpiW((LPOWSTR)piVar15[0xc],L"kernel32.dll"), iVar5 == 0)) 154 break.</pre>   | 258 if (iVar5 != 3) goto LAB_004186f9;  |
| 71  | <pre>69 uVar16 = FUN_00010a4(extraout_ECX,extraout_EDX,0x42f0fc,0xd,(undefined *)&amp;DAT_0042f47c,3);</pre> | <pre>155 } while (piVarl != piVarl5);</pre>  | 259 DVar7 = 0;  |
| 22       DAT_00432f80 = DAT_0042f510;       DAT_0042f510 = DAT_0042f510;  | 71 (undefined *)&DAT 0042f47c.3):  |  | 261 FUN_0041bc60();   |
| 74       DAT_0042F518 = 0;       Control Contrelectica Control Control Contrelectica Control Control  | 72 DAT 00425510 - DAT 00425514   | 158 DAT_00434178 = FUN_00415700(iVar5,0xcaa2a74c,0x10,0x5bfa915f);   |   |
| /> // DAT_0042f514 = 0;       // DAT_0042f542ee4)(mWar6,austack_380;;       // DAT_0042f542ee4)(mWar6,austack_370;;       // DAT_0042f542ee4)(mWar6,austack_370;;       // DAT_0042f5418,0xe);       // DAT_0042f5418,0xe   | 74DAT_0042f518 = 0;  | 160 DAT 0042fee4 - (code *)EIN 00415690(0v9h4c2d0 &DAT 0042f0e8 0v13)  | 264 uVarl0 = extraout_ECX_18;   |
| 77       Warl6 = FLN_004010a4(extraout_ECX_01,(int)((ulonglong)uVarl6 >> 0x20),0x42f138,0x4,       163       (iVar5 = FLN_00419b00(), iVar5 1 = 0)) {       267       Warl6 = extraout_ECX_01,(int)((ulonglong)uVarl6 >> 0x20),0x42f138,0x4,       163       (iVar5 = FLN_00419b00(), iVar5 1 = 0)) {       268 LBA_004186f5:       278 LBA_004186f5:   | 70 DIT 00100000 0  | 161 (* DAT 0042fee4)(pWVar6,auStack 368);  |   |
| Me         Underline *Jubla_0042/f4/c,3);         Int Ood32/f12         Code *FLW_00412609(x)2.13bb3_00AT_0042/f14(x,0x);         Zoe LAB_00412609(x)2.13bb3_00AT_0042/f14(x,0x);           0 AT_0042/f510 = 0AT_0042/f510;         166         Apt_0042/f10(x)005000(x).1xb1, x(x,0x)(x)(x)(x)(x)(x)(x)(x)(x)(x)(x)(x)(x)(x  | <pre>77 uVar16 = FUN_004010a4(extraout_ECX_01, (int) ((ulonglong)uVar16 &gt;&gt; 0x20), 0x42f138, 0xd,</pre> | 163 (ivar5 = FIN_00419b0(), ivar5 != 0)) {   | 267 uVar10 = extraout_ECX_19;   |
| 80       DAT_0042f510 = DAT_0042f514;       166       6puStack_370);       270       FLN_0041709();         10       DAT_0042f514 = 0;       167       DVar7 = GetModuleFileNemeW(!+MODULE)0x0,aWStack_310,0t104);       271       DAT_0043693e1;       272       DAT_0043693e2;         10       DVar7 = 0;       168       if (Dvar7 = 0) {       273       pCV4  | 78 (undefined *)&DAT_0042f4/c,3);<br>79 DAT_00436f80 = DAT_0042f510;   | <pre>164 DAT_00434214 = (code *)FUN_00415690(0x2a13bb3,&amp;DAT_0042f418,0xe);</pre>   |   |
| 82       DAT_0042f514 = 0;       272       FUN_0041e650();         9       if (Dvar7 = 0) {       273       pc/v2 = (code +)swi(3);         9       u/arife + bit (dualphane(+traout_ECX_02, (int)((ulonglong)u/arife >> 0x20), 0x42f418, 0xd,       169       if (Dvar7 = 0) {         84       (undefined +)6DAT_0042f47c, 3);       170       >       274       (*pcVar2)();         9       DAT_0042f510 = DAT_0042f510;       171       else {       275       return;         9       DAT_0042f510 = DAT_0042f510;       172       i/ar5 = 0;       276   | 80 DAT_0042f510 = DAT_0042f514;  | 66 SpuStack_370)   | 270 FUN_00417d90();   |
| 83       uVar16 = FUN_004010a4(extraout_ECX_02, (int) ((ulongiong)uVar16 >> 0x20), 0x42f418, 0xd, 106       106       FUN_0041b960(0);       273       pcVar2 = (code *)swi(3);         84       (undefined *)&DAT_0042f47c, 3);       170       }       274       (*pcVar2) ();         85       DAT_00436f80 = DAT_0042f510;       171       else {       275       retur;         86       DAT_0042f510;       171       else {       276       276         97       DAT_0042f510;       172       iVar5 = 0;       276  |  | 167 DVar7 = GetModuleFileNameW((HMODULE)0x0,aWStack 310,0x104);  |   |
| $172$ $1/ar^5 = 0;$   | 83 uVar16 = FUN_004010a4(extraout_ECX_02, (int)((ulonglong)uVar16 >> 0x20),0x42f418,0xd,                     | $\frac{1}{169} = \frac{1}{FUN_0041b980(0);}$   | 273 pcVar2 = (code *)swi(3);  |
| $172$ $1/ar^5 = 0;$   | 84 (undefined *)&DAT_0042f47c,3);<br>85 _DAT_00436f80 = DAT_0042f510;  | 170 }  | 275 return;   |
|   |  | 172 iVar5 = 0;   | 276 }   |
|   |  | 173 iVarl1 = 0;  |   |

```
200
                         208
                       } while (uVar12 < 5);</pre>
                  209 if (5 < _DAT_004340e4) {
                        FUN 0041e130();
                  210
                  211
                       }
                  212
                       pWVar3 = DAT 00436928;
                       DAT 004342a8 = (code *)FUN 00415690(0x76a4fa0b,&DAT 0042f0fc,0xb);
                  213
                  214 iVar5 = (* DAT 004342a8)(0x1f0001,0,pWVar3);
                  215
                       pWVar3 = DAT 00436928;
                  216
                       if (iVar5 != 0) {
                  217
                                         /* WARNING: Subroutine does not return */
                  218
                         ExitProcess(0);
                  219
                       }
                  220
                       DAT 00434240 = (code *)FUN 00415690(0xed3e7aaa,&DAT 0042f0fc,0xd);
                  221
                       (* DAT 00434240)(0,0,pWVar3);
                  222 FUN 00402030();
                  223 FUN 00415690(0xa38e7de7,&DAT 0042f0fc,0x1b);
                  224 FUN 00415690(0x97622898,&DAT 0042f0fc,10);
                  225 bVar4 = FUN 004192e0();
                       iVar5 = CONCAT31(extraout var,bVar4);
                  226
                  227
                       if (iVar5 == 0) {
                  228
                         return;
                  229
                       }
                       DAT 0042f880 = (code *)FUN 00415690(0x48b64e68,&DAT 0042f0fc,0x12);
                  230
                  231
                       uVarl0 = (* DAT 0042f880)();
                       DAT 004342b0 = (code *)FUN 00415690(0xf0df0lcc,&DAT 0042f0fc,0x11);
                  232
                  233
                       (* DAT 004342b0)(uVarl0,0x80);
                  234
                       PTR DAT 0042f038 = (undefined *)FUN 00415650(0x100);
                       wsprintfW((LPWSTR)PTR DAT 0042f038,L".%s.%s%s",DAT 00436928,L"anobtanium@tutanota.com",L".pandora"
                  235
                  236
                                );
                  237
                       iVarll = FUN 00419b00();
                  238
                       if (iVarll != 0) {
                  239
                         FUN 004198a0(&DAT 0042f160,0x2b5);
                  240
                       }
                  241 iVarl1 = lstrlenW((LPCWSTR)PTR DAT 0042f038);
                  242
                       DAT 00436930 = iVarll * 2 + 2;
                  243 DAT 00434200 = FUN 00415690(0xac74699e,&DAT 0042f0fc,0xc);
                  244 DAT 0043692c = (LPCSTR)FUN 00415650(0x1fa0);
                  245 FUN 00415610(0x1000);
                  246
                       dwBytes = 0x100;
                  247
                       DVar7 = 8;
                  248
                       hHeap = GetProcessHeap();
                       lpMultiByteStr = (LPSTR)HeapAlloc(hHeap,DVar7,dwBytes);
                  249
                       if (iVar5 == 1) {
                  250
                  251
                        FUN 0041bc60();
                  252
                         FUN_0041be00(extraout_ECX_15,&DAT_00436720);
                         pcVarl3 = "<br></strong></div></body>";
                  253
Information Classifica
                  254
                         uVar10 = extraout ECX 16;
```

255 }

### Summary

- 1. Initializing and manipulating some global variables.
- 2. Gathering version information about the operating system.
- 3. Performing some repetitive calls to `FUN\_004010a4` with different parameters.
- 4. Checking certain conditions based on the operating system version and other data.
- 5. Interacting with the `kernel32.dll` module.
- 6. Formatting strings, potentially building paths or filenames.
- 7. Conditionally executing certain blocks of code based on computed or retrieved values.
- Making use of the `ShellExecuteExW` function, which suggests it may be launching external processes or commands.
- 9. Interacting with the system's heap for memory allocations.
- 0. Possibly interacting with the system's registry (based on the presence of strings like
  - `"Software\\Microsoft\\Windows\\CurrentVersion\\Run"`).
- 1. Formatting and building an email address.
- 2. Using **`HeapAlloc`** to allocate memory on the heap.
- 3. Converting wide characters to multibyte character strings.
- 4. Handling various conditions based on the value of the variable `iVar5`.

Hey there it looks like you're trying to reverse engineer a function...



A

### **Step-by-Step Description**

- 1. Initializes a memory region (`memset(&DAT\_004340e4,0,0×118)`).
- 2. Retrieves the version of the operating system using `GetVersionExW`.
- 3. Calls the function `FUN\_004010a4` multiple times with various parameters.
- 4. Manipulates and checks certain global variables and data points.
- 5. Interacts with the `kernel32.dll` module and retrieves function pointers.
- 6. Formats and builds strings, especially paths or filenames.
- 7. Checks for certain conditions, including operating system version and other computed data.
- 8. Potentially executes shell commands or processes using `ShellExecuteExW`.
- 9. Retrieves the command line of the current process using `GetCommandLineW`.
- 10. Makes additional calls to external functions (`FUN\_00415690`, `FUN\_0041c220`, etc.).
- 11. Builds an email address using `wsprintfW`.
- 12. Checks certain conditions and interacts with the system's registry.
- 13. Allocates memory on the heap using `HeapAlloc`.
- 14. Converts wide character strings to multibyte strings using `WideCharToMultiByte`.
- 15. Further manipulates and processes strings and data.
- 16. Makes additional function calls and possibly ends the process using `ExitProcess`.

Hey there it looks like you're trying to reverse engineer a function...



Hey there it looks like you're trying to reverse engineer a function...



### **Suggested Function Names**

Based on the operations performed, some suitable new names for the function could be:

- `initializeAndExecuteShellCommands`
- `performSystemOperations`
- \* `handleRegistryAndShellCommands`
- \* `processKernel320perations`

### **Observations and Recommendations**

- The function appears to be part of a larger program, and without additional context, its exact purpose remains somewhat obscured.
- The repetitive calls to `FUN\_004010a4` and interaction with `kernel32.dll` suggest this might be some initialization or setup routine.
- The usage of `ShellExecuteExW` and interaction with the registry implies that this function has the potential to make significant changes to the system.
- It's recommended to analyze the external functions like `FUN\_004010a4`, `FUN\_00415690`, etc., to get a clearer picture of the overall functionality and goal of this function.

Hey there it looks like you're trying to reverse engineer a function...



| uint *puVar1;  |
|--|
| int32_t iVar2;   |
| uint uStack72;   |
| uint uStack68;   |
| uint uStack64;   |
| uint uStack60;   |
| uint uStack56;   |
| uint uStack52;   |
| uint uStack48;   |
| uint uStack44;   |
| uint uStack40;   |
| uint uStack36;   |
| uint uStack32;   |
| uint uStack28;   |
| uint uStack24;   |
| uint uStack20;   |
| uint uStack16;   |
| uint uStack12;   |
|  |
| The second s |

if ((arg\_ch | arg\_10h uStack56 = \*arg uStack52 = arg\_1 uStack48 = arg 1uStack44 = arg\_1 uStack40 = arg 1uStack36 = arg 1 uStack32 = arg\_18h uStack28 = arg\_18h[7]; uStack72 = 0x61707865; uStack20 = \*arg 14h; uStack16 = arg\_14h[1]; uStack12 = arg\_14h[2]; uStack68 = 0x3320646e; uStack64 = 0x79622d32; uStack60 = 0x6b206574; uStack24 = 0; sub.msvcrt.dll\_memset(arg\_8h, 0, arg\_ch); fcn.004028f0(arg\_8h, arg\_ch, arg\_10h); puVar1 = &uStack72; iVar2 = 0x40;do { \*puVar1 = 0; puVar1 = puVar1 + 1; iVar2 = iVar2 + -1; } while (iVar2 != 0);

return 0;

Close Analyze

### Recognizes encryption constants

uint \_\_cdecl fcn.004038f0(uint arg\_8h, uint32\_t arg\_ch, uint32\_t arg\_10h, uint \*arg\_14h, uint \*arg\_18h)

| Step-by-step | description |
|--------------|-------------|
|--------------|-------------|

|  | <ul> <li>OITS</li> </ul> |
|--|--------------------------|
| <ol> <li>The function takes five arguments: arg_8h, arg_ch, arg_10h, arg_14h and<br/>arg_18h.</li> </ol>                         | • nam                    |
|  | <ul> <li>size</li> </ul> |
| <ol><li>arg_8h is a memory address, arg_ch and arg_10h are integers, arg_14h and<br/>arg_18h are pointers to integers.</li></ol> | • is-p                   |
| 3. First, the function checks if arg_ch or arg_10h are not equal to zero.  | <ul> <li>real</li> </ul> |
|  | <ul> <li>nore</li> </ul> |
| 4. If either arg_ch or arg_10h is not zero, function proceeds to do the following  | • stac                   |
| operations. Otherwise, it skips to the end, returning 0.   | • call                   |
| 5. It pulls integer values from the memory locations pointed to by arg_18h and   | <ul> <li>cost</li> </ul> |
| arg_14h and stores them in local variables (uStack56 to uStack28 and   | 100000                   |
| uStack20 to uStack12 respectively)   | • cc: 3                  |
| 6. It sets uStack72 to 0x61707865 (which is "expa" in ASCII form), uStack68 to   | <ul> <li>bits</li> </ul> |
|  | <ul> <li>type</li> </ul> |
| 0x3320646e ("2 dn" in ASCII), uStack64 to 0x79622d32 ("-2yb" in ASCII), and  | • nbb                    |
| uStack60 to 0x6b206574 (" te k" in ASCII)  | • is-li                  |
| 7. It sets uStack24 to 0   | • nins                   |
| 8. The function then clears memory starting from $\arg\_8h$ and extending to $\arg\_ch$  | <ul> <li>eda</li> </ul>  |
| through memset().  | • ebb                    |
| 9. Then fcn. 004028f0 is executed with arg_8h, arg_ch, arg_10h as its  |                          |
| arguments.   | <ul> <li>sign</li> </ul> |
| 10. After fcn. 004028f0 is executed, function clears variables in the stack from   | int3                     |
|  | • min                    |
| uStack72 to uStack28 and uStack20 to uStack12, by setting them to zero.  | • max                    |
| <ol> <li>Finally, the function returns 0.</li> </ol>   | <ul> <li>inde</li> </ul> |
|  |                          |

### Summary

This function seems to perform some form of initial setup or configuration and then cleanup. It seems to be doing some cryptographic operations given the constants that are in ASCII form which resembles constants for Salsa20 / ChaCha encryption algorithms. It doesn't directly seem to engage in networking, lateral movement, or direct exploitation.

### Function Name:

prepare\_and\_cleanup\_crypto\_config

### Interesting Constants

- "expa" (0x61707865)
- "2 dn" (0x3320646e)
- "-2yb" (0x79622d32)
- "te k" (0x6b206574)

### Interesting Memory Addresses to dereference

It would be interesting to see the values stored at these memory addresses:

arg\_8harg\_14h

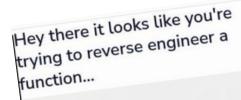
• arg\_18h

### Function Info

- offset: 4208880
  name: fcn.004038f0
- name: icn.004038f
   size: 211
- is-pure: false
- realsz: 211
- noreturn: false
- stackframe; 88
- calltype: cdecl
- cost 73
  - c: 3
- bits: 32
- type: fcnnbbs: 4
- is-lineal: true
- ninstrs: 65
- edges: 5
- ebbs: 1
- signature: fcn.004038f0 (int32\_t arg\_8h, int32\_t arg\_ch, int32\_t arg\_ int32\_t arg\_14h, int32\_t arg\_18h);
- minbound: 4208880
- maxbound: 4209091
- indegree: 0
- outdegree: 2
- nlocals: 16
- nargs: 5
- difftype: new

### Callrefs

| Addr    | Туре | At      |  |
|---------|------|---------|--|
| 4209083 | CODE | 4208899 |  |
| 4327813 | CALL | 4209033 |  |
| 4204784 | CALL | 4209052 |  |
| 4209072 | CODE | 4209081 |  |





# LLMs in the Lab

LLM empowered Malware Analysis.

### **Function Name:**

prepare\_and\_cleanup\_crypto\_config

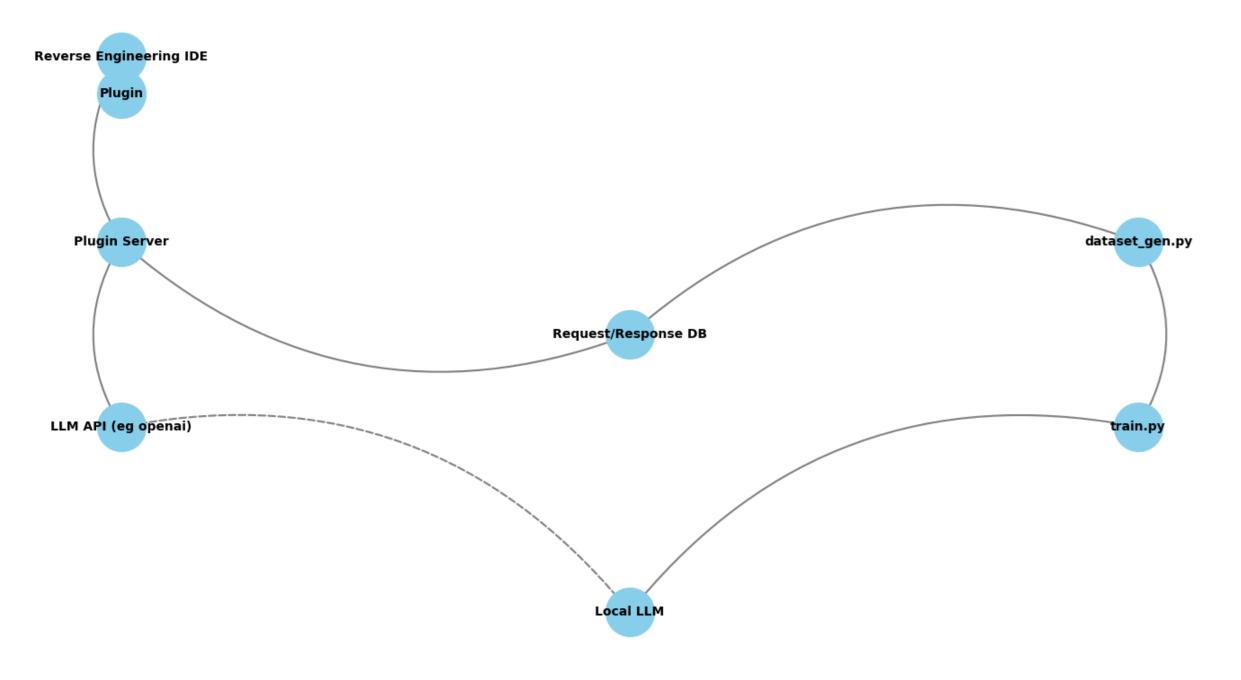
### Summary

This function seems to perform some form of initial setup or configuration and then cleanup. It seems to be doing some cryptographic operations given the constants that are in ASCII form which resembles constants for Salsa20 / ChaCha encryption algorithms. It doesn't directly seem to engage in networking, lateral movement, or direct exploitation.

## Step-by-step description

- 1. The function takes five arguments: arg\_8h, arg\_ch, arg\_10h, arg\_14h and arg\_18h.
- 2. arg\_8h is a memory address, arg\_ch and arg\_10h are integers, arg\_14h and arg\_18h are pointers to integers.
- 3. First, the function checks if arg\_ch or arg\_10h are not equal to zero.
- 4. If either arg\_ch or arg\_10h is not zero, function proceeds to do the following operations. Otherwise, it skips to the end, returning 0.
- 5. It pulls integer values from the memory locations pointed to by arg\_18h and arg\_14h and stores them in local variables (uStack56 to uStack28 and uStack20 to uStack12 respectively)
- 6. It sets uStack72 to 0x61707865 (which is "expa" in ASCII form), uStack68 to 0x3320646e ("2 dn" in ASCII), uStack64 to 0x79622d32 ("-2yb" in ASCII), and uStack60 to 0x6b206574 (" te k" in ASCII)
- 7. It sets uStack24 to 0
- The function then clears memory starting from arg\_8h and extending to arg\_ch through memset().
- 9. Then fcn.004028f0 is executed with arg\_8h, arg\_ch, arg\_10h as its arguments.
- After fcn.004028f0 is executed, function clears variables in the stack from uStack72 to uStack28 and uStack20 to uStack12, by setting them to zero.
   Finally, the function returns 0.

Project Lupine Flowchart



## **Plugins**

### **Ghidra Plugins**

•There are three Ghidra plugins each with their own configured shortcut keys

#### .llm.py (CTRL-ALT-L)

•This script calls your local LLM. It requests a new function name and function description. It renames the function and updates the somments with the description. Your cursor can be anywhere inside the decompiled function that you're interested in.

•The plugin expects api\_server.py to be running on localhost on port 8000. Documentation for the API server can be found below.

#### .llm\_remote.pt (CTRL-ALT-O)

•This script calls the Project Lupine community server. It requests a new function name and function description. It renames the function and updates the somments with the description. Note that it sends the hash, function offset, and decompiled code to the community server.

#### .llm\_suggest (CTRL-SHIFT-K)

•This script is useful for contributing back to the community. If you get a summary, function name or step-by-step description that you don't like you can edit the content directly in Ghidra and send your edits back. Note that it sends the hash, function offset, and decompiled code, function name, function comment to the community server

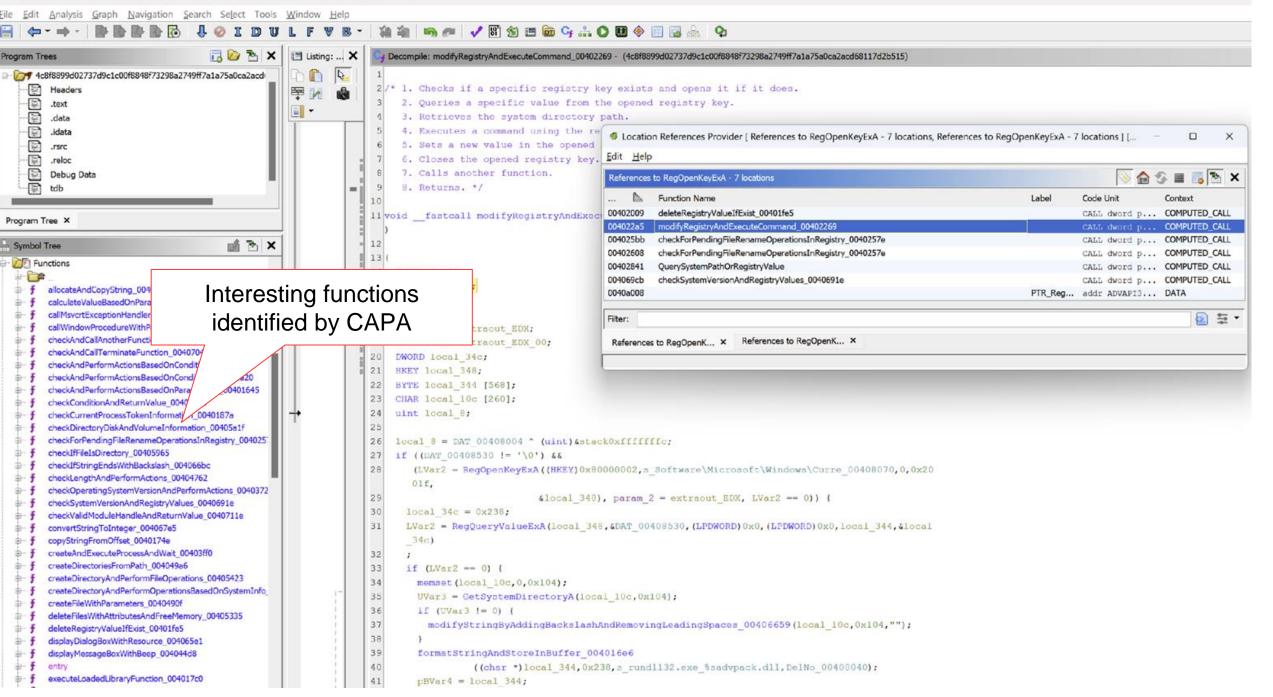
#### 

|   | J L F W B | 9 ~   4        | i i i i i i i i i i i i i i i i i i i  |   |                                   |  |
|---|-----------|----------------|--|---|-----------------------------------|--|
| Program Trees 🗊 🖄 🗙   | _         |                | Decompile: entry - (4c8f8899d02737d9c1c00f8848f73298a                        | 2749ff7a1a75a0ca2acd68117d2b515)  |                                   |  |
| Kosf8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd     Kosf8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd     Kosfata     Kosfata     Kosfata |           | 2              | /* WARNING: Function:SEH_prolog4_GS<br>/* WARNING: Globals starting with '_' |   | address */                        |  |
| ···)당 .rsrc<br>···)당 .rsic<br>···)당 .reloc  |           | 5              | 4  | References to RegOpenKeyExA - 7 locations [<br>Edit Help                | Xref RegOpenKey fror<br>functions |  |
| Debug Data  |           | 8<br>9<br>10   | void *pvVar2;  | References to RegOpenKeyExA - 7 locations                               | Label                             | Code Unit Context  |
| Program Tree X  | 1         | 11 12          | <pre>code *pcVar4;<br/>void *pvVar5;</pre>                                   | 00402009 FUN_00401fe5<br>004022a5 FUN_00402269                          |                                   | CALL dword p COMPUTED_CALL<br>CALL dword p COMPUTED_CALL                               |
| Symbol Tree 📝 🏝 🗙   | -         | 13<br>14<br>15 | BOOL BVar7;<br>uint uVar8;   | 004025bb FUN_0040257e<br>00402608 FUN_0040257e<br>00402841 FUN_0040278b | $\bigvee$                         | CALL dword p COMPUTED_CALL<br>CALL dword p COMPUTED_CALL<br>CALL dword p COMPUTED_CALL |
| CABINET.DLL<br>COMCTL32.DLL<br>GDI32.DLL  |           | 16<br>17<br>18 | undefined *extraout_ECX;   | 004069cb FUN_0040591e<br>0040a008                                       | PTR_Reg.                          | CALL dword p COMPUTED_CALL<br>addr ADVAPI3 DATA  |
| KERNEL32.DLL<br>MSVCRT.DLL<br>USER32.DLL  |           | 19<br>20       | undefined4 extraout_EDX;<br>undefined4 extraout_EDX_00;                      | Filter:   |                                   | <u>图 节 ·</u>   |
| ia-<br>→<br>→<br>→<br>→<br>→<br>→<br>→<br>→<br>→<br>→   |           | 21<br>22<br>23 | undefined8 uVarll;<br>undefined4 uVarl2;                                     |   |                                   |  |
|   |           | 24<br>25<br>26 | undefined *puVar14;  |   |                                   |  |
| È- ͡͡͡͡ FUN_0040<br>È- ͡͡͡ FUN_00401<br>È- ͡͡ FUN_00401645  |           | 27<br>28<br>29 | uint local_28;   |   |                                   |  |
|   | →         | 30<br>31       | undefined4 uStack_c;<br>undefined4 local_8;                                  |   |                                   |  |
| B·· f FUN_00401795<br>B·· f FUN_004017c0<br>B·· f FUN_0040187a  |           | 32<br>33<br>34 | <pre>FUN_004072d8();<br/>uStack_c = 0x406beb;</pre>                          |   |                                   |  |
| ⊕··· f FUN_004019c0     ⊕··· f FUN_00401a69     ⊕··· f FUN_00401a62     ⊕··· f FUN_00401a62     ⊕··· f FUN_00401a9a                                 |           | 35<br>36<br>37 | <pre>local_8 = 0;<br/>GetStartupInfoA(&amp;local_70);</pre>                  |   |                                   |  |
|   |           | 38<br>39<br>40 | local_8 = 1;   |   |                                   |  |
| PUN_00402 ii fun_00402033   |           | 41<br>42<br>43 | LOCK ();   |   |                                   |  |
|   |           | 44             | if (DAT_004088ac != (void *)0x0) (   |   |                                   |  |
|   |           |                |  |   |                                   |  |

CodeBrowser: test1:/4c8f8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd68117d2b515

Eile Edit Analysis Graph Navigation Search Select Tools Window Help 🗸 🕅 🐒 🖻 🚾 🗣 🚠 🔕 🛄 🔶 🔛 🔗 論 論 5 0 🖥 🕑 🏝 🗙 Listing: ... X Decompile: modifyRegistryAndExecuteCommand\_00402269 - (4c8f8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd68117d2b515) Program Trees 1 274c8f8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd 2/\* 1. Checks if a specific registry key exists and opens it if it does. Headers E ŵ 1 .text 2. Queries a specific value from the opened registry key. . 3. Retrieves the system directory path. .data .idata 4. Executes a command using the re Location References Provider [ References to RegO X LLM generated names 11 5. Sets a new value in the opened .rsrc Edit Help 6. Closes the opened registry key. .reloc in API xref view 7. Calls another function. 🕉 🔳 📑 🔁 🗙 Debug Data References to RegOpenKeyExA - 7 locations 8. Returns. \*/ tdb - 1 A Function Name Label Code Unit ..... Context 00402009 deleteRegistryValueIfExist\_00401fe5 CALL dword p... COMPUTED\_CALL 11 void fastcall modifyRegistryAndExec Program Tree × CALL dword p... COMPUTED\_CALL 004022a5 modifyRegistryAndExecuteCommand\_00402269 checkForPendingFileRenameOperationsInRegistry\_00402 004025bb CALL dword p... COMPUTED\_CALL 📫 🔁 🗙 12 Symbol Tree 00402608 checkForPendingFileRenameOperationsInRegistry\_0040 CALL dword p... COMPUTED\_CALL 13 E Functions 00402841 QuerySystemPathOrRegistryValue CALL dword p... COMPUTED CALL 14 BYTE BVarl; CPP 004069cb checkSystemVersionAndRegistryValues\_0040691e CALL dword p... COMPUTED\_CALL LSTATUS LVar2; 15 allocateAndCopyString\_00404870 a- **f** 0040a008 PTR\_Reg... addr ADVAPI3... DATA UINT UVar3; 16 calculateValueBasedOnParameters\_00401795 callMsvcrtExceptionHandler\_00407400 17 BYTE \*pBVar4; Ð- 🕈 Filter: callWindowProcedureWithParams 00403100 18 undefined4 extraout EDX; checkAndCallAnotherFunctionWithParameter 0040167d References to RegOpenK ... × References to RegOpenK ... × undefined4 extraout EDX 00; 19 1- f checkAndCallTerminateFunction 00407040 20 DWORD local 34c; checkAndPerformActionsBasedOnCondition\_004070af ±- ₹ 21 HKEY local 348; checkAndPerformActionsBasedOnConditions 00404a20 ⊕-**f** 22 BYTE local 344 [568]; checkAndPerformActionsBasedOnParamValue 00401645 23 CHAR local 10c [260]; checkConditionAndReturnValue 00401e9a 24 uint local 8; checkCurrentProcessTokenInformation\_0040187a • f 25 checkDirectoryDiskAndVolumeInformation\_00405a1f checkForPendingFileRenameOperationsInRegistry\_004025 26 local 8 = DAT 00408004 ^ (uint)&stack0xfffffffc; checkIfFileIsDirectory\_00405965 ±- ₹ 27 if ((DAT 00408530 != '\0') && checkIfStringEndsWithBackslash\_004066bc 28 (LVar2 = RegOpenKeyExA ((HKEY)0x80000002,s Software\Microsoft\Windows\Curre 00408070,0,0x20 checkLengthAndPerformActions\_00404762 a- f 01f, checkOperatingSystemVersionAndPerformActions\_0040372 29 Glocal 348), param 2 = extraout EDX, LVar2 == 0)) { checkSystemVersionAndRegistryValues\_0040691e 30 local 34c = 0x238; checkValidModuleHandleAndReturnValue 0040711e 31 LVar2 = RegQueryValueExA(local 348, &DAT 00408530, (LPDWORD)0x0, (LPDWORD)0x0, local 344, &local convertStringToInteger\_004067e5 34c) copyStringFromOffset\_0040174e ±- ₹ createAndExecuteProcessAndWait 00403ff0 32 1 createDirectoriesFromPath\_004049a6 33 if (LVar2 -- 0) { createDirectoryAndPerformFileOperations\_00405423 ÷f 34 memset (local 10c, 0, 0x104); createDirectoryAndPerformOperationsBasedOnSystemInfo 35 UVar3 = GetSystemDirectoryA(local 10c,0x104); createFileWithParameters 0040490f 36 if (UVar3 != 0) { deleteFilesWithAttributesAndFreeMemory\_00405335 i⊇- **f** 37 modifyStringByAddingBackslashAndRemovingLeadingSpaces 00406659 (local 10c,0x104,""); deleteRegistryValueIfExist\_00401fe5 38 - f displayDialogBoxWithResource 004065e1 displayMessageBoxWithBeep\_004044d8 39 formatStringAndStoreInBuffer 004016e6 Ŧ 40 ÷- 🕈 entry ((char \*)local 344,0x238,s rundl132.exe %sadvpack.dll,DelNo 00408040); executeLoadedLibraryFunction 004017c0 a-f pBVar4 = local 344;

CodeBrowser: test1:/4c8f8899d02737d9c1c00f8848f73298a2749ff7a1a75a0ca2acd68117d2b515



# **Ghidra Plugin**

Analyst in the loop

• This function name isn't very descriptive...

Decompile: MaliciousFunction - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c) 2 /\* WARNING: Globals starting with '\_' overlap smaller symbols at the same address \*/ 3 /\* This function is a malicious function that is designed to perform a variety of malicious activities, including: 4 6 1. Loading and executing a kernel32.dll function 2. Enumerating logical drives and creating threads to scan for specific files 7 3. Enumerating network resources and creating threads to scan for specific files 8 9 4. Enumerating network shares and creating threads to scan for specific files 5. Creating a new thread to perform a malicious activity \*/ 10 11 12 void MaliciousFunction(void) 13 14 { 15 WCHAR WVarl; int \*piVar2; 16 bool bVar3; 17 18 int iVar4; 19 DWORD nBufferLength; LPWSTR lpBuffer; 20 21 UINT UVar5; 22 int iVar6; 23 LPWSTR lpStringl; 24 HANDLE pvVar7; 25 int iVar8; 26 undefined4 extraout\_ECX; undefined4 extraout ECX 00; 27 28 undefined4 uVar9; undefined4 extraout EDX; 29 undefined4 extraout EDX 00; 30 31 undefined4 uVar10; 32 LPCWSTR lpRootPathName; 33 int iVarll; 34 uint uVarl2; 35 int \*piVarl3; 36 undefined4 uStack le0; 37 WCHAR aWStack ldc [34]; 38 undefined local 198 [4]; undefined auStack 194 [400]; 39 40 41 memset(local 198,0,400); DAT\_0042f894 = LoadAndExecuteKernel32Function(0x16d34cd7, &DAT\_0042f0fc, 0x14); 42 iVar4 = Ordinal\_115(0x202,auStack\_194); 43 44 if (iVar4 != 0) { 45 Ordinal 116(); 46 } 47 nBufferLength = GetLogicalDriveStringsW(0, (LPWSTR)0x0); lpBuffer = (LPWSTR)FUN\_00415650(nBufferLength \* 2); 48 GetLogicalDriveStringsW(nBufferLength, lpBuffer); 49 WVarl = \*lpBuffer; 50 51 lpRootPathName = lpBuffer; 52 while (WVarl != L'\0') { 53 bVar3 = false; 54 UVar5 = GetDriveTypeW(lpRootPathName); 55 wsprintfW(aWStack\_ldc,L"\\??\\%.2s\\",lpRootPathName); 56 if (((UVar5 == 2) || (UVar5 == 3)) || (UVar5 == 4)) { 57 if ((DAT 004342e8 != 0) && (iVar4 = 0, 0 < DAT 004342e8)) { 58 do {

# **Ghidra Plugin**

Analyst in the loop

 The Analyst can modify the function name and/or summary description and submit suggestions to the community server.

| Information | Craiss Wick Notwocker | All rights | reserved. Public |
|-------------|-----------------------|------------|------------------|
| muormanon   | Ciassification, Ger   |            |                  |

| <pre>Decompile: entry_enumeration_shell_execution_strings - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f42dd0611830764c6 //* WARNING: Globals starting with '_' overlap smaller symbols at the same address */ //* This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware. 1. The function initializes some variables and retrieves the version information of the operating system. 2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory. 3. Depending on the value of a variable, it executes different code paths to perform specific actions. 4. It converts some strings from wide character format to multi-byte format. 5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.] 6. Finally, it calls the system interrupt to exit the process. */ 17 void entry_enumeration_shell_execution_strings(void) 18 19 20 10 11 *piVar1; 12 code *pcVar2; 12 LPONSTR pWWar3; 13 bool bWar4; 14 th iVar5; 12 LPONSTR pWWar3; 13 bool bWar4; 14 th iVar5; 12 LPONSTR pWWar3; 13 bool bWar4; 14 th iVar5; 12 LPONSTR pWWar3; 13 bool bWar4; 14 th iVar5; 12 LPONSTR pWWar3; 13 bool bWar4; 14 th iVar5; 12 LPONSTR pWWar3; 14 DATE Converts some strings from vice the string strings s</pre>   |
|--|
| <pre>1 2 /* WARNING: Globals starting with '_' overlap smaller symbols at the same address */ 3/* This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware. 1. The function initializes some variables and retrieves the version information of the operating system. 2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory. 3. Depending on the value of a variable, it executes different code paths to perform specific actions. 4. It converts some strings from wide character format to multi-byte format. 5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.] 6. Finally, it calls the system interrupt to exit the process. */ void entry_enumeration_shell_execution_strings(void) [9 {     int *piVar1;     code *pcVar2;     LPOKSTR pWWar3;     bool bWar4;     int iVar5;     LPWSTR pWWar6;     DWORD DVar7; BOL BWar8;     ushort *puVar9;     undefined4 uVar10;     HADDLE hHeap; L PSTR LpMUltiByteStr;     undefined4 evtraout_FCY. </pre>   |
| <pre>2 /* WARNING: Globals starting with '_' overlap smaller symbols at the same address */ 3 /* This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware. 1. The function initializes some variables and retrieves the version information of the operating system. 2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory. 3. Depending on the value of a variable, it executes different code paths to perform specific actions. 4. It converts some strings from wide character format to multi-byte format. 5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.] 6. Finally, it calls the system interrupt to exit the process. */ 7 void entry_enumeration_shell_execution_strings(void) 8 8 9 { 1     int 'piVar1; 1     code *pcVar2; 2     LPCWSTR pWWar6; 2     DWORD DVar7; 8     Bool BVar4; 8     int iVar5; 1     LMWSTR pWWar6; 9     DWORD DVar7; 9     undefined3 extraout_var; 9     undefined4 uVar10; 1     HANDLE hHeap; 1     LFSTR lpMultiByteStr; 9     undefined4 evenut ECY. </pre>   |
| <pre>This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware. 1. The function initializes some variables and retrieves the version information of the operating system. 2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory. 3. Depending on the value of a variable, it executes different code paths to perform specific actions. 4. It converts some strings from wide character format to multi-byte format. 5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.] 6. Finally, it calls the system interrupt to exit the process. */ 7 void entry_enumeration_shell_execution_strings(void) 8 { 10. t *piVar1; 10. code *peVar2; 12. LPCWSTR pWVar3; 13. bool bVar4; 13. int iVar5; 12. LPWSTR pWVar6; 13. MONED OVAr7; 14. MONED OVAr7; 15. BOOL EVAr6; 15. Just *piVar1; 16. Code *peVar2; 17. Worde extraout_var; 17. undefinedd evtraout_var; 18. undefinedd evtraout_var; 19. undefinedd evtraout_var; 10. undefinedd uvar10; 14. HANDLE hHeap; 15. LPWSTR IPMUltByteStr; 15. UPSTR IPMULTERY IP IPMULTBY IP IPMUL</pre>   |
| <pre>to the execution of the malware.<br/>1. The function initializes some variables and retrieves the version information of the operating<br/>system.<br/>2. It calls several other functions to perform different tasks, such as setting up registry keys,<br/>executing shell commands, and allocating memory.<br/>3. Depending on the value of a variable, it executes different code paths to perform specific<br/>actions.<br/>4. It converts some strings from wide character format to multi-byte format.<br/>5. It calls additional functions to perform various operations, such as printing output and<br/>calculating string lengths.]<br/>6. Finally, it calls the system interrupt to exit the process. */<br/>7 void entry_enumeration_shell_execution_strings(void)<br/>8 {<br/>1 int *piVarl;<br/>1 code *pcVar2;<br/>2 LPCWSTR pWVar3;<br/>3 bool bVar4;<br/>4 int iVar5;<br/>5 LIPWSTR pWVar6;<br/>5 DWORD DVar7;<br/>8 ushort *puVar9;<br/>1 undefined3 extraout_var;<br/>1 undefined4 extraout_var;<br/>1 undefined4 extraout_var;<br/>2 LPSTR IpMULTByteStr;<br/>2 LPSTR IpMULTByteStr;<br/>3 undefined4 extraout_var5;<br/>5 LPSTR IpMULTByteStr;<br/>3 undefined4 extraout_FCX;<br/>5 LPSTR IpMULTByteStr;<br/>5 LPSTR</pre>   |
| <pre>6 1. The function initializes some variables and retrieves the version information of the operating 7 system. 2. It calls several other functions to perform different tasks, such as setting up registry keys, 9 executing shell commands, and allocating memory. 3. Depending on the value of a variable, it executes different code paths to perform specific 1 actions. 4. It converts some strings from wide character format to multi-byte format. 5. It calls additional functions to perform various operations, such as printing output and 4 calculating string lengths.] 6. Finally, it calls the system interrupt to exit the process. */ 7 void entry_enumeration_shell_execution_strings(void) 8 9 { 1 int *piVar1; 1 code *pcVar2; 2 LPOWSTR pWVar3; 3 bool bVar4; 3 int ivar5; 5 LPWSTR pWVar6; 6 DWORD DVar7; 8 BOOL EVAr8; 8 ushort *puVar9; 9 undefined3 extraout_var; 9 undefined4 uvarious to factors 9 context as the set of the set o</pre>   |
| <pre>7 system.<br/>8 2. It calls several other functions to perform different tasks, such as setting up registry keys,<br/>9 executing shell commands, and allocating memory.<br/>3. Depending on the value of a variable, it executes different code paths to perform specific<br/>1 actions.<br/>4. It converts some strings from wide character format to multi-byte format.<br/>5. It calls additional functions to perform various operations, such as printing output and<br/>calculating string lengths.]<br/>6. Finally, it calls the system interrupt to exit the process. */<br/>7 void entry_enumeration_shell_execution_strings(void)<br/>8<br/>{<br/>1 code *pcVar2;<br/>2 LPCWSTR pWVar3;<br/>3 bool bVar4;<br/>4 int iVar5;<br/>5 LPWSTR pWvar6;<br/>6 DWORD DVar7;<br/>8 BOL EWar8;<br/>8 ushort *puVar9;<br/>9 undefined4 uvar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultByteStr;<br/>9 undefined4 uvar10;<br/>1 HANDLE hHeap;<br/>1 Heap;<br/>1 H</pre>   |
| <pre>9 executing shell commands, and allocating memory.<br/>3. Depending on the value of a variable, it executes different code paths to perform specific<br/>actions.<br/>4. It converts some strings from wide character format to multi-byte format.<br/>5. It calls additional functions to perform various operations, such as printing output and<br/>calculating string lengths.]<br/>6. Finally, it calls the system interrupt to exit the process. */<br/>7 void entry_enumeration_shell_execution_strings(void)<br/>8<br/>9 {<br/>1 int *piVarl;<br/>1 code *pcVar2;<br/>2 LPOWSTR pWVar3;<br/>3 bool bVar4;<br/>4 int iVar5;<br/>5 LPWSTR pWVar6;<br/>6 DWORD DVar7;<br/>7 BOOL BVar8;<br/>8 ushort *puVar9;<br/>9 undefined3 extraout_var;<br/>9 undefined4 evtraout_var;<br/>9 undefined4 evtraout_var;<br/>9 undefined4 evtraout_Var10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>9 undefined4 evtraout_ECX;</pre>   |
| <pre>3. Depending on the value of a variable, it executes different code paths to perform specific<br/>actions.<br/>4. It converts some strings from wide character format to multi-byte format.<br/>5. It calls additional functions to perform various operations, such as printing output and<br/>calculating string lengths.]<br/>6. Finally, it calls the system interrupt to exit the process. */<br/>7 void entry_enumeration_shell_execution_strings(void)<br/>8<br/>9{<br/>1</pre>  |
| <pre>actions.<br/>4. It converts some strings from wide character format to multi-byte format.<br/>5. It calls additional functions to perform various operations, such as printing output and<br/>calculating string lengths.]<br/>6. Finally, it calls the system interrupt to exit the process. */<br/>7 void entry_enumeration_shell_execution_strings(void)<br/>9 {<br/>10 int *piVar1;<br/>11 code *pcVar2;<br/>12 LPCMSTR pWVar3;<br/>13 bool bVar4;<br/>14 int iVar5;<br/>15 LPWSTR pWVar6;<br/>16 DWORD DVar7;<br/>17 BOOL BVar8;<br/>18 ushort *puVar9;<br/>19 undefined3 extraout_var;<br/>10 undefined4 uVar10;<br/>11 HANDLE hHeap;<br/>12 LPSTR lpMultiByteStr;<br/>12 undefined4 extraout_ECX;</pre>  |
| <pre>3 5. It calls additional functions to perform various operations, such as printing output and<br/>calculating string lengths. <br/>6. Finally, it calls the system interrupt to exit the process. */<br/>7 void entry_enumeration_shell_execution_strings(void)<br/>8 {<br/>10 int *piVarl;<br/>10 code *pcVar2;<br/>12 LPOWSTR pWVar3;<br/>13 bool bVar4;<br/>14 int iVar5;<br/>15 LPWSTR pWVar6;<br/>16 DWORD DVar7;<br/>17 BOOL BVar8;<br/>18 ushort *puVar9;<br/>19 undefined3 extraout_var;<br/>10 undefined4 uVar10;<br/>11 HANDLE hHeap;<br/>12 LPSTR lpMultiByteStr;<br/>13 undefined4 extraout_ECX;</pre>  |
| <pre>4 calculating string lengths.  5 6. Finally, it calls the system interrupt to exit the process. */ 6 7 void entry_enumeration_shell_execution_strings(void) 8 9 { 0 int *piVar1; 1 code *pcVar2; 2 LPOWSTR pWVar3; 3 bool bVar4; 1 int iVar5; 3 LPWSTR pWVar6; 6 DWORD DVar7; 7 BOOL BVar8; 8 ushort *puVar9; 9 undefined3 extraout_var; 0 undefined4 uVar10; 1 HANDLE hHeap; 2 LPSTR lpMultiByteStr; 3 undefined4 extraout_ECX; 6 Comparison of the string o</pre>   |
| <pre>5 6. Finally, it calls the system interrupt to exit the process. */ 6 7 void entry_enumeration_shell_execution_strings(void) 8 9 { 0 int *piVar1; 1 code *pcVar2; 2 LPOKSTR pWvar3; 3 bool bVar4; 4 int iVar5; 5 LPWSTR pWvar6; 6 DWORD DVar7; 7 BOOL BVar8; 8 ushort *puVar9; 9 undefined3 extraout_var; 1 undefined4 uVar10; 1 HANDLE hHeap; 2 LPSTR lpMultiByteStr; 3 undefined4 extraout_ECX; </pre>  |
| <pre>7 void entry_enumeration_shell_execution_strings(void) 8 9 { 0 int *piVar1; 1 code *pcVar2; 2 LPOWSTR pWVar3; 3 bool bVar4; 4 int iVar5; 5 LPWSTR pWVar6; 6 DWORD DVar7; 7 BOOL BVar8; 8 ushort *puVar9; 9 undefined3 extraout_var; 0 undefined4 uVar10; 1 HANDLE hHeap; 2 LPSTR lpMultByteStr; 3 undefined4 extraout ECX; </pre>   |
| <pre>8 9 { int *piVarl; code *pcVar2; LPCWSTR pWVar3; bool bVar4; int iVar5; 5 LPWSTR pWVar6; 6 DWORD DVar7; 7 BOOL BVar8; 9 undefined3 extraout_var; 9 undefined4 uVarl0; 1 HANDLE hHeap; 2 LPSTR lpMultiByteStr; 3 undefined4 extraout EX:</pre>   |
| <pre>9 { 9 { 9 { 9 } 1</pre>   |
| <pre>0 int *piVarl;<br/>code *pcVar2;<br/>LPCWSTR pWVar3;<br/>bool bVar4;<br/>int iVar5;<br/>LPWSTR pWVar6;<br/>6 DWORD DVar7;<br/>BOOL BVar8;<br/>ushort *puVar9;<br/>undefined3 extraout_var;<br/>undefined4 uvar10;<br/>HANDLE hHeap;<br/>LPSTR lpMultiByteStr;<br/>undefined4 extraout ECX;</pre>  |
| <pre>2 LPOWSTR pWVar3;<br/>bool bVar4;<br/>int iVar5;<br/>5 LPWSTR pWVar6;<br/>6 DWORD DVar7;<br/>7 BOOL BVar8;<br/>9 undefined3 extraout_var;<br/>9 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout_ECX;</pre>   |
| <pre>Big bool bvar4;<br/>bool bvar4;<br/>big tri tvar5;<br/>buewsTR pWvar6;<br/>bowOpD Dvar7;<br/>BOOL Bvar8;<br/>ushort *puvar9;<br/>undefined3 extraout_var;<br/>undefined4 uvar10;<br/>HANDLE hHeap;<br/>buewstriputation<br/>HANDLE hHeap;<br/>buewstriputation<br/>HANDL</pre> |
| <pre>int iVar5;<br/>LPWSTR pWVar6;<br/>DWORD DVar7;<br/>BOOL BVar8;<br/>undefined3 extraout_var;<br/>undefined4 uVar10;<br/>HANDLE hHeap;<br/>LPSTR lpMultiByteStr;<br/>updfined4 extraout_FCX;</pre>  |
| <pre>5 LPWSTR pWVar6;<br/>6 DWORD DVar7;<br/>7 BOOL BVar8;<br/>8 ushort *puVar9;<br/>9 undefined3 extraout_var;<br/>9 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout_ECX;</pre>  |
| <pre>7 BOOL BVar8;<br/>8 ushort *puVar9;<br/>9 undefined3 extraout_var;<br/>0 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout ECX;</pre>  |
| <pre>8 ushort *puVar9;<br/>9 undefined3 extraout_var;<br/>0 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout ECX;</pre>  |
| <pre>9 undefined3 extraout_var;<br/>0 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout ECX;</pre>  |
| 0 undefined4 uVarl0;<br>1 HANDLE hHeap;<br>2 LPSTR lpMultiByteStr;<br>3 undefined4 extraout ECX;   |
| 1 HANDLE hHeap;<br>2 LPSTR lpMultiByteStr;<br>3 undefined4 extraout ECX:   |
| 3 undefined4 extraout ECX  |
| BL undefined4 extraout FCX: //   |
| undefined4 extraout ECX 00; Set Comment(s) at Address 00417f50   |
| 5 undefined4 extraout ECX 01;  |
| 6 undefined4 extraout_ECX_02; EOL Comment Pre Comment  |
| 7 undefined4 extraout_ECX_03; Post Comment Plate Comment Repeatable Comment  |
| 8 undefined4 extraout_ECX_04;<br>9 undefined4 extraout_ECX_05;   |
| 0 undefined4 extraout ECX 06; This code is a function that is part of a malware sample. It performs various oper   |
| 1 undefined4 extraout ECX 07; related to the execution of the malware.   |
| 2 undefined4 extraout_ECX_08; 1. The function initializes some variables and retrieves the version information or  |
| 3 undefined4 extraout ECX 09; operating system.<br>4 undefined4 extraout ECX 10: 2. It calls several other functions to perform different tasks, such as setting up  |
| 4 undefined4 extraout_ECX_10; 2. It calls several other functions to perform different tasks, such as setting up<br>5 undefined4 extraout ECX_11; registry keys, executing shell commands, and allocating memory.  |
| 6 undefined4 extraout ECX 12: 3. Depending on the value of a variable, it executes different code paths to perfo   |
|  |
|  |
| <ul> <li>and interfaced extraout_ecx_1s;</li> <li>int iVarll;</li> <li>T calls additional functions to perform various operations, such as printing of</li> </ul>  |
| <ul> <li>and inderined extraout_ECX_1s;</li> <li>int ivarl;</li> <li>undefined4 extraout_ECX_14;</li> <li>It converts some strings from wide character format to multi-byte format.</li> <li>It calls additional functions to perform various operations, such as printing or and calculating string lengths.</li> </ul>   |
| <ul> <li>a int iVarll;</li> <li>a undefined4 extraout_ECX_14;</li> <li>a undefined4 extraout_ECX_14;</li> <li>a undefined4 extraout_ECX_14;</li> <li>a undefined4 extraout_ECX_15;</li> <li>a calculating string lengths.</li> <li>a finite fini</li></ul>   |
| <ul> <li>interined4 extraout_ECX_1s;</li> <li>int iVarll;</li> <li>undefined4 extraout_ECX_14;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_16;</li> <li>Finally, it calls the system interrupt to exit the process.</li> </ul>  |
| <ul> <li>4. It converts some strings from wide character format to multi-byte format.</li> <li>9. undefined4 extraout_ECX_14;</li> <li>9. undefined4 extraout_ECX_15;</li> <li>1. undefined4 extraout_ECX_15;</li> <li>1. undefined4 extraout_ECX_16;</li> <li>2. undefined4 extraout_ECX_17;</li> <li>2. undefined4 extraout_ECX_17;</li> <li>3. tr calls additional functions to perform various operations, such as printing or and calculating string lengths.</li> <li>6. Finally, it calls the system interrupt to exit the process.</li> <li>4. It converts some strings from wide character format to multi-byte format.</li> <li>5. It calls additional functions to perform various operations, such as printing or and calculating string lengths.</li> <li>6. Finally, it calls the system interrupt to exit the process.</li> <li>7. The string lengths.</li> <li>8. Finally, it calls the system interrupt to exit the process.</li> </ul>   |
| <ul> <li>int iverified4 extraout_ECX_1s;</li> <li>int iverified4 extraout_ECX_14;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_18;</li> </ul>  |
| <ul> <li>4. It converts some strings from wide character format to multi-byte format.</li> <li>5. It calls additional functions to perform various operations, such as printing or<br/>and calculating string lengths.</li> <li>6. Finally, it calls the system interrupt to exit the process.</li> <li>mudefined4 extraout_ECX_16;<br/>undefined4 extraout_ECX_16;<br/>undefined4 extraout_ECX_16;<br/>undefined4 extraout_ECX_18;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;</li> <li>mudefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;</li> <li>mudefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefine4 ext</li></ul>   |
| <ul> <li>4. It converts some strings from wide character format to multi-byte format.</li> <li>5. It calls additional functions to perform various operations, such as printing or<br/>undefined4 extraout_ECX_15;<br/>undefined4 extraout_ECX_16;<br/>undefined4 extraout_ECX_17;<br/>undefined4 extraout_ECX_17;<br/>undefined4 extraout_ECX_18;<br/>undefined4 extraout_ECX_18;<br/>undefined4 extraout_ECX_18;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;<br/>undefined4 extraout_ECX_19;</li> <li>6. Finally, it calls the system interrupt to exit the process.</li> <li>7. It converts some strings from wide character format to multi-byte format.</li> <li>7. It calls additional functions to perform various operations, such as printing or<br/>undefined4 extraout_ECX_19;</li> <li>8. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It converts some strings from wide character format to multi-byte format.</li> <li>9. It calls additional function format.</li> <li>9. It calls the system interrupt to exit the process.</li> <li>9. It calls the system interrupt to exit the process.</li> <li>9. It calls the system interrupt to exit the process.</li> <li>9. It calls the system interrupt to exit the process.</li> <li>9. It calls the system interrupt to exit the proces.</li></ul>   |
| <ul> <li>int ivarl;</li> <li>undefined4 extraout_ECX_13;</li> <li>undefined4 extraout_ECX_14;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_17;</li> <li>undefined4 extraout_ECX_17;</li> <li>undefined4 extraout_ECX_17;</li> <li>undefined4 extraout_ECX_18;</li> <li>undefined4 extraout_ECX_18;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_EDX;</li> <li< td=""></li<></ul>  |
| <ul> <li>int ivarl;</li> <li>undefined4 extraout_ECX_12;</li> <li>undefined4 extraout_ECX_12;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_17;</li> <li>undefined4 extraout_ECX_17;</li> <li>undefined4 extraout_ECX_18;</li> <li>undefined4 extraout_ECX_18;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_1</li></ul>   |
| <ul> <li>int iVarll;</li> <li>int iVarll;</li> <li>undefined4 extraout_ECX_14;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_18;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_EDX;</li> <li>undefined4 extraout_EDX_00;</li> <li>undefined4 extraout_EDX_01;</li> <li>undefined4 extraout_EDX_02;</li> <li>Enter accepts comment</li> </ul>  |
| <ul> <li>int iVarll;</li> <li>int iVarll;</li> <li>undefined4 extraout_ECX_14;</li> <li>undefined4 extraout_ECX_15;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_16;</li> <li>undefined4 extraout_ECX_18;</li> <li>undefined4 extraout_ECX_19;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_ECX_10;</li> <li>undefined4 extraout_EDX;</li> <li>undefined4 extraout_EDX,00;</li> <li>undefined4 extraout_EDX_00;</li> <li>undefined4 extraout_EDX_00;</li> <li>undefined4 extraout_EDX_00;</li> <li>undefined4 extraout_EDX_02;</li> </ul>  |

## **Ghidra Plugin: Feedback**

Analyst in the loop

- The Analyst can modify the function name and/or summary description and submit suggestions to the community server.
- CTRL-SHIFT-K

#### 🖳 Console - Scripting

- 3. Depending on the value of a variable, it executes different code paths to perform specific actions.
- 4. It converts some strings from wide character format to multi-byte format.
- 5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.
- 6. Finally, it calls the system interrupt to exit the process.

### Constants

This function refrences a registry path commonly associated with malware persistence <Response [200]> Server Response: {'status': 'success', 'message': 'Suggestion added successfully.'} llm\_suggest.py> Finished!

Information Classification, Ceneral All rights reserved. Public

|  | _shell_execution_strings [CodeBrowser: ghidra test project 1:/c373d37b5a9427a18dbf93d519   |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|
| ille <u>E</u> dit <u>N</u> avigation <u>S</u> earch Select <u>H</u> elp  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  | shell_execution_strings - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c)   |  |  |  |  |  |  |
| 1<br>2 /* WARNING: Globals starting  | with '_' overlap smaller symbols at the same address */  |  |  |  |  |  |  |
| 3 /*   | the is and of a solution could. The solutions countries and the  |  |  |  |  |  |  |
| <ul> <li>This code is a function t</li> <li>to the execution of the m</li> </ul>   | hat is part of a malware sample. It performs various operations related<br>alware.   |  |  |  |  |  |  |
| 6 1. The function initializ  | es some variables and retrieves the version information of the operating   |  |  |  |  |  |  |
| system.  |  |  |  |  |  |  |  |
| executing shell commands, and allocating memory.   |  |  |  |  |  |  |  |
| 3. Depending on the value of a variable, it executes different code paths to perform specific  |  |  |  |  |  |  |  |
| 11 actions.<br>12 4. It converts some strin  | gs from wide character format to multi-byte format.  |  |  |  |  |  |  |
|  | nctions to perform various operations, such as printing output and   |  |  |  |  |  |  |
| 14 calculating string length<br>5 Finally it calls the   |  |  |  |  |  |  |  |
| 5 6. Finally, it calls the .   | system interrupt to exit the process. */   |  |  |  |  |  |  |
| .7 void entry_enumeration_shell  | _execution_strings(void)   |  |  |  |  |  |  |
| 8<br>9 {   |  |  |  |  |  |  |  |
| 20 int *piVarl;  |  |  |  |  |  |  |  |
| 1 code *pcVar2;  |  |  |  |  |  |  |  |
| 2 LPCWSTR pWVar3;<br>23 bool bVar4;  |  |  |  |  |  |  |  |
| 4 int iVar5;   |  |  |  |  |  |  |  |
| 25 LPWSTR pWVar6;  |  |  |  |  |  |  |  |
| 26 DWORD DVar7;<br>27 BOOL BVar8;  |  |  |  |  |  |  |  |
| BUUL DIGIO,  |  |  |  |  |  |  |  |
| 28 ushort *puVar9;   |  |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>29 undefined3 extraout_var;  |  |  |  |  |  |  |  |
| 28 ushort *puVar9;   |  |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVarl0;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> </ul>  |  |  |  |  |  |  |  |
| <ul> <li>28 ushort *puVar9;</li> <li>29 undefined3 extraout_var;</li> <li>30 undefined4 uVar10;</li> <li>31 HANDLE hHeap;</li> <li>32 LPSTR lpMultiByteStr;</li> <li>33 undefined4 extraout_ECX;</li> </ul>  | Set Comment(s) at Address 00417f50   |  |  |  |  |  |  |
| <ul> <li>28 ushort *puVar9;</li> <li>29 undefined3 extraout_var;</li> <li>30 undefined4 uVar10;</li> <li>31 HANDLE hHeap;</li> <li>32 LPSTR lpMultiByteStr;</li> <li>33 undefined4 extraout_ECX;</li> <li>34 undefined4 extraout_ECX_00</li> <li>35 undefined4 extraout_ECX_01</li> </ul>  |  |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVarl0;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_00</li> <li>undefined4 extraout_ECX_01</li> <li>undefined4 extraout_ECX_02</li> </ul>  | EOL Comment Pre Comment  |  |  |  |  |  |  |
| <ul> <li>28 ushort *puVar9;</li> <li>29 undefined3 extraout_var;</li> <li>30 undefined4 uVar10;</li> <li>31 HANDLE hHeap;</li> <li>32 LPSTR lpMultiByteStr;</li> <li>33 undefined4 extraout_ECX;</li> <li>34 undefined4 extraout_ECX_00</li> <li>35 undefined4 extraout_ECX_01</li> </ul>  | EOL Comment Pre Comment<br>Post Comment Repeatable Comment   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_02</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> </ul>  | EOL Comment Pre Comment Post Comment Plate Comment Repeatable Comment  |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVarl0;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_00</li> <li>undefined4 extraout_ECX_02</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> </ul>  | EOL Comment<br>Post Comment<br>Post Comment<br>This code is a function that is part of a malware sample. It performs various operations  |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_00</li> <li>undefined4 extraout_ECX_01</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_064</li> <li>undefined4 extraout_ECX_064</li> <li>undefined4 extraout_ECX_065</li> <li>undefined4 extraout_ECX_065</li> <li>undefined4 extraout_ECX_065</li> </ul>   | EOL Comment<br>Post Comment<br>Post Comment<br>Plate Comment<br>Repeatable Comment<br>This code is a function that is part of a malware sample. It performs various operations<br>related to the execution of the malware.   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_00</li> <li>undefined4 extraout_ECX_02</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_08</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> </ul>  | EOL Comment         Pre Comment           Post Comment         Plate Comment           Repeatable Comment         Repeatable Comment   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX[0]</li> <li>undefined4 extraout_ECX[1]</li> </ul>  | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up   |  |  |  |  |  |  |
| <pre>8 ushort *puVar9;<br/>9 undefined3 extraout_var;<br/>0 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout_ECX;<br/>4 undefined4 extraout_ECX_00<br/>5 undefined4 extraout_ECX_02<br/>7 undefined4 extraout_ECX_03<br/>8 undefined4 extraout_ECX_03<br/>9 undefined4 extraout_ECX_04<br/>9 undefined4 extraout_ECX_05<br/>1 undefined4 extraout_ECX_07<br/>2 undefined4 extraout_ECX_08<br/>3 undefined4 extraout_ECX_08<br/>3 undefined4 extraout_ECX_08<br/>4 undefined4 extraout_ECX_08<br/>5 undefined4 extraout_ECX_08<br/>5 undefined4 extraout_ECX_08</pre>   | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX_01</li> <li>undefined4 extraout_ECX_02</li> <li>undefined4 extraout_ECX_02</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_08</li> <li>undefined4 extraout_ECX_09</li> <li>undefined4 extraout_ECX_09</li> <li>undefined4 extraout_ECX_09</li> <li>undefined4 extraout_ECX_09</li> <li>undefined4 extraout_ECX_09</li> <li>undefined4 extraout_ECX_12</li> <li>undefined4 extraout_ECX_112</li> </ul>   | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.  |  |  |  |  |  |  |
| <pre>8 ushort *puVar9;<br/>9 undefined3 extraout_var;<br/>0 undefined4 uVar10;<br/>1 HANDLE hHeap;<br/>2 LPSTR lpMultiByteStr;<br/>3 undefined4 extraout_ECX;<br/>4 undefined4 extraout_ECX_00<br/>0 undefined4 extraout_ECX_01<br/>6 undefined4 extraout_ECX_02<br/>7 undefined4 extraout_ECX_03<br/>8 undefined4 extraout_ECX_03<br/>0 undefined4 extraout_ECX_04<br/>9 undefined4 extraout_ECX_05<br/>1 undefined4 extraout_ECX_05<br/>1 undefined4 extraout_ECX_06<br/>1 undefined4 extraout_ECX_06<br/>3 undefined4 extraout_ECX_07<br/>9 undefined4 extraout_ECX_08<br/>3 undefined4 extraout_ECX_09<br/>4 undefined4 extraout_ECX_09<br/>4 undefined4 extraout_ECX_02<br/>5 undefined4 extraout_ECX_10<br/>5 undefined4 extraout_ECX_12<br/>7 undefined4 extraout_ECX_12</pre>  | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         1.  |  |  |  |  |  |  |
| <ul> <li>8 ushort *puVar9;</li> <li>9 undefined3 extraout_var;</li> <li>10 undefined4 uVar10;</li> <li>11 HANDLE hHeap;</li> <li>22 LPSTR lpMultiByteStr;</li> <li>33 undefined4 extraout_ECX;</li> <li>44 undefined4 extraout_ECX;</li> <li>44 undefined4 extraout_ECX_03</li> <li>45 undefined4 extraout_ECX_03</li> <li>48 undefined4 extraout_ECX_03</li> <li>48 undefined4 extraout_ECX_03</li> <li>49 undefined4 extraout_ECX_04</li> <li>49 undefined4 extraout_ECX_05</li> <li>40 undefined4 extraout_ECX_06</li> <li>41 undefined4 extraout_ECX_07</li> <li>41 undefined4 extraout_ECX_06</li> <li>42 undefined4 extraout_ECX_07</li> <li>43 undefined4 extraout_ECX_07</li> <li>44 undefined4 extraout_ECX_07</li> <li>44 undefined4 extraout_ECX_07</li> <li>44 undefined4 extraout_ECX_07</li> <li>44 undefined4 extraout_ECX_10</li> <li>54 undefined4 extraout_ECX_11</li> <li>54 undefined4 extraout_ECX_12</li> <li>54 undefined4 extraout_ECX_13</li> <li>55 int iVar11;</li> </ul>   | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         I. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX,01</li> <li>undefined4 extraout_ECX,02</li> <li>undefined4 extraout_ECX,02</li> <li>undefined4 extraout_ECX,03</li> <li>undefined4 extraout_ECX,04</li> <li>undefined4 extraout_ECX,05</li> <li>undefined4 extraout_ECX,06</li> <li>undefined4 extraout_ECX,06</li> <li>undefined4 extraout_ECX,06</li> <li>undefined4 extraout_ECX,06</li> <li>undefined4 extraout_ECX,07</li> <li>undefined4 extraout_ECX,08</li> <li>undefined4 extraout_ECX,09</li> <li>undefined4 extraout_ECX,09</li> <li>undefined4 extraout_ECX,09</li> <li>undefined4 extraout_ECX,09</li> <li>undefined4 extraout_ECX,10</li> <li>undefined4 extraout_ECX,11</li> <li>undefined4 extraout_ECX,13</li> <li>int iVar11;</li> <li>undefined4 extraout_ECX,14</li> <li>undefined4 extraout_ECX,14</li> <li>undefined4 extraout_ECX,14</li> </ul>   | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.           6. Finally, it calls the system interrupt to exit the process.         1  |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;<br/>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX[0]</li> <li>undefined4 extraout_ECX[1]</li> <li>undefined4 extraout_ECX[1]</li> <li>undefined4 extraout_ECX[1]</li> <li>int iVar1];</li> <li>undefined4 extraout_ECX[1]</li> <li>undefined4 extraout_ECX[1]</li> <li>undefined4 extraout_ECX[1]</li> </ul>  | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         I. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.           6. Finally, it calls the system interrupt to exit the process.         ************************************   |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>undefined3 extraout_var;<br>30 undefined4 uVar10;<br>HANDLE hHeap;<br>32 LPSTR lpMultiByteStr;<br>33 undefined4 extraout_ECX;<br>34 undefined4 extraout_ECX;<br>36 undefined4 extraout_ECX_03<br>37 undefined4 extraout_ECX_03<br>38 undefined4 extraout_ECX_03<br>39 undefined4 extraout_ECX_03<br>40 undefined4 extraout_ECX_04<br>90 undefined4 extraout_ECX_05<br>40 undefined4 extraout_ECX_06<br>41 undefined4 extraout_ECX_07<br>42 undefined4 extraout_ECX_07<br>43 undefined4 extraout_ECX_07<br>44 undefined4 extraout_ECX_07<br>45 undefined4 extraout_ECX_07<br>46 undefined4 extraout_ECX_07<br>47 undefined4 extraout_ECX_08<br>48 undefined4 extraout_ECX_08<br>49 undefined4 extraout_ECX_08<br>40 undefined4 extraout_ECX_08<br>40 undefined4 extraout_ECX_10<br>41 undefined4 extraout_ECX_11<br>42 undefined4 extraout_ECX_12<br>43 undefined4 extraout_ECX_13<br>44 undefined4 extraout_ECX_14<br>45 undefined4 extraout_ECX_15<br>45 undefined4 extraout_ECX_14<br>46 undefined4 extraout_ECX_14<br>47 undefined4 extraout_ECX_15<br>47 undefined4 extraout_ECX_16<br>47 undefined4 extraout                         | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         I. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         S. It calls additional functions to perform various operations, such as printing output and calculating string lengths.           6. Finally, it calls the system interrupt to exit the process.         ### Constants   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_105</li> <li>undefined4 extraout_ECX_116</li> <li>undefined4 extraout_ECX_12</li> <li>undefined4 extraout_ECX_12</li> <li>undefined4 extraout_ECX_13</li> <li>int iVar11;</li> <li>undefined4 extraout_ECX_15</li> <li>undefined4 extraout_ECX_15</li> <li>undefined4 extraout_ECX_15</li> <li>undefined4 extraout_ECX_16</li> <li>undefined4 extraout_ECX_17</li> <li>undefined4 extraout_ECX_18</li> <li>undefined4 extraout_ECX_17</li> <li>undefined4 extraout_ECX_18</li> <li>undefined4 extraout_ECX_18</li> <li>undefined4 extraout_ECX_18</li> <li>undefined4 extraout_ECX_18</li> </ul>   | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.           6. Finally, it calls the system interrupt to exit the process.         ### Constants           This function refrences a registry path commonly associated with malware persistence   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX 02</li> <li>undefined4 extraout_ECX.02</li> <li>undefined4 extraout_ECX.02</li> <li>undefined4 extraout_ECX.03</li> <li>undefined4 extraout_ECX.04</li> <li>undefined4 extraout_ECX.05</li> <li>undefined4 extraout_ECX.06</li> <li>undefined4 extraout_ECX.06</li> <li>undefined4 extraout_ECX.06</li> <li>undefined4 extraout_ECX.06</li> <li>undefined4 extraout_ECX.06</li> <li>undefined4 extraout_ECX.07</li> <li>undefined4 extraout_ECX.08</li> <li>undefined4 extraout_ECX.09</li> <li>undefined4 extraout_ECX.09</li> <li>undefined4 extraout_ECX.10</li> <li>undefined4 extraout_ECX.11</li> <li>undefined4 extraout_ECX.12</li> <li>undefined4 extraout_ECX.13</li> <li>int iVar11;</li> <li>undefined4 extraout_ECX.14</li> <li>undefined4 extraout_ECX.15</li> <li>undefined4 extraout_ECX.16</li> <li>undefined4 extraout_ECX.17</li> <li>undefined4 extraout_ECX.18</li> <li>undefined4 extraout_ECX.19</li> <li>undefined4 extraout_ECX.14</li> <li>undefined4 extraout_ECX.15</li> <li>undefined4 extraout_ECX.14</li> <li>undefined4 extraout_ECX.15</li> <li>undefined4 extraout_ECX.16</li> <li>undefined4 extraout_ECX.17</li> <li>undefined4 extraout_ECX.18</li> <li>undefined4 extraout_ECX.14</li> <li>undefined4 extraout_ECX.15</li> <li>undefined4 extraout_ECX.16</li> <li>undefined4 extraout_ECX.17</li> <li>undefined4 extraout_ECX.18</li> <li>undefined4 extraout_ECX.19</li> <li>undefined4 extraout_ECX.14</li> <li>undefined4 extraout_ECX.15</li> </ul>   | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment           This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.         1. The function initializes some variables and retrieves the version information of the operating system.           2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.           4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.           6. Finally, it calls the system interrupt to exit the process.         ### Constants           This function refrences a registry path commonly associated with malware persistence   |  |  |  |  |  |  |
| <ul> <li>ushort *puVar9;</li> <li>undefined3 extraout_var;</li> <li>undefined4 uVar10;</li> <li>HANDLE hHeap;</li> <li>LPSTR lpMultiByteStr;</li> <li>undefined4 extraout_ECX;</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_03</li> <li>undefined4 extraout_ECX_04</li> <li>undefined4 extraout_ECX_05</li> <li>undefined4 extraout_ECX_06</li> <li>undefined4 extraout_ECX_07</li> <li>undefined4 extraout_ECX_08</li> <li>undefined4 extraout_ECX_08</li> <li>undefined4 extraout_ECX_10</li> <li>undefined4 extraout_ECX_11</li> <li>undefined4 extraout_ECX_12</li> <li>undefined4 extraout_ECX_13</li> <li>int iVar11;</li> <li>undefined4 extraout_ECX_140</li> <li>undefined4 extraout_ECX_151</li> <li>undefined4 extraout_ECX_162</li> <li>undefined4 extraout_ECX_1733</li> <li>undefined4 extraout_ECX_1743</li> <li>undefined4 extraout_ECX_1753</li> </ul>  | EOL Comment         Pre Comment           Post Comment         Plate Comment         Repeatable Comment   This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware. 1. The function initializes some variables and retrieves the version information of the operating system. 2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory. 3. Depending on the value of a variable, it executes different code paths to perform specific actions. 4. It converts some strings from wide character format to multi-byte format. 5. It calls additional functions to perform various operations, such as printing output and calculating string lengths. 6. Finally, it calls the system interrupt to exit the process. ### Constants This function refrences a registry path commonly associated with malware persistence   |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>undefined3 extraout_var;<br>0 undefined4 uVar10;<br>11 HANDLE hHeap;<br>22 LPSTR lpMultiByteStr;<br>33 undefined4 extraout_ECX,01<br>undefined4 extraout_ECX,02<br>undefined4 extraout_ECX,02<br>undefined4 extraout_ECX,03<br>undefined4 extraout_ECX,03<br>undefined4 extraout_ECX,04<br>undefined4 extraout_ECX,04<br>undefined4 extraout_ECX,05<br>undefined4 extraout_ECX,05<br>undefined4 extraout_ECX,05<br>undefined4 extraout_ECX,05<br>undefined4 extraout_ECX,05<br>undefined4 extraout_ECX,03<br>undefined4 extraout_ECX,03<br>undefined4 extraout_ECX,04<br>undefined4 extraout_ECX,05<br>undefined4 extraout_ECX,04<br>undefined4 extraout_ECX,04<br>undefined4 extraout_ECX,12<br>undefined4 extraout_ECX,13<br>int iVar11;<br>undefined4 extraout_ECX,14<br>undefined4 extraout_ECX,15<br>undefined4 extraout_ECX,16<br>undefined4 extraout_ECX,17<br>undefined4 extraout_ECX,18<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,19<br>undefined4 extraout_ECX,10<br>undefined4 extraout_ECX,10<br>unde | EOL Comment       Pre Comment         Post Comment       Plate Comment       Repeatable Comment         This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.       1. The function initializes some variables and retrieves the version information of the operating system.         2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.         4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.         6. Finally, it calls the system interrupt to exit the process.         ### Constants         This function refrences a registry path commonly associated with malware persistence         Add Annotation       Address                        |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>undefined3 extraout_var;<br>0 undefined4 uVar10;<br>11 HANDLE hHeap;<br>32 LPSTR lpMultiByteStr;<br>33 undefined4 extraout_ECX;<br>44 undefined4 extraout_ECX;<br>54 undefined4 extraout_ECX;<br>55 undefined4 extraout_ECX;<br>65 undefined4 extraout_ECX;<br>76 undefined4 extraout_ECX;<br>77 undefined4 extraout_ECX;<br>78 undefined4 extraout_ECX;<br>79 undefined4 extraout_ECX;<br>70 undefined4 extraout_ECX;<br>70 undefined4 extraout_ECX;<br>70 undefined4 extraout_ECX;<br>70 undefined4 extraout_ECX;<br>71 undefined4 extraout_ECX;<br>72 undefined4 extraout_ECX;<br>73 undefined4 extraout_ECX;<br>74 undefined4 extraout_ECX;<br>75 undefined4 extraout_ECX;<br>75 undefined4 extraout_ECX;<br>76 undefined4 extraout_ECX;<br>77 undefined4 extraout_ECX;<br>70 undefined4 extraout_ECX;<br>73 undefined4 extraout_ECX;<br>74 undefined4 extraout_ECX;<br>75 undefined4 extraout_ECX;<br>75 undefined4 extraout_ECX;<br>76 undefined4 extraout_ECX;<br>77 undefined4 extraout_ECX;<br>78 undefined4 extraout_ECX;<br>79 undefined4 extraout_ECX;<br>70 undefined4 extraout_ECX;<br>71 undefined4 extraout_ECX;<br>72 undefined4 extraout_ECX;<br>73 undefined4 extraout_ECX;<br>74 undefined4 extraout_ECX;<br>75 undefined4 extraout_ECX;<br>75 undefined4 extraout_ECX;<br>76 undefined4 extraout_ECX;<br>77 undefined4 extraout_EDX;<br>76 undefined4 extraout_EDX;<br>77 undefined4 extraout_EDX;<br>78 undefined4 extraout_EDX;<br>79 undefined4 extraout_EDX;<br>70 u          | EOL Comment       Pre Comment         Post Comment       Plate Comment       Repeatable Comment         This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.       I. The function initializes some variables and retrieves the version information of the operating system.         2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.       3. Depending on the value of a variable, it executes different code paths to perform specific actions.         4. It converts some strings from wide character format to multi-byte format.       5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.         6. Finally, it calls the system interrupt to exit the process.       ### Constants         This function refrences a registry path commonly associated with malware persistence       Add Annotation         Add Annotation       Address       • |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>undefined3 extraout_var;<br>undefined4 evtraout_var;<br>11 HANDLE hHeap;<br>22 LPSTR lpMultiByteStr;<br>23 undefined4 extraout_ECX;<br>24 undefined4 extraout_ECX_03<br>undefined4 extraout_ECX_03<br>26 undefined4 extraout_ECX_03<br>27 undefined4 extraout_ECX_03<br>28 undefined4 extraout_ECX_04<br>29 undefined4 extraout_ECX_05<br>20 undefined4 extraout_ECX_05<br>20 undefined4 extraout_ECX_06<br>21 undefined4 extraout_ECX_07<br>21 undefined4 extraout_ECX_07<br>22 undefined4 extraout_ECX_07<br>23 undefined4 extraout_ECX_07<br>24 undefined4 extraout_ECX_07<br>25 undefined4 extraout_ECX_07<br>26 undefined4 extraout_ECX_07<br>26 undefined4 extraout_ECX_07<br>27 undefined4 extraout_ECX_07<br>28 undefined4 extraout_ECX_07<br>29 undefined4 extraout_ECX_10<br>20 undefined4 extraout_ECX_11<br>20 undefined4 extraout_ECX_12<br>20 undefined4 extraout_ECX_13<br>28 undefined4 extraout_ECX_14<br>29 undefined4 extraout_ECX_15<br>20 undefined4 extraout_ECX_15<br>20 undefined4 extraout_ECX_16<br>20 undefined4 extraout_ECX_17<br>30 undefined4 extraout_ECX_16<br>31 undefined4 extraout_ECX_17<br>32 undefined4 extraout_ECX_16<br>33 undefined4 extraout_ECX_17<br>34 undefined4 extraout_ECX_16<br>35 undefined4 extraout_ECX_17<br>36 undefined4 extraout_ECX_16<br>37 undefined4 extraout_ECX_17<br>38 undefined4 extraout_ECX_18<br>39 undefined4 extraout_ECX_19<br>30 undefined4 extraout_EDX_00<br>30 undefined4 extraout_EDX_00<br>30 undefined4 extraout_EDX_00   | EOL Comment       Pre Comment         Post Comment       Plate Comment       Repeatable Comment         This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.       I. The function initializes some variables and retrieves the version information of the operating system.         2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.       3. Depending on the value of a variable, it executes different code paths to perform specific actions.         4. It converts some strings from wide character format to multi-byte format.       5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.         6. Finally, it calls the system interrupt to exit the process.       ### Constants         This function refrences a registry path commonly associated with malware persistence       Add Annotation         Add Annotation       Address       • |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>undefined3 extraout_var;<br>30 undefined4 uVar10;<br>31 HANDLE hHeap;<br>32 LPSTR lpMultiByteStr;<br>33 undefined4 extraout_ECX;<br>34 undefined4 extraout_ECX;<br>36 undefined4 extraout_ECX;<br>37 undefined4 extraout_ECX;<br>38 undefined4 extraout_ECX;<br>39 undefined4 extraout_ECX;<br>39 undefined4 extraout_ECX;<br>30 undefined4 extraout_ECX;<br>30 undefined4 extraout_ECX;<br>31 undefined4 extraout_ECX;<br>32 undefined4 extraout_ECX;<br>33 undefined4 extraout_ECX;<br>34 undefined4 extraout_ECX;<br>35 undefined4 extraout_ECX;<br>36 undefined4 extraout_ECX;<br>37 undefined4 extraout_ECX;<br>39 undefined4 extraout_ECX;<br>30 undefined4 extraout_ECX;<br>31 undefined4 extraout_ECX;<br>32 undefined4 extraout_ECX;<br>33 undefined4 extraout_ECX;<br>34 undefined4 extraout_ECX;<br>35 undefined4 extraout_ECX;<br>36 undefined4 extraout_ECX;<br>36 undefined4 extraout_ECX;<br>37 undefined4 extraout_ECX;<br>38 undefined4 extraout_ECX;<br>39 undefined4 extraout_ECX;<br>30 undefined4 extraout_ECX;<br>30 undefined4 extraout_EDX;<br>30           | EOL Comment       Pre Comment         Post Comment       Plate Comment         Repeatable Comment       Repeatable Comment         This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.       I. The function initializes some variables and retrieves the version information of the operating system.         2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.       3. Depending on the value of a variable, it executes different code paths to perform specific actions.         4. It converts some strings from wide character format to multi-byte format.       5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.         6. Finally, it calls the system interrupt to exit the process.         ### Constants         This function refrences a registry path commonly associated with malware persistence         Add Annotation       Address |  |  |  |  |  |  |
| 28 ushort *puVar9;<br>undefined3 extraout_var;<br>30 undefined4 uVar10;<br>31 HANDLE hHeap;<br>32 LPSTR lpMultiByteStr;<br>33 undefined4 extraout_ECX;<br>44 undefined4 extraout_ECX_00<br>54 undefined4 extraout_ECX_01<br>36 undefined4 extraout_ECX_03<br>38 undefined4 extraout_ECX_03<br>39 undefined4 extraout_ECX_04<br>40 undefined4 extraout_ECX_05<br>40 undefined4 extraout_ECX_06<br>41 undefined4 extraout_ECX_07<br>42 undefined4 extraout_ECX_07<br>43 undefined4 extraout_ECX_07<br>44 undefined4 extraout_ECX_07<br>45 undefined4 extraout_ECX_07<br>46 undefined4 extraout_ECX_07<br>47 undefined4 extraout_ECX_07<br>48 undefined4 extraout_ECX_07<br>49 undefined4 extraout_ECX_07<br>40 undefined4 extraout_ECX_07<br>40 undefined4 extraout_ECX_07<br>41 undefined4 extraout_ECX_10<br>42 undefined4 extraout_ECX_11<br>43 undefined4 extraout_ECX_12<br>44 undefined4 extraout_ECX_13<br>55 undefined4 extraout_ECX_15<br>51 undefined4 extraout_ECX_15<br>52 undefined4 extraout_ECX_15<br>53 undefined4 extraout_ECX_15<br>54 undefined4 extraout_ECX_15<br>55 undefined4 extraout_ECX_15<br>56 unt_Var12;<br>57 undefined4 extraout_EDX_00<br>59 undefined4 extraout_EDX_00  | EOL Comment       Pre Comment         Post Comment       Plate Comment       Repeatable Comment         This code is a function that is part of a malware sample. It performs various operations related to the execution of the malware.       I. The function initializes some variables and retrieves the version information of the operating system.         2. It calls several other functions to perform different tasks, such as setting up registry keys, executing shell commands, and allocating memory.         3. Depending on the value of a variable, it executes different code paths to perform specific actions.         4. It converts some strings from wide character format to multi-byte format.         5. It calls additional functions to perform various operations, such as printing output and calculating string lengths.         6. Finally, it calls the system interrupt to exit the process.         ### Constants         This function refrences a registry path commonly associated with malware persistence         Add Annotation       Address                        |  |  |  |  |  |  |

64

STZE T dwBvtes:

## **Future Work**

Analyst in the loop

- YARA/Sigma/Snort
- AutoAnalyst: Using a predefined list of Windows APIs, the auto analyst iterates, gets calling cross xrefs, and predecomiles & generates IIm analysis results, follows LLM generated tasks from seed exploration, calls dynamic analysis as needed.
- Reporting: Generate/export a report in markdown for threat intelligence analysis
- Smaller models: Quantize existing, explore 7 & 13 B versions.

```
Decompile: MaliciousFunction - (c373d37b5a9427a18dbf93d519968d9fda04f2a262f424d0611830764c8cc69c)
2 /* WARNING: Globals starting with '_' overlap smaller symbols at the same address */
   /* This function is a malicious function that is designed to perform a variety of malicious
3
     activities, including:
4
6
     1. Loading and executing a kernel32.dll function
     2. Enumerating logical drives and creating threads to scan for specific files
     3. Enumerating network resources and creating threads to scan for specific files
     4. Enumerating network shares and creating threads to scan for specific files
     5. Creating a new thread to perform a malicious activity */
10
11
12 void MaliciousFunction(void)
13
14 {
15
    WCHAR WVarl;
    int *piVar2;
16
17
    bool bVar3;
18
    int iVar4;
    DWORD nBufferLength;
19
    LPWSTR lpBuffer;
20
21
    UINT UVar5;
22
    int iVar6;
23
    LPWSTR lpStringl;
24
     HANDLE pvVar7;
25
    int iVar8:
26
     undefined4 extraout_ECX;
     undefined4 extraout ECX 00;
27
28
     undefined4 uVar9;
    undefined4 extraout EDX;
29
     undefined4 extraout EDX 00;
30
31
    undefined4 uVar10;
32
    LPCWSTR lpRootPathName;
33
    int iVarll:
34
     uint uVar12;
     int *piVarl3;
     undefined4 uStack le0;
     WCHAR aWStack ldc [34];
37
     undefined local 198 [4];
38
39
     undefined auStack_194 [400].
40
41
     memset(local 198,0,400);
42
     DAT_0042f894 = LoadAndExecuteKernel32Function(0x16d34cd7, &DAT_0042f0fc, 0x14);
     iVar4 = Ordinal 115(0x202, auStack 194);
43
44
    if (iVar4 != 0) {
45
      Ordinal 116();
46
47
     nBufferLength = GetLogicalDriveStringsW(0, (LPWSTR)0x0);
48
     lpBuffer = (LPWSTR)FUN_00415650(nBufferLength * 2);
     GetLogicalDriveStringsW(nBufferLength, lpBuffer);
49
50
     WVarl = *lpBuffer;
51
     lpRootPathName = lpBuffer;
52
     while (WVarl != L'\0') {
53
      bVar3 = false:
54
      UVar5 = GetDriveTypeW(lpRootPathName);
55
      wsprintfW(aWStack ldc,L"\\??\\%.2s\\",lpRootPathName);
56
      if (((UVar5 == 2) || (UVar5 == 3)) || (UVar5 == 4)) {
57
        if ((DAT 004342e8 != 0) && (iVar4 = 0, 0 < DAT 004342e8)) {
58
           do {
```

### Conclusion

Did it work?

- We can fine-tune a model to improve results for reverse engineering tasks
- Model size matters and to get decent results for summary and step-by-step tasks 34B parameter models were required. This may change with better training data, more training steps at higher context limit.
- Matching the pre-training prompts is important, so is finding a decent semi-working starting point.
- Context should be treated like a budget where we maximize the context on all calls. When there is additional space we can add xrefs, dref memory, include dynamic analysis tracing.
- When context is tight due to large functions chunking & summarization helps.
- Analyst in the loop for continuous learning. Make the labeling process an improvement to workflow to drive adoption. Reward contribution.
- Confident hallucinations on Community model (also repeating, not knowing when to stop), susceptible to injection / misdirection.

```
56 FUN_00419450();
```

- 57 local\_28 = (PVOID) 0x417ce5;
- 58 FUN\_00419490(local\_8e8,0,0x820);
- 59 local\_28 = (PVOID) 0x417cfc;
- 60 DVar1 = GetModuleFileNameW((HMODULE)0x0,local\_1928,0x40f);

```
61 local 28 = (PVOID) 0x0;
```

- 62 pvVar2 = (HANDLE) (\*DAT\_0043b498) ();
- 63 nNumberOfBytesToRead = GetFileSize(pvVar2, (LPDWORD)0x0);
- 64 lpBuffer = (LPVOID) FUN\_00418870(nNumberOfBytesToRead);
- 65 ReadFile (pvVar2, lpBuffer, nNumberOfBytesToRead, (LPDWORD) & stack0xffffffe8, (LPOVERLAPPED) 0x0);

```
66 CloseHandle(pvVar2);
```

```
67 if (unaff EDI != 0) {
```

68 iVar3 = 0;

```
69 iVar6 = 0;
```

```
70 if (0 < (int)DVar1) {
```

```
71 do {
```

73

74

75

77

```
72 if (local_1928[iVar3] == L'\\') {
```

iVar6 = iVar3;

}

iVar3 = iVar3 + 1;

```
76 } while (iVar3 < (int)DVar1);</pre>
```

78 RegOpenKeyExW((HKEY)0x80000001,L"Software\\Microsoft\\Windows\\CurrentVersion\\Run",0,0xf003f

#### •FAILURES

| Pecompile: `malware_dropper` |  | (0a1dbcff63619c4d9072484bb17b3d06300504e836e42df98eadf57e7ad0d0be) |
|------------------------------|--|--|
|------------------------------|--|--|

3 /\* This function is a malware dropper that downloads and executes a payload from a remote server. uses the Windows API to download the payload using the `URLDownloadToFileW` function, and then 4 executes it using the `ShellExecuteExW` function. The payload is encrypted using the 5 `CryptDecrypt` function, and the decryption key is stored in the `DAT 0043af64` variable. 6 7 8 The function first calls the `FUN 00419450` function, which is not shown in the decompiled cod This function is likely responsible for setting up the malware's environment, such as creating 9 10 new process or thread. 11 The function then calls the `GetModuleFileNameW` function to get the path of the current modul 12 (i.e., the malware executable). It then calls the `GetFileSize` function to get the size of th 13 14 module, and uses this size to allocate a buffer for the payload. 15 The function then calls the `URLDownloadToFileW` function to download the payload from the rem 16 17 server. The `URLDownloadToFileW` function takes the URL of the payload as an argument, and wri 18 the downloaded payload to the buffer allocated by the `FUN 00418870` function. 19 The function then calls the `CryptDecrypt` function to decrypt the payload using the decryption 20 21 key stored in `DAT 0043af64`. The decrypted payload is then executed using the `ShellExecuteEx' 22 function. 23 The function also uses the `RegOpenKeyExW` and `RegSetValueExW` functions to set a registry va 24

that points to the payload. This registry value is used to persist the malware on the system.

#### void FUN\_140001000(void)

```
HRESULT HVar1;
```

```
longlong lVar2; uses the `URLDownloadToFileA` API to download the file, and if the download is successful, i
char *pcVar3;
CHAR *pCVar4;
undefined auStackY_88 [32];
CHAR local_50 [24];
CHAR local_38 [24];
ulonglong local_20;
local 20 = DAT 140003078 ^ (ulonglong)auStackY 88;
pcVar3 = s_https___www_google_com_140003000;
pCVar4 = local_38;
for (lVar2 = 0x17; lVar2 != 0; lVar2 = lVar2 + -1) {
  *pCVar4 = *pcVar3;
 pcVar3 = pcVar3 + 1;
  pCVar4 = pCVar4 + 1;
3
pcVar3 = s_C_mpgoogle_html_140003018;
pCVar4 = local 50;
for (lVar2 = 0x11; lVar2 != 0; lVar2 = lVar2 + -1) {
 *pCVar4 = *pcVar3;
 pcVar3 = pcVar3 + 1;
 pCVar4 = pCVar4 + 1;
HVar1 = URLDownloadToFileA((LPUNKNOWN)0x0,local_38,local_50,0,(LPBINDSTATUSCALLBACK)0x0);
if (HVar1 == 0) {
  printf(s_File_downloaded_successfully__140003030);
else {
 printf(s_Failed_to_download_file__140003050);
FUN_1400011f0(local_20 ^ (ulonglong)auStackY_88);
return;
```

### **Function Summary:**

This function appears to be a malicious payload that is designed to download a file from a remote server and execute it on the target system. The function uses the URLDownloadToFileA API to download the file, and if the download is successful, it prints a message to console indicating that the file was downloaded successfully. If the download fails, it prints a message indicating that the file could n downloaded.

The function also includes a call to another function, FUN\_1400011F0, which is likely used to clean up any resources or data that were during the execution of the payload.

### **New Function Name:**

Payload\_DownloadAndExecute

### Demo?

#### How it's done

- Briefly demonstrate how your plugin and LLM assist in reverse engineering.
- . Show a few examples or case studies.

(1) abuse\_api\_thread32first.exe, checkremotedebugger--5d,

(2) /review

(3) /curate

•FAILURES:

• • •

### Q&A

Any questions?

• http://lupine.richards.ai:9001

