ОБРПО. Лекция 3 Переполнение буфера

ТЮРИН КАЙ АНДРЕЕВИЧ

Было

№	Название	Баллы
1	Лаб. 1 (бин. без.)	13
2	Лаб. 2 (бин. без.)	13
3	Лаб. 3 (сетевая без.)	13
4	Тест 1	11
5	Лаб. 4 (web)	13
6	Лаб. 5 (привилегии)	13
7	Лаб. 6 (аутентификация)	13
8	Тест 2	11

Стало

№	Название	Баллы
1	Лаб. 1 (бин. без.)	13
2	Лаб. 2 (сетевая без.)	13
3	Лаб. 3 (web)	13
4	Тест 1	11
5	Лаб. 4 (web)	13
6	Лаб. 5 (привилегии)	13
7	Лаб. 6 (аутентификация)	13
8	Тест 2	11

Сегодня мы поговорим про

DEP

ASLR

ROP

DEP

DATA EXECUTION PREVENTION

Non-executable memory

RWX

W^X

Железо следит

В отличие от операционной системы, аппаратное обеспечение машины контролирует выполнение всех проводимых операций.

Зачем WX?

Одним из допустимых примеров использования страниц, доступных и на запись и на исполнение, является JIT (Just In Time Compilation)

Пример работы DEP

Name	Start	End	R	W	X
.text	0000000140001000	0000000140075000	R	.	X
.rdata	0000000140075000	000000014008A000	R	.	.
.data	000000014008A000	000000014008E000	R	W	.
.pdata	000000014008E000	0000000140094000	R	.	.
.idata	0000000140094000	00000001400943F8	R	.	.
.gfids	0000000140096000	0000000140097000	R	.	.
.00cfg	0000000140097000	0000000140098000	R	.	.

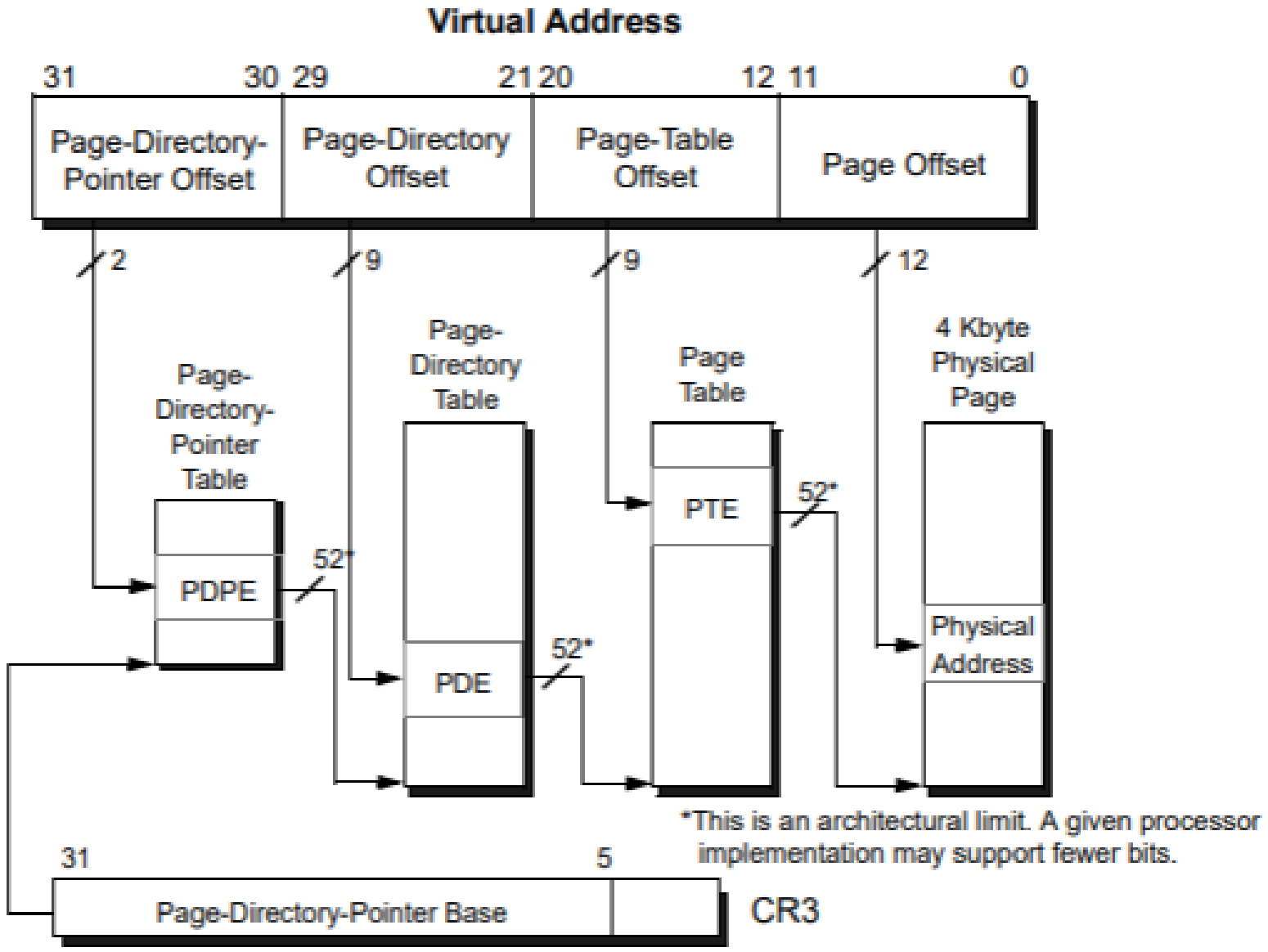


Figure 5-9. 4-Kbyte PAE Page Translation—Legacy Mode

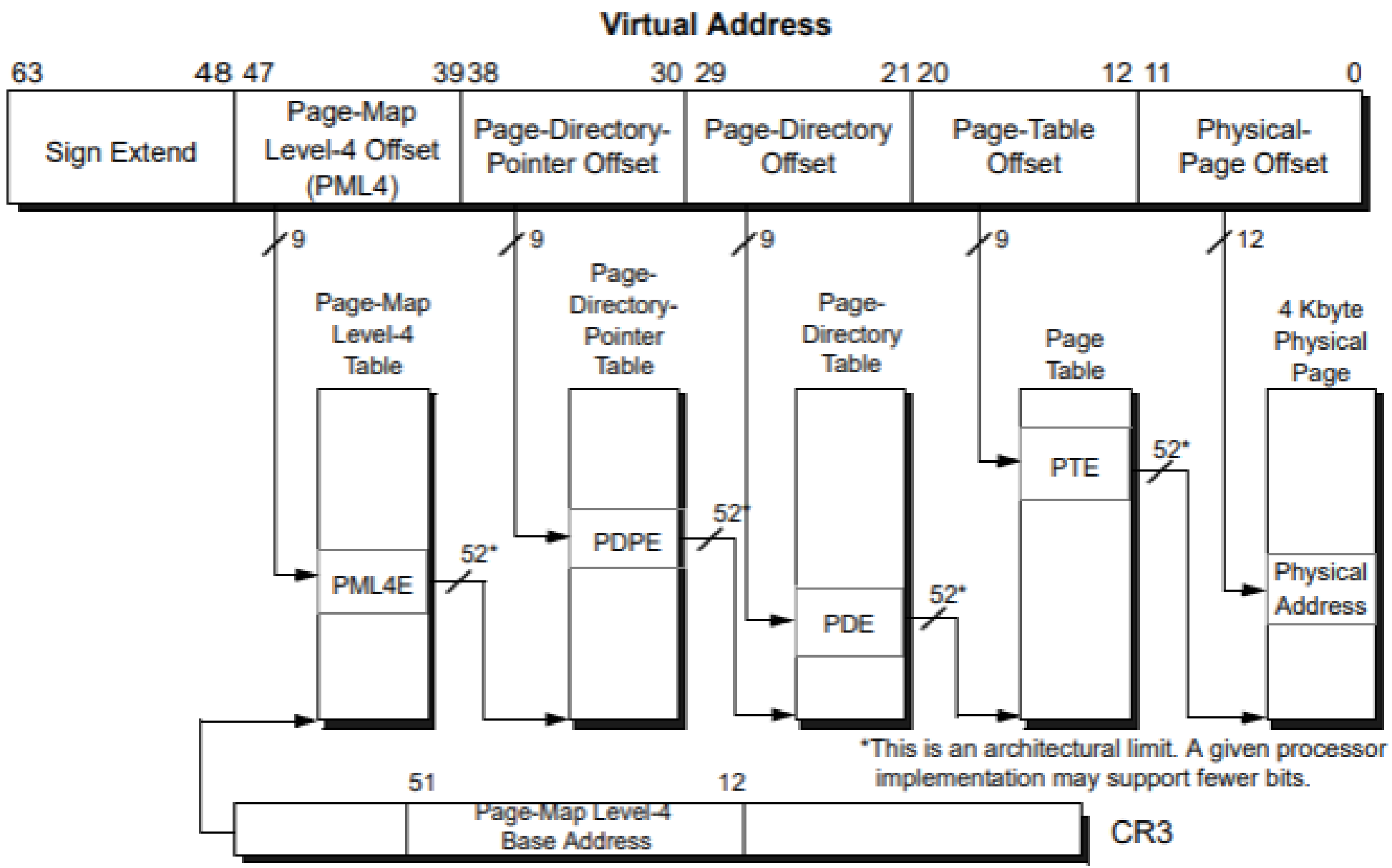


Figure 5-17. 4-Kbyte Page Translation—Long Mode

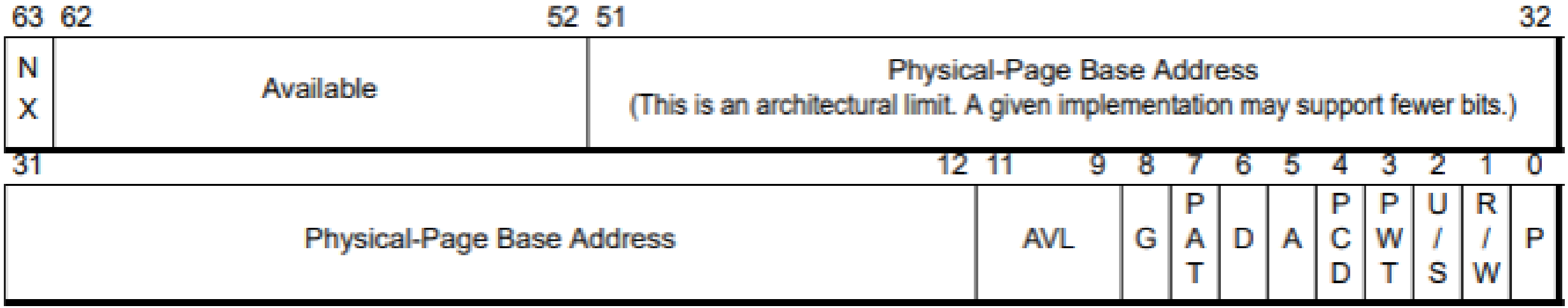


Figure 5-21. 4-Kbyte PTE—Long Mode

Значение отдельных битов

Present (P) Bit. – отвечает за то, находится ли страница в оперативной памяти (физической)

Read/Write (R/W) Bit. – доступна ли страница на чтение/запись (1 – доступно и то и другое).

User/Supervisor (U/S) Bit. – бит принадлежности страницы пространству пользоваля или ядра (1 – и пользователю и ядру)

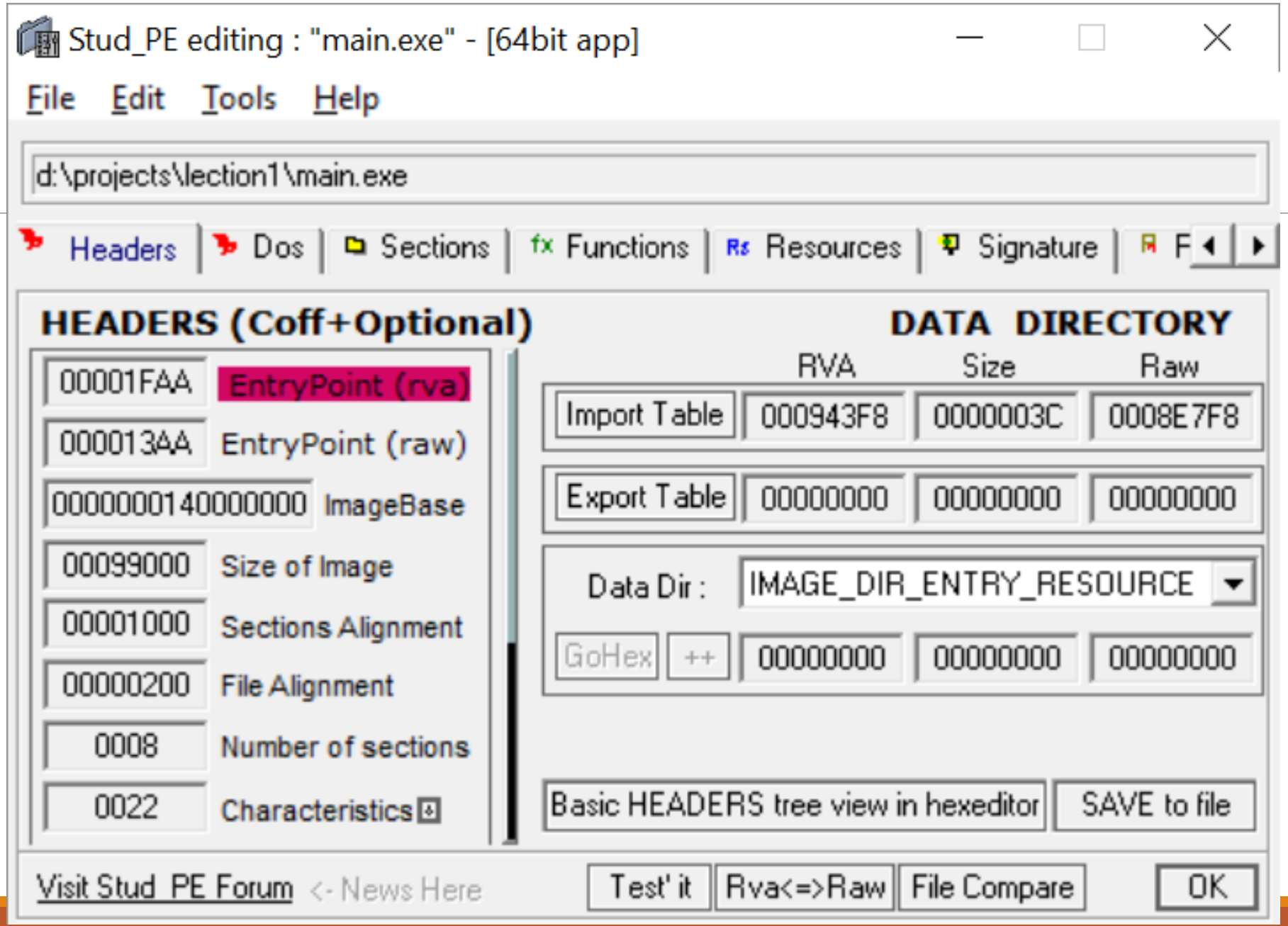
No Execute (NX) Bit. – отвечает за запрет исполнения (1 – запрет)

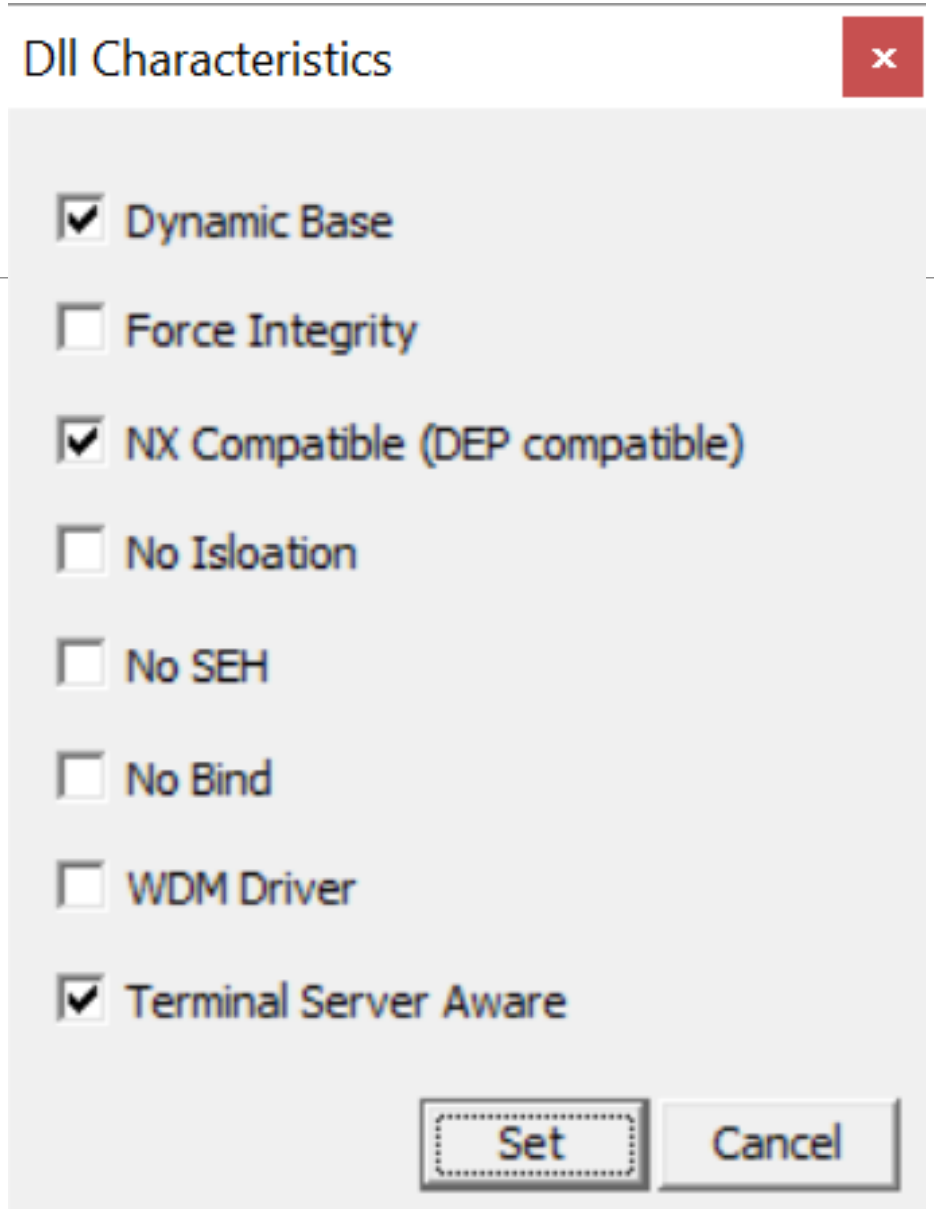
Источники

AMD64 Architecture Programmer’s Manual
Volume 2: System Programming

ASLR

ADDRESS SPACE LAYOUT RANDOMIZATION


Stud_PE editing : "main.exe" - [64bit app]
File Edit Tools Help
d:\projects\lection1\main.exe
Headers
Dos
Sections
Functions
Resources
Signature
HEADERS (Coff+Optional)
00001FAA EntryPoint (rva)
000013AA EntryPoint (raw)
0000000140000000 ImageBase
00099000 Size of Image
00001000 Sections Alignment
00000200 File Alignment
0008 Number of sections
0022 Characteristics
DATA DIRECTORY
RVA Size Raw
Import Table 000943F8 0000003C 0008E7F8
Export Table 00000000 00000000 00000000
Data Dir : IMAGE_DIR_ENTRY_RESOURCE
GoHex ++ 00000000 00000000 00000000
Basic HEADERS tree view in hexeditor
SAVE to file
Visit Stud_PE Forum <- News Here
Test' it
Rva<=>Raw
File Compare
OK


Dll Characteristics
Dynamic Base
Force Integrity
NX Compatible (DEP compatible)
No Isloation
No SEH
No Bind
WDM Driver
Terminal Server Aware
Set
Cancel

PIE

Исполнимые файлы, которые могут загружаться по произвольному адресу, называются Position Independent Executables (PIE).

Как бороться с ASLR?

Один из способов – наплевать на случайность!

Допустим, злоумышленник может сделать память выглядящей вот так:

```
0x90 0x90 0x90. . . 0x90 0x90 *code*
0x90 0x90 0x90. . . 0x90 0x90 *code*
0x90 0x90 0x90. . . 0x90 0x90 *code*
```

NOP sled и Heap spraying

NOP sled – конструкция вида

<много операторов NOP><шелкод>

Heap spraying – процедура размещения в памяти нужных данных

Используя NOP sled и Heap spraying возможно передавать управление на случайный адрес.

ROP

RETURN-ORIENTED PROGRAMMING

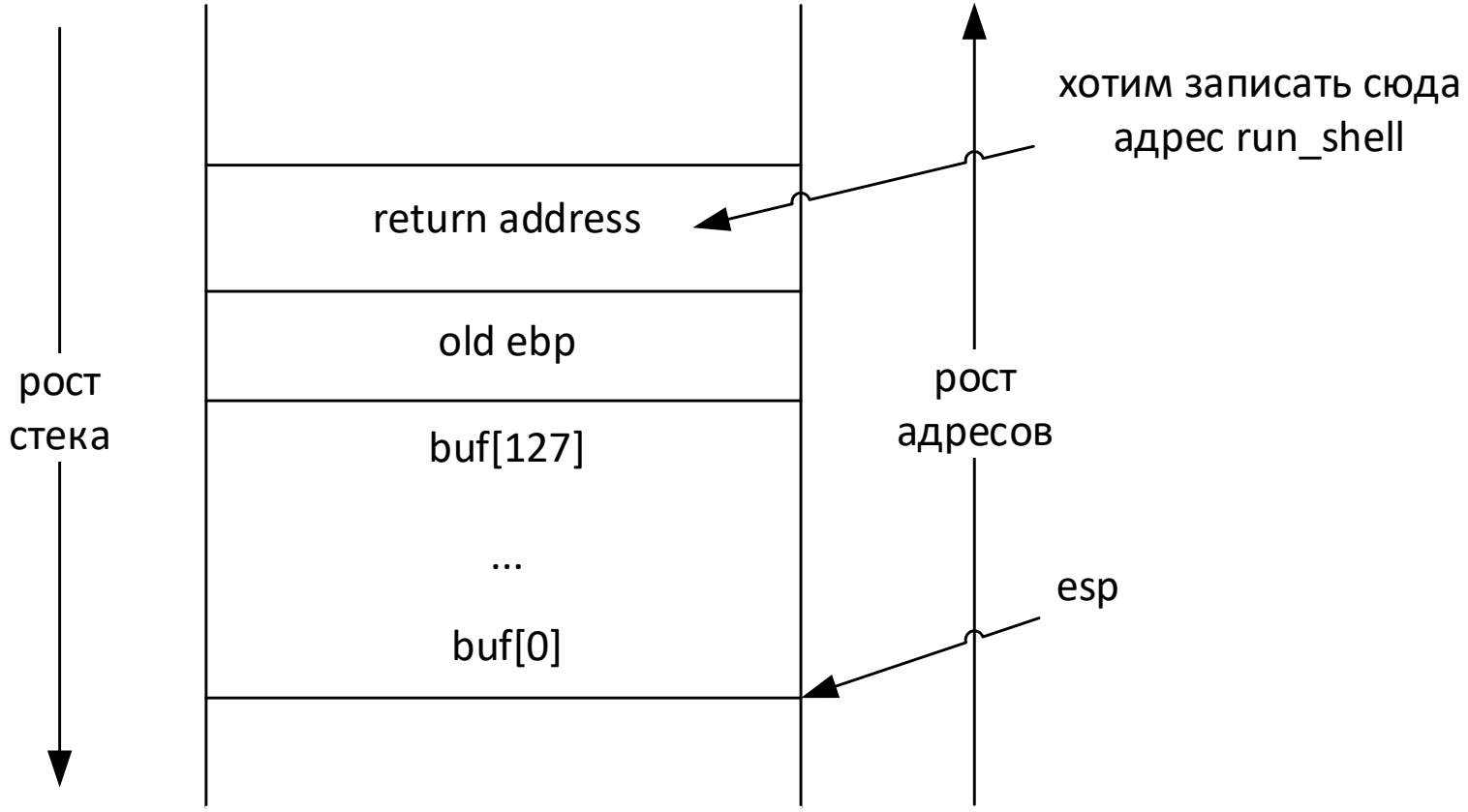
Почему Stack Canaries, DEP и ASLR не панацея?

Потому что хакеры каждый день думают, как взламывать программы!

Пример кода

```
void run_shell(){
    system("/bin/bash");
}
void process_msg(){
    char buf[128];
    gets(buf);
}
```

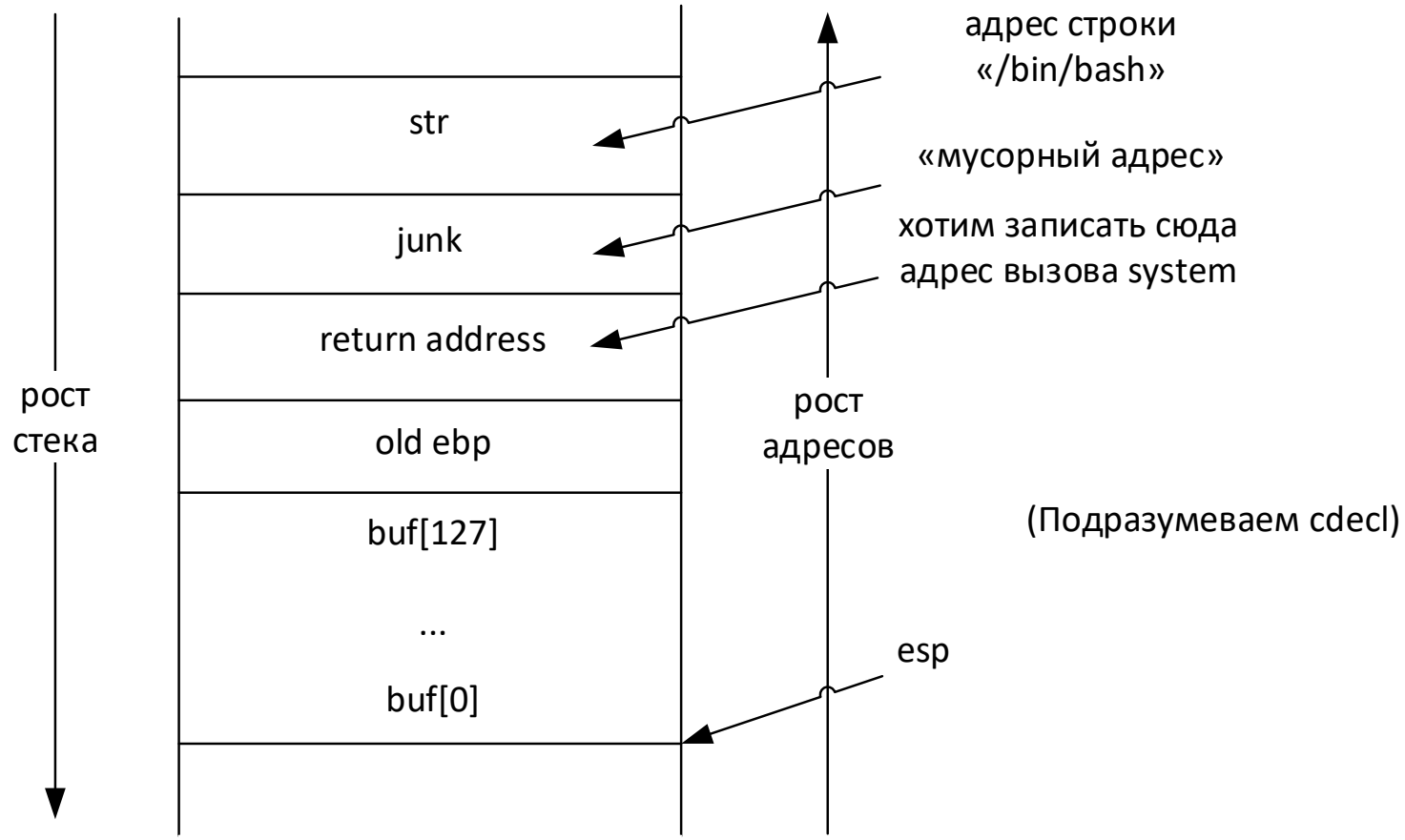
Пример эксплуатации



Пример кода

```
char* str = "/bin/bash";
void run_boring(){
    system("/bin/ls");
}
void process_msg(){
    char buf[128];
    gets(buf);
}
```

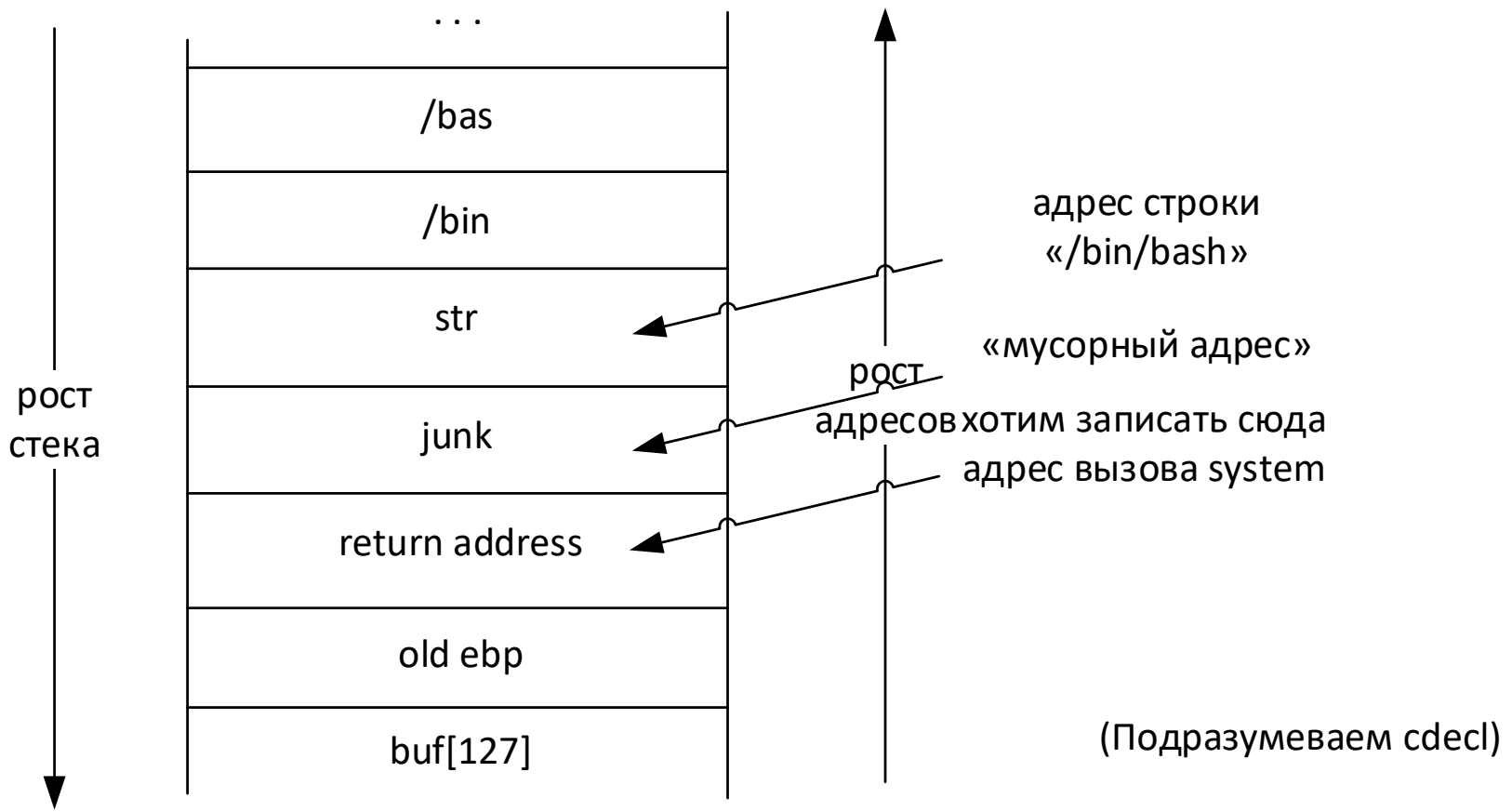
Пример эксплуатации



Пример кода

```
void run_boring(){
    system("/bin/ls");
}
void process_msg(){
    char buf[128];
    gets(buf);
}
```

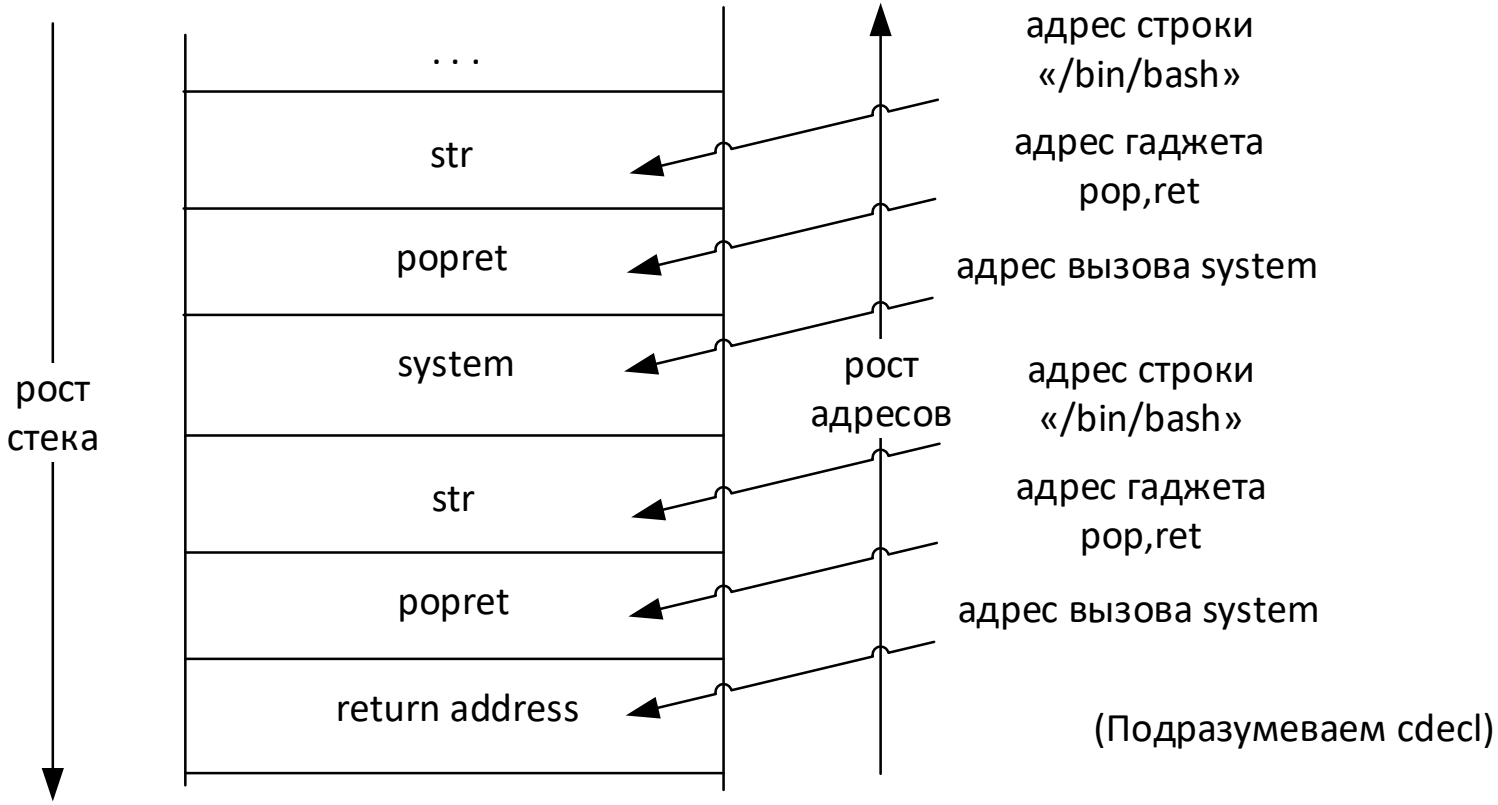
Пример эксплуатации



ROP-gadget (пример)

```
pop eax
ret
```

Пример эксплуатации



Как победить stack canary?

Допустим, что

1. Есть buffer overflow

2. Сервер падает при плохой проверке канарейки

3. Канарейка не меняется при перезапуске (такое может произойти из-за вызова fork)