DCS-7517_B1_FW_v2.02.01 Weak Password Vulnerability

firmware version

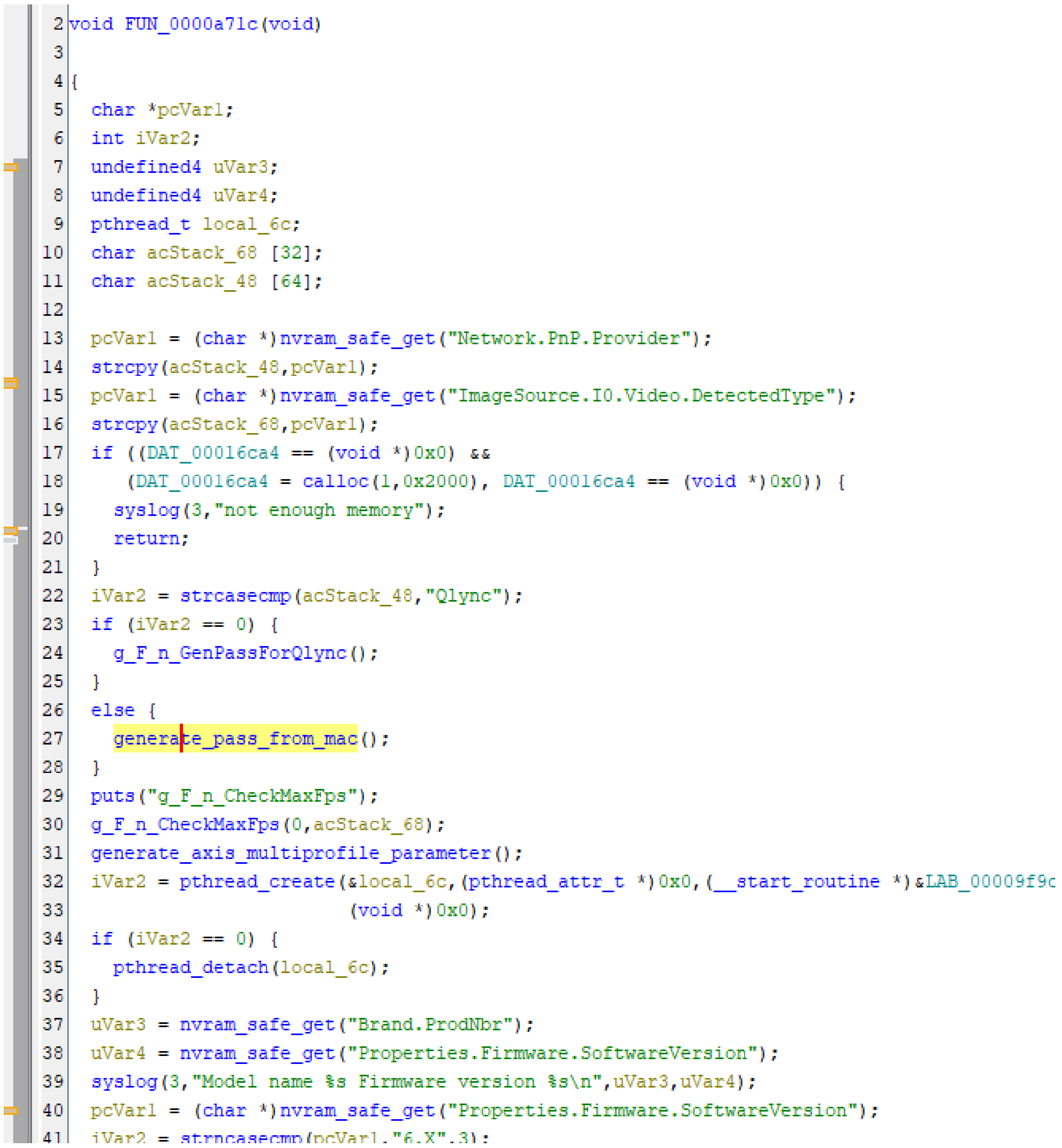
- vendor: dlink_ipcamera
- product: DCS-7517B1
- version: below or equal v2.02.01
- firmware download url: https://files.dlink.com.au/products/DCS-7517/REV_B/Firmware/Firmware_2.02.01/

description

In D-link-ipcamera DCS-7517B1 firmware, binary `/bin/httpd` contains weak password vulnerability. The root user, which has superuser privileges (UID=0, GID=0), uses a password generated via a predictable mechanism based on the devices's MAC address. Attackers can reproduce the password generation and gain unauthorized administrative access to the device.

details

The vulerability resides in `/bin/httpd` binary, where the function `FUN_0000a71c` calls library function `generate_pass_from_mac` to generate the root password.



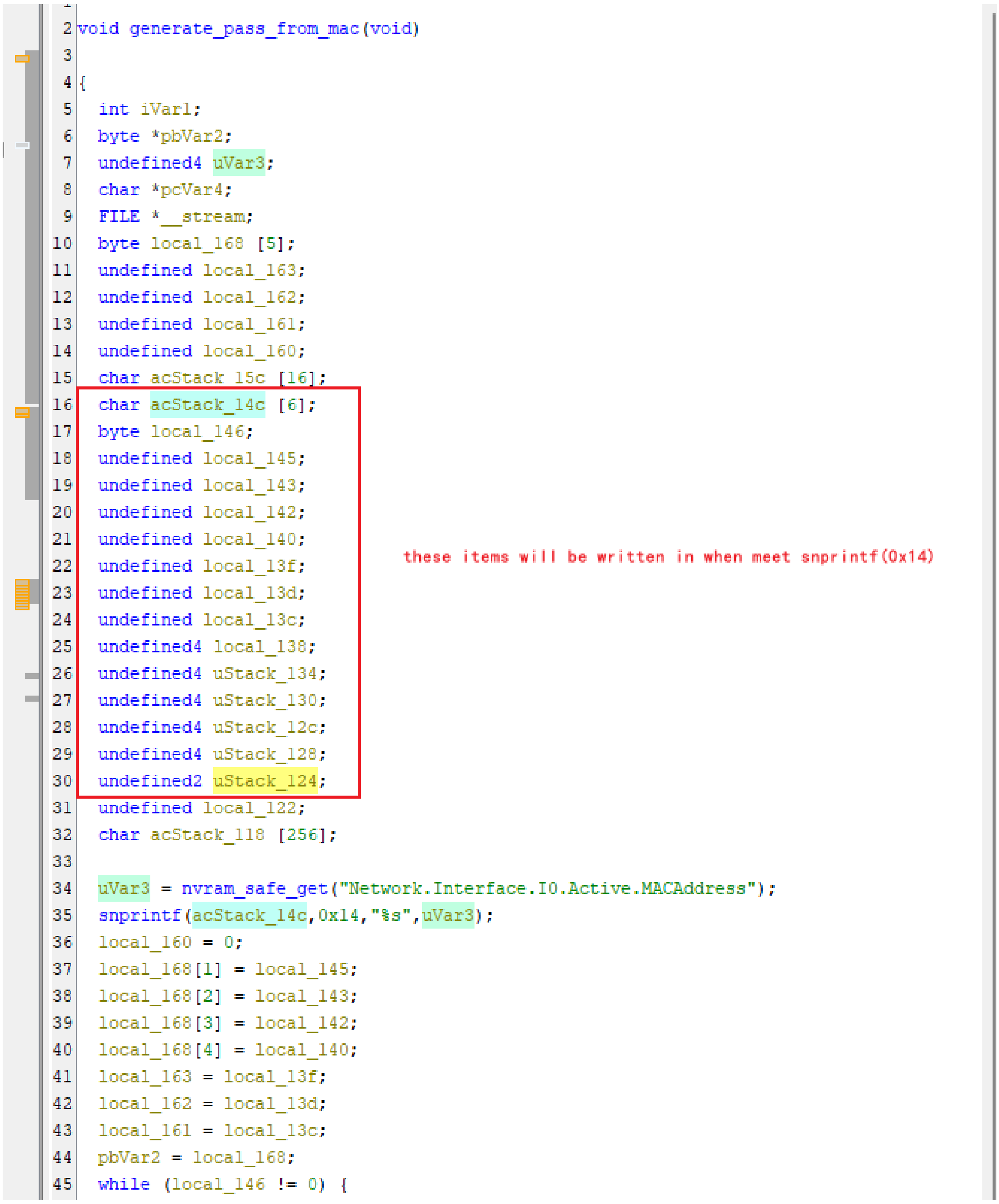
```
void FUN_0000a71c(void)

{
  char *pcVar1;
  int iVar2;
  undefined4 uVar3;
  undefined4 uVar4;
  pthread_t local_6c;
  char acStack_68 [32];
  char acStack_48 [64];

  pcVar1 = (char *)nvram_safe_get("Network.PnP.Provider");
  strcpy(acStack_48,pcVar1);
  pcVar1 = (char *)nvram_safe_get("ImageSource.I0.Video.DetectedType");
  strcpy(acStack_68,pcVar1);
  if ((DAT_00016ca4 == (void *)0x0) &&
     (DAT_00016ca4 = calloc(1,0x2000), DAT_00016ca4 == (void *)0x0)) {
    syslog(3,"not enough memory");
    return;
  }
  iVar2 = strcasecmp(acStack_48,"Qlync");
  if (iVar2 == 0) {
    g_F_n_GenPassForQlync();
  }
  else {
    generate_pass_from_mac();
  }
  puts("g_F_n_CheckMaxFps");
  g_F_n_CheckMaxFps(0,acStack_68);
  generate_axis_multiprofile_parameter();
  iVar2 = pthread_create(&local_6c,(pthread_attr_t *)0x0,(__start_routine *)&LAB_00009f9c
                         (void *)0x0);
  if (iVar2 == 0) {
    pthread_detach(local_6c);
  }
  uVar3 = nvram_safe_get("Brand.ProdNbr");
  uVar4 = nvram_safe_get("Properties.Firmware.SoftwareVersion");
  syslog(3,"Model name %s Firmware version %s\n",uVar3,uVar4);
  pcVar1 = (char *)nvram_safe_get("Properties.Firmware.SoftwareVersion");
  iVar2 = strncasecmp(pcVar1,"6.X",3);
```

The `generate_pass_from_mac` function (in /lib/libnvarm.so) calls crypt() to perform password hashing with a fixed salt value ("AM"), making it vulnerable to brute-force attacks or raindow table precomputation.

An attacker with access to the device's MAC address(which can be obtained through network scanning or physical access) can reproduce the password generation login and gain unauthorized access ti the device.

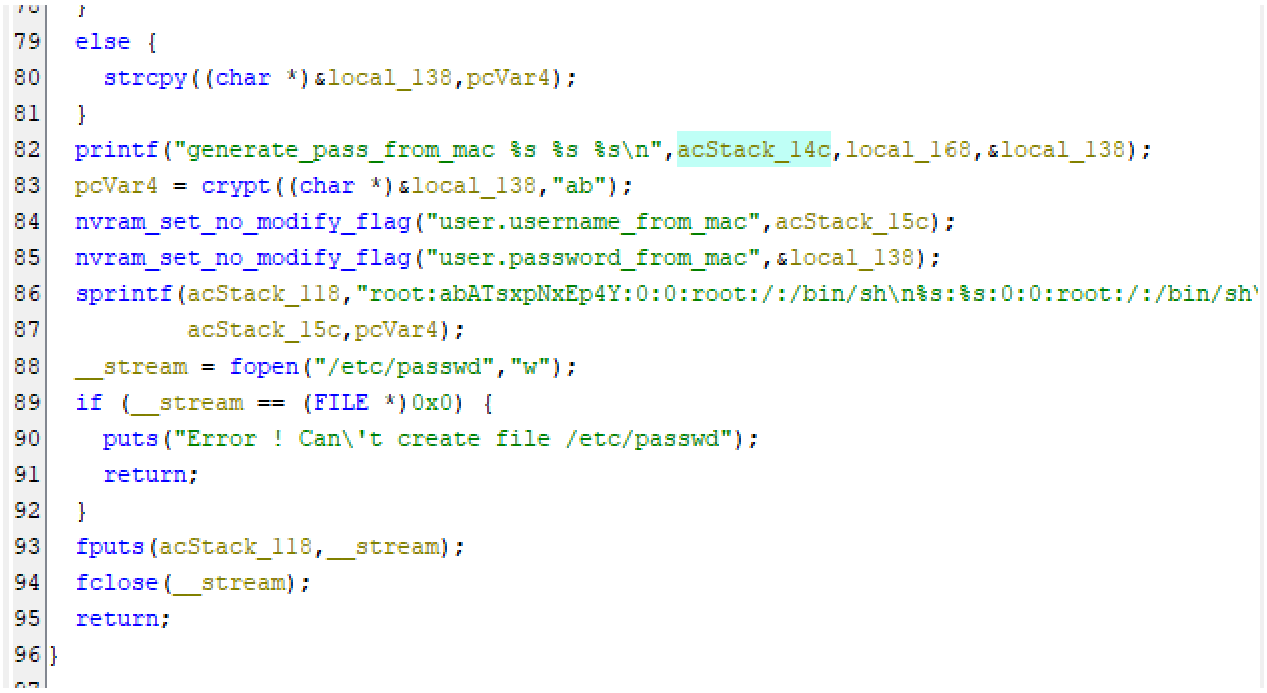


```
void generate_pass_from_mac(void)

{
  int iVar1;
  byte *pbVar2;
  undefined4 uVar3;
  char *pcVar4;
  FILE *__stream;
  byte local_168 [5];
  undefined local_163;
  undefined local_162;
  undefined local_161;
  undefined local_160;
  char acStack_15c [16];
  char acStack_14c [6];
  byte local_146;
  undefined local_145;
  undefined local_143;
  undefined local_142;
  undefined local_140;
  undefined local_13f;
  undefined local_13d;
  undefined local_13c;
  undefined4 local_138;
  undefined4 uStack_134;
  undefined4 uStack_130;
  undefined4 uStack_12c;
  undefined4 uStack_128;
  undefined2 uStack_124;
  undefined local_122;
  char acStack_118 [256];

  uVar3 = nvram_safe_get("Network.Interface.I0.Active.MACAddress");
  snprintf(acStack_14c,0x14,"%s",uVar3);
  local_160 = 0;
  local_168[1] = local_145;
  local_168[2] = local_143;
  local_168[3] = local_142;
  local_168[4] = local_140;
  local_163 = local_13f;
  local_162 = local_13d;
  local_161 = local_13c;
  pbVar2 = local_168;
  while (local_146 != 0) {
```

these items will be written in when meet snprintf(0x14)



```
  }
  else {
    strcpy((char *)&local_138,pcVar4);
  }
  printf("generate_pass_from_mac %s %s %s\n",acStack_14c,local_168,&local_138);
  pcVar4 = crypt((char *)&local_138,"ab");
  nvram_set_no_modify_flag("user.username_from_mac",acStack_15c);
  nvram_set_no_modify_flag("user.password_from_mac",&local_138);
  sprintf(acStack_118,"root:abATsxpNxEp4Y:0:0:root:/:/bin/sh\n%s:%s:0:0:root:/:/bin/sh\
          acStack_15c,pcVar4);
  __stream = fopen("/etc/passwd","w");
  if (__stream == (FILE *)0x0) {
    puts("Error ! Can\'t create file /etc/passwd");
    return;
  }
  fputs(acStack_118,__stream);
  fclose(__stream);
  return;
}
```

The password is written to `/etc/passwd` file, granting root-level superuser privileges to the attacker.