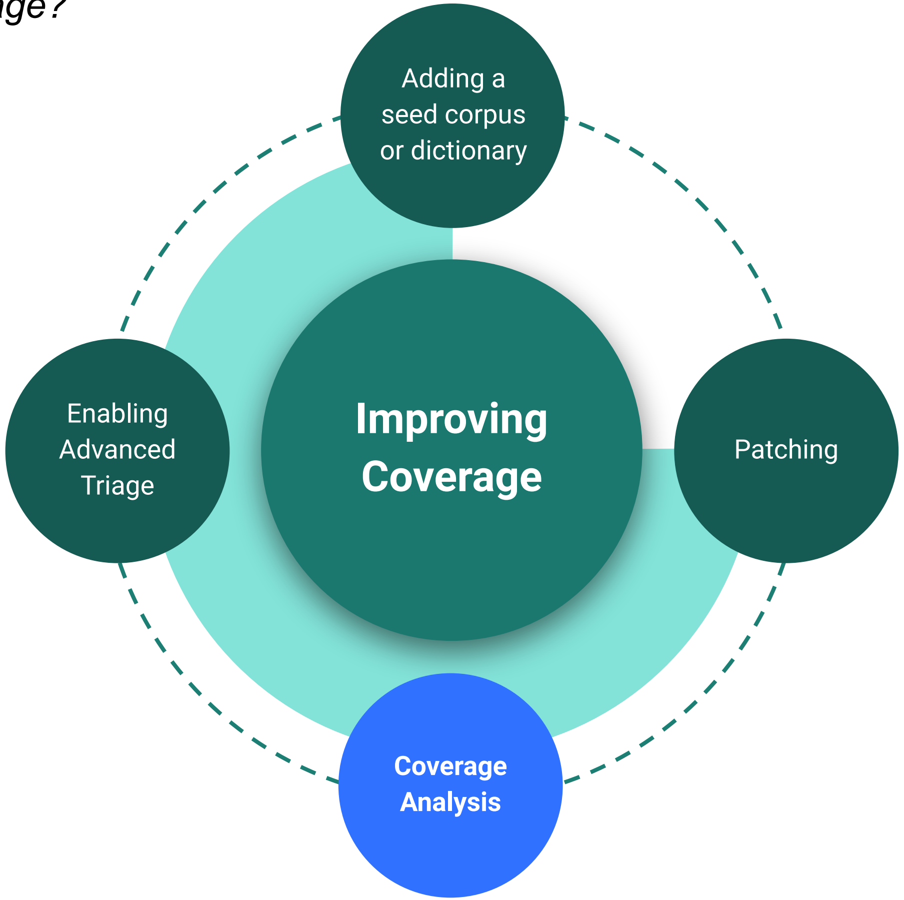
ForAllSecure

Binary Analysis

And Patching!

How can I fuzz better?

And improve overall coverage?



What is Coverage Analysis?

Line Coverage ⓘ
*requires debug symbols

- ➔ Represents the percentage of source code lines hit by test cases out of the total

Function Coverage ⓘ

- ➔ Represents the percentage of functions hit by test cases out of the total

Dynamic Block Coverage ⓘ

- ➔ Represents the percentage of code blocks (sections of code with one entry and one exit point) hit by test cases out of the total

What is Coverage Analysis?

```
project: ffmpeg
target: ffmpeg
image: ghcr.io/xansec/ffmpeg:latest
advanced_triage: true


tasks:
 - name: exploitability_factors
 - name: regression_testing
 - name: behavior_testing
 - name: coverage_analysis

cmds:

 - cmd: /ffmpeg -i @@ -f null ignore.mp4
   env:
     LD LIBRARY PATH: /ffmpeg-libs
   dictionary: /dictionaries/mp4.dict
```

Viewing Coverage in Ghidra

1. Install Ghidra:

https://ghidra-sre.org/

https://ghidra-sre.org/InstallationGuide.html#Platforms

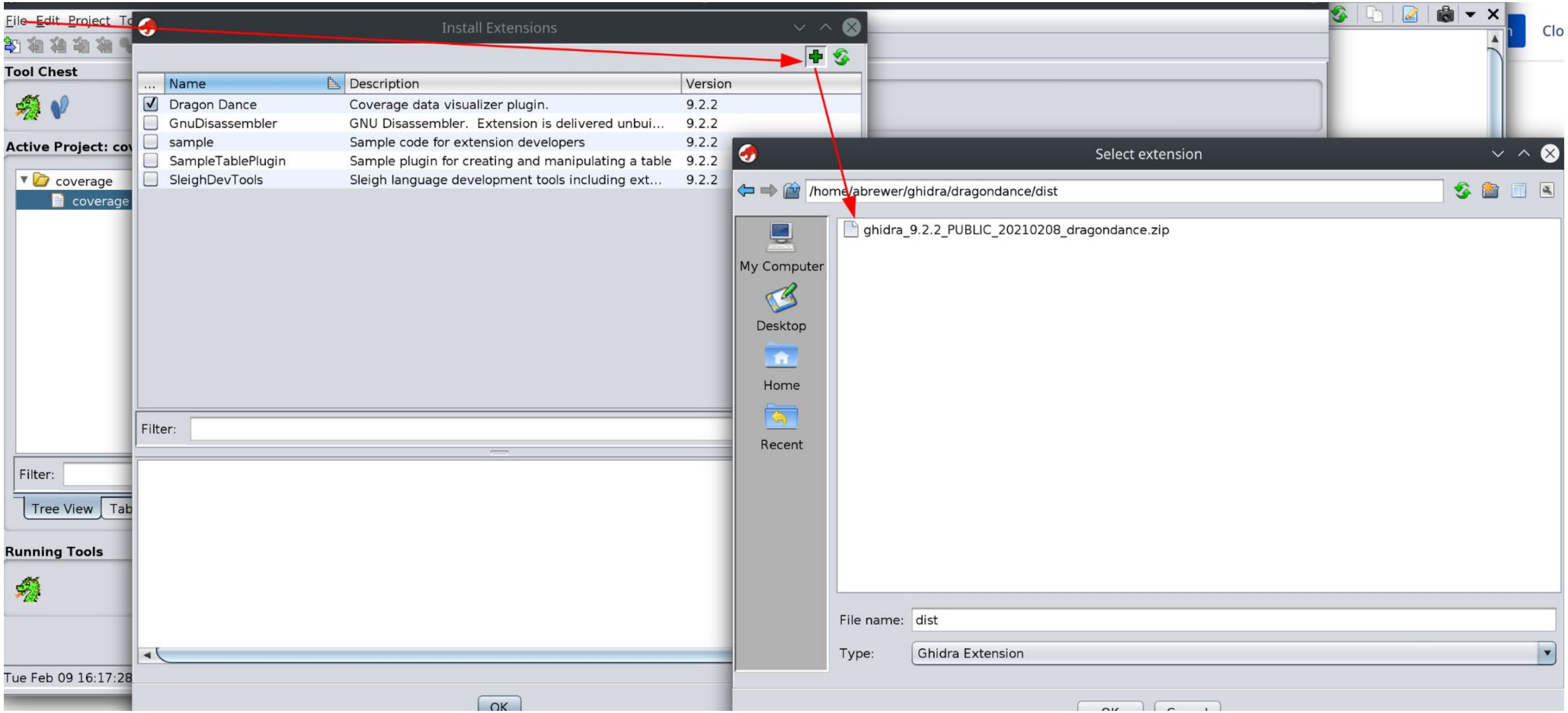
2. Install Dragon Dance.

```
$ git clone https://github.com/0ffffffffh/dragondance.git
$ cd dragondance
$ gradle -PGHIDRA_INSTALL_DIR=<your_ghidra_installation_dir>
```

3. Run Ghidra:

```
$ ./ghidraRun
```

Viewing Coverage in Ghidra



Install Extensions

...	Name	Description	Version
✓	Dragon Dance	Coverage data visualizer plugin.	9.2.2
	GnuDisassembler	GNU Disassembler. Extension is delivered unbui...	9.2.2
	sample	Sample code for extension developers	9.2.2
	SampleTablePlugin	Sample plugin for creating and manipulating a table	9.2.2
	SleighDevTools	Sleigh language development tools including ext...	9.2.2

Select extension

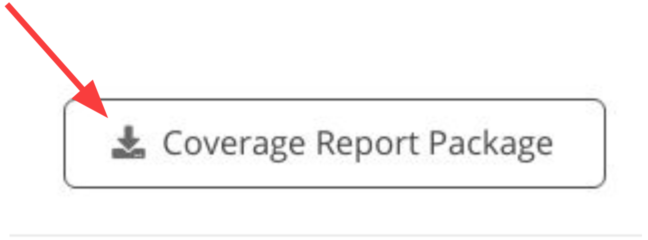
/home/abrewer/ghidra/dragondance/dist

ghidra_9.2.2_PUBLIC_20210208_dragondance.zip

File name: dist

Type: Ghidra Extension

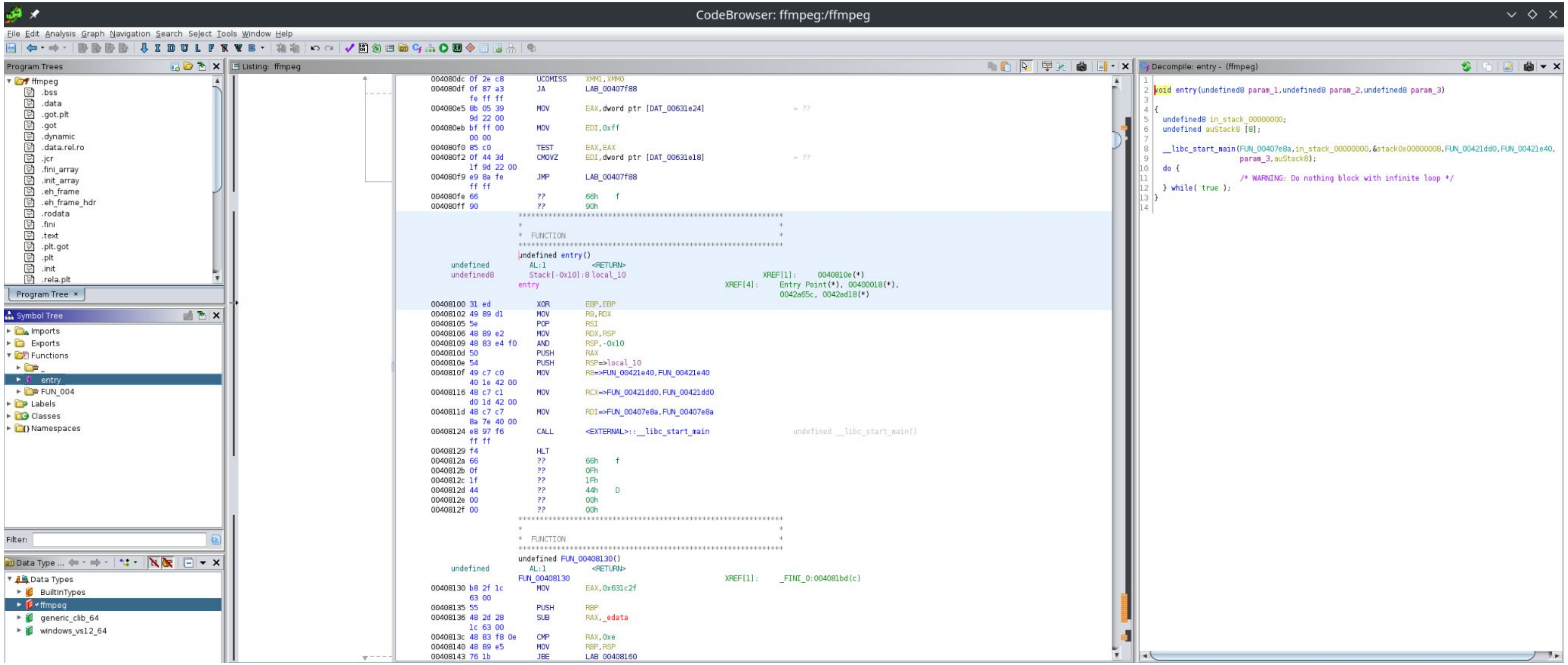
Viewing Coverage in Ghidra



or

```
$ mayhem sync .
```

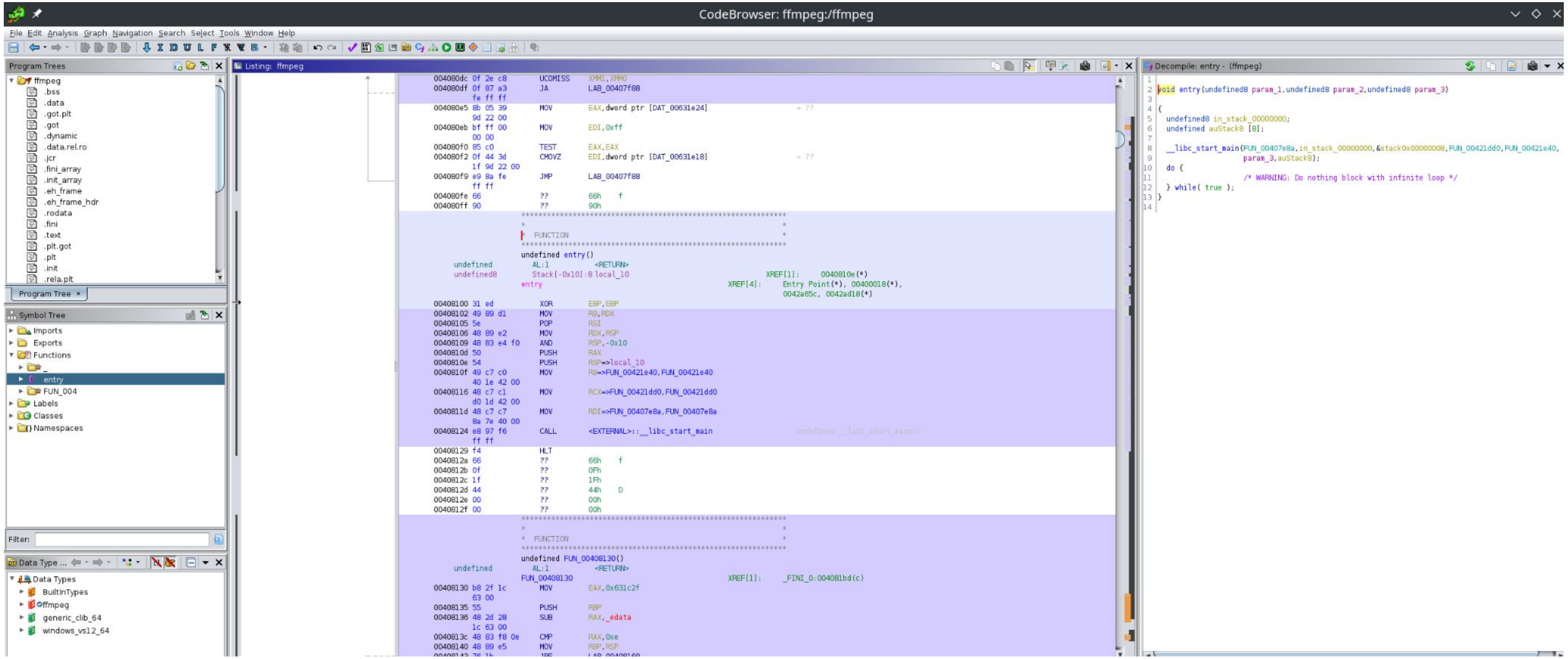
Viewing Coverage in Ghidra



```
004080dc 0f 2e c8       UCOMISS    XMM1,XMM0
004080df 0f 87 a3       JA         LAB_00407f88
         fe ff ff
004080e5 8b 05 39       MOV        EAX,dword ptr [DAT_00631e24]              = ??
         9d 22 00
004080eb bf ff 00       MOV        EDI,0xff
         00 00
004080f0 85 c0          TEST       EAX,EAX
004080f2 0f 44 3d       CMOVZ      EDI,dword ptr [DAT_00631e18]              = ??
         1f 9d 22 00
004080f9 e9 8a fe       JMP        LAB_00407f88
         ff ff
004080fe 66             ??         66h    f
004080ff 90             ??         90h
                    **************************************************************
                    *                                                            *
                    *  FUNCTION                                                  *
                    **************************************************************
                    undefined entry()
     undefined          AL:1           <RETURN>
     undefined8         Stack[-0x10]:8 local_10                         XREF[1]:     0040810e(*)
                    entry                                         XREF[4]:     Entry Point(*), 00400018(*),
                                                                               0042a65c, 0042ad18(*)
00408100 31 ed          XOR        EBP,EBP
00408102 49 89 d1       MOV        R9,RDX
00408105 5e             POP        RSI
00408106 48 89 e2       MOV        RDX,RSP
00408109 48 83 e4 f0    AND        RSP,-0x10
0040810d 50             PUSH       RAX
0040810e 54             PUSH       RSP=>local_10
0040810f 49 c7 c0       MOV        R8=>FUN_00421e40,FUN_00421e40
         40 1e 42 00
00408116 48 c7 c1       MOV        RCX=>FUN_00421dd0,FUN_00421dd0
         d0 1d 42 00
0040811d 48 c7 c7       MOV        RDI=>FUN_00407e8a,FUN_00407e8a
         8a 7e 40 00
00408124 e8 97 f6       CALL       <EXTERNAL>::__libc_start_main             undefined __libc_start_main()
         ff ff
00408129 f4             HLT
0040812a 66             ??         66h    f
0040812b 0f             ??         0Fh
0040812c 1f             ??         1Fh
0040812d 44             ??         44h    D
0040812e 00             ??         00h
0040812f 00             ??         00h
                    **************************************************************
                    *                                                            *
                    *  FUNCTION                                                  *
                    **************************************************************
                    undefined FUN_00408130()
     undefined          AL:1           <RETURN>
                    FUN_00408130                                  XREF[1]:     _FINI_0:004081bd(c)
00408130 b8 2f 1c       MOV        EAX,0x631c2f
         63 00
00408135 55             PUSH       RBP
00408136 48 2d 28       SUB        RAX,_edata
         1c 63 00
0040813c 48 83 f8 0e    CMP        RAX,0xe
00408140 48 89 e5       MOV        RBP,RSP
00408143 76 1b          JBE        LAB_00408160
```

```c

void entry(undefined8 param_1,undefined8 param_2,undefined8 param_3)

{
  undefined8 in_stack_00000000;
  undefined auStack8 [8];

  __libc_start_main(FUN_00407e8a,in_stack_00000000,&stack0x00000008,FUN_00421dd0,FUN_00421e40,
                    param_3,auStack8);
  do {
                    /* WARNING: Do nothing block with infinite loop */
  } while( true );
}

```

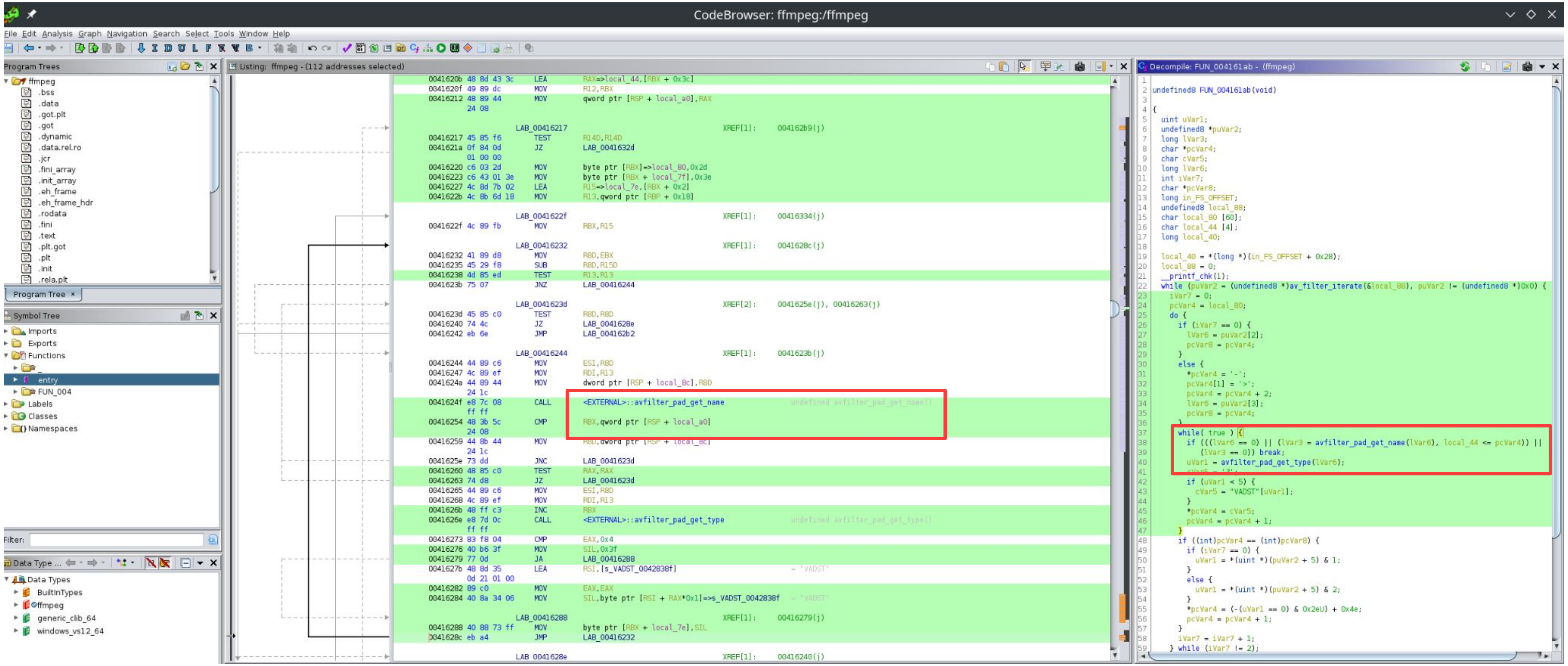
Viewing Coverage in Ghidra



```
004080dc 0f 2e c8       UCOMISS   XMM1,XMM0
004080df 0f 87 a3       JA        LAB_00407f88
         fe ff ff
004080e5 8b 05 39       MOV       EAX,dword ptr [DAT_00631e24]          = ??
         9d 22 00
004080eb bf ff 00       MOV       EDI,0xff
         00 00
004080f0 85 c0          TEST      EAX,EAX
004080f2 0f 44 3d       CMOVZ     EDI,dword ptr [DAT_00631e18]          = ??
         1f 9d 22 00
004080f9 e9 8a fe       JMP       LAB_00407f88
         ff ff
004080fe 66             ??        66h    f
004080ff 90             ??        90h
                    **************************************************************
                    *                                                            *
                    *  FUNCTION                                                  *
                    **************************************************************
                    undefined entry()
     undefined        AL:1           <RETURN>
     undefined8       Stack[-0x10]:8 local_10                    XREF[1]:     0040810e(*)
                    entry                                      XREF[4]:     Entry Point(*), 00400018(*),
                                                                            0042a65c, 0042ad18(*)
00408100 31 ed          XOR       EBP,EBP
00408102 49 89 d1       MOV       R9,RDX
00408105 5e             POP       RSI
00408106 48 89 e2       MOV       RDX,RSP
00408109 48 83 e4 f0    AND       RSP,-0x10
0040810d 50             PUSH      RAX
0040810e 54             PUSH      RSP=>local_10
0040810f 49 c7 c0       MOV       R8=>FUN_00421e40,FUN_00421e40
         40 1e 42 00
00408116 48 c7 c1       MOV       RCX=>FUN_00421dd0,FUN_00421dd0
         d0 1d 42 00
0040811d 48 c7 c7       MOV       RDI=>FUN_00407e8a,FUN_00407e8a
         8a 7e 40 00
00408124 e8 97 f6       CALL      <EXTERNAL>::__libc_start_main          undefined __libc_start_main()
         ff ff
00408129 f4             HLT
0040812a 66             ??        66h    f
0040812b 0f             ??        0Fh
0040812c 1f             ??        1Fh
0040812d 44             ??        44h    D
0040812e 00             ??        00h
0040812f 00             ??        00h
                    **************************************************************
                    *                                                            *
                    *  FUNCTION                                                  *
                    **************************************************************
                    undefined FUN_00408130()
     undefined        AL:1           <RETURN>
                    FUN_00408130                               XREF[1]:     _FINI_0:004081bd(c)
00408130 b8 2f 1c       MOV       EAX,0x631c2f
         63 00
00408135 55             PUSH      RBP
00408136 48 2d 28       SUB       RAX,_edata
         1c 63 00
0040813c 48 83 f8 0e    CMP       RAX,0xe
00408140 48 89 e5       MOV       RBP,RSP
```

```

void entry(undefined8 param_1,undefined8 param_2,undefined8 param_3)

{
  undefined8 in_stack_00000000;
  undefined auStack8 [8];

  __libc_start_main(FUN_00407e8a,in_stack_00000000,&stack0x00000008,FUN_00421dd0,FUN_00421e40,
                    param_3,auStack8);
  do {
                    /* WARNING: Do nothing block with infinite loop */
  } while( true );
}

```

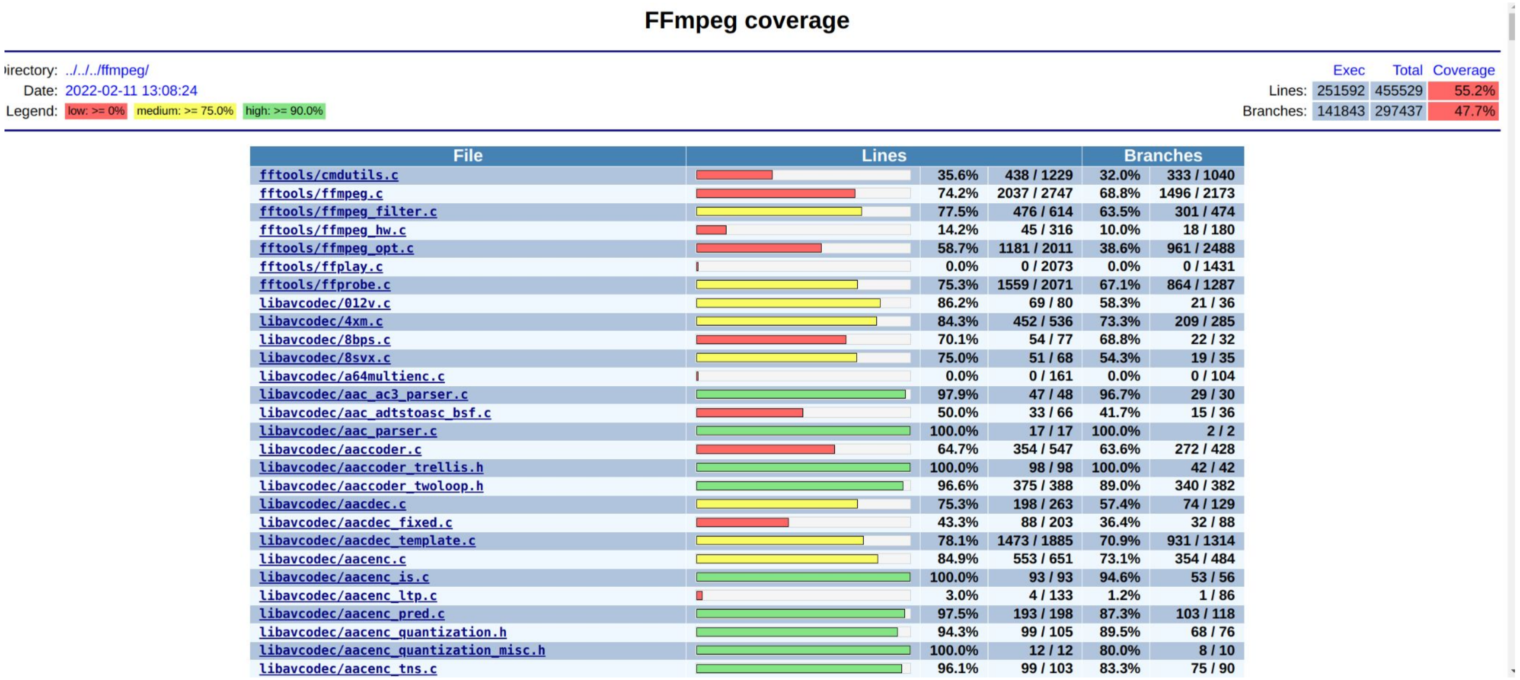
Viewing Coverage in Ghidra



```
0041620b 48 8d 43 3c     LEA     RAX=>local_44,[RBX + 0x3c]
0041620f 49 89 dc        MOV     R12,RBX
00416212 48 89 44        MOV     qword ptr [RSP + local_a0],RAX
         24 08

                   LAB_00416217                          XREF[1]:     004162b9(j)
00416217 45 85 f6        TEST    R14D,R14D
0041621a 0f 84 0d        JZ      LAB_0041632d
         01 00 00
00416220 c6 03 2d        MOV     byte ptr [RBX]=>local_80,0x2d
00416223 c6 43 01 3e     MOV     byte ptr [RBX + local_7f],0x3e
00416227 4c 8d 7b 02     LEA     R15=>local_7e,[RBX + 0x2]
0041622b 4c 8b 6d 18     MOV     R13,qword ptr [RBP + 0x18]

                   LAB_0041622f                          XREF[1]:     00416334(j)
0041622f 4c 89 fb        MOV     RBX,R15

                   LAB_00416232                          XREF[1]:     0041628c(j)
00416232 41 89 d8        MOV     R8D,EBX
00416235 45 29 f8        SUB     R8D,R15D
00416238 4d 85 ed        TEST    R13,R13
0041623b 75 07           JNZ     LAB_00416244

                   LAB_0041623d                          XREF[2]:     0041625e(j), 00416263(j)
0041623d 45 85 c0        TEST    R8D,R8D
00416240 74 4c           JZ      LAB_0041628e
00416242 eb 6e           JMP     LAB_004162b2

                   LAB_00416244                          XREF[1]:     0041623b(j)
00416244 44 89 c6        MOV     ESI,R8D
00416247 4c 89 ef        MOV     RDI,R13
0041624a 44 89 44        MOV     dword ptr [RSP + local_8c],R8D
         24 1c
0041624f e8 7c 08        CALL    <EXTERNAL>::avfilter_pad_get_name          undefined avfilter_pad_get_name()
         ff ff
00416254 48 3b 5c        CMP     RBX,qword ptr [RSP + local_a0]
         24 08
00416259 44 8b 44        MOV     R8D,dword ptr [RSP + local_8c]
         24 1c
0041625e 73 dd           JNC     LAB_0041623d
00416260 48 85 c0        TEST    RAX,RAX
00416263 74 d8           JZ      LAB_0041623d
00416265 44 89 c6        MOV     ESI,R8D
00416268 4c 89 ef        MOV     RDI,R13
0041626b 48 ff c3        INC     RBX
0041626e e8 7d 0c        CALL    <EXTERNAL>::avfilter_pad_get_type          undefined avfilter_pad_get_type()
         ff ff
00416273 83 f8 04        CMP     EAX,0x4
00416276 40 b6 3f        MOV     SIL,0x3f
00416279 77 0d           JA      LAB_00416288
0041627b 48 8d 35        LEA     RSI,[s_VADST_0042838f]                     = "VADST"
         0d 21 01 00
00416282 89 c0           MOV     EAX,EAX
00416284 40 8a 34 06     MOV     SIL,byte ptr [RSI + RAX*0x1]=>s_VADST_0042838f   = "VADST"

                   LAB_00416288                          XREF[1]:     00416279(j)
00416288 40 88 73 ff     MOV     byte ptr [RBX + local_7e],SIL
0041628c eb a4           JMP     LAB_00416232

                   LAB_0041628e                          XREF[1]:     00416240(j)
```

```c

undefined8 FUN_004161ab(void)

{
  uint uVar1;
  undefined8 *puVar2;
  long lVar3;
  char *pcVar4;
  char cVar5;
  long lVar6;
  int iVar7;
  char *pcVar8;
  long in_FS_OFFSET;
  undefined8 local_88;
  char local_80 [60];
  char local_44 [4];
  long local_40;

  local_40 = *(long *)(in_FS_OFFSET + 0x28);
  local_88 = 0;
  __printf_chk(1);
  while (puVar2 = (undefined8 *)av_filter_iterate(&local_88), puVar2 != (undefined8 *)0x0) {
    iVar7 = 0;
    pcVar4 = local_80;
    do {
      if (iVar7 == 0) {
        lVar6 = puVar2[2];
        pcVar8 = pcVar4;
      }
      else {
        *pcVar4 = '-';
        pcVar4[1] = '>';
        pcVar4 = pcVar4 + 2;
        lVar6 = puVar2[3];
        pcVar8 = pcVar4;
      }
      while( true ) {
        if (((lVar6 == 0) || (lVar3 = avfilter_pad_get_name(lVar6), local_44 <= pcVar4)) ||
           (lVar3 == 0)) break;
        uVar1 = avfilter_pad_get_type(lVar6);
        cVar5 = '?';
        if (uVar1 < 5) {
          cVar5 = "VADST"[uVar1];
        }
        *pcVar4 = cVar5;
        pcVar4 = pcVar4 + 1;
      }
      if ((int)pcVar4 == (int)pcVar8) {
        if (iVar7 == 0) {
          uVar1 = *(uint *)(puVar2 + 5) & 1;
        }
        else {
          uVar1 = *(uint *)(puVar2 + 5) & 2;
        }
        *pcVar4 = (-(uVar1 == 0) & 0x2eU) + 0x4e;
        pcVar4 = pcVar4 + 1;
      }
      iVar7 = iVar7 + 1;
    } while (iVar7 != 2);
```

Bonus: Google-Hacking



FFmpeg coverage

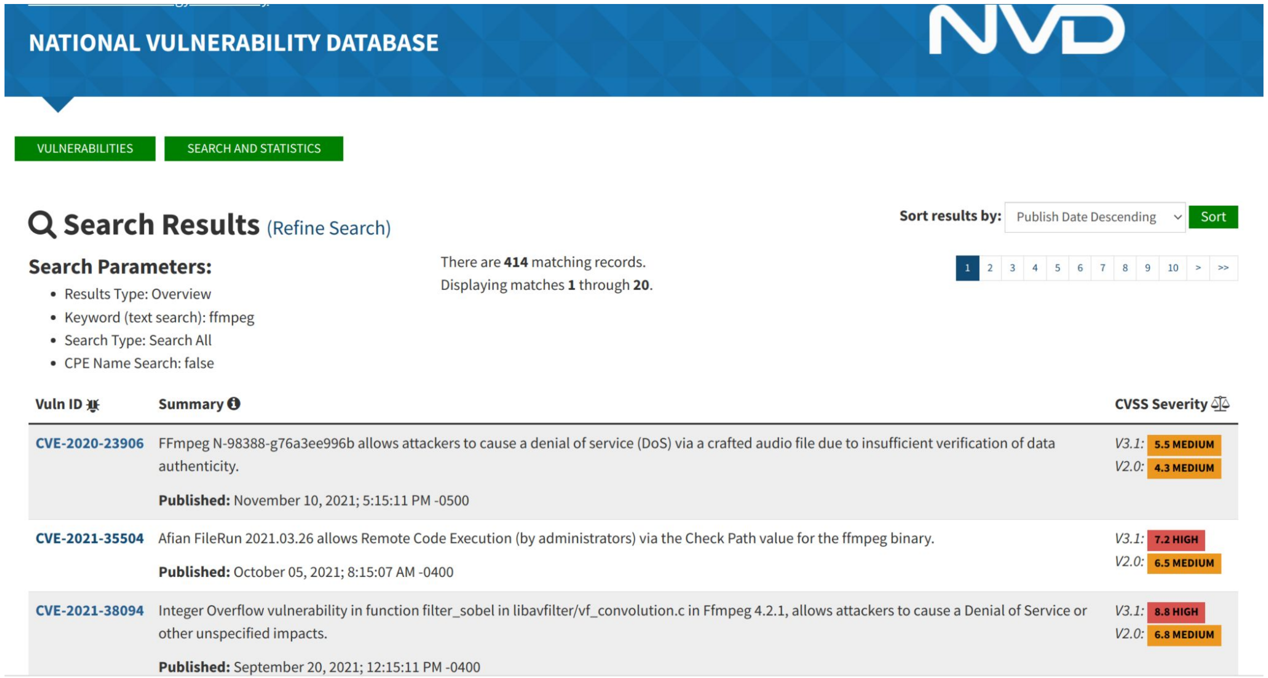
irectory: ../../../ffmpeg/
Date: 2022-02-11 13:08:24
Legend: low: >= 0% medium: >= 75.0% high: >= 90.0%

	Exec	Total	Coverage
Lines:	251592	455529	55.2%
Branches:	141843	297437	47.7%

File	Lines			Branches	
fftools/cmdutils.c	🟥	35.6%	438 / 1229	32.0%	333 / 1040
fftools/ffmpeg.c	🟥	74.2%	2037 / 2747	68.8%	1496 / 2173
fftools/ffmpeg_filter.c	🟨	77.5%	476 / 614	63.5%	301 / 474
fftools/ffmpeg_hw.c	🟥	14.2%	45 / 316	10.0%	18 / 180
fftools/ffmpeg_opt.c	🟥	58.7%	1181 / 2011	38.6%	961 / 2488
fftools/ffplay.c	🟥	0.0%	0 / 2073	0.0%	0 / 1431
fftools/ffprobe.c	🟨	75.3%	1559 / 2071	67.1%	864 / 1287
libavcodec/012v.c	🟨	86.2%	69 / 80	58.3%	21 / 36
libavcodec/4xm.c	🟨	84.3%	452 / 536	73.3%	209 / 285
libavcodec/8bps.c	🟥	70.1%	54 / 77	68.8%	22 / 32
libavcodec/8svx.c	🟨	75.0%	51 / 68	54.3%	19 / 35
libavcodec/a64multienc.c	🟥	0.0%	0 / 161	0.0%	0 / 104
libavcodec/aac_ac3_parser.c	🟩	97.9%	47 / 48	96.7%	29 / 30
libavcodec/aac_adtstoasc_bsf.c	🟥	50.0%	33 / 66	41.7%	15 / 36
libavcodec/aac_parser.c	🟩	100.0%	17 / 17	100.0%	2 / 2
libavcodec/aaccoder.c	🟥	64.7%	354 / 547	63.6%	272 / 428
libavcodec/aaccoder_trellis.h	🟩	100.0%	98 / 98	100.0%	42 / 42
libavcodec/aaccoder_twoloop.h	🟩	96.6%	375 / 388	89.0%	340 / 382
libavcodec/aacdec.c	🟨	75.3%	198 / 263	57.4%	74 / 129
libavcodec/aacdec_fixed.c	🟥	43.3%	88 / 203	36.4%	32 / 88
libavcodec/aacdec_template.c	🟨	78.1%	1473 / 1885	70.9%	931 / 1314
libavcodec/aacenc.c	🟨	84.9%	553 / 651	73.1%	354 / 484
libavcodec/aacenc_is.c	🟩	100.0%	93 / 93	94.6%	53 / 56
libavcodec/aacenc_ltp.c	🟥	3.0%	4 / 133	1.2%	1 / 86
libavcodec/aacenc_pred.c	🟩	97.5%	193 / 198	87.3%	103 / 118
libavcodec/aacenc_quantization.h	🟩	94.3%	99 / 105	89.5%	68 / 76
libavcodec/aacenc_quantization_misc.h	🟩	100.0%	12 / 12	80.0%	8 / 10
libavcodec/aacenc_tns.c	🟩	96.1%	99 / 103	83.3%	75 / 90

source: http://coverage.ffmpeg.org/

Bonus: Google-Hacking



NATIONAL VULNERABILITY DATABASE

Search Results (Refine Search)

Sort results by: Publish Date Descending Sort

Search Parameters:

- Results Type: Overview
- Keyword (text search): ffmpeg
- Search Type: Search All
- CPE Name Search: false

There are **414** matching records.
Displaying matches **1** through **20**.

1 2 3 4 5 6 7 8 9 10 > >>

Vuln ID	Summary	CVSS Severity
CVE-2020-23906	FFmpeg N-98388-g76a3ee996b allows attackers to cause a denial of service (DoS) via a crafted audio file due to insufficient verification of data authenticity. **Published:** November 10, 2021; 5:15:11 PM -0500	V3.1: 5.5 MEDIUM V2.0: 4.3 MEDIUM
CVE-2021-35504	Afian FileRun 2021.03.26 allows Remote Code Execution (by administrators) via the Check Path value for the ffmpeg binary. **Published:** October 05, 2021; 8:15:07 AM -0400	V3.1: 7.2 HIGH V2.0: 6.5 MEDIUM
CVE-2021-38094	Integer Overflow vulnerability in function filter_sobel in libavfilter/vf_convolution.c in Ffmpeg 4.2.1, allows attackers to cause a Denial of Service or other unspecified impacts. **Published:** September 20, 2021; 12:15:11 PM -0400	V3.1: 8.8 HIGH V2.0: 6.8 MEDIUM

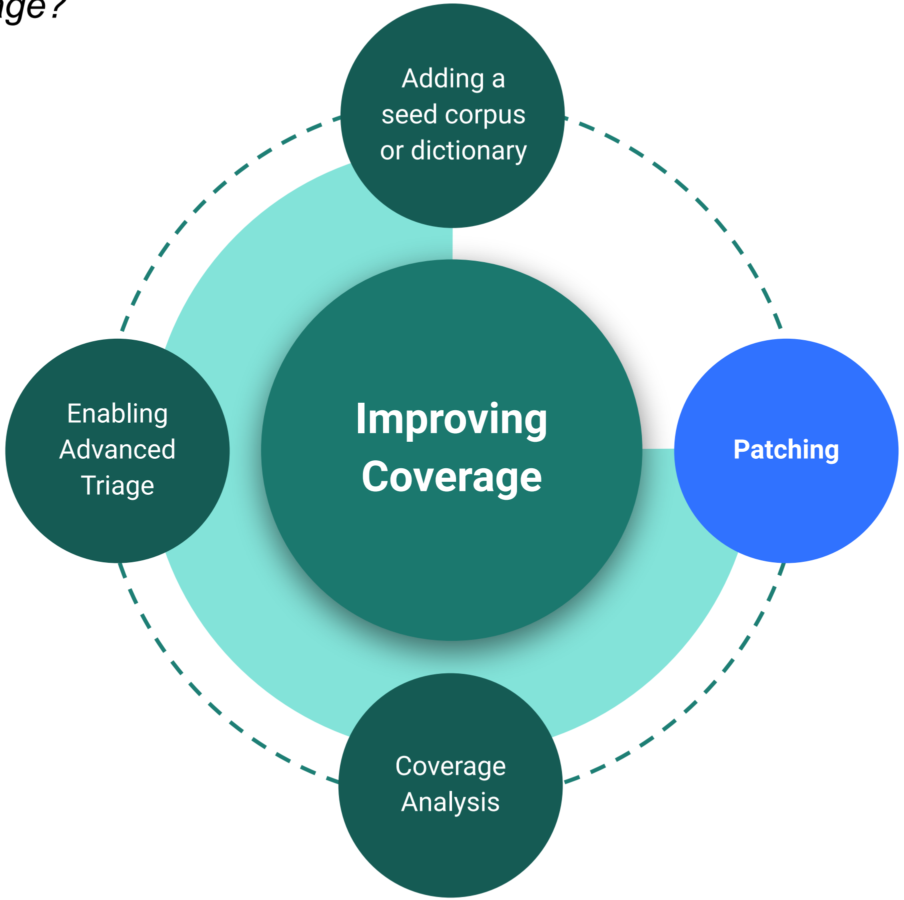
source: https://nvd.nist.gov/

Fuzzing with Options

```
project: ffmpeg
target: ffmpeg
image: ghcr.io/xansec/ffmpeg:latest
advanced_triage: true


tasks:
 - name: exploitability_factors
 - name: regression_testing
 - name: behavior_testing

cmds:

 - cmd: /ffmpeg -i @@ -vf "split [main][tmp]; [tmp] crop=iw:ih/2:0:0, vflip [flip];
[main][flip] overlay=0:H/2" -f null ignore.mp4

   env:
     LD_LIBRARY_PATH: /ffmpeg-libs
   dictionary: /dictionaries/mp4.dict
```

How can I fuzz better?

And improve overall coverage?



What You’ll Need

- Ghidra Installed (or a reverse engineering tool of your choice)
    - https://github.com/NationalSecurityAgency/ghidra/releases
- Ghidra “SavePatch.py” tool
    - https://github.com/schlafwandler/ghidra_SavePatch

The Problem

```
$ ./convert corpus/nsa-insignia-sm.png
/tmp/out.png
```

Success!

```
$ ./convert corpus/nsa-insignia-crc-error.png
/tmp/out.png

convert: IDAT: CRC error
`corpus/nsa-insignia-crc-error.png' @
error/png.c/MagickPNGErrorHandler/1713.

convert: no images defined `/tmp/out.png' @
error/convert.c/ConvertImageCommand/3322.
```

Fail!

Investigating the Binary

```
$ ldd ./convert
        linux-vdso.so.1 (0x00007ffd497a9000)
        libMagickCore-7.Q16HDRI.so.10 => /usr/lib/libMagickCore-7.Q16HDRI.so.10 (0x00007f04676c8000)
        libMagickWand-7.Q16HDRI.so.10 => /usr/lib/libMagickWand-7.Q16HDRI.so.10 (0x00007f0467589000)
        libc.so.6 => /usr/lib/libc.so.6 (0x00007f04673bd000)
        liblcms2.so.2 => /usr/lib/liblcms2.so.2 (0x00007f046735b000)
        libraqm.so.0 => /usr/lib/libraqm.so.0 (0x00007f0467354000)
        liblqr-1.so.0 => /usr/lib/liblqr-1.so.0 (0x00007f0467344000)
        libxml2.so.2 => /usr/lib/libxml2.so.2 (0x00007f04671d4000)
        libfontconfig.so.1 => /usr/lib/libfontconfig.so.1 (0x00007f0467185000)
        libfreetype.so.6 => /usr/lib/libfreetype.so.6 (0x00007f04670bb000)
        libXext.so.6 => /usr/lib/libXext.so.6 (0x00007f04670a6000)
        libX11.so.6 => /usr/lib/libX11.so.6 (0x00007f0466f65000)
        libbz2.so.1.0 => /usr/lib/libbz2.so.1.0 (0x00007f0466f52000)
        libz.so.1 => /usr/lib/libz.so.1 (0x00007f0466f36000)
        libltdl.so.7 => /usr/lib/libltdl.so.7 (0x00007f0466f2e000)
```

Bash Wizardry

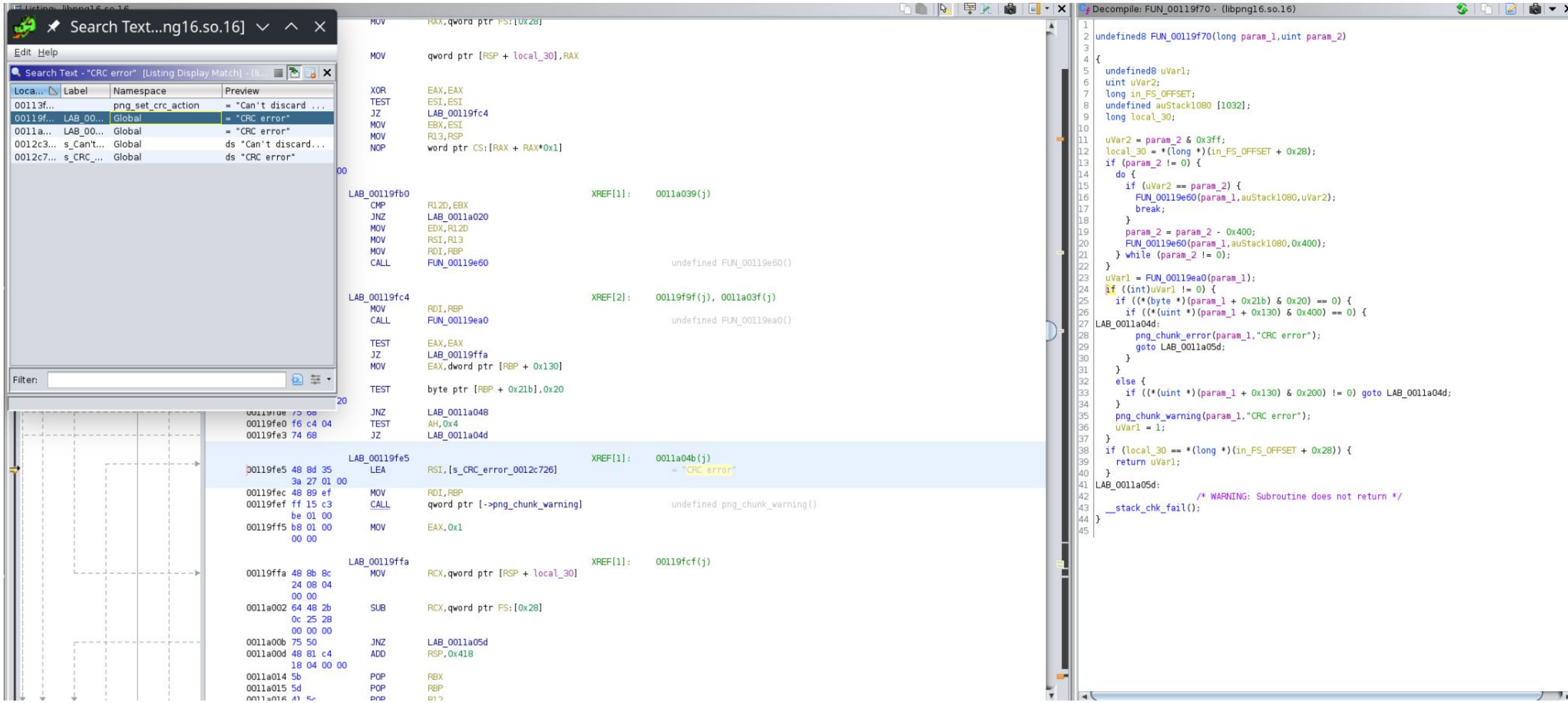
```
$ for file in $(ldd ./convert | sed -e 's/.*> \(.*\) (.*/\1/g'); do grep "CRC error" $file;
done

grep: linux-vdso.so.1: No such file or directory

grep: (0x00007fff8434c000): No such file or directory

grep: /usr/lib/libpng16.so.16: binary file matches
```

(this is just lazy stream editing)

hey! there it is

Let's use Ghidra to analyze this library!

Finding the Error Code



Search Text...ng16.so.16]

Search Text - "CRC error" [Listing Display Match]

Loca...	Label	Namespace	Preview
00113f...		png_set_crc_action	= "Can't discard ...
00119f...	LAB_00...	Global	= "CRC error"
0011a...	LAB_00...	Global	= "CRC error"
0012c3...	s_Can't...	Global	ds "Can't discard...
0012c7...	s_CRC_...	Global	ds "CRC error"

Filter:

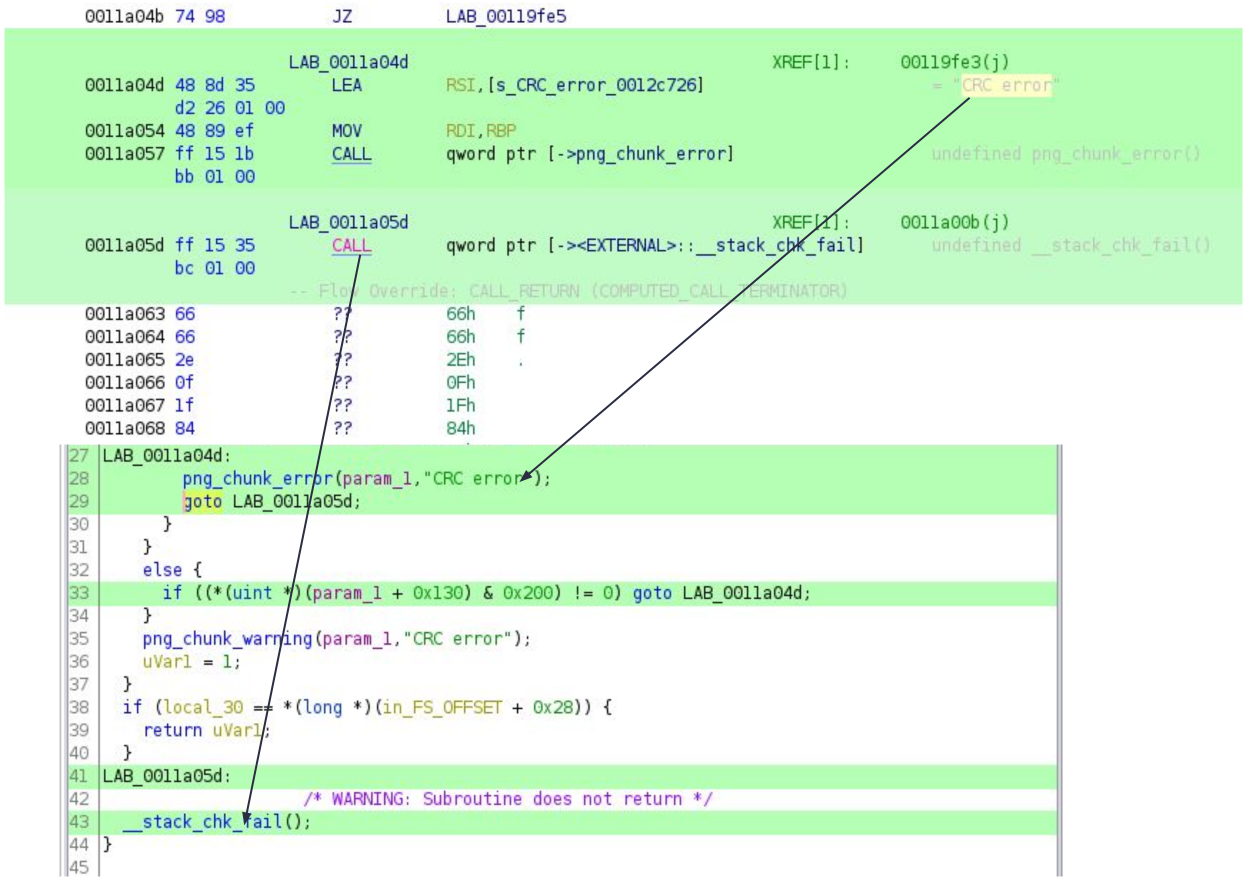
```
                  MOV      RAX,qword ptr FS:[0x28]

                  MOV      qword ptr [RSP + local_30],RAX

                  XOR      EAX,EAX
                  TEST     ESI,ESI
                  JZ       LAB_00119fc4
                  MOV      EBX,ESI
                  MOV      R13,RSP
                  NOP      word ptr CS:[RAX + RAX*0x1]
      00

               LAB_00119fb0                                  XREF[1]:     0011a039(j)
                  CMP      R12D,EBX
                  JNZ      LAB_0011a020
                  MOV      EDX,R12D
                  MOV      RSI,R13
                  MOV      RDI,RBP
                  CALL     FUN_00119e60                           undefined FUN_00119e60()

               LAB_00119fc4                                  XREF[2]:     00119f9f(j), 0011a03f(j)
                  MOV      RDI,RBP
                  CALL     FUN_00119ea0                           undefined FUN_00119ea0()

                  TEST     EAX,EAX
                  JZ       LAB_00119ffa
                  MOV      EAX,dword ptr [RBP + 0x130]

                  TEST     byte ptr [RBP + 0x21b],0x20
      20
00119fde 75 68    JNZ      LAB_0011a048
00119fe0 f6 c4 04 TEST     AH,0x4
00119fe3 74 68    JZ       LAB_0011a04d

               LAB_00119fe5                                  XREF[1]:     0011a04b(j)
00119fe5 48 8d 35 LEA      RSI,[s_CRC_error_0012c726]                = "CRC error"
         3a 27 01 00
00119fec 48 89 ef MOV      RDI,RBP
00119fef ff 15 c3 CALL     qword ptr [->png_chunk_warning]        undefined png_chunk_warning()
         be 01 00
00119ff5 b8 01 00 MOV      EAX,0x1
         00 00

               LAB_00119ffa                                  XREF[1]:     00119fcf(j)
00119ffa 48 8b 8c MOV      RCX,qword ptr [RSP + local_30]
         24 08 04
         00 00
0011a002 64 48 2b SUB      RCX,qword ptr FS:[0x28]
         0c 25 28
         00 00 00
0011a00b 75 50    JNZ      LAB_0011a05d
0011a00d 48 81 c4 ADD      RSP,0x418
         18 04 00 00
0011a014 5b       POP      RBX
0011a015 5d       POP      RBP
0011a016 41 5c    POP      R12
```

Decompile: FUN_00119f70 - (libpng16.so.16)

```

undefined8 FUN_00119f70(long param_1,uint param_2)

{
  undefined8 uVar1;
  uint uVar2;
  long in_FS_OFFSET;
  undefined auStack1080 [1032];
  long local_30;

  uVar2 = param_2 & 0x3ff;
  local_30 = *(long *)(in_FS_OFFSET + 0x28);
  if (param_2 != 0) {
    do {
      if (uVar2 == param_2) {
        FUN_00119e60(param_1,auStack1080,uVar2);
        break;
      }
      param_2 = param_2 - 0x400;
      FUN_00119e60(param_1,auStack1080,0x400);
    } while (param_2 != 0);
  }
  uVar1 = FUN_00119ea0(param_1);
  if ((int)uVar1 != 0) {
    if ((*(byte *)(param_1 + 0x21b) & 0x20) == 0) {
      if ((*(uint *)(param_1 + 0x130) & 0x400) == 0) {
LAB_0011a04d:
        png_chunk_error(param_1,"CRC error");
        goto LAB_0011a05d;
      }
    }
    else {
      if ((*(uint *)(param_1 + 0x130) & 0x200) != 0) goto LAB_0011a04d;
    }
    png_chunk_warning(param_1,"CRC error");
    uVar1 = 1;
  }
  if (local_30 == *(long *)(in_FS_OFFSET + 0x28)) {
    return uVar1;
  }
LAB_0011a05d:
                    /* WARNING: Subroutine does not return */
  __stack_chk_fail();
}

```

Finding the Error Code

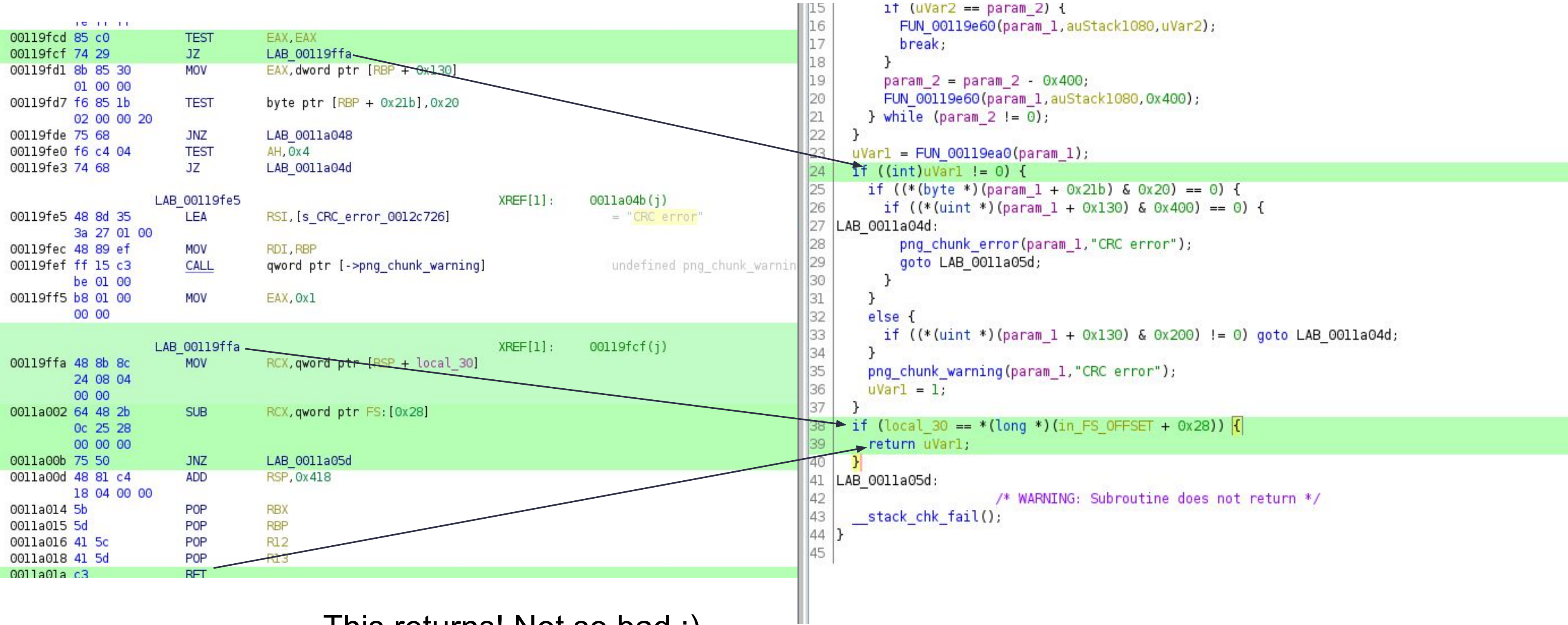
This looks bad :(



```
0011a04b 74 98          JZ        LAB_00119fe5

                      LAB_0011a04d                              XREF[1]:     00119fe3(j)
0011a04d 48 8d 35       LEA       RSI,[s_CRC_error_0012c726]                  = "CRC error"
         d2 26 01 00
0011a054 48 89 ef       MOV       RDI,RBP
0011a057 ff 15 1b       CALL      qword ptr [->png_chunk_error]               undefined png_chunk_error()
         bb 01 00

                      LAB_0011a05d                              XREF[1]:     0011a00b(j)
0011a05d ff 15 35       CALL      qword ptr [-><EXTERNAL>::__stack_chk_fail]  undefined __stack_chk_fail()
         bc 01 00
                      -- Flow Override: CALL_RETURN (COMPUTED_CALL_TERMINATOR)
0011a063 66             ??        66h    f
0011a064 66             ??        66h    f
0011a065 2e             ??        2Eh    .
0011a066 0f             ??        0Fh
0011a067 1f             ??        1Fh
0011a068 84             ??        84h
```

```
LAB_0011a04d:
        png_chunk_error(param_1,"CRC error");
        goto LAB_0011a05d;
      }
    }
    else {
      if ((*(uint *)(param_1 + 0x130) & 0x200) != 0) goto LAB_0011a04d;
    }
    png_chunk_warning(param_1,"CRC error");
    uVar1 = 1;
  }
  if (local_30 == *(long *)(in_FS_OFFSET + 0x28)) {
    return uVar1;
  }
LAB_0011a05d:
                    /* WARNING: Subroutine does not return */
  __stack_chk_fail();
}
```

Finding the Error Code

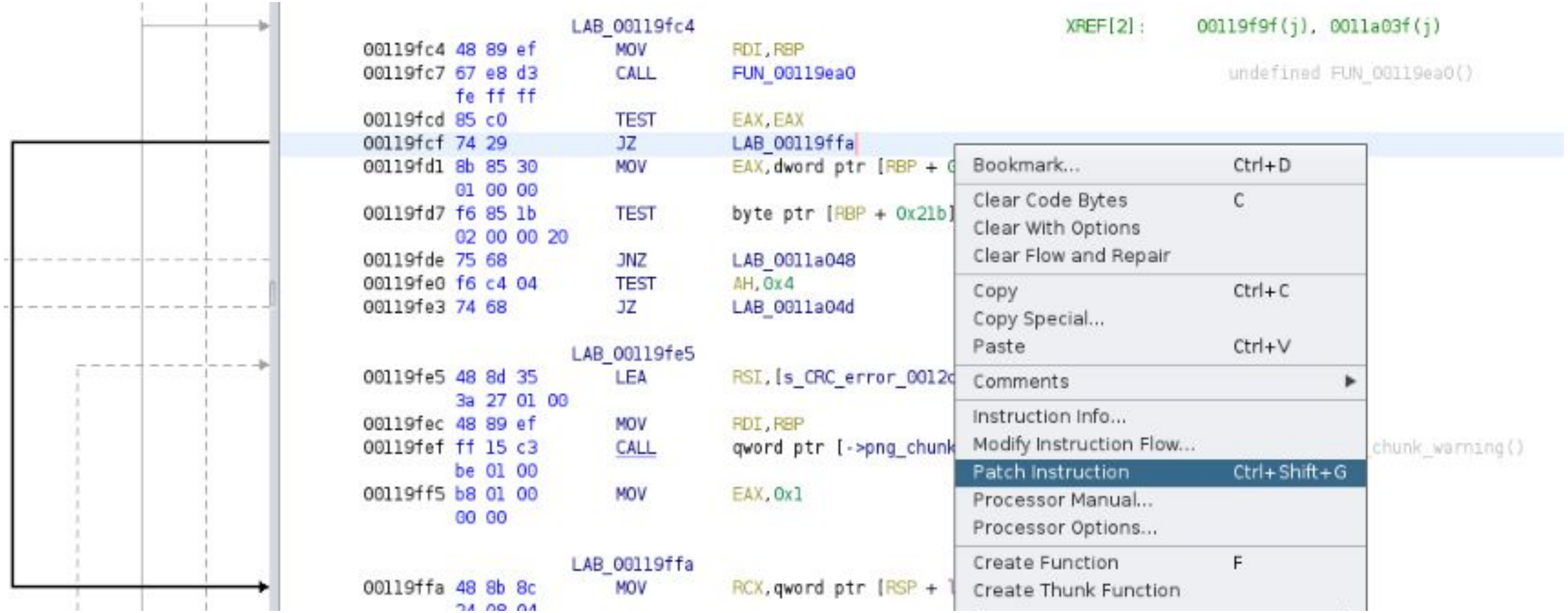


```
00119fcd 85 c0          TEST    EAX,EAX
00119fcf 74 29          JZ      LAB_00119ffa
00119fd1 8b 85 30       MOV     EAX,dword ptr [RBP + 0x130]
         01 00 00
00119fd7 f6 85 1b       TEST    byte ptr [RBP + 0x21b],0x20
         02 00 00 20
00119fde 75 68          JNZ     LAB_0011a048
00119fe0 f6 c4 04       TEST    AH,0x4
00119fe3 74 68          JZ      LAB_0011a04d

                    LAB_00119fe5                          XREF[1]:     0011a04b(j)
00119fe5 48 8d 35       LEA     RSI,[s_CRC_error_0012c726]                = "CRC error"
         3a 27 01 00
00119fec 48 89 ef       MOV     RDI,RBP
00119fef ff 15 c3       CALL    qword ptr [->png_chunk_warning]           undefined png_chunk_warnin
         be 01 00
00119ff5 b8 01 00       MOV     EAX,0x1
         00 00

                    LAB_00119ffa                          XREF[1]:     00119fcf(j)
00119ffa 48 8b 8c       MOV     RCX,qword ptr [RSP + local_30]
         24 08 04
         00 00
0011a002 64 48 2b       SUB     RCX,qword ptr FS:[0x28]
         0c 25 28
         00 00 00
0011a00b 75 50          JNZ     LAB_0011a05d
0011a00d 48 81 c4       ADD     RSP,0x418
         18 04 00 00
0011a014 5b             POP     RBX
0011a015 5d             POP     RBP
0011a016 41 5c          POP     R12
0011a018 41 5d          POP     R13
0011a01a c3             RET
```

```
      if (uVar2 == param_2) {
        FUN_00119e60(param_1,auStack1080,uVar2);
        break;
      }
      param_2 = param_2 - 0x400;
      FUN_00119e60(param_1,auStack1080,0x400);
    } while (param_2 != 0);
  }
  uVar1 = FUN_00119ea0(param_1);
  if ((int)uVar1 != 0) {
    if ((*(byte *)(param_1 + 0x21b) & 0x20) == 0) {
      if ((*(uint *)(param_1 + 0x130) & 0x400) == 0) {
LAB_0011a04d:
        png_chunk_error(param_1,"CRC error");
        goto LAB_0011a05d;
      }
    }
    else {
      if ((*(uint *)(param_1 + 0x130) & 0x200) != 0) goto LAB_0011a04d;
    }
    png_chunk_warning(param_1,"CRC error");
    uVar1 = 1;
  }
  if (local_30 == *(long *)(in_FS_OFFSET + 0x28)) {
    return uVar1;
  }
LAB_0011a05d:
                    /* WARNING: Subroutine does not return */
  __stack_chk_fail();
}
```

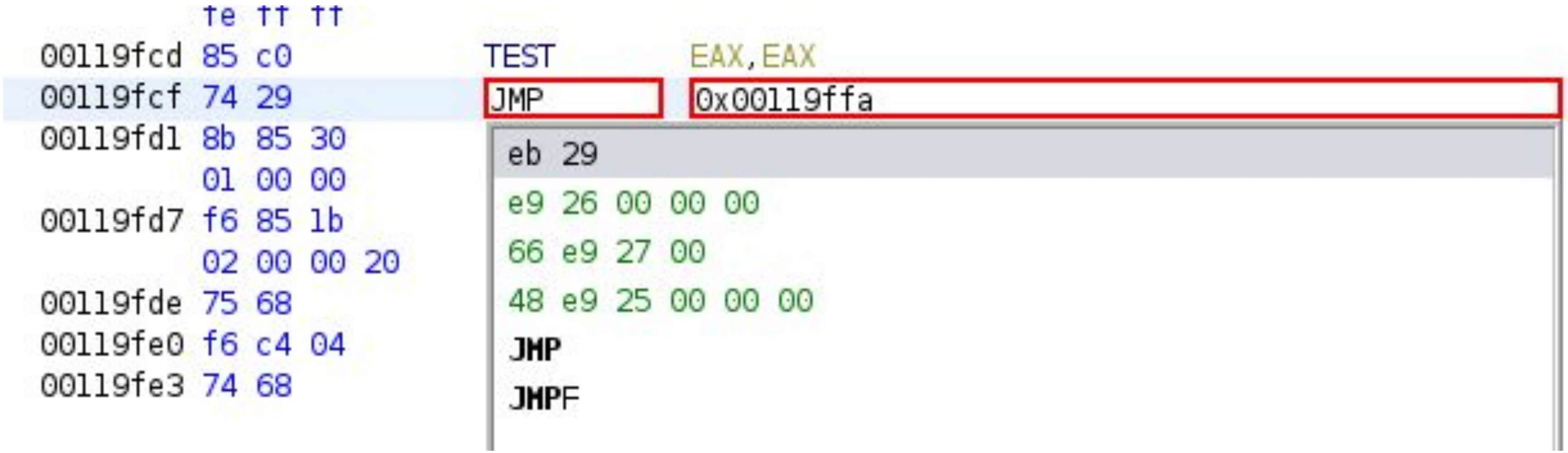
This returns! Not so bad :)
Can we make the program *always* do this?

Patching the Instruction



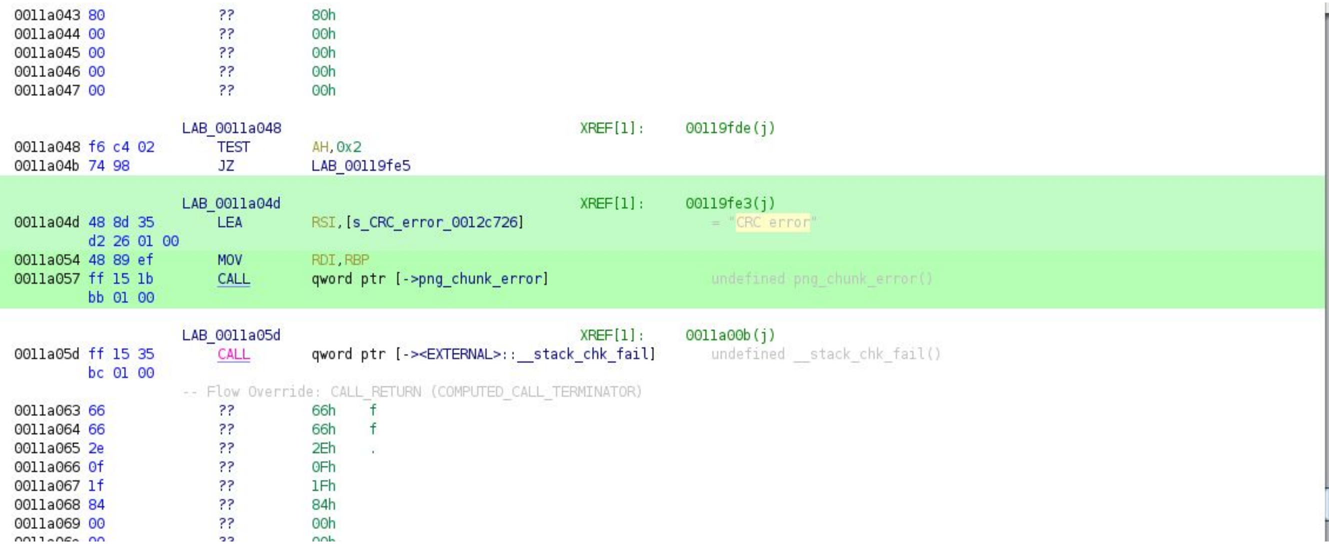
```
                                  LAB_00119fc4                                          XREF[2]:     00119f9f(j), 0011a03f(j)
00119fc4 48 89 ef        MOV         RDI,RBP
00119fc7 67 e8 d3        CALL        FUN_00119ea0                                                    undefined FUN_00119ea0()
         fe ff ff
00119fcd 85 c0           TEST        EAX,EAX
00119fcf 74 29           JZ          LAB_00119ffa
00119fd1 8b 85 30        MOV         EAX,dword ptr [RBP + 
         01 00 00
00119fd7 f6 85 1b        TEST        byte ptr [RBP + 0x21b]
         02 00 00 20
00119fde 75 68           JNZ         LAB_0011a048
00119fe0 f6 c4 04        TEST        AH,0x4
00119fe3 74 68           JZ          LAB_0011a04d

                                  LAB_00119fe5
00119fe5 48 8d 35        LEA         RSI,[s_CRC_error_0012
         3a 27 01 00
00119fec 48 89 ef        MOV         RDI,RBP
00119fef ff 15 c3        CALL        qword ptr [->png_chunk                                          chunk_warning()
         be 01 00
00119ff5 b8 01 00        MOV         EAX,0x1
         00 00

                                  LAB_00119ffa
00119ffa 48 8b 8c        MOV         RCX,qword ptr [RSP + 
```

Patching the Instruction

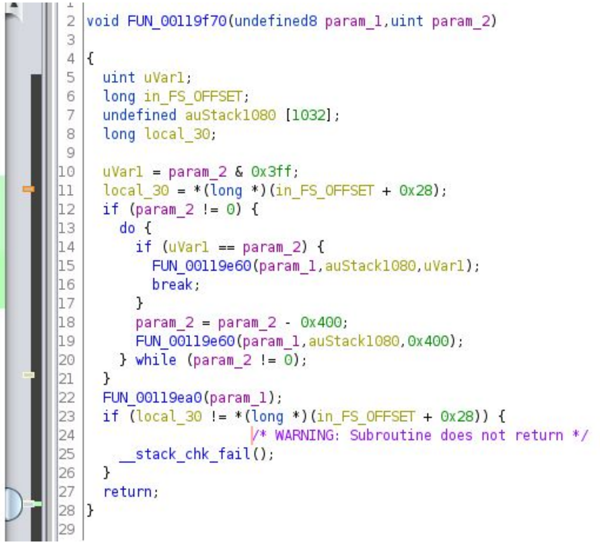


```
         fe ff ff
00119fcd 85 c0          TEST        EAX,EAX
00119fcf 74 29          JMP         0x00119ffa
00119fd1 8b 85 30        eb 29
         01 00 00        e9 26 00 00 00
00119fd7 f6 85 1b        66 e9 27 00
         02 00 00 20     48 e9 25 00 00 00
00119fde 75 68           JMP
00119fe0 f6 c4 04        JMPF
00119fe3 74 68
```

Patching the Instruction

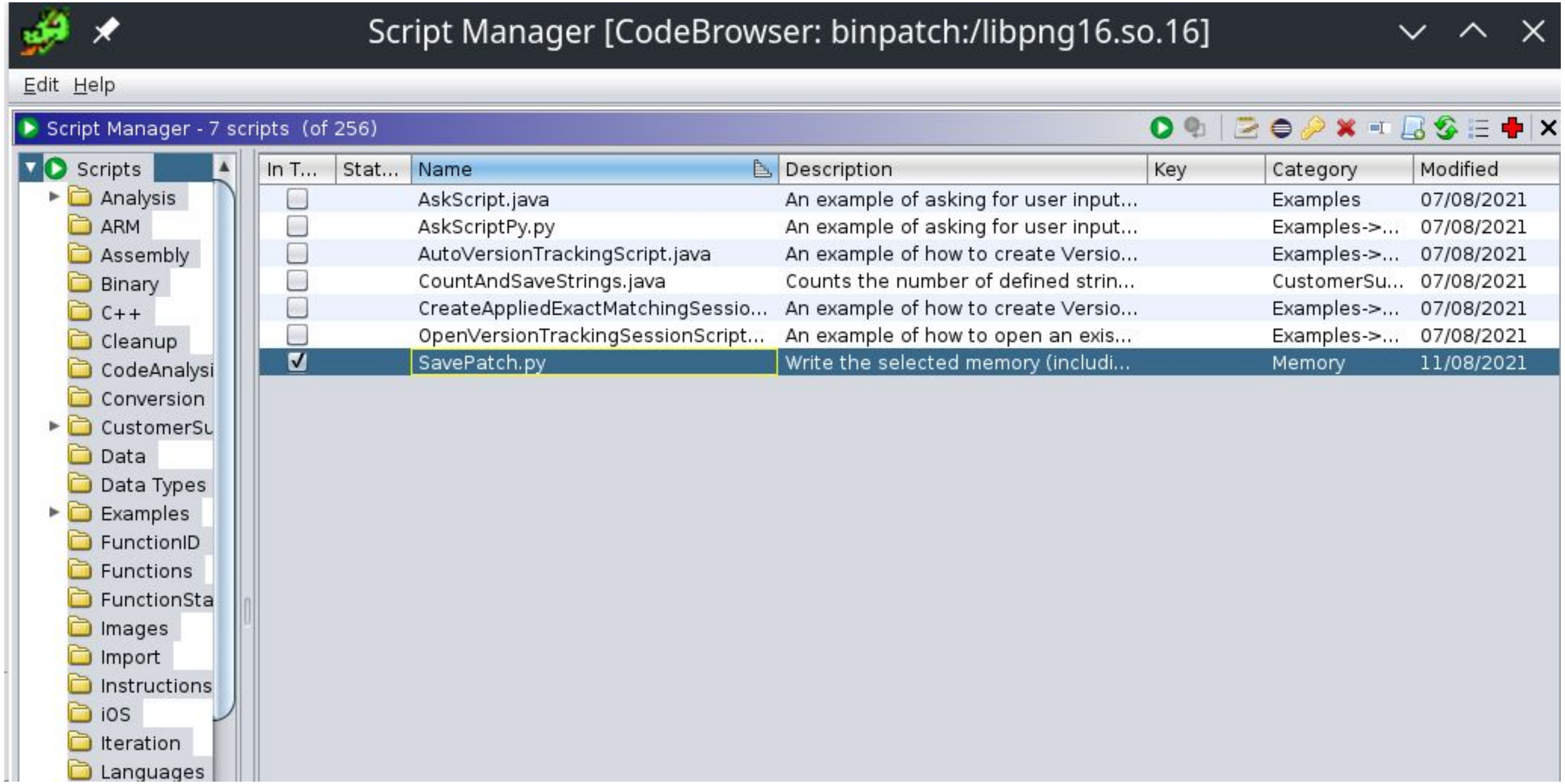


```
0011a043 80             ??         80h
0011a044 00             ??         00h
0011a045 00             ??         00h
0011a046 00             ??         00h
0011a047 00             ??         00h

               LAB_0011a048                                XREF[1]:     00119fde(j)
0011a048 f6 c4 02       TEST       AH,0x2
0011a04b 74 98          JZ         LAB_00119fe5

               LAB_0011a04d                                XREF[1]:     00119fe3(j)
0011a04d 48 8d 35       LEA        RSI,[s_CRC_error_0012c726]                = "CRC error"
         d2 26 01 00
0011a054 48 89 ef       MOV        RDI,RBP
0011a057 ff 15 1b       CALL       qword ptr [->png_chunk_error]             undefined png_chunk_error()
         bb 01 00

               LAB_0011a05d                                XREF[1]:     0011a00b(j)
0011a05d ff 15 35       CALL       qword ptr [-><EXTERNAL>::__stack_chk_fail]    undefined __stack_chk_fail()
         bc 01 00
               -- Flow Override: CALL_RETURN (COMPUTED_CALL_TERMINATOR)
0011a063 66             ??         66h    f
0011a064 66             ??         66h    f
0011a065 2e             ??         2Eh    .
0011a066 0f             ??         0Fh
0011a067 1f             ??         1Fh
0011a068 84             ??         84h
0011a069 00             ??         00h
```



```
void FUN_00119f70(undefined8 param_1,uint param_2)

{
  uint uVar1;
  long in_FS_OFFSET;
  undefined auStack1080 [1032];
  long local_30;

  uVar1 = param_2 & 0x3ff;
  local_30 = *(long *)(in_FS_OFFSET + 0x28);
  if (param_2 != 0) {
    do {
      if (uVar1 == param_2) {
        FUN_00119e60(param_1,auStack1080,uVar1);
        break;
      }
      param_2 = param_2 - 0x400;
      FUN_00119e60(param_1,auStack1080,0x400);
    } while (param_2 != 0);
  }
  FUN_00119ea0(param_1);
  if (local_30 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return;
}
```

Writing the File



Script Manager [CodeBrowser: binpatch:/libpng16.so.16]

Edit Help

Script Manager - 7 scripts (of 256)

In T...	Stat...	Name	Description	Key	Category	Modified
☐		AskScript.java	An example of asking for user input...		Examples	07/08/2021
☐		AskScriptPy.py	An example of asking for user input...		Examples->...	07/08/2021
☐		AutoVersionTrackingScript.java	An example of how to create Versio...		Examples->...	07/08/2021
☐		CountAndSaveStrings.java	Counts the number of defined strin...		CustomerSu...	07/08/2021
☐		CreateAppliedExactMatchingSessio...	An example of how to create Versio...		Examples->...	07/08/2021
☐		OpenVersionTrackingSessionScript...	An example of how to open an exis...		Examples->...	07/08/2021
☑		SavePatch.py	Write the selected memory (includi...		Memory	11/08/2021

ForAllSecure

questions?

www.forallsecure.com

info@forallsecure.com

@forallsecure

ForAllSecure

thank you

ForAllSecure