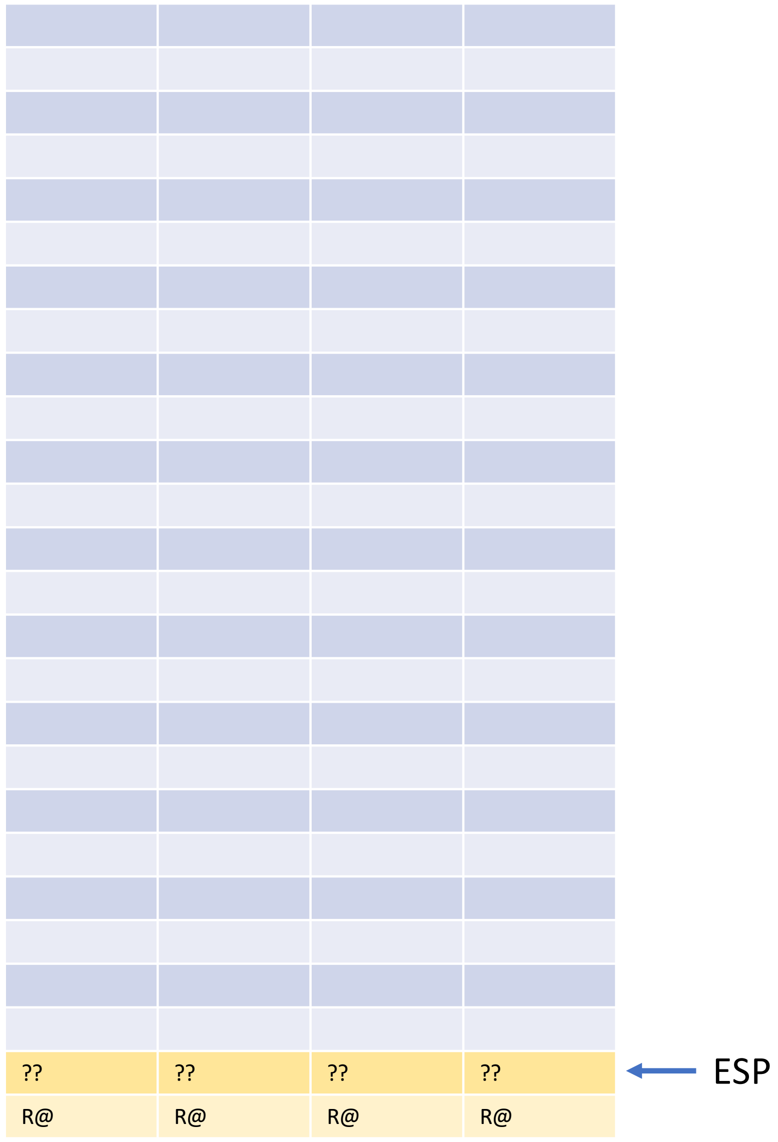
Stack & Function call

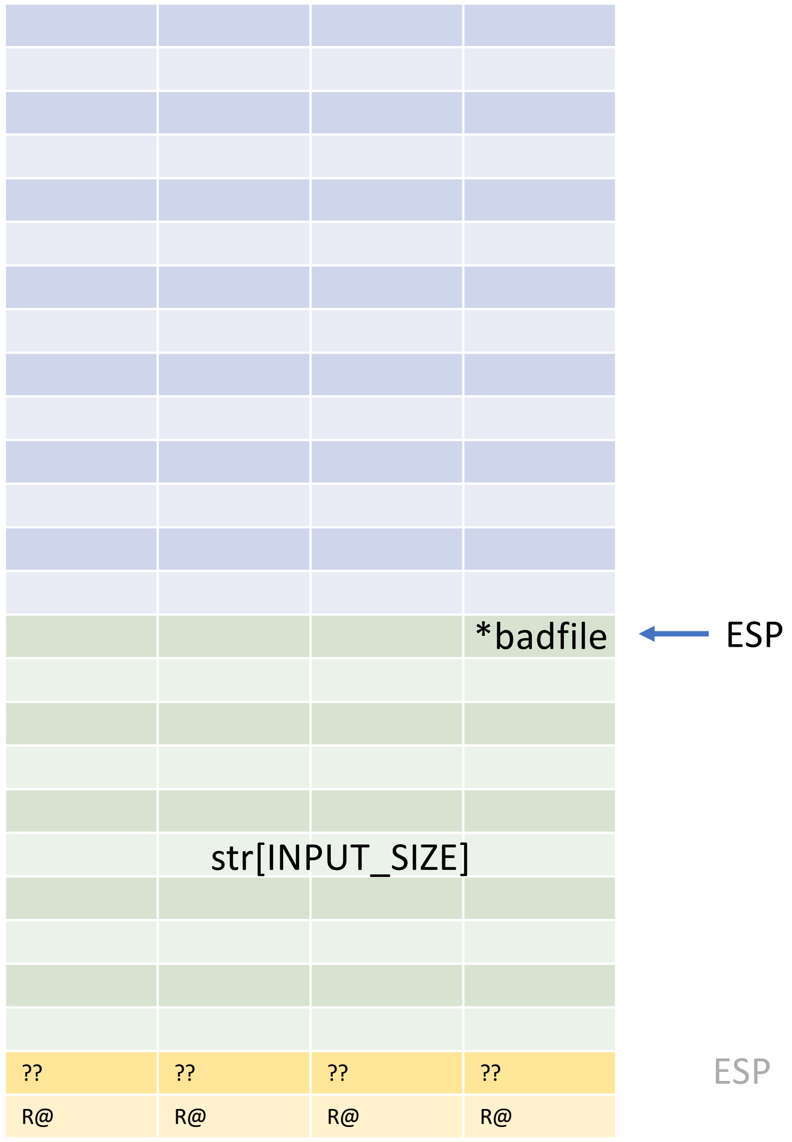
Source code

```
int bof(char *str) {

       char buffer[BUFFER_SIZE];
       strcpy(buffer,str);
}


int main(int argc, char* argv[]) {

       char str[INPUT_SIZE];
       FILE *badfile;
       badfile = fopen("badfile","r");
       fread(str, sizeof(char),INPUT_SIZE, badfile);
       fclose( badfile );

       bof(str);

       printf("Returned Properly\n");
       return 1;
}


// gcc –m32 -z execstack -fno-stack-protector -o stack stack.c
```

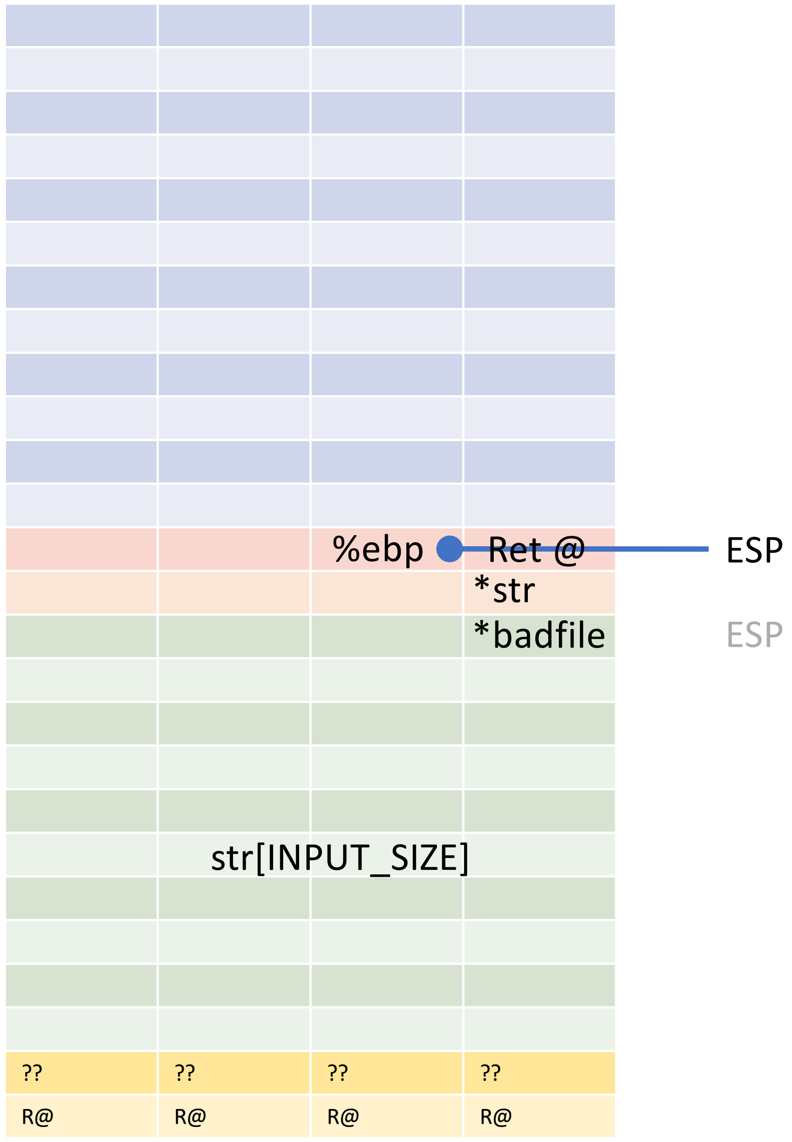
Buffer overflow could be exploited if
sizeof(str) >> sizeof(buffer)



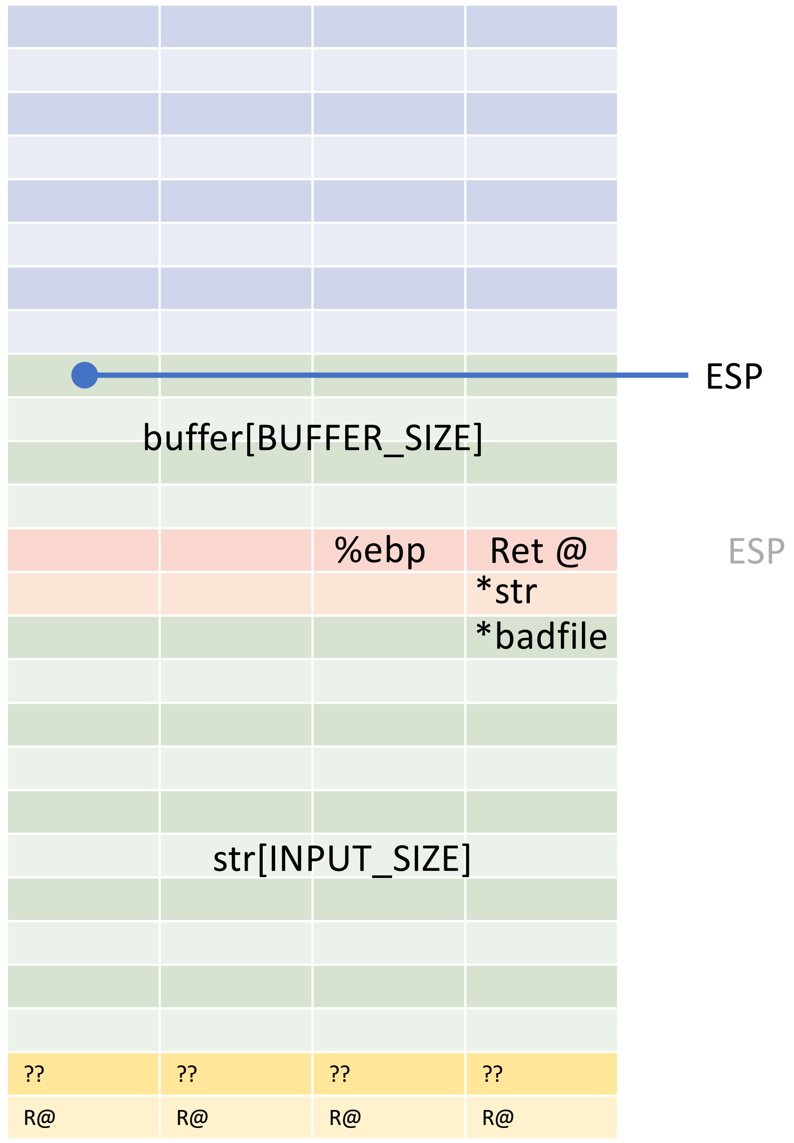
```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```



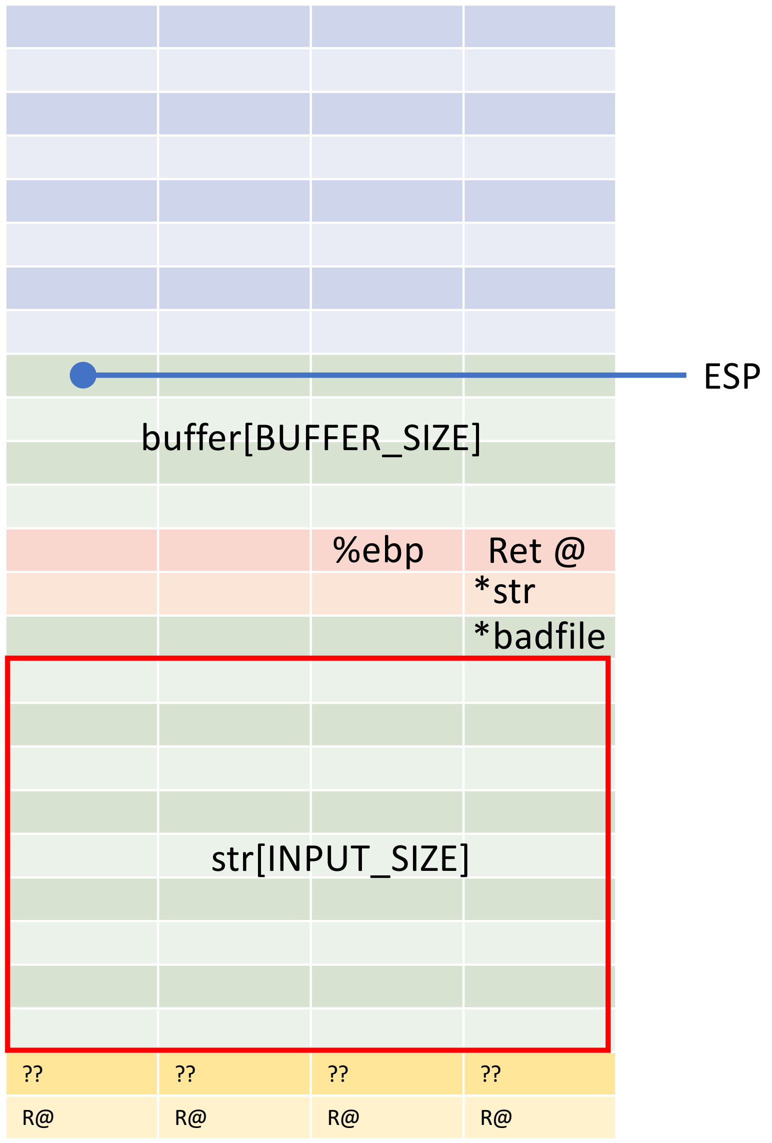
```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```



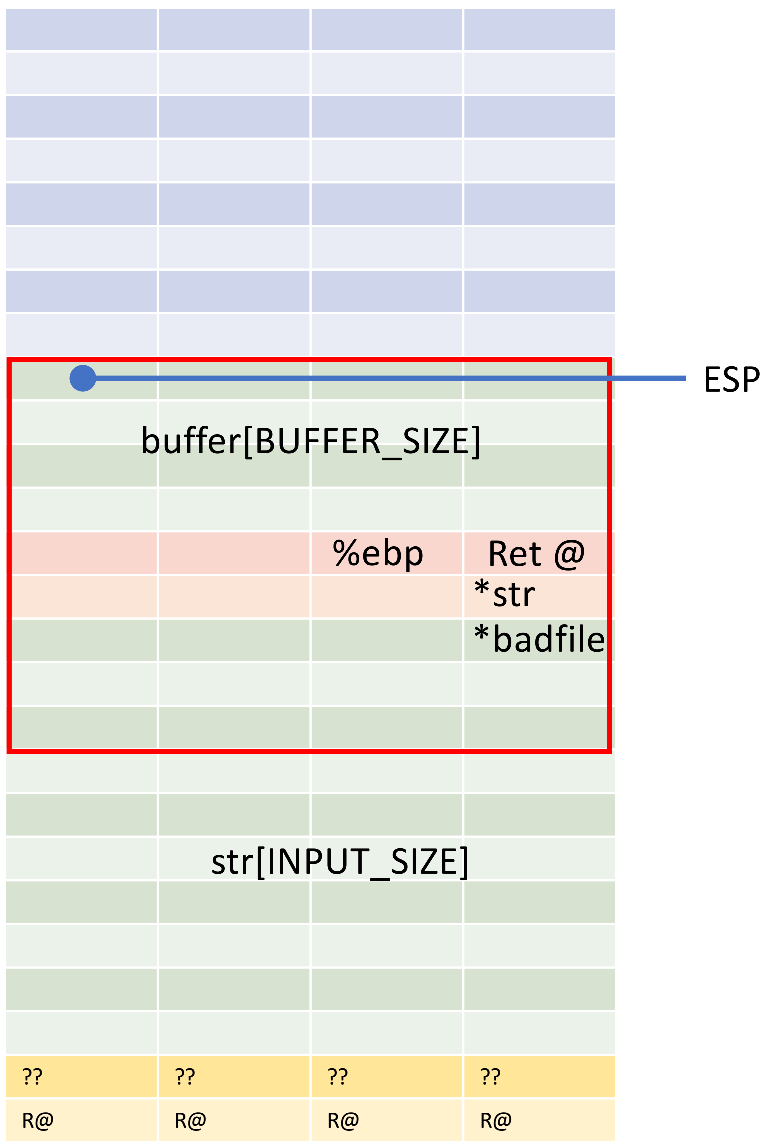
```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```



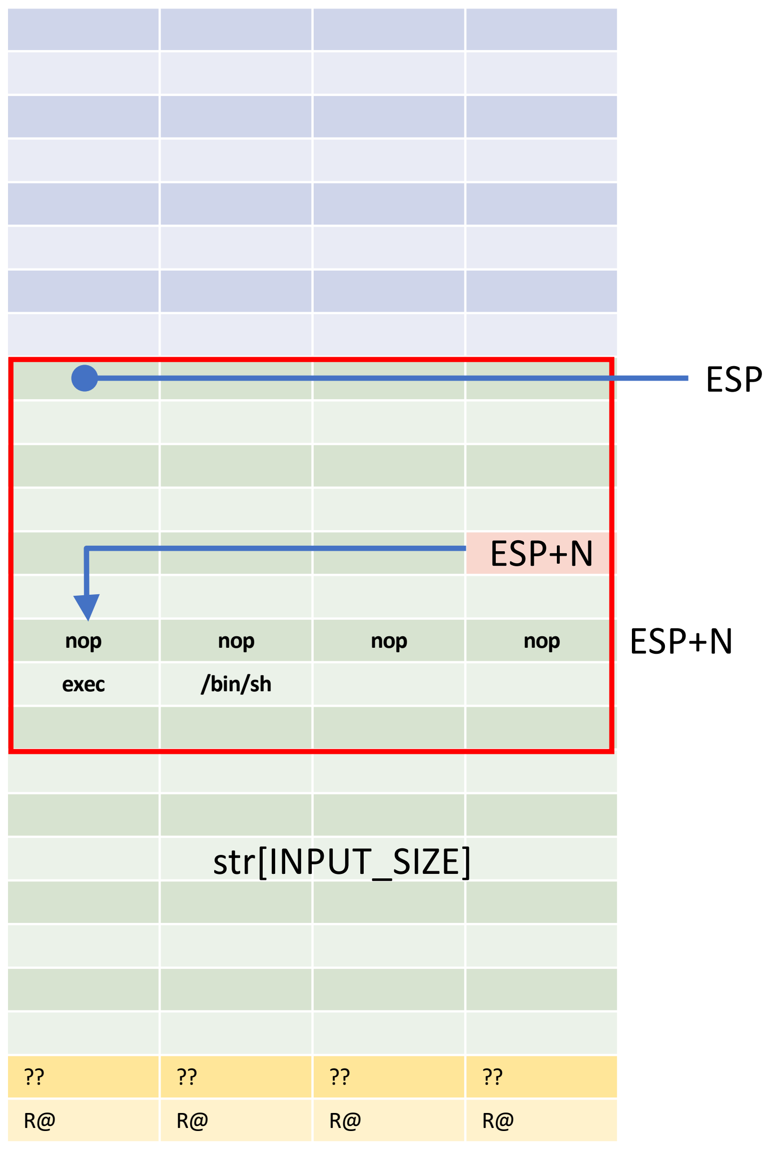
```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```



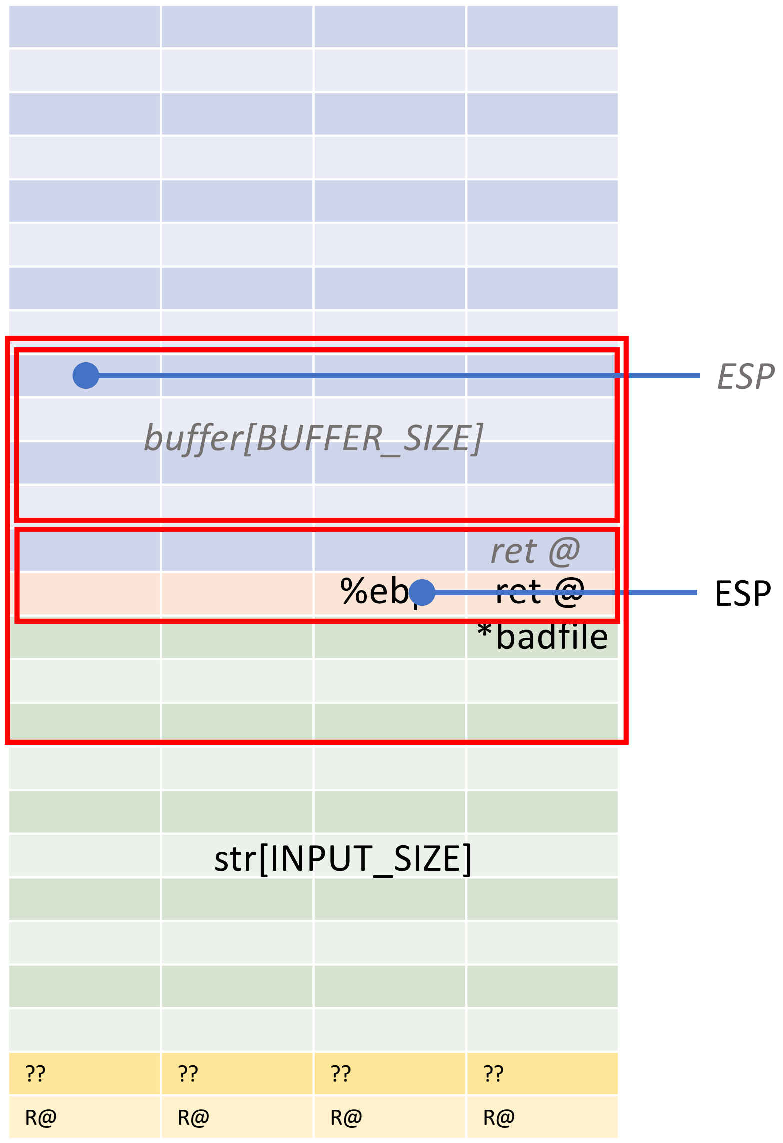
```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```



```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```

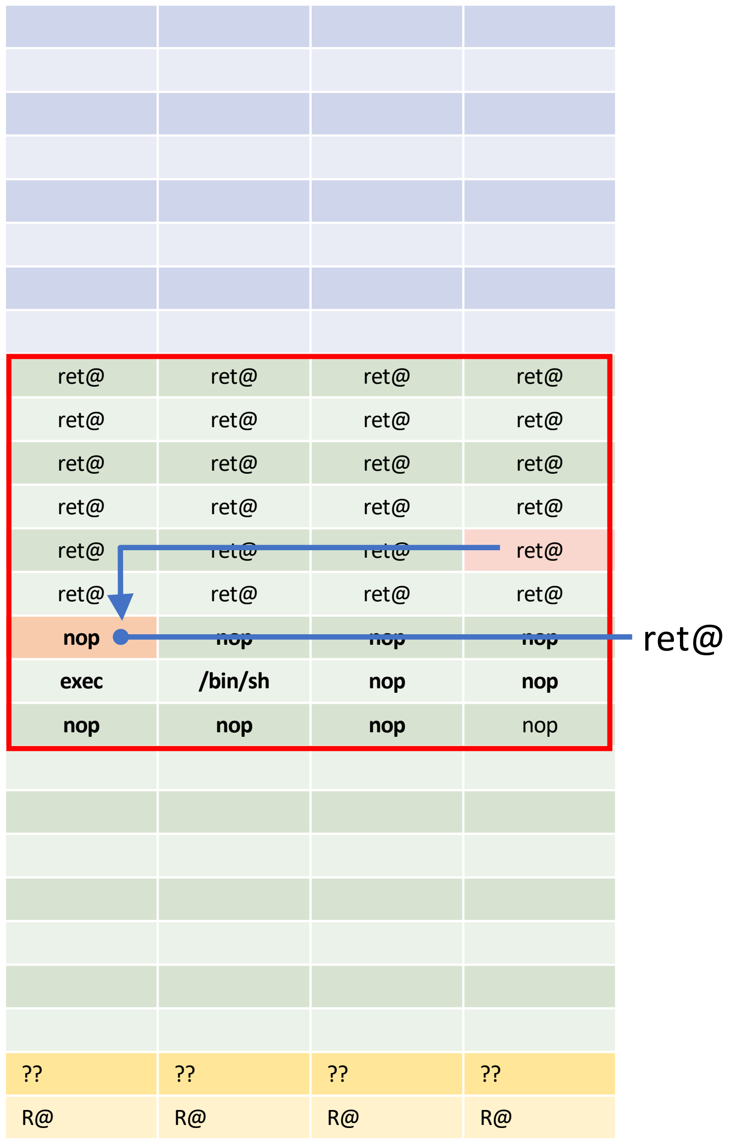


```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```



```
unsigned long get_sp(void) {
        __asm__("movl %esp,%eax");
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        printf("%.8lx\n", get_sp());
        badfile = fopen("badfile", »w");
        …
        fwrite(str, sizeof(char),INPUT_SIZE, badfile)
        fclose( badfile );
        return 1;
}
```

1. Fill str[] with nop
2. Compute return@ = ESP+n
3. Fill str[0.. BUFFER_SIZE+16] with return@
4. Copy shell code at str[BUFFER_SIZE+16+N] (N>n)



```
int bof(char *str) {
        char buffer[BUFFER_SIZE];
        strcpy(buffer,str);
        return 1;
}

int main(int argc, char* argv[]) {
        char str[INPUT_SIZE];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char),INPUT_SIZE, badfile);
        fclose( badfile );

        bof(str);

        printf("Returned Properly\n");
        return 1;
}
```