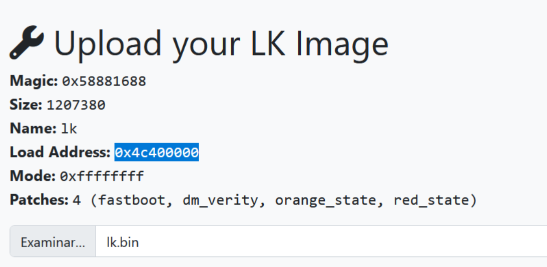
Creating patches for LK image(s)

1. Get Ghidra

Download Ghidra from the **official webpage** or use my **automated script (Windows).**

2. Figure out the load address

The next step is to figure out the correct load address of the LK image. You can do this by using my **web patcher** which will print the information of the image. Alternatively, you can use this **Python script**.



Upload your LK Image

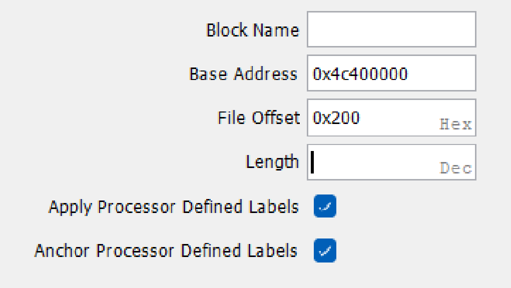
Magic: `0x58881688`
Size: `1207380`
Name: `lk`
Load Address: `0x4c400000`
Mode: `0xffffffff`
Patches: `4 (fastboot, dm_verity, orange_state, red_state)`

lk.bin

3. Load the image into Ghidra

Once you know the correct load address and you've got the image, load it with the following options:

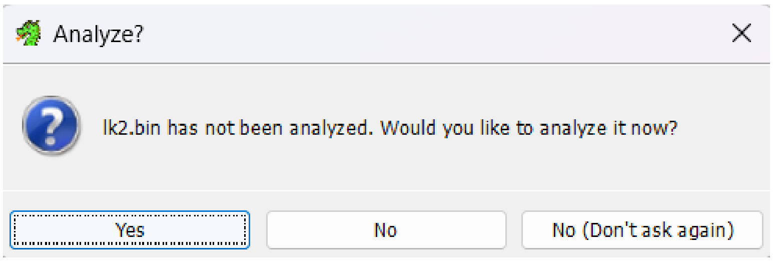
- **Language:** ARM *(processor)* v7 *(variant)* 32 *(size)* little *(endian)* default *(compiler)*.
- **Block Name:** *Empty.*
- **Base Address:** *Your Load Address*.
- **File Offset:** `0x200`.
- **Length:** *Empty* (otherwise it will raise an error).



Block Name:
Base Address: 0x4c400000
File Offset: 0x200 Hex
Length: Dec
Apply Processor Defined Labels ☑
Anchor Processor Defined Labels ☑

4. Let Ghidra auto-analyze the image

Once you load the image, you'll be prompted with the "*Would you like to analyze the file now?*". Press *Yes* and let it use the defaults.



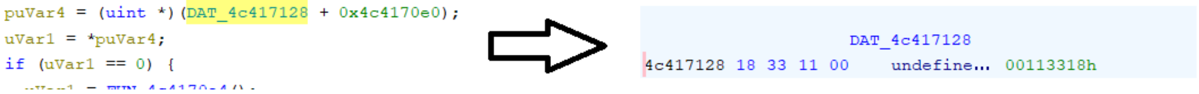
Analyze?

lk2.bin has not been analyzed. Would you like to analyze it now?

5. Change data mutability

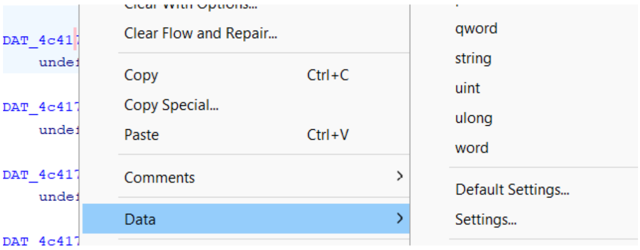
Next step is to change data mutability to "*Constant*" so Ghidra properly shows and lists all the strings. In order to do so, open any random function and click over any of the `DAT_*` values.

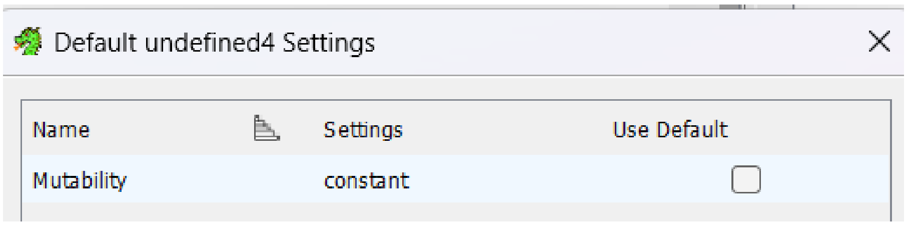
Then right click on it and set "*Constant"* in *Data → Default Settings.* Once you do this, go back to any function and press "*ctrl+S*" for the changes to take effect.



```
puVar4 = (uint *)(DAT_4c417128 + 0x4c4170e0);
uVar1 = *puVar4;
if (uVar1 == 0) {
```

```
                 DAT_4c417128
4c417128 18 33 11 00     undefine...  00113318h
```



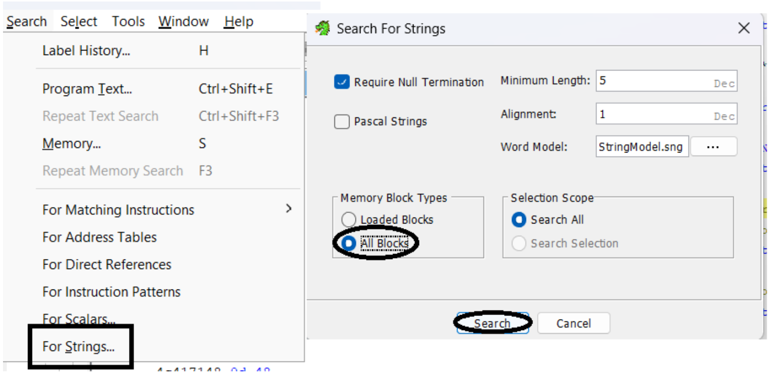


Default undefined4 Settings

Name	Settings	Use Default
Mutability	constant	☐

6. Find the function you want to patch

In this case, we’re going to patch the “dm verity” warning function so we’ll use string search.

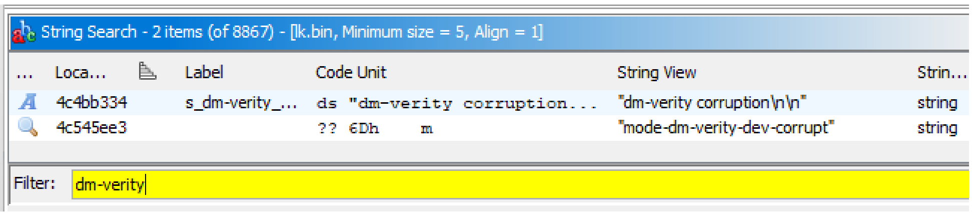


Search For Strings

☑ Require Null Termination
☐ Pascal Strings
Minimum Length: 5 Dec
Alignment: 1 Dec
Word Model: StringModel.sng

Memory Block Types: ○ Loaded Blocks ● All Blocks

Selection Scope: ● Search All ○ Search Selection

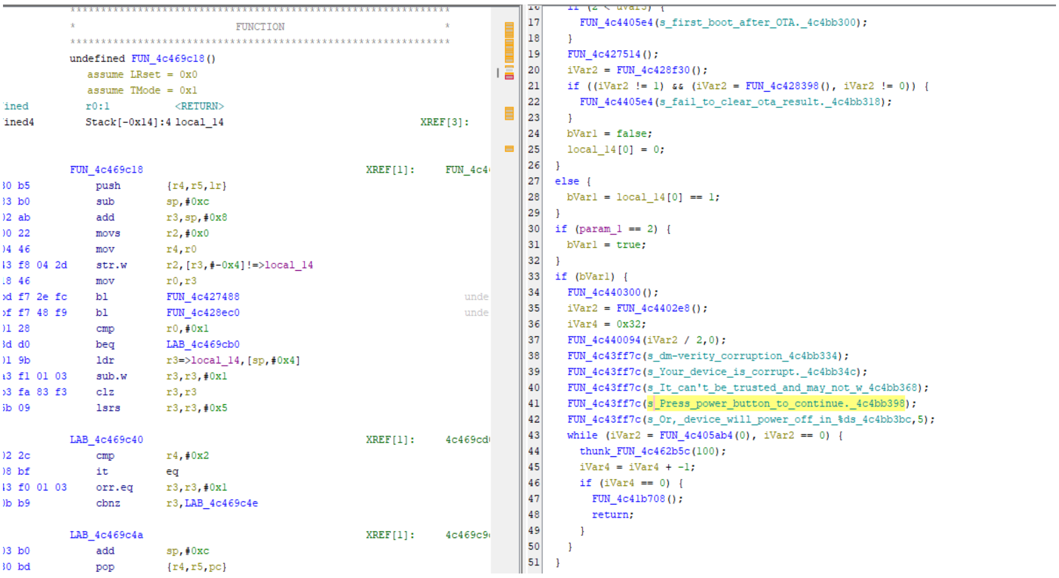


String Search - 2 items (of 8867) - [lk.bin, Minimum size = 5, Align = 1]

...	Loca...	Label	Code Unit	String View	Strin...
A	4c4bb334	s_dm-verity_...	ds "dm-verity corruption...	"dm-verity corruption\n\n"	string
	4c545ee3		?? 6Dh m	"mode-dm-verity-dev-corrupt"	string

Filter: dm-verity

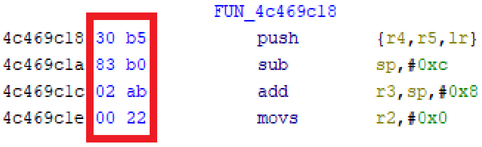
In this case, the function is `FUN_4c469c18` (it will be different depending on the image).



```
              ************************************************************
              *                         FUNCTION                         *
              ************************************************************
              undefined FUN_4c469c18()
                 assume LRset = 0x0
                 assume TMode = 0x1
ined             r0:1           <RETURN>
ined4            Stack[-0x14]:4 local_14                        XREF[3]:

              FUN_4c469c18                              XREF[1]:     FUN_4c4
0 b5                push       {r4,r5,lr}
3 b0                sub        sp,#0xc
2 ab                add        r3,sp,#0x8
0 22                movs       r2,#0x0
4 46                mov        r4,r0
3 f8 04 2d          str.w      r2,[r3,#-0x4]!=>local_14
8 46                mov        r0,r3
d f7 2e fc          bl         FUN_4c427488                             unde
f f7 48 f9          bl         FUN_4c428ec0                             unde
1 28                cmp        r0,#0x1
d d0                beq        LAB_4c469cb0
1 9b                ldr        r3=>local_14,[sp,#0x4]
3 f1 01 03          sub.w      r3,r3,#0x1
3 fa 83 f3          clz        r3,r3
b 09                lsrs       r3,r3,#0x5

              LAB_4c469c40                              XREF[1]:     4c469cd
2 2c                cmp        r4,#0x2
8 bf                it         eq
3 f0 01 03          orr.eq     r3,r3,#0x1
b b9                cbnz       r3,LAB_4c469c4e

              LAB_4c469c4a                              XREF[1]:     4c469c9
3 b0                add        sp,#0xc
0 bd                pop        {r4,r5,pc}
```

```
    if (2 < uVar3) {
      FUN_4c4405e4(s_first_boot_after_OTA._4c4bb300);
    }
    FUN_4c427514();
    iVar2 = FUN_4c428f30();
    if ((iVar2 != 1) && (iVar2 = FUN_4c428398(), iVar2 != 0)) {
      FUN_4c4405e4(s_fail_to_clear_ota_result._4c4bb318);
    }
    bVar1 = false;
    local_14[0] = 0;
  }
  else {
    bVar1 = local_14[0] == 1;
  }
  if (param_1 == 2) {
    bVar1 = true;
  }
  if (bVar1) {
    FUN_4c440300();
    iVar2 = FUN_4c4402e8();
    iVar4 = 0x32;
    FUN_4c440094(iVar2 / 2,0);
    FUN_4c43ff7c(s_dm-verity_corruption_4c4bb334);
    FUN_4c43ff7c(s_Your_device_is_corrupt._4c4bb34c);
    FUN_4c43ff7c(s_It_can't_be_trusted_and_may_not_w_4c4bb368);
    FUN_4c43ff7c(s_Press_power_button_to_continue._4c4bb398);
    FUN_4c43ff7c(s_Or,_device_will_power_off_in_%ds_4c4bb3bc,5);
    while (iVar2 = FUN_4c405ab4(0), iVar2 == 0) {
      thunk_FUN_4c462b5c(100);
      iVar4 = iVar4 + -1;
      if (iVar4 == 0) {
        FUN_4c41b708();
        return;
      }
    }
  }
```

We’ll click in the name of the function and we’ll note down the first hex values.

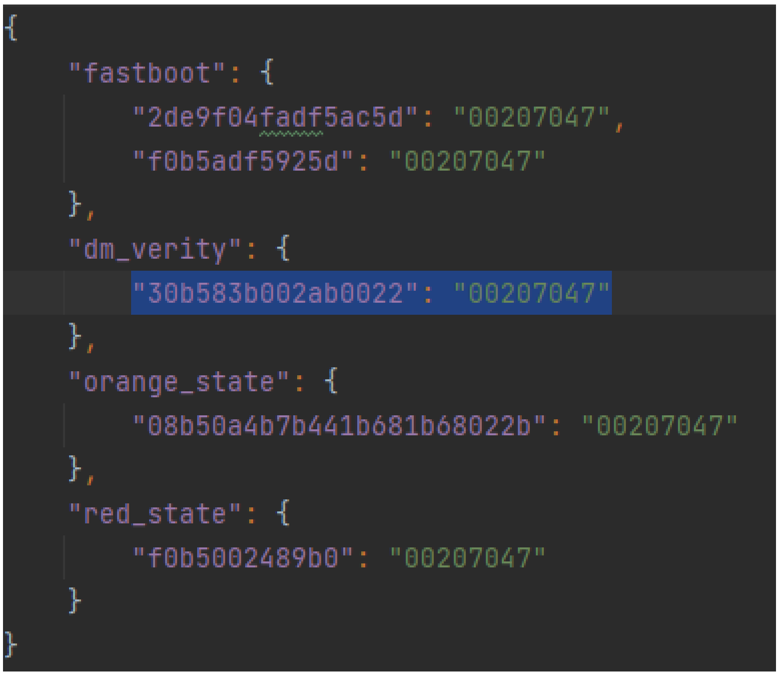


```
                      FUN_4c469c18
4c469c18 30 b5           push       {r4,r5,lr}
4c469c1a 83 b0           sub        sp,#0xc
4c469c1c 02 ab           add        r3,sp,#0x8
4c469c1e 00 22           movs       r2,#0x0
```

30b583b002ab022

7. Updating the JSON schema

We know our sequence is ***`30b583b002ab022`***. If we want the function to just `return 0`, we'll use the HEX equivalent to ARM's `mov r0, #0x0` which is `00207047`. If you want it to do something else, use [this online converter]() (for example, `return 1` would be `0100A0E3`). That being said, the update schema will look like this



```
{
    "fastboot": {
        "2de9f04fadf5ac5d": "00207047",
        "f0b5adf5925d": "00207047"
    },
    "dm_verity": {
        "30b583b002ab0022": "00207047"
    },
    "orange_state": {
        "08b50a4b7b441b681b68022b": "00207047"
    },
    "red_state": {
        "f0b5002489b0": "00207047"
    }
}
```