Vulnerabilidades de Software

Motivação

Exemplos Práticos de Vulnerabilidades de SW

Injeção de SQL

Teen Tells DEF CON How He Hacked Millions of Student Records From Popular Education Software [Update]

Alyse Stanley
Yesterday 12:21am • Filed to: HACKING

13.9K 9 1

"Hello from Bill Demirkapi :)" read the message sent to thousands of parents, students, and teachers in his school district after the aforementioned teenager hacked his school's education software. It was one of many bugs Demirkapi discovered over the last three years—another exposed millions of student records—that he presented on at this year's DEF CON, a hacker convention in Vegas.

The software belonged to two of the biggest names in education tech: Blackboard and Follett. Combined, these tech firms provide online education products for more than half the schools in America.

During Demirkapi's freshman year, a mixture of boredom and aimless ambition led him to start investigating the companies' interfaces. In Blackboard's Community Engagement software alone, he was able to access records for roughly 5 million students, everything from their phone numbers to their class schedules, by exploiting common bugs like "so-called SQL-injection and cross-site-scripting vulnerabilities," Wired reported. He found similar bugs in Follett's Student Information System, including student passwords that some genius left unencrypted for any fledgling security researcher like him to see.

Texas.gov & Florida.gov Hacked by New World Hackers, Access To Various Databases Leaked Online

BY BRIAN DUNN ON DECEMBER 31, 2018 • (0)

This afternoon, December 31st 2018, "Qurlla" of *New World Hackers* announced the breach of two databases tied to the States of Florida and Texas. The first hack targeted the state website of Florida, exposing information of what appears to either be various state contractors or employees, including their names, addresses, emails and phone numbers – roughly 1,066 lines of data. The second leak effected the state website of Texas and contains much more detailed/sensitive information, including 83 website administrators full names, email addresses, user login names and passwords. Additional information leaked from the site exposes access to the sites user database contents, as well as video files.

In a message to *Rogue Media Labs*, Qurlla explained how his group was able to hack both the websites via SQL injection, presumably through a vulnerability tied to the sites back end. However, the exact URL's effected or payloads delivered were not disclosed online. Additionally and perhaps most importantly, Qurlla did tag the states of Texas and Florida in the leak, meaning that there is no discernible timetable as to how long the credentials exposed online will remain valid – if you were interested in poking around, that is 😉.

Websites Targeted:

hxxp://texas.gov/
hxxp://florida.gov/

Texas.gov Leak: https://ghostbin.com/paste/ceeom
Florida.gov Leak: https://ghostbin.com/paste/zaq2y

> The State of Texas website database has been leaked by New World Hackers.
> You should really update your security @texasgov, please add load balancers.
> Targets: https://t.co/3EWYfOrn8f
> View Database: https://t.co/J0Emv8R2zK #Security #TangoDown
> pic.twitter.com/YVjQ5dHYEK

SQL Injection – como funciona...

- Uma aplicação recebe uma entrada de um usuário
  - Por exemplo, num campo de busca
- A entrada do usuário não recebe tratamento adequado
- O usuário envia um comando SQL
- O comando SQL é repassado para o banco de dados e executado
  - Esse comando pode exibir dados confidenciais, alterar campos, apagar tabelas inteiras...

```php
<?php
if($_SERVER['REQUEST_METHOD'] == 'POST') {
        $conexao=mysqli_connect('localhost', 'root', '[redacted]')  or die("Erro ao conectar ao banco de dados");
        mysqli_select_db($conexao,'testeDB');
        $nome = $_POST['nome'];
        $senha = $_POST['senha'];
        $query = "SELECT * FROM usuarios WHERE nome='$nome' AND senha='$senha'"; // xxx' OR 'a'='a
        echo $query;
        $result = mysqli_query($conexao,$query);
        if (mysqli_num_rows($result) > 0){
                echo "<br> <<<  Logado com sucesso  >>> <br>";
        }
        else {
                echo "<br> <<<  Não logou. Tente novamente.  >>> <br>";
        }
}
?>

<!DOCTYPE html>
<html>
<head>
        <title>Demonstrando Injection</title>
</head>
<body>
<form action="teste-SQLi.php" method="POST">
  <h2>Demonstrando SQL Injection</h2><br>
  Usuário:<br>
  <input type="text" name="nome"><br><br>
  Senha:<br>
  <input type="text" name="senha"><br><br>
  <input type="submit" value="Logar">
</form>
</body>
</html>
```

```
mysql> SELECT * FROM usuarios;
+-----------+-------------+
| nome      | senha       |
+-----------+-------------+
| Raphael   | Senha       |
| Joaozinho | OutraSenha  |
+-----------+-------------+
2 rows in set (0,00 sec)

mysql> SELECT * FROM usuarios WHERE nome='Raphael' AND senha='Senha';
+---------+-------+
| nome    | senha |
+---------+-------+
| Raphael | Senha |
+---------+-------+
1 row in set (0,00 sec)

mysql> SELECT * FROM usuarios WHERE nome='Joaozinho' AND senha='OutraSenha';
+-----------+-------------+
| nome      | senha       |
+-----------+-------------+
| Joaozinho | OutraSenha  |
+-----------+-------------+
1 row in set (0,00 sec)

mysql> SELECT * FROM usuarios WHERE nome='Joaozinho' AND senha='Senha';
Empty set (0,00 sec)

mysql> SELECT * FROM usuarios WHERE nome='Joaozinho' AND senha='Senha' OR 'a'='a';
+-----------+-------------+
| nome      | senha       |
+-----------+-------------+
| Raphael   | Senha       |
| Joaozinho | OutraSenha  |
+-----------+-------------+
2 rows in set (0,00 sec)

mysql>
```

Demonstrando SQL Injection

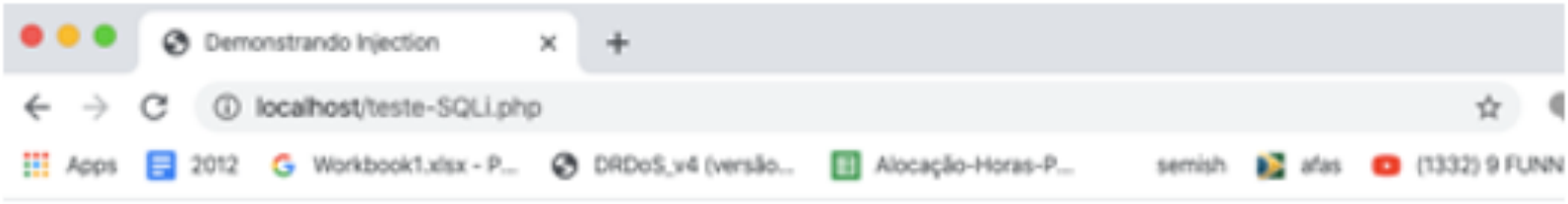
Usuário:

xzxzxzxz

Senha:

xxx' OR 'a'='a

Logar



SELECT * FROM usuarios WHERE nome='xzxzxzxz' AND senha='xxx' OR 'a'='a'
<<< Logado com sucesso >>>

Demonstrando SQL Injection

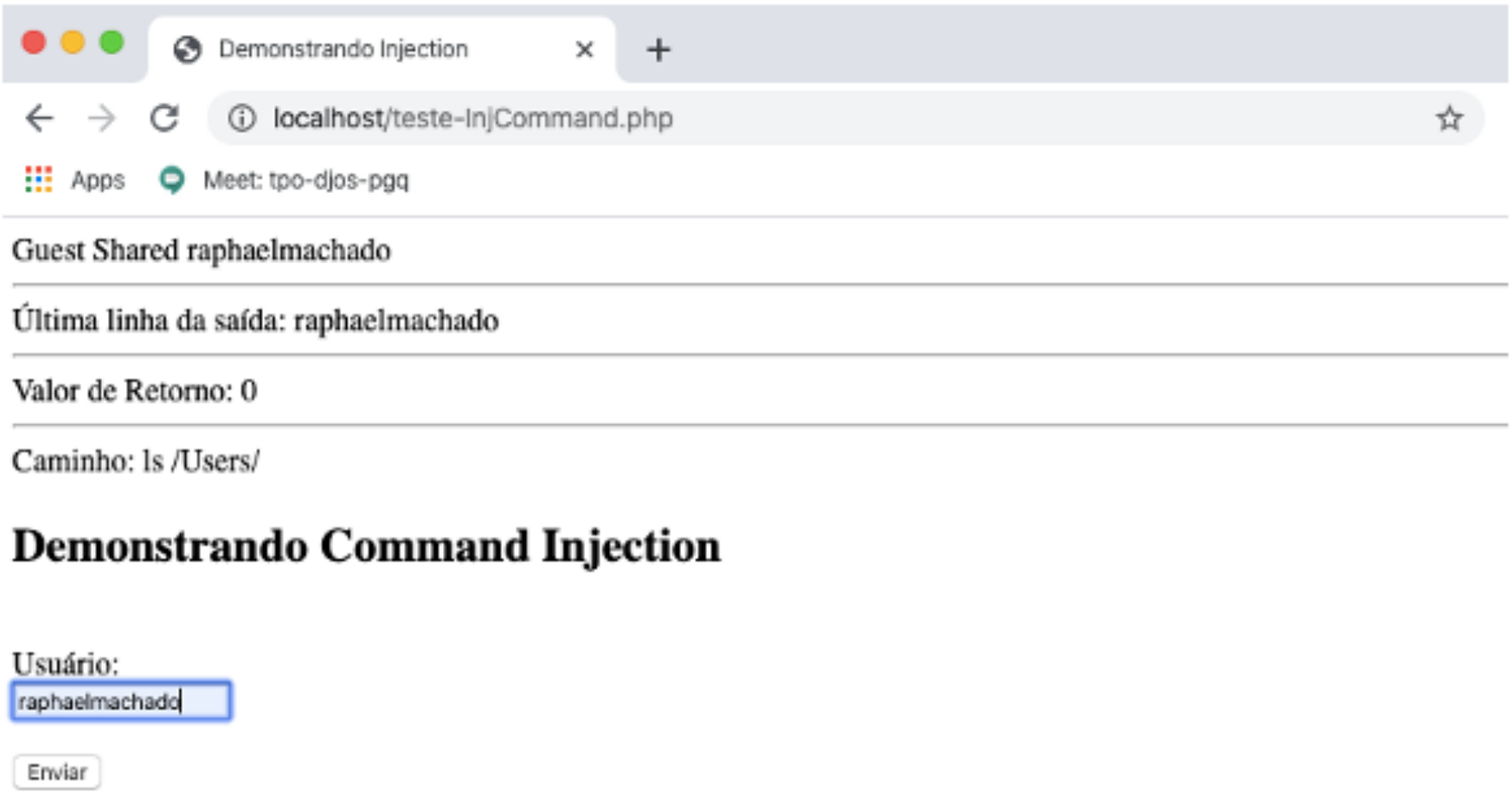
Usuário:

Senha:

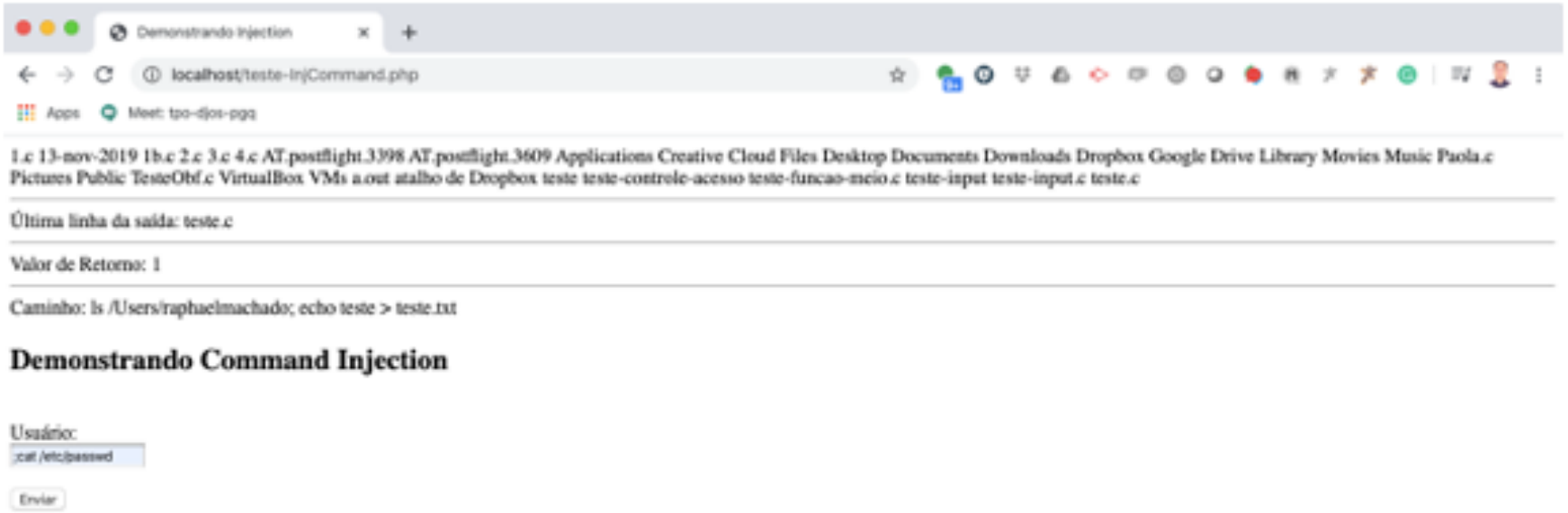
Logar

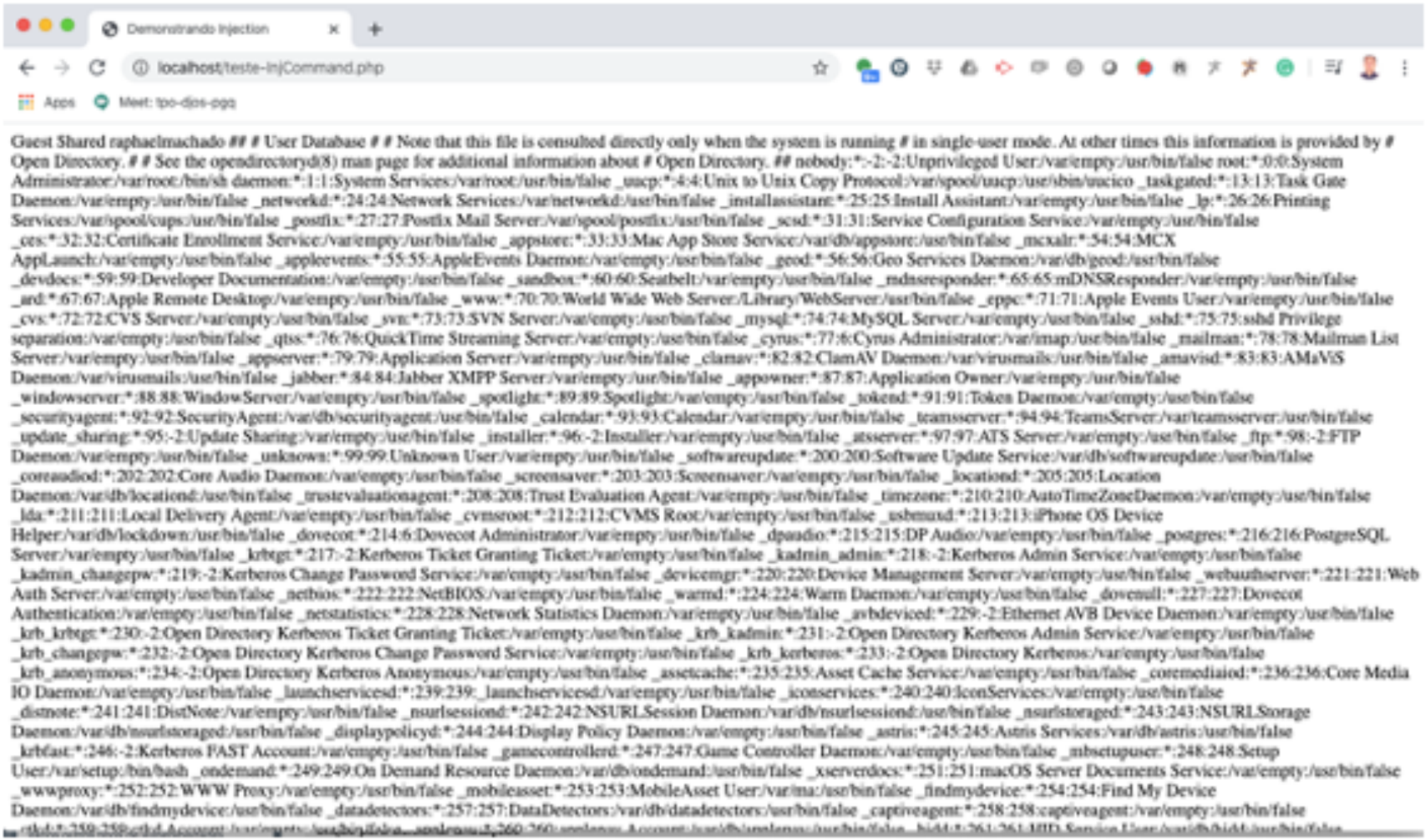
Injeção de Comandos

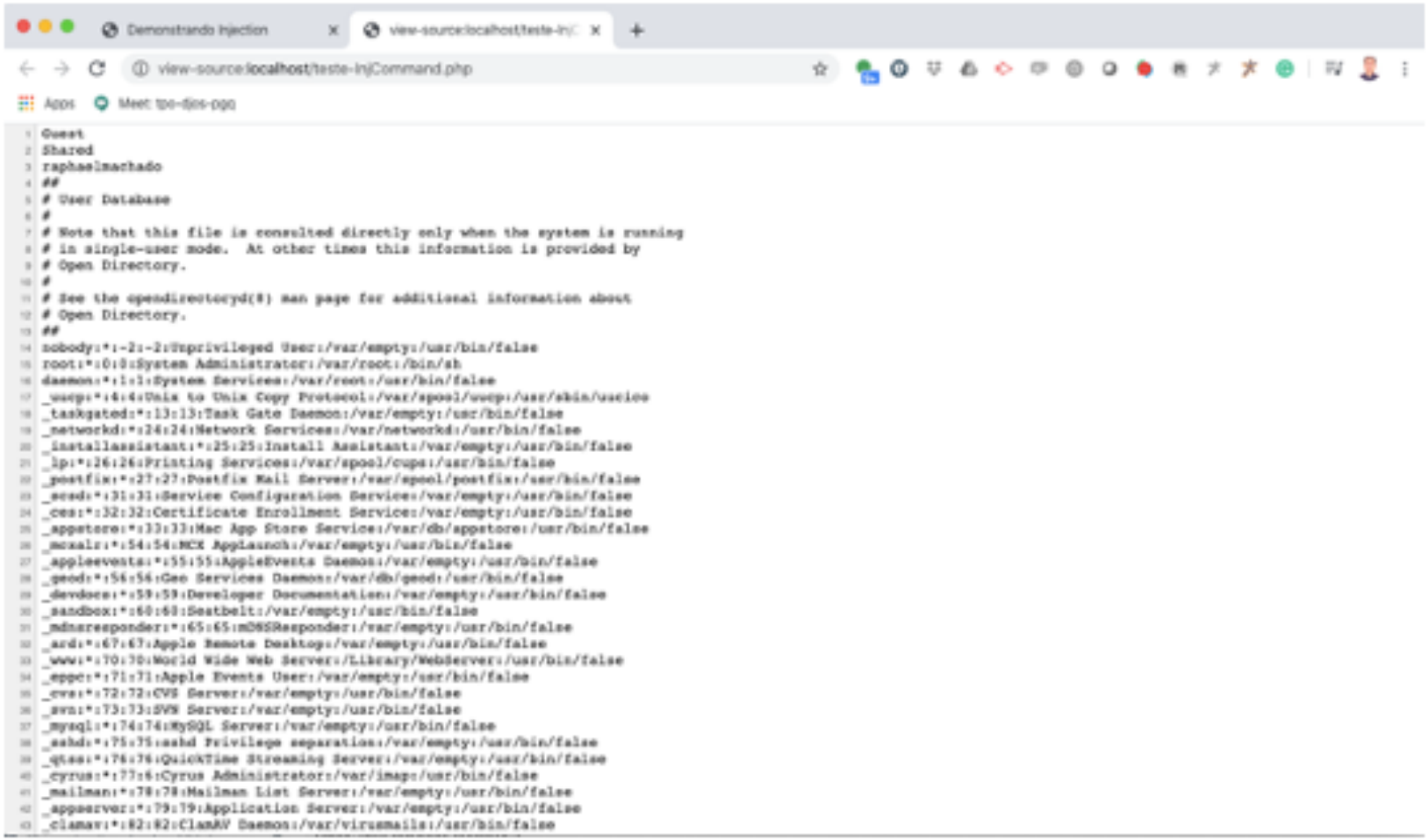
```php
<?php

        $userName = $_POST["nome"];

        $caminho='ls /Users/'.$userName;
        $last_line = system('ls /Users/'.$userName, $retval);

        // Mostrando informação adicional
        echo '
        </pre>
        <hr />Última linha da saída: '.$last_line.'
        <hr />Valor de Retorno: '.$retval.'
        <hr />Caminho: '.$caminho;

?>

<!DOCTYPE html>
<html>
<head>
        <title>Demonstrando Injection</title>
</head>
<body>
<form action="teste-InjCommand.php" method="POST">
  <h2>Demonstrando Command Injection</h2><br>
  Usuário:<br>
  <input type="text" name="nome"><br><br>
  <input type="submit" value="Enviar">
</form>
</body>
</html>
```


Demonstrando Injection
localhost/teste-InjCommand.php
Apps
Meet: tpo-djos-pgq
Guest Shared raphaelmachado
Última linha da saída: raphaelmachado
Valor de Retorno: 0
Caminho: ls /Users/
Demonstrando Command Injection
Usuário:
raphaelmachado
Enviar


Demonstrando Injection
localhost/teste-InjCommand.php
Apps
Meet: tpo-djos-pgq
1.c 13-nov-2019 1b.c 2.c 3.c 4.c AT.postflight.3398 AT.postflight.3609 Applications Creative Cloud Files Desktop Documents Downloads Dropbox Google Drive Library Movies Music Paola.c Pictures Public TesteObf.c VirtualBox VMs a.out atalho de Dropbox teste teste-controle-acesso teste-funcao-meio.c teste-input teste-input.c teste.c
Última linha da saída: teste.c
Valor de Retorno: 0
Caminho: ls /Users/raphaelmachado
Demonstrando Command Injection
Usuário:
Enviar

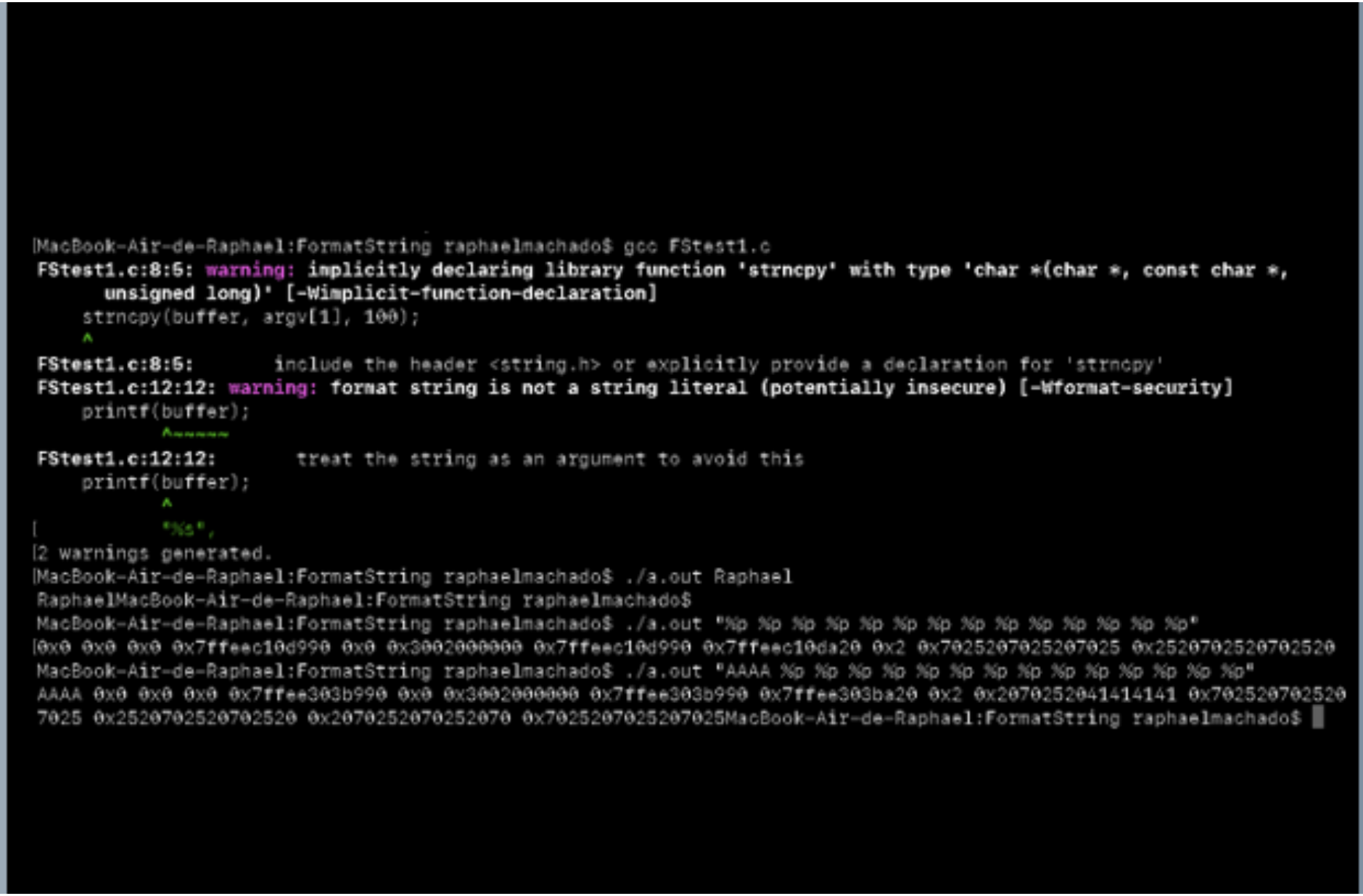

Demonstrando Injection
localhost/teste-InjCommand.php
Apps
Meet: tpo-djos-pgq
1.c 13-nov-2019 1b.c 2.c 3.c 4.c AT.postflight.3398 AT.postflight.3609 Applications Creative Cloud Files Desktop Documents Downloads Dropbox Google Drive Library Movies Music Paola.c Pictures Public TesteObf.c VirtualBox VMs a.out atalho de Dropbox teste teste-controle-acesso teste-funcao-meio.c teste-input teste-input.c teste.c
Última linha da saída: teste.c
Valor de Retorno: 1
Caminho: ls /Users/raphaelmachado; echo teste > teste.txt
Demonstrando Command Injection
Usuário:
;cat /etc/passwd
Enviar


Demonstrando Injection
localhost/teste-InjCommand.php
Apps
Meet: tpo-dos-ogq
Guest Shared raphaelmachado ## # User Database # # Note that this file is consulted directly only when the system is running # in single-user mode. At other times this information is provided by # Open Directory. # # See the opendirectoryd(8) man page for additional information about # Open Directory. ## nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false root:*:0:0:System Administrator:/var/root:/bin/sh daemon:*:1:1:System Services:/var/root:/usr/bin/false _uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/uucico _taskgated:*:13:13:Task Gate Daemon:/var/empty:/usr/bin/false _networkd:*:24:24:Network Services:/var/networkd:/usr/bin/false _installassistant:*:25:25:Install Assistant:/var/empty:/usr/bin/false _lp:*:26:26:Printing Services:/var/spool/cups:/usr/bin/false _postfix:*:27:27:Postfix Mail Server:/var/spool/postfix:/usr/bin/false _scsd:*:31:31:Service Configuration Service:/var/empty:/usr/bin/false _ces:*:32:32:Certificate Enrollment Service:/var/empty:/usr/bin/false _appstore:*:33:33:Mac App Store Service:/var/db/appstore:/usr/bin/false _mcxalr:*:54:54:MCX AppLaunch:/var/empty:/usr/bin/false _appleevents:*:55:55:AppleEvents Daemon:/var/empty:/usr/bin/false _geod:*:56:56:Geo Services Daemon:/var/db/geod:/usr/bin/false _devdocs:*:59:59:Developer Documentation:/var/empty:/usr/bin/false _sandbox:*:60:60:Seatbelt:/var/empty:/usr/bin/false _mdnsresponder:*:65:65:mDNSResponder:/var/empty:/usr/bin/false _ard:*:67:67:Apple Remote Desktop:/var/empty:/usr/bin/false _www:*:70:70:World Wide Web Server:/Library/WebServer:/usr/bin/false _eppc:*:71:71:Apple Events User:/var/empty:/usr/bin/false _cvs:*:72:72:CVS Server:/var/empty:/usr/bin/false _svn:*:73:73:SVN Server:/var/empty:/usr/bin/false _mysql:*:74:74:MySQL Server:/var/empty:/usr/bin/false _sshd:*:75:75:sshd Privilege separation:/var/empty:/usr/bin/false _qtss:*:76:76:QuickTime Streaming Server:/var/empty:/usr/bin/false _cyrus:*:77:6:Cyrus Administrator:/var/imap:/usr/bin/false _mailman:*:78:78:Mailman List Server:/var/empty:/usr/bin/false _appserver:*:79:79:Application Server:/var/empty:/usr/bin/false _clamav:*:82:82:ClamAV Daemon:/var/virusmails:/usr/bin/false _amavisd:*:83:83:AMaViS Daemon:/var/virusmails:/usr/bin/false _jabber:*:84:84:Jabber XMPP Server:/var/empty:/usr/bin/false _appowner:*:87:87:Application Owner:/var/empty:/usr/bin/false _windowserver:*:88:88:WindowServer:/var/empty:/usr/bin/false _spotlight:*:89:89:Spotlight:/var/empty:/usr/bin/false _tokend:*:91:91:Token Daemon:/var/empty:/usr/bin/false _securityagent:*:92:92:SecurityAgent:/var/db/securityagent:/usr/bin/false _calendar:*:93:93:Calendar:/var/empty:/usr/bin/false _teamsserver:*:94:94:TeamsServer:/var/teamsserver:/usr/bin/false _update_sharing:*:95:-2:Update Sharing:/var/empty:/usr/bin/false _installer:*:96:-2:Installer:/var/empty:/usr/bin/false _atsserver:*:97:97:ATS Server:/var/empty:/usr/bin/false _ftp:*:98:-2:FTP Daemon:/var/empty:/usr/bin/false _unknown:*:99:99:Unknown User:/var/empty:/usr/bin/false _softwareupdate:*:200:200:Software Update Service:/var/db/softwareupdate:/usr/bin/false _coreaudiod:*:202:202:Core Audio Daemon:/var/empty:/usr/bin/false _screensaver:*:203:203:Screensaver:/var/empty:/usr/bin/false _locationd:*:205:205:Location Daemon:/var/db/locationd:/usr/bin/false _trustevaluationagent:*:208:208:Trust Evaluation Agent:/var/empty:/usr/bin/false _timezone:*:210:210:AutoTimeZoneDaemon:/var/empty:/usr/bin/false _lda:*:211:211:Local Delivery Agent:/var/empty:/usr/bin/false _cvmsroot:*:212:212:CVMS Root:/var/empty:/usr/bin/false _usbmuxd:*:213:213:iPhone OS Device Helper:/var/db/lockdown:/usr/bin/false _dovecot:*:214:6:Dovecot Administrator:/var/empty:/usr/bin/false _dpaudio:*:215:215:DP Audio:/var/empty:/usr/bin/false _postgres:*:216:216:PostgreSQL Server:/var/empty:/usr/bin/false _krbtgt:*:217:-2:Kerberos Ticket Granting Ticket:/var/empty:/usr/bin/false _kadmin_admin:*:218:-2:Kerberos Admin Service:/var/empty:/usr/bin/false _kadmin_changepw:*:219:-2:Kerberos Change Password Service:/var/empty:/usr/bin/false _devicemgr:*:220:220:Device Management Server:/var/empty:/usr/bin/false _webauthserver:*:221:221:Web Auth Server:/var/empty:/usr/bin/false _netbios:*:222:222:NetBIOS:/var/empty:/usr/bin/false _warmd:*:224:224:Warm Daemon:/var/empty:/usr/bin/false _dovenull:*:227:227:Dovecot Authentication:/var/empty:/usr/bin/false _netstatistics:*:228:228:Network Statistics Daemon:/var/empty:/usr/bin/false _avbdeviced:*:229:-2:Ethernet AVB Device Daemon:/var/empty:/usr/bin/false _krb_krbtgt:*:230:-2:Open Directory Kerberos Ticket Granting Ticket:/var/empty:/usr/bin/false _krb_kadmin:*:231:-2:Open Directory Kerberos Admin Service:/var/empty:/usr/bin/false _krb_changepw:*:232:-2:Open Directory Kerberos Change Password Service:/var/empty:/usr/bin/false _krb_kerberos:*:233:-2:Open Directory Kerberos:/var/empty:/usr/bin/false _krb_anonymous:*:234:-2:Open Directory Kerberos Anonymous:/var/empty:/usr/bin/false _assetcache:*:235:235:Asset Cache Service:/var/empty:/usr/bin/false _coremediaiod:*:236:236:Core Media IO Daemon:/var/empty:/usr/bin/false _launchservicesd:*:239:239:_launchservicesd:/var/empty:/usr/bin/false _iconservices:*:240:240:IconServices:/var/empty:/usr/bin/false _distnote:*:241:241:DistNote:/var/empty:/usr/bin/false _nsurlsessiond:*:242:242:NSURLSession Daemon:/var/db/nsurlsessiond:/usr/bin/false _nsurlstoraged:*:243:243:NSURLStorage Daemon:/var/db/nsurlstoraged:/usr/bin/false _displaypolicyd:*:244:244:Display Policy Daemon:/var/empty:/usr/bin/false _astris:*:245:245:Astris Services:/var/db/astris:/usr/bin/false _krbfast:*:246:-2:Kerberos FAST Account:/var/empty:/usr/bin/false _gamecontrollerd:*:247:247:Game Controller Daemon:/var/empty:/usr/bin/false _mbsetupuser:*:248:248:Setup User:/var/setup:/bin/bash _ondemand:*:249:249:On Demand Resource Daemon:/var/db/ondemand:/usr/bin/false _xserverdocs:*:251:251:macOS Server Documents Service:/var/empty:/usr/bin/false _wwwproxy:*:252:252:WWW Proxy:/var/empty:/usr/bin/false _mobileasset:*:253:253:MobileAsset User:/var/ma:/usr/bin/false _findmydevice:*:254:254:Find My Device Daemon:/var/db/findmydevice:/usr/bin/false _datadetectors:*:257:257:DataDetectors:/var/db/datadetectors:/usr/bin/false _captiveagent:*:258:258:captiveagent:/var/empty:/usr/bin/false


Demonstrando Injection
view-source:localhost/teste-In
view-source:localhost/teste-Inj/Command.php
Apps
Meet tpe-djos-pgo
Guest
Shared
raphaelmachado
##
User Database
#
Note that this file is consulted directly only when the system is running
in single-user mode. At other times this information is provided by
Open Directory.
#
See the opendirectoryd(8) man page for additional information about
Open Directory.
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/uucico
_taskgated:*:13:13:Task Gate Daemon:/var/empty:/usr/bin/false
_networkd:*:24:24:Network Services:/var/networkd:/usr/bin/false
_installassistant:*:25:25:Install Assistant:/var/empty:/usr/bin/false
_lp:*:26:26:Printing Services:/var/spool/cups:/usr/bin/false
_postfix:*:27:27:Postfix Mail Server:/var/spool/postfix:/usr/bin/false
_scsd:*:31:31:Service Configuration Service:/var/empty:/usr/bin/false
_ces:*:32:32:Certificate Enrollment Service:/var/empty:/usr/bin/false
_appstore:*:33:33:Mac App Store Service:/var/db/appstore:/usr/bin/false
_mcxalr:*:54:54:MCX AppLaunch:/var/empty:/usr/bin/false
_appleevents:*:55:55:AppleEvents Daemon:/var/empty:/usr/bin/false
_geod:*:56:56:Geo Services Daemon:/var/db/geod:/usr/bin/false
_devdocs:*:59:59:Developer Documentation:/var/empty:/usr/bin/false
_sandbox:*:60:60:Seatbelt:/var/empty:/usr/bin/false
_mdnsresponder:*:65:65:mDNSResponder:/var/empty:/usr/bin/false
_ard:*:67:67:Apple Remote Desktop:/var/empty:/usr/bin/false
_www:*:70:70:World Wide Web Server:/Library/WebServer:/usr/bin/false
_eppc:*:71:71:Apple Events User:/var/empty:/usr/bin/false
_cvs:*:72:72:CVS Server:/var/empty:/usr/bin/false
_svn:*:73:73:SVN Server:/var/empty:/usr/bin/false
_mysql:*:74:74:MySQL Server:/var/empty:/usr/bin/false
_sshd:*:75:75:sshd Privilege separation:/var/empty:/usr/bin/false
_qtss:*:76:76:QuickTime Streaming Server:/var/empty:/usr/bin/false
_cyrus:*:77:6:Cyrus Administrator:/var/imap:/usr/bin/false
_mailman:*:78:78:Mailman List Server:/var/empty:/usr/bin/false
_appserver:*:79:79:Application Server:/var/empty:/usr/bin/false
_clamav:*:82:82:ClamAV Daemon:/var/virusmails:/usr/bin/false

Format String

```c
// Exemplo de programa simples em C com
// vulnerabilidade format string
#include<stdio.h>

int main(int argc, char** argv)
{
    char buffer[100];
    strncpy(buffer, argv[1], 100);

    // Passando argumento de linha de comando
    // diretamente para o printf
    printf(buffer);

    return 0;
}
```


MacBook-Air-de-Raphael:FormatString raphaelmachado$ gcc FStest1.c
FStest1.c:8:5: warning: implicitly declaring library function 'strncpy' with type 'char *(char *, const char *, unsigned long)' [-Wimplicit-function-declaration]
strncpy(buffer, argv[1], 100);
FStest1.c:8:5: include the header <string.h> or explicitly provide a declaration for 'strncpy'
FStest1.c:12:12: warning: format string is not a string literal (potentially insecure) [-Wformat-security]
printf(buffer);
FStest1.c:12:12: treat the string as an argument to avoid this
printf(buffer);
"%s",
2 warnings generated.
MacBook-Air-de-Raphael:FormatString raphaelmachado$ ./a.out Raphael
RaphaelMacBook-Air-de-Raphael:FormatString raphaelmachado$
MacBook-Air-de-Raphael:FormatString raphaelmachado$ ./a.out "%p %p %p %p %p %p %p %p %p %p %p %p %p %p"
0x0 0x0 0x0 0x7ffeec10d990 0x0 0x3002000000 0x7ffeec10d990 0x7ffeec10da20 0x2 0x702520702520702520 0x2520702520702520
MacBook-Air-de-Raphael:FormatString raphaelmachado$ ./a.out "AAAA %p %p %p %p %p %p %p %p %p %p %p %p %p %p"
AAAA 0x0 0x0 0x0 0x7ffee303b990 0x0 0x3002000000 0x7ffee303b990 0x7ffee303ba20 0x2 0x2070252041414141 0x702520702520
7025 0x2520702520702520 0x2070252070252070 0x702520702520702MacBook-Air-de-Raphael:FormatString raphaelmachado$

Estouro de Limites de Memória

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {

        char senha[7] = "";
        char status[100]= ""; //I incorreta, C correta

        strcpy(status,"I");

        printf("Digite senha: ");
        scanf("%s",senha);

        if (!strcmp(senha, "Raphael")) {
                        strcpy(status,"C");
        }

        printf("\nSenha Digitada: %s.\nStatus: %s.\n", senha, status);

        if (!strcmp(status, "I")) {
                printf("Senha incorreta. Conexão Terminada.\n");
        } else {
                printf("Senha correta. Acesso concedido.\n");
        }

//      for(int i=0;i<20;i++)printf("%c,%c;",senha[i],status[i]);

        return(0);
}
```

```
MacBook-Air-de-Raphael:BufferOverflow raphaelmachado$ gcc EstouroMemoria.c
MacBook-Air-de-Raphael:BufferOverflow raphaelmachado$ ./a.out
Digite senha: rrr

Senha Digitada: rrr.
Status: I.
Senha incorreta. Conexão Terminada.
MacBook-Air-de-Raphael:BufferOverflow raphaelmachado$ ./a.out
Digite senha: Raphael

Senha Digitada: Raphael.
Status: C.
Senha correta. Acesso concedido.
MacBook-Air-de-Raphael:BufferOverflow raphaelmachado$ ./a.out
Digite senha: 123456789

Senha Digitada: 123456789.
Status: I.
Senha incorreta. Conexão Terminada.
MacBook-Air-de-Raphael:BufferOverflow raphaelmachado$ ./a.out
Digite senha: 12345678901234567890

Senha Digitada: 12345678901234567890.
Status: 234567890.
Senha correta. Acesso concedido.
MacBook-Air-de-Raphael:BufferOverflow raphaelmachado$
```

Um pouco mais sobre as “falhas”
dos programas anteriores...

Falha de implementação

- Nenhum requisitos de segurança foi mal-especificado
- Nenhuma premissa de segurança foi assumida erroneamente
- Nenhum modelo de ataque plausível foi ignorado
- Simplesmente, a implementação de uma funcionalidade abriu portas para um ataque

Falha: Injeção de SQL

- xxx

Exemplo de Injeção de SQL

› Código em PHP

```
$id = $_REQUEST[“id”];
$pass = $_REQUEST[“password”];
$qry = “SELECT ccnum FROM cust WHERE id = ‘$id’ AND pass=‘$pass’”;
```

Atacante inclui id do usuário alvo

Para a senha, entrega: `' OR 1=1 –`

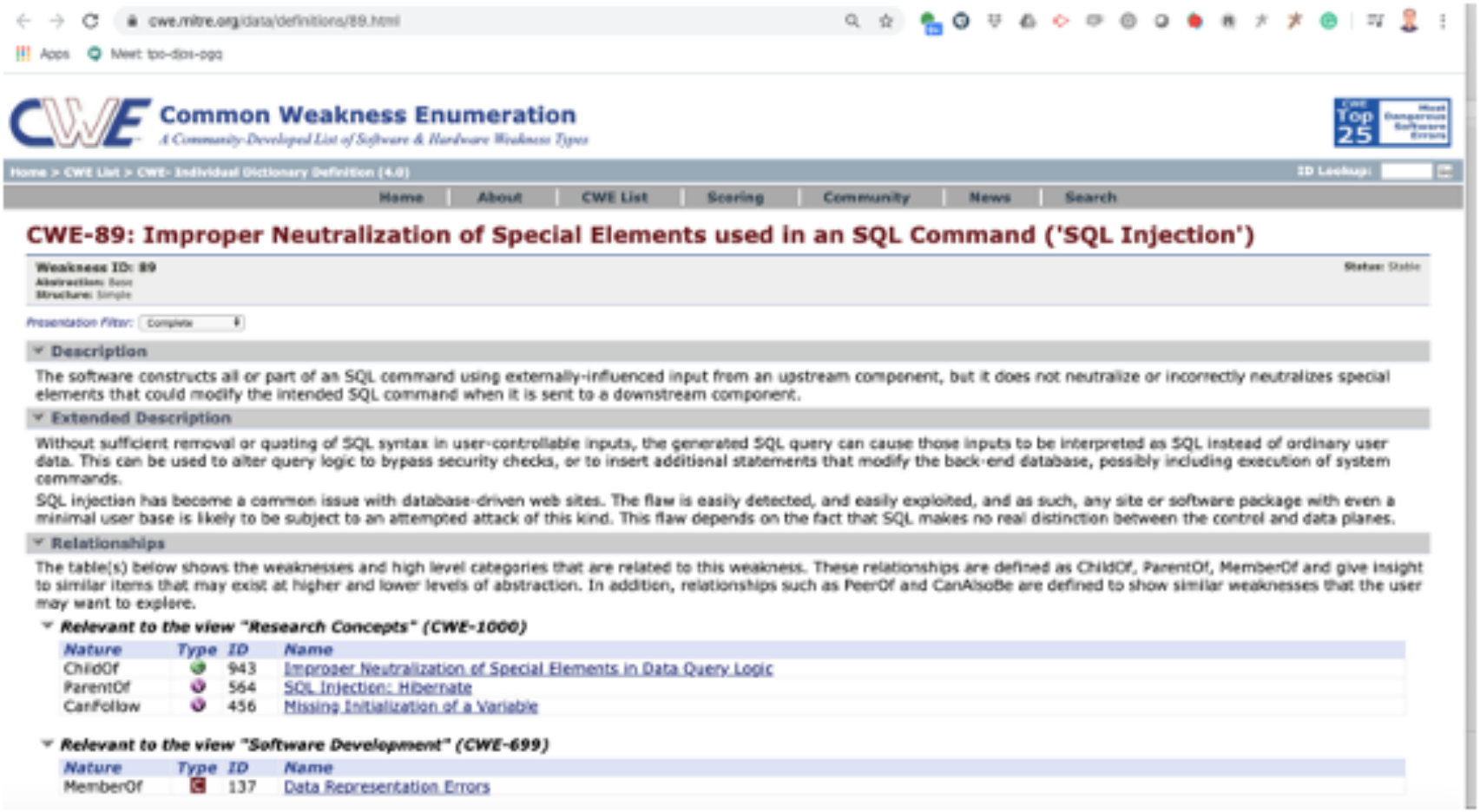
onde – ‘é o operador de comenário (ignora o que vem depois)

Exemplos

- CVE-2020-8841
  - An issue was discovered in TestLink 1.9.19. The relation_type parameter of the lib/requirements/reqSearch.php endpoint is vulnerable to authenticated SQL Injection.

- CVE-2020-3154
  - A vulnerability in the web UI of Cisco Cloud Web Security (CWS) could allow an authenticated, remote attacker to execute arbitrary SQL queries. The vulnerability exists because the web-based management interface improperly validates SQL values. An authenticated attacker could exploit this vulnerability sending malicious requests to the affected device. An exploit could allow the attacker to modify values on or return values from the underlying database.

Definição de Injeção de SQL



Falha: Injeção de comandos

› XXX

Falha: Injeção de comandos

- Em 1994, você poderia obter acesso de root num shel de IRIX enviando o seguinte comando para uma impressora:

```
FRED; xterm&
```

- Code:

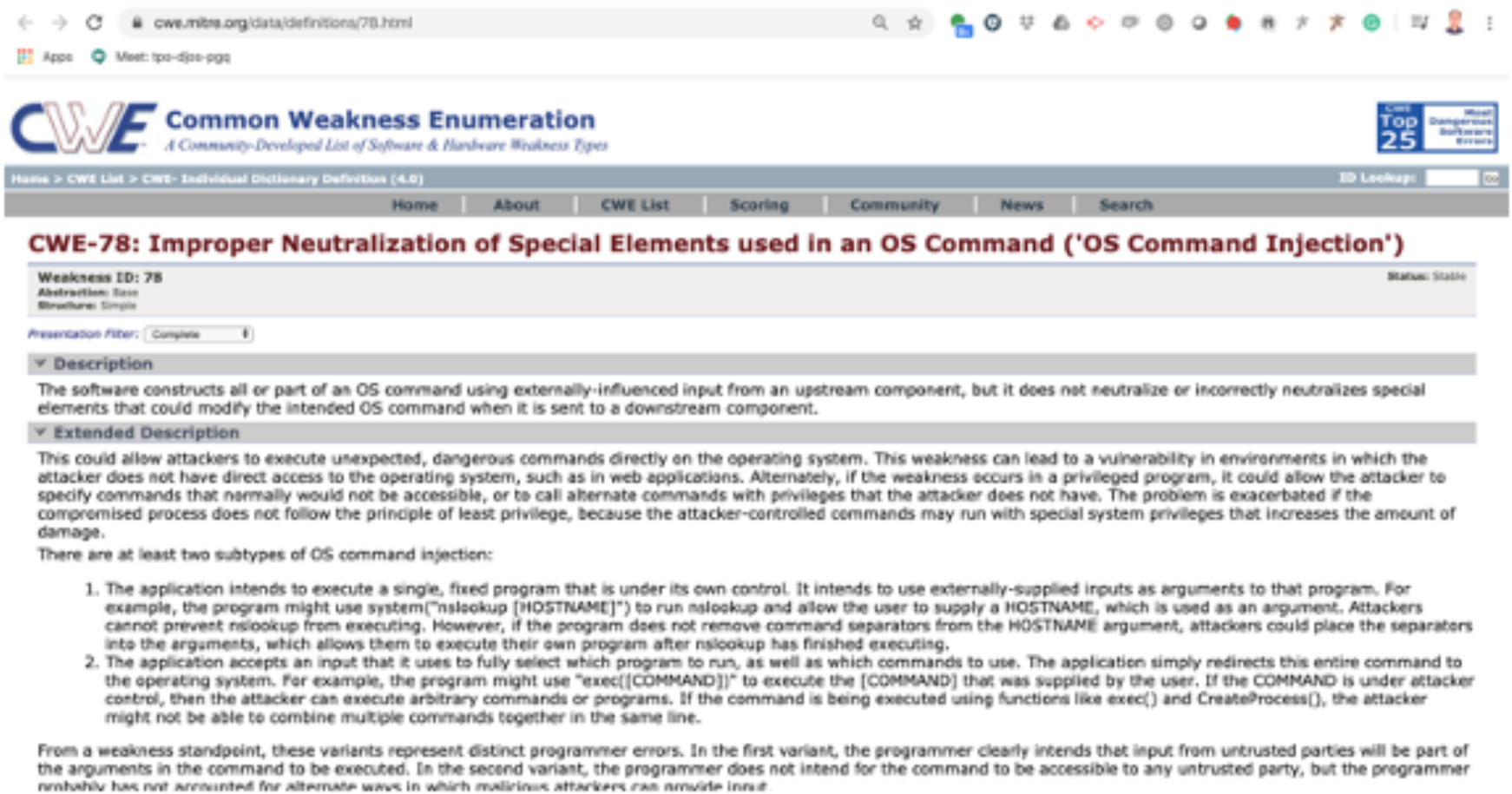
```
char buf[1024];
snprintf(buf, “system lpr –P %s”, user_input, sizeof(buf) -1);
system(buf);
```

Exemplos

- CVE-2017-14127
  - Command Injection in the Ping Module in the Web Interface on Technicolor TD5336 OI_Fw_v7 devices allows remote attackers to execute arbitrary OS commands as root via shell metacharacters in the pingAddr parameter to mnt_ping.cgi.

- CVE-2019-11829
  - OS command injection vulnerability in drivers_syno_import_user.php in Synology Calendar before 2.3.1-0617 allows remote attackers to execute arbitrary commands via the crafted 'X-Real-IP' header.

Definição de Injeção de Comando



Falha: Format String

- Um problema típico de C/C++
  - Mas o "conceito" se repete em outros ambientes
- String de entrada avaliada como código
  - Espécie de "injeção"de código"
- Exemplo

```
void main(int argc, char * argv[])
{
        printf(argv[1]);
}
```

- Simples – o que pode dar errado?

Falha: Format String

› E se o programa é executado como :

<executável> “%x %x”

› A saída é algo como...

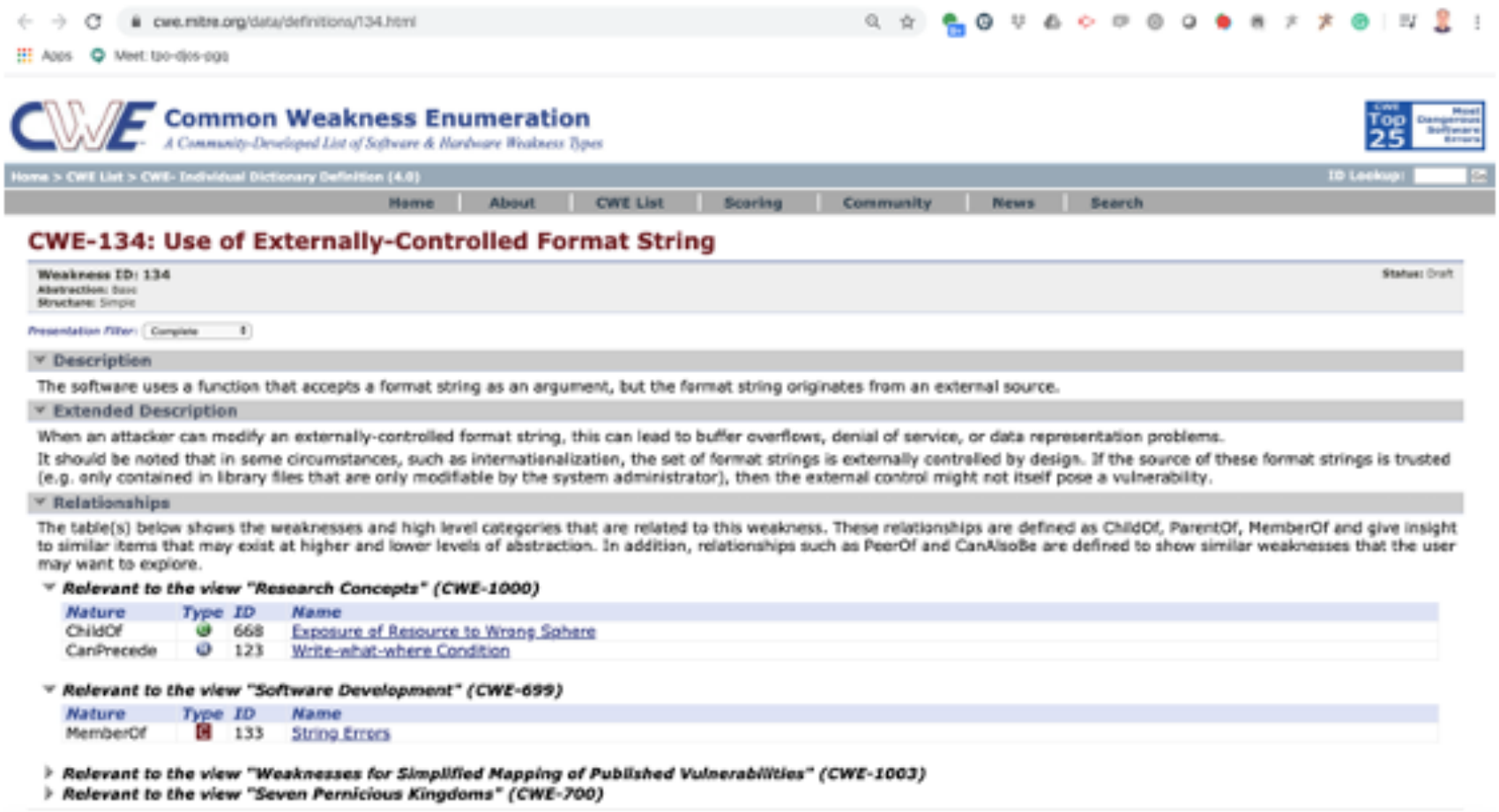
12FFC0 4011E5

O %x lê a pilha 4 bytes por vez e imprime
Vaza informação importante para um atacante

Exemplos

- CVE-2019-12297
  - An issue was discovered in scopd on Motorola routers CX2 1.01 and M2 1.01. There is a Use of an Externally Controlled Format String, reachable via TCP port 8010 or UDP port 8080.
- CVE-2019-13318
  - This vulnerability allows remote attackers to disclose sensitive information on affected installations of Foxit Reader 9.5.0.20723. User interaction is required to exploit this vulnerability in that the target must visit a malicious page or open a malicious file. The specific flaw exists within the processing of the util.printf Javascript method. The application processes the %p parameter in the format string, allowing heap addresses to be returned to the script. An attacker can leverage this in conjunction with other vulnerabilities to execute code in the context of the current process.

Definição de Format String



Falha: Estouro de buffer

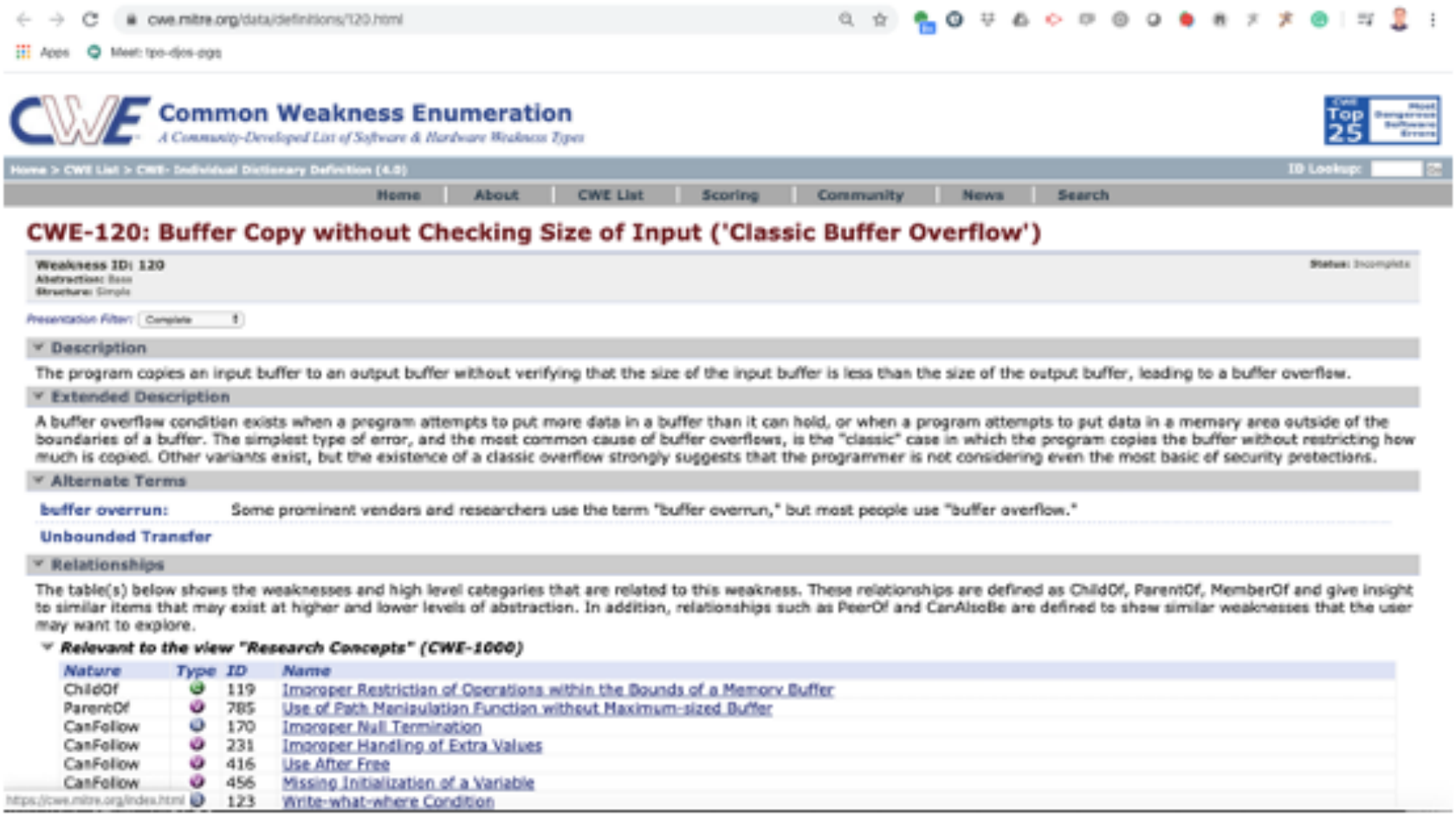
- Certamente, você já escutou muitas vezes
- Acontece quando um programa permite ao usuário escrever dados além dos limites de memória alocada
  - O programa pode travar – ou um atacante pode tomar controle da aplicação
  - Ainda é possível em linguagens como C# e Java (pois usam bibliotecas escritas em C/C++), mas é improvável.

Exemplos de Estouro de Buffer

- CVE-2019-14698
  - An issue was discovered on MicroDigital N-series cameras with firmware through 6400.0.8.5. In a CGI program running under the HTTPD web server, a buffer overflow in the param parameter leads to remote code execution in the context of the nobody account.

- CVE-2019-13537
  - The IEC870IP driver for AVEVA's Vijeo Citect and Citect SCADA and Schneider Electric's Power SCADA Operation has a buffer overflow vulnerability that could result in a server-side crash.

Definição de Estouro de Memória (Buff.Ov.)

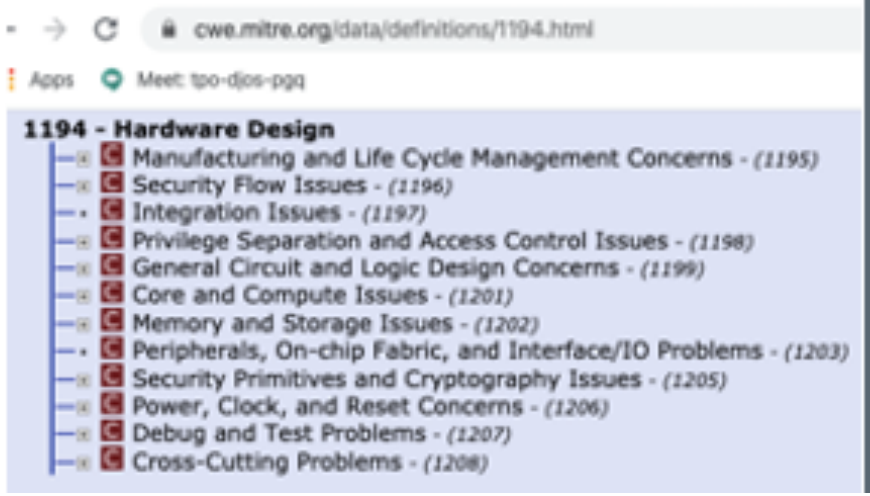


Vários outros tipos de vulnerabilidades...

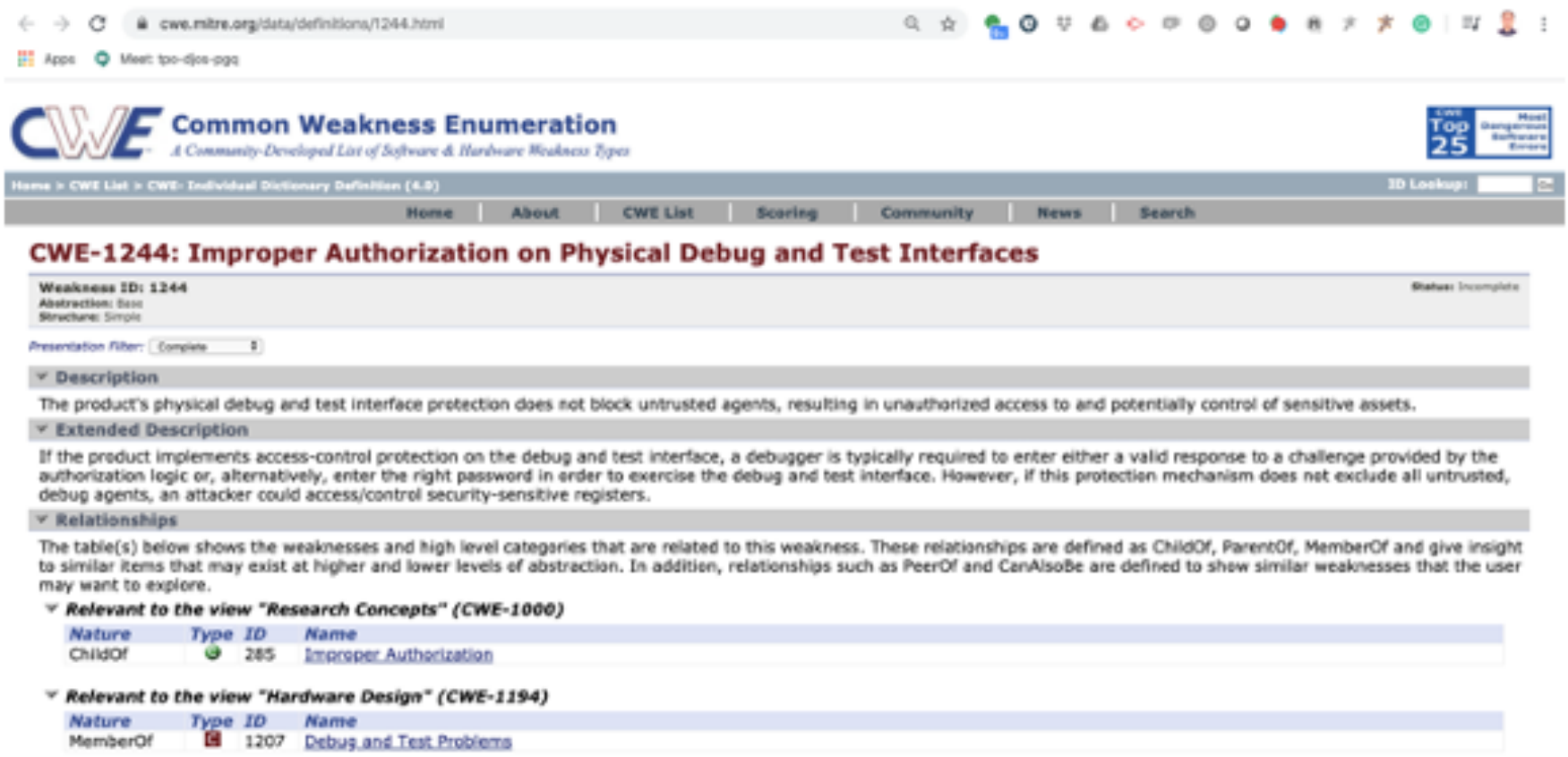
- Muitas vezes, associados àqueles ”warnings” que o compilador nos passa
- Gerenciamento inadequado de regiões de memória
- Controle inadequado de limites de variáveis
- Tratamento inadequado de entradas de usuário
- Gerenciamento inadequado de contas e credenciais
- vários outros...

Acontecem em todo tipo de software

- Software “de prateleira”: sistemas operacionais, bancos de dados, servidores, aplicativos,...
- Sistemas de informação customizados
- Sistemas de suporte a infraestrutura críticas
- Sistemas embarcados
- ...
- ... e também em hardware



Exemplo de vulnerabilidade de hardware



cwe.mitre.org/data/definitions/1244.html

Apps Meet: tpo-djos-pgq

Common Weakness Enumeration
A Community-Developed List of Software & Hardware Weakness Types

CWE Top 25 Most Dangerous Software Errors

Home > CWE List > CWE- Individual Dictionary Definition (4.0) ID Lookup:

Home | About | CWE List | Scoring | Community | News | Search

CWE-1244: Improper Authorization on Physical Debug and Test Interfaces

Weakness ID: 1244
Abstraction: Base
Structure: Simple
Status: Incomplete

Presentation Filter: Complete

Description

The product's physical debug and test interface protection does not block untrusted agents, resulting in unauthorized access to and potentially control of sensitive assets.

Extended Description

If the product implements access-control protection on the debug and test interface, a debugger is typically required to enter either a valid response to a challenge provided by the authorization logic or, alternatively, enter the right password in order to exercise the debug and test interface. However, if this protection mechanism does not exclude all untrusted, debug agents, an attacker could access/control security-sensitive registers.

Relationships

The table(s) below shows the weaknesses and high level categories that are related to this weakness. These relationships are defined as ChildOf, ParentOf, MemberOf and give insight to similar items that may exist at higher and lower levels of abstraction. In addition, relationships such as PeerOf and CanAlsoBe are defined to show similar weaknesses that the user may want to explore.

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf		285	Improper Authorization

Relevant to the view "Hardware Design" (CWE-1194)

Nature	Type	ID	Name
MemberOf		1207	Debug and Test Problems

Vulnerabilidades de Software: definição

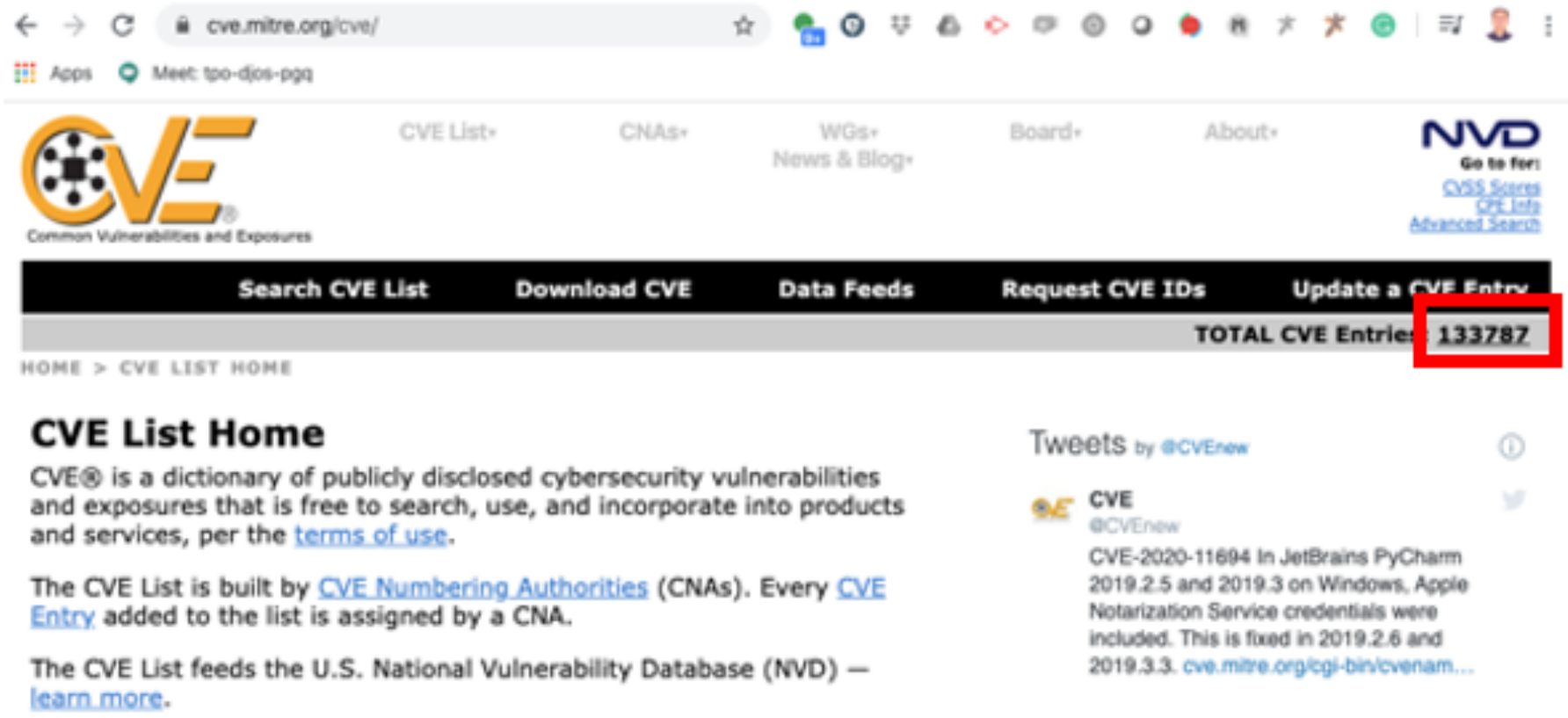
- Falhas que concepção, projeto ou implementação que podem (potencialmente) ser exploradas por atacantes
- Permitem que seu software passe por condições não previstas – e que podem ser exploradas por um atacante para comprometer a segurança
- Podem ter diferentes tipos de impacto e diferentes graus de dificuldade de descoberta e exploração
  - Impacto e explorabilidade são critérios típicos de classificação
- Veremos alguns exemplos nos próximos slides

Bases de Riscos e Vulnerabilidades

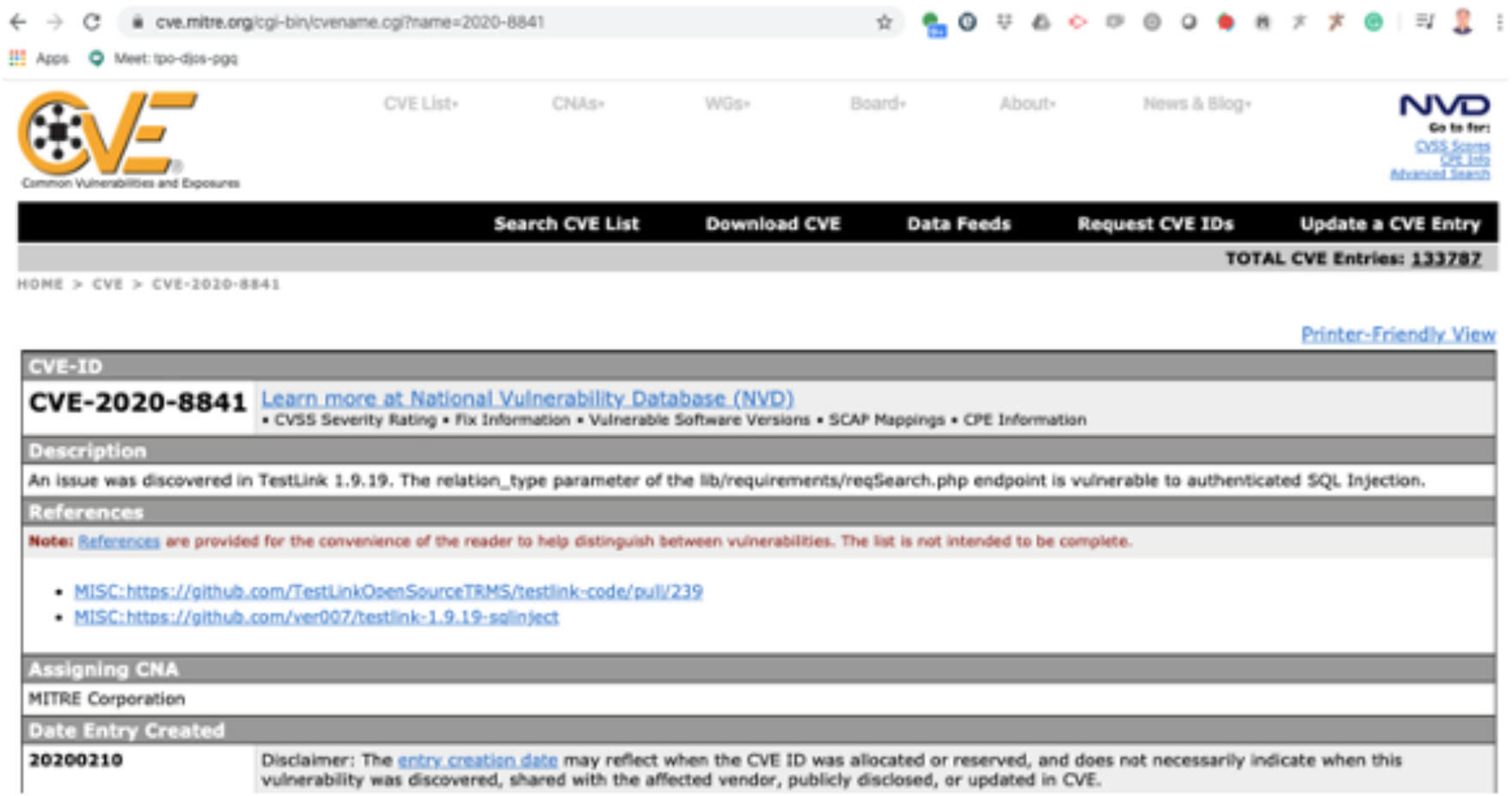
Bases de riscos e vulnerabilidades

- Grandes organizações dedicam-se a catalogar e organizar riscos vulnerabilidades de software
- Exemplos
  - CVE
  - NVDB
  - CWE
- Listas de riscos mais relevantes
  - SANS/CWE Top 25
  - OWASP Top 10

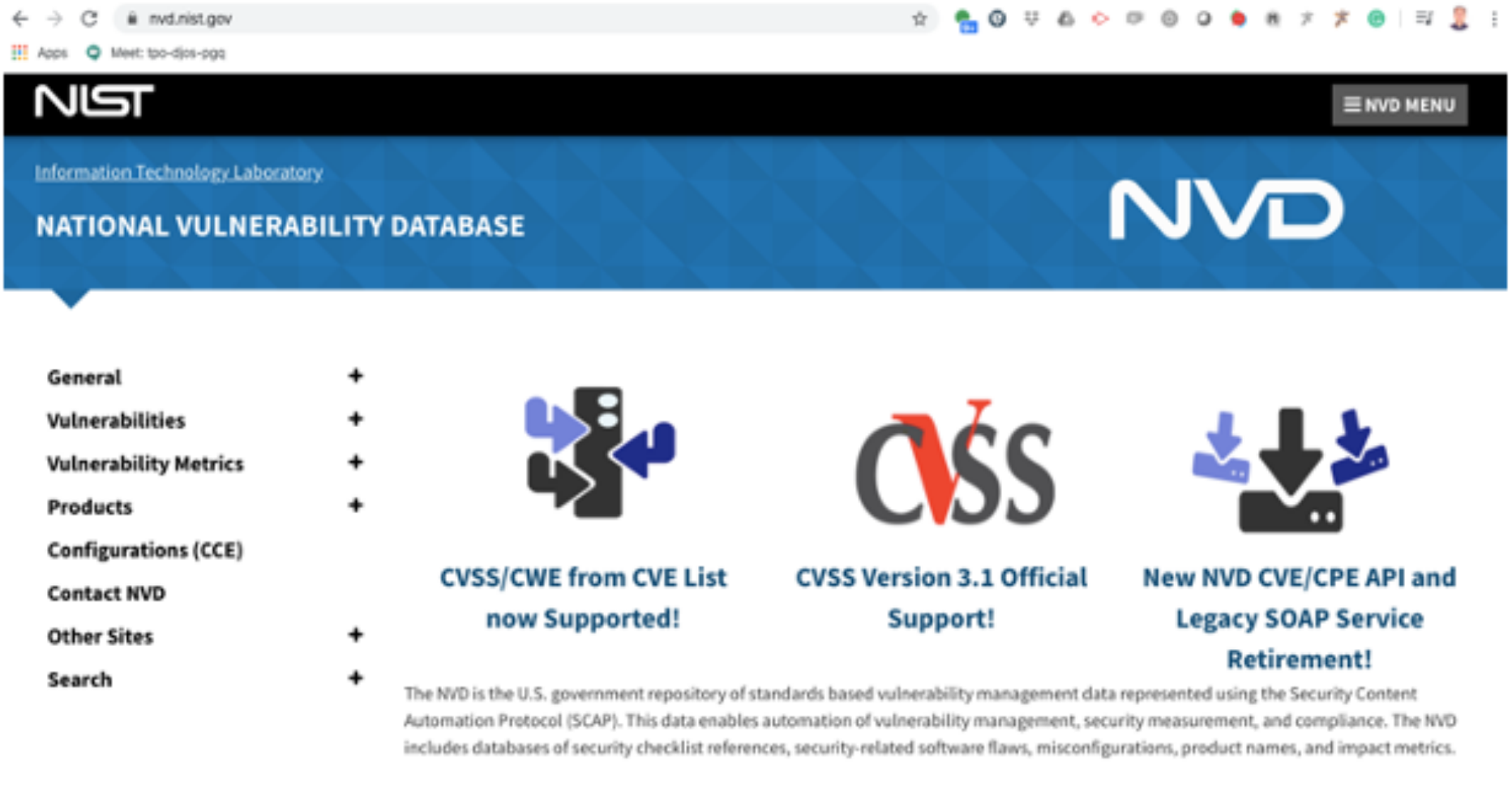
CVE: lista de vulnerabilidades “concretas”



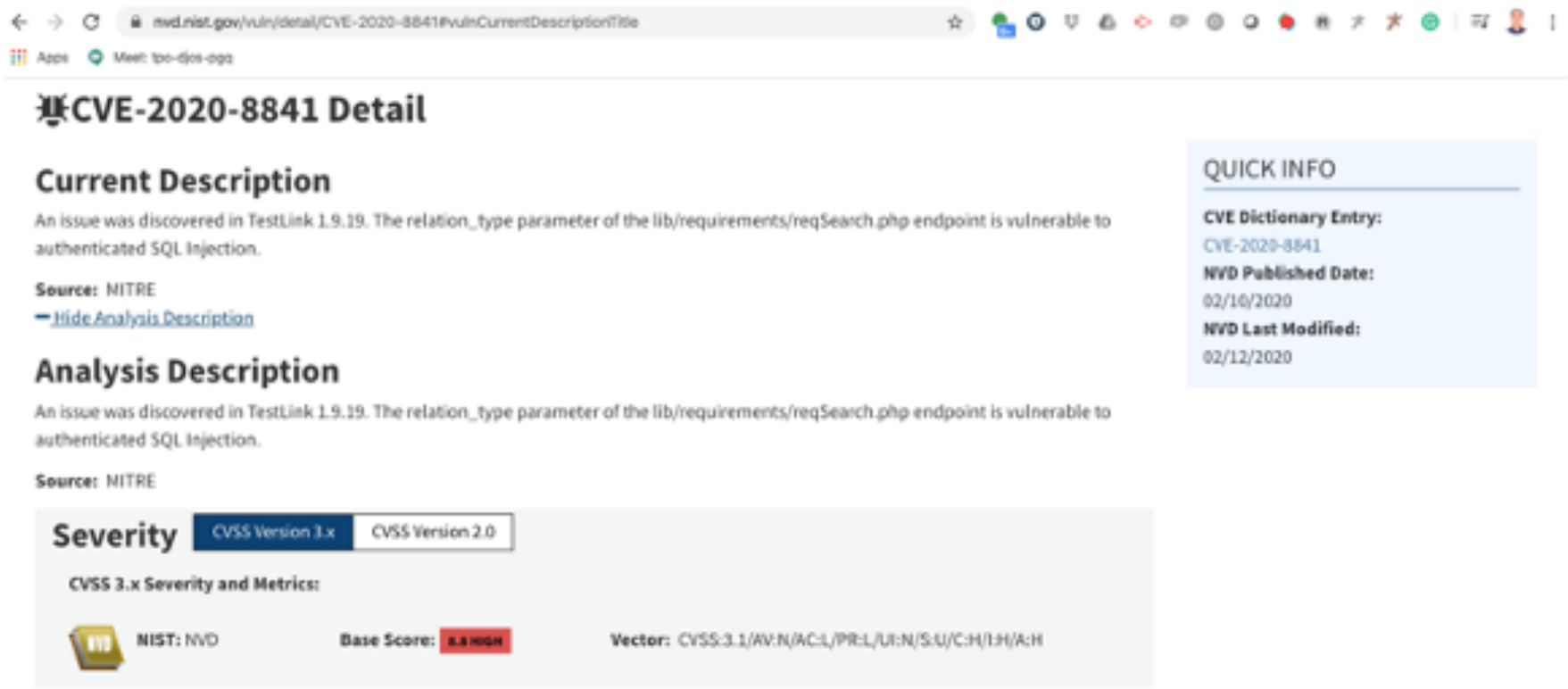
Exemplo: SQLi CVE-2020-8841



NVD: Base nacional americana de vulns



Exemplo: SQLi CVE-2020-8841



Common Vulnerabilities and Exposures (CVE)

- Fornece um método de referência para vulnerabilidades e exposições de segurança da informação conhecidas publicamente.
- A National Cybersecurity FFRDC, operada pela Mitre Corporation, mantém o sistema
  - financiamento da Divisão Nacional de Segurança Cibernética do Departamento de Segurança Interna dos Estados Unidos.
- O Security Content Automation Protocol usa o CVE, e os CVE IDs estão listados no sistema do MITRE, bem como no National National Vulnerability Database.

Exemplo de CVE

CVE-ID	
CVE-2019-0801	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

A remote code execution vulnerability exists when Microsoft Office fails to properly handle certain files.To exploit the vulnerability, an attacker would have to convince a user to open a specially crafted URL file that points to an Excel or PowerPoint file that was also downloaded.The update addresses the vulnerability by correcting how Office handles these files., aka 'Office Remote Code Execution Vulnerability'.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- MISC:https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0801
- MISC:https://www.zerodayinitiative.com/advisories/ZDI-19-358/

Assigning CNA

Microsoft Corporation

Exemplo de CVE

Observe que o CVE é bastante específico (falha em particular do Excel ou PowerPoint)

CVE-ID

CVE-2019-0801 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

A remote code execution vulnerability exists when Microsoft Office fails to properly handle certain files.To exploit the vulnerability, an attacker would have to convince a user to open a specially crafted URL file that points to an Excel or PowerPoint file that was also downloaded.The update addresses the vulnerability by correcting how Office handles these files., aka 'Office Remote Code Execution Vulnerability'.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- MISC:https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0801
- MISC:https://www.zerodayinitiative.com/advisories/ZDI-19-358/

Assigning CNA

Microsoft Corporation

Exemplo de CVE

Exploitability Assessment

The following table provides an exploitability assessment for this vulnerability at the time of original publication.

Publicly Disclosed	Exploited	Latest Software Release	Older Software Release	Denial of Service
No	No	1 - Exploitation More Likely	1 - Exploitation More Likely	Not Applicable

Security Updates

To determine the support life cycle for your software version or edition, see the Microsoft Support Lifecycle.

Product ▲	Platform	Article	Download	Impact	Severity	Supersedence
Microsoft Office 2010 Service Pack 2 (32-bit editions)		4462223	Security Update	Remote Code Execution	Important	4462174
Microsoft Office 2010 Service Pack 2 (64-bit editions)		4462223	Security Update	Remote Code Execution	Important	4462174
Microsoft Office 2013 RT Service Pack 1		4464504	Security Update	Remote Code Execution	Important	4462138
Microsoft Office 2013 Service Pack 1 (32-bit editions)		4464504	Security Update	Remote Code Execution	Important	4462138
Microsoft Office 2013 Service Pack 1 (64-bit editions)		4464504	Security Update	Remote Code Execution	Important	4462138
Microsoft Office 2016 (32-bit edition)		4462242	Security Update	Remote Code Execution	Important	4462146
Microsoft Office 2016 (64-bit edition)		4462242	Security Update	Remote Code Execution	Important	4462146
Microsoft Office 2019 for 32-bit editions		Click to Run	Security Update	Remote Code Execution	Important	
Microsoft Office 2019 for 64-bit editions		Click to Run	Security Update	Remote Code Execution	Important	
Office 365 ProPlus for 32-bit Systems		Click to Run	Security Update	Remote Code Execution	Important	
Office 365 ProPlus for 64-bit Systems		Click to Run	Security Update	Remote Code Execution	Important	

National Vulnerability Database (NVD)

- O NVD é o repositório do governo dos EUA de dados de gerenciamento de vulnerabilidade baseados em padrões representados usando o SCAP (Security Content Automation Protocol).
- Esses dados permitem a automação do gerenciamento de vulnerabilidades, medidas de segurança e conformidade.
- O NVD inclui bancos de dados de listas de verificação de segurança, falhas de software relacionadas à segurança, configurações incorretas, nomes de produtos e métricas de impacto.
- NVD vs CVE: O NVD é o dicionário CVE, acrescido de análises adicionais, um banco de dados e um mecanismo de busca refinado.
  - O NVD é um superconjunto do CVE. O NVD é sincronizado com o CVE, de forma que qualquer atualização do CVE apareça imediatamente no NVD.

Common Weakness Enumeration (CWE)

- Sistema de categorização para fraquezas e vulnerabilidades de software.
- Também é patrocinado pela MITRE
- CWE foca nas fraquezas ”genéricas”, CVE lista as instâncias de falha
  - Exemplo: *buffer overflow* versus ***o*** *buffer overflow da versão X.Y.Z do software ABX*

Exemplo de CWE

CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

Weakness ID: 89
Abstraction: Base
Structure: Simple

Status: Draft

Presentation Filter: High Level

Description

The software constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended SQL command when it is sent to a downstream component.

Extended Description

Without sufficient removal or quoting of SQL syntax in user-controllable inputs, the generated SQL query can cause those inputs to be interpreted as SQL instead of ordinary user data. This can be used to alter query logic to bypass security checks, or to insert additional statements that modify the back-end database, possibly including execution of system commands.

SQL injection has become a common issue with database-driven web sites. The flaw is easily detected, and easily exploited, and as such, any site or software package with even a minimal user base is likely to be subject to an attempted attack of this kind. This flaw depends on the fact that SQL makes no real distinction between the control and data planes.

Exemplo de CWE

Observe que o CWE é genérico
(falha em comando SQL)

CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

Weakness ID: 89
Abstraction: Base
Structure: Simple

Status: Draft

Presentation Filter: High Level

Description

The software constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended SQL command when it is sent to a downstream component.

Extended Description

Without sufficient removal or quoting of SQL syntax in user-controllable inputs, the generated SQL query can cause those inputs to be interpreted as SQL instead of ordinary user data. This can be used to alter query logic to bypass security checks, or to insert additional statements that modify the back-end database, possibly including execution of system commands.

SQL injection has become a common issue with database-driven web sites. The flaw is easily detected, and easily exploited, and as such, any site or software package with even a minimal user base is likely to be subject to an attempted attack of this kind. This flaw depends on the fact that SQL makes no real distinction between the control and data planes.

Exemplo de CWE

Relationships

The table(s) below shows the weaknesses and high level categories that are related to this weakness. These relationships are defined as ChildOf, ParentOf, MemberOf and give insight to similar items that may exist at higher and lower levels of abstraction. In addition, relationships such as PeerOf and CanAlsoBe are defined to show similar weaknesses that the user may want to explore.

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf		943	Improper Neutralization of Special Elements in Data Query Logic
ParentOf		564	SQL Injection: Hibernate
CanFollow		456	Missing Initialization of a Variable

Relevant to the view "Weaknesses for Simplified Mapping of Published Vulnerabilities" (CWE-1003)

Nature	Type	ID	Name
ChildOf		943	Improper Neutralization of Special Elements in Data Query Logic

Relevant to the view "Architectural Concepts" (CWE-1008)

Nature	Type	ID	Name
MemberOf	C	1019	Validate Inputs

Relevant to the view "Development Concepts" (CWE-699)

Nature	Type	ID	Name
ChildOf		943	Improper Neutralization of Special Elements in Data Query Logic
ParentOf		564	SQL Injection: Hibernate

Relevant to the view "Weaknesses in OWASP Top Ten (2013)" (CWE-928)

Nature	Type	ID	Name
ParentOf		564	SQL Injection: Hibernate

Exemplo de CWE

Modes Of Introduction

The different Modes of Introduction provide information about how and when this weakness may be introduced. The Phase identifies a point in the software life cycle at which introduction may occur, while the Note provides a typical scenario related to introduction during the given phase.

Phase	Note
Architecture and Design	This weakness typically appears in data-rich applications that save user inputs in a database.
Implementation	REALIZATION: This weakness is caused during implementation of an architectural security tactic.
Operation	

Common Consequences

The table below specifies different individual consequences associated with the weakness. The Scope identifies the application security area that is violated, while the Impact describes the negative technical impact that arises if an adversary succeeds in exploiting this weakness. The Likelihood provides information about how likely the specific consequence is expected to be seen relative to the other consequences in the list. For example, there may be high likelihood that a weakness will be exploited to achieve a certain impact, but a low likelihood that it will be exploited to achieve a different impact.

Scope	Impact	Likelihood
Confidentiality	**Technical Impact:** *Read Application Data* Since SQL databases generally hold sensitive data, loss of confidentiality is a frequent problem with SQL injection vulnerabilities.	
Access Control	**Technical Impact:** *Bypass Protection Mechanism* If poor SQL commands are used to check user names and passwords, it may be possible to connect to a system as another user with no previous knowledge of the password.	
Access Control	**Technical Impact:** *Bypass Protection Mechanism* If authorization information is held in a SQL database, it may be possible to change this information through the successful exploitation of a SQL injection vulnerability.	
Integrity	**Technical Impact:** *Modify Application Data* Just as it may be possible to read sensitive information, it is also possible to make changes or even delete this information with a SQL injection attack.	

Likelihood Of Exploit

High

Exemplo de CWE

Likelihood Of Exploit

High

Memberships

This MemberOf Relationships table shows additional CWE Categories and Views that reference this weakness as a member. This information is often useful in understanding where a weakness fits within the context of external information sources.

Nature	Type	ID	Name
MemberOf	V	635	Weaknesses Originally Used by NVD from 2008 to 2016
MemberOf	C	713	OWASP Top Ten 2007 Category A2 - Injection Flaws
MemberOf	C	722	OWASP Top Ten 2004 Category A1 - Unvalidated Input
MemberOf	C	727	OWASP Top Ten 2004 Category A6 - Injection Flaws
MemberOf	C	751	2009 Top 25 - Insecure Interaction Between Components
MemberOf	C	801	2010 Top 25 - Insecure Interaction Between Components
MemberOf	C	810	OWASP Top Ten 2010 Category A1 - Injection
MemberOf	C	864	2011 Top 25 - Insecure Interaction Between Components
MemberOf	V	884	CWE Cross-section
MemberOf	C	929	OWASP Top Ten 2013 Category A1 - Injection
MemberOf	C	990	SFP Secondary Cluster: Tainted Input to Command
MemberOf	C	1005	7PK - Input Validation and Representation
MemberOf	C	1027	OWASP Top Ten 2017 Category A1 - Injection
MemberOf	C	1131	CISQ Quality Measures - Security

Notes

Relationship

SQL injection can be resultant from special character mismanagement, MAID, or blacklist/whitelist problems. It can be primary to authentication errors.

More information is available — Please select a different filter.

Exemplo (do CWE)

The following code dynamically constructs and executes a SQL query that searches for items matching a specified name. The query restricts the items displayed to those where owner matches the user name of the currently-authenticated user.

Example Language: **C#** *(bad code)*

```
...
string userName = ctx.getAuthenticatedUserName();
string query = "SELECT * FROM items WHERE owner = '" + userName + "' AND itemname = '" + ItemName.Text + "'";
sda = new SqlDataAdapter(query, conn);
DataTable dt = new DataTable();
sda.Fill(dt);
...
```

The query that this code intends to execute follows:

(informative)

```
SELECT * FROM items WHERE owner = <userName> AND itemname = <itemName>;
```

However, because the query is constructed dynamically by concatenating a constant base query string and a user input string, the query only behaves correctly if itemName does not contain a single-quote character. If an attacker with the user name wiley enters the string:

(attack code)

```
name' OR 'a'='a
```

for itemName, then the query becomes the following:

(attack code)

```
SELECT * FROM items WHERE owner = 'wiley' AND itemname = 'name' OR 'a'='a';
```

The addition of the:

(attack code)

```
OR 'a'='a
```

condition causes the WHERE clause to always evaluate to true, so the query becomes logically equivalent to the much simpler query:

(attack code)

```
SELECT * FROM items;
```

SANS/CWE Top 25

- CWE/SANS Top 25 Most Dangerous Software Errors (2011)
  - Erros de software mais comuns/críticos
- Geralmente são fáceis de encontrar e fáceis de explorar.
- Frequentemente permitem que invasores assumam completamente o software, roubem dados ou impeçam o software de funcionar.
- A lista é uma ferramenta de educação e conscientização
  - Ajuda programadores a evitar as vulnerabilidades, identificando e evitando erros comuns
- A lista é resultado da colaboração entre o SANS Institute, o MITRE e especialistas nos EUA e Europa.

SANS/CWE Top 25

- O Common Weakness Enumeration (CWE) é uma lista formal de tipos de fraquezas de software e é patrocinado pela Divisão Nacional de Segurança Cibernética do Departamento de Segurança Interna dos EUA.
  - O Instituto SANS (SysAdmin, Audit, Network, Security) foi estabelecido em 1989 como uma organização de pesquisa e educação.
- Fonte: http://www.sans.org/top25errors/

Reconhecimento

- National Security Agency's Information Assurance Directorate
  - "The publication of a list of programming errors that enable cyber espionage and cyber crime is an important first step in managing the vulnerability of our networks and technology. There needs to be a move away from reacting to thousands of individual vulnerabilities, and to focus instead on a relatively small number of software flaws that allow vulnerabilities to occur, each with a general root cause. Such a list allows the targeting of improvements in software development practices, tools, and requirements to manage these problems earlier in the life cycle, where they can be solved on a large scale and cost-effectively.“
  - Tony Sager, National Security Agency's Information Assurance Directorate

Reconhecimento

- Microsoft:
  - "The 2009 CWE/SANS Top 25 Programming Errors project is a great resource to help software developers identify which security vulnerabilities are the most important to understand, prevent and fix.“
  - Michael Howard, Principal Security Program Manager, Security Development Lifecycle Team, Microsoft Corp.
- Symantec:
  - "The 2009 CWE/SANS Top 25 Programming Errors reflects the kinds of issues we've seen in application software and helps provide us with actionable direction to continuously improve the security of our software."
  - - Wesley H. Higaki, Director, Software Assurance, Office of the CTO, Symantec Corporation

A lista Top 25

Rank	Score	ID	Name
[1]	93.8	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2]	83.3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3]	79.0	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
[4]	77.7	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5]	76.9	CWE-306	Missing Authentication for Critical Function
[6]	76.8	CWE-862	Missing Authorization
[7]	75.0	CWE-798	Use of Hard-coded Credentials
[8]	75.0	CWE-311	Missing Encryption of Sensitive Data
[9]	74.0	CWE-434	Unrestricted Upload of File with Dangerous Type
[10]	73.8	CWE-807	Reliance on Untrusted Inputs in a Security Decision
[11]	73.1	CWE-250	Execution with Unnecessary Privileges
[12]	70.1	CWE-352	Cross-Site Request Forgery (CSRF)
[13]	69.3	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
[14]	68.5	CWE-494	Download of Code Without Integrity Check
[15]	67.8	CWE-863	Incorrect Authorization
[16]	66.0	CWE-829	Inclusion of Functionality from Untrusted Control Sphere
[17]	65.5	CWE-732	Incorrect Permission Assignment for Critical Resource
[18]	64.6	CWE-676	Use of Potentially Dangerous Function
[19]	64.1	CWE-327	Use of a Broken or Risky Cryptographic Algorithm
[20]	62.4	CWE-131	Incorrect Calculation of Buffer Size
[21]	61.5	CWE-307	Improper Restriction of Excessive Authentication Attempts
[22]	61.1	CWE-601	URL Redirection to Untrusted Site ('Open Redirect')
[23]	61.0	CWE-134	Uncontrolled Format String
[24]	60.3	CWE-190	Integer Overflow or Wraparound
[25]	59.9	CWE-759	Use of a One-Way Hash without a Salt

Organização em Categorias:

- **Insecure Interaction Between Components**
  - Essas fraquezas estão relacionadas a formas inseguras nas quais os dados são enviados e recebidos entre componentes, módulos, programas, processos, threads ou sistemas separados.
- **Risky Resource Management**
  - Os pontos fracos dessa categoria estão relacionados às maneiras pelas quais o software gerencia (inadequadamente) a criação, o uso, a transferência ou a destruição de recursos importantes do sistema.
- **Porous Defenses**
  - As fraquezas nessa categoria estão relacionadas a técnicas defensivas que são frequentemente mal utilizadas, abusadas ou simplesmente ignoradas.

Insecure Interaction Among Components

- CWE-20: Improper Input Validation
- CWE-116: Improper Encoding or Escaping of Output
- CWE-89: Failure to Preserve SQL Query Structure (aka 'SQL Injection')

Insecure Interaction Among Components

- CWE-79: Failure to Preserve Web Page Structure (aka 'Cross-site Scripting')
- CWE-78: Failure to Preserve OS Command Structure (aka 'OS Command Injection')
- CWE-319: Cleartext Transmission of Sensitive Information

Insecure Interaction Among Components

- CWE-352: Cross-Site Request Forgery (CSRF)
- CWE-362: Race Condition
- CWE-209: Error Message Information Leak

Risky Resource Management

- CWE-119: Failure to Constrain Operations within the Bounds of a Memory Buffer
- CWE-642: External Control of Critical State Data
- CWE-73: External Control of File Name or Path

Risky Resource Management

- CWE-426: Untrusted Search Path
- CWE-94: Failure to Control Generation of Code (aka 'Code Injection')
- CWE-494: Download of Code Without Integrity Check

Risky Resource Management

- CWE-404: Improper Resource Shutdown or Release
- CWE-665: Improper Initialization
- CWE-682: Incorrect Calculation

Porous Defenses

- CWE-285: Improper Access Control (Authorization)
- CWE-327: Use of a Broken or Risky Cryptographic Algorithm
- CWE-259: Hard-Coded Password

Porous Defenses

- CWE-732: Insecure Permission Assignment for Critical Resource
- CWE-330: Use of Insufficiently Random Values

Porous Defenses

- CWE-250: Execution with Unnecessary Privileges
- CWE-602: Client-Side Enforcement of Server-Side Security

OWASP Top 10

- O Open Web Application Security Project (OWASP) é uma organização internacional dedicada a melhorar a segurança de aplicações web.
- O projeto OWASP Top 10 publica uma lista dos considerados 10 principais riscos de segurança de aplicações da web em todo o mundo.
- A lista descreve cada vulnerabilidade, fornece exemplos e oferece sugestões sobre como evitá-la.
- A versão mais recente da lista dos 10 melhores, publicada oficialmente em 2017.
- O OWASP priorizou o top 10 de acordo com sua prevalência e sua relativa capacidade de exploração, detectabilidade e impacto.

A Lista OWASP Top 10

OWASP Top 10 Application Security Risks - 2017

A1:2017-Injection
Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

A2:2017-Broken Authentication
Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.

A3:2017-Sensitive Data Exposure
Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

A4:2017-XML External Entities (XXE)
Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

A5:2017-Broken Access Control
Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

A6:2017-Security Misconfiguration
Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.

A7:2017-Cross-Site Scripting (XSS)
XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

A8:2017-Insecure Deserialization
Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

A9:2017-Using Components with Known Vulnerabilities
Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

A10:2017-Insufficient Logging&Monitoring
Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

A Lista OWASP Top 10

- A1:2017-Injection
- A2:2017-Broken Authentication
- A3:2017-Sensitive Data Exposure
- A4:2017-XML External Entities (XXE)
- A5:2017-Broken Access Control
- A6:2017-Security Misconfiguration
- A7:2017-Cross-Site Scripting (XSS)
- A8:2017-Insecure Deserialization
- A9:2017-Using Components with Known Vulnerabilities
- A10:2017-Insufficient Logging&Monitoring

OWASP Top 10 versus CWE Top 25

- OWASP originalmente focado em aplicações web
  - Atualmente, inclui vulnerabilidades mais "genéricas"
- OWASP é mais geral do que o CWE
  - Exemplo: vulnerabilidade OWASP A1 (Injection) está associada a seis vulnerabilidades CWE
- CWE é uma lista de vulnerabilidades enquanto OWASP é uma lista de riscos

OWASP A1: Injection

- CWE-78: Improper Neutralization of Special Elements Used in an OS Command ('OS Command Injection')
- CWE-89: SQL Injection
- CWE-94: Code Injection
- CWE-434: Unrestricted Upload of File with Dangerous Type
- CWE-494: Download of Code Without Integrity Check
- CWE-829: Inclusion of Functionality from Untrusted Control Sphere

OWASP A2: Broken Authentication

- CWE-306: Missing Authentication for Critical Function
- CWE-307: Improper Restriction of Excessive Authentication Attempts
- CWE-798: Use of Hard-coded Credentials
- CWE-807: Reliance on Untrusted Inputs in a Security Decision
- CWE-862: Missing Authorization
- CWE-863: Incorrect Authorization

OWASP A3: Sensitive Data Exposure

- CWE-311: Missing Encryption of Sensitive Data
- CWE-319: Cleartext Transmission of Sensitive Information

OWASP A4: XML External Entities

- Nenhum

OWASP A5: Broken Access Control

- CWE-73: External Control of File Name or Path
- CWE-285: Improper Authorization

OWASP A6: Security Misconfiguration

- CWE-250: Execution with Unnecessary Privileges
- CWE-676: Use of Potentially Dangerous Function
- CWE-732: Incorrect Permission Assignment for Critical Resource

OWASP A7: Cross-Site Scripting (XSS)

- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-Site Scripting')

OWASP A8: Insecure Deserialization

› CWE-134: Use of Externally-Controlled Format String

OWASP A9: Using Components with Known Vulnerabilities

- CWE-190: Integer Overflow or Wraparound
- CWE-327: Use of a Broken or Risky Cryptographic Algorithm
- CWE-759: Use of a One-way Hash Without a Salt

OWASP A10: Insufficient Logging and Monitoring

- Nenhum

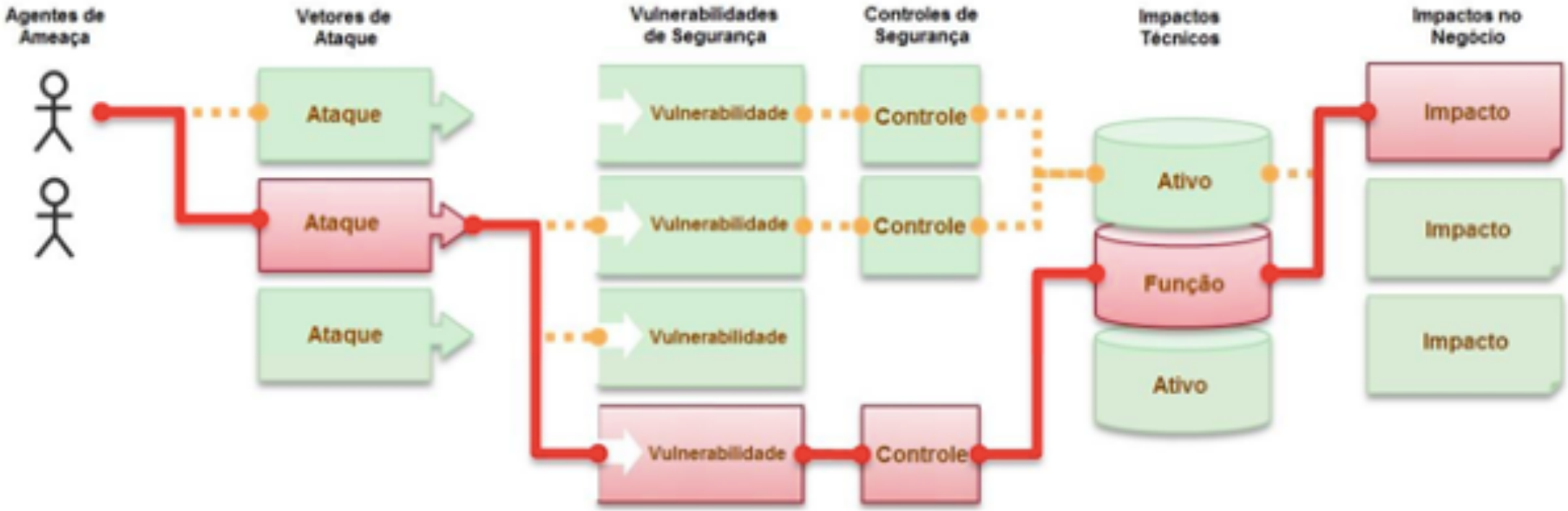
As vulnerabilidades da OWASP TOP 10

- Desenvolvido para descrever as 10 vulnerabilidades mais comuns em Aplicações Web e seus mecanismos de prevenção
  - Define de forma clara quais vulnerabilidades ocorrem com maior frequência nas Aplicações Web.
  - Apresenta e direciona propostas para medidas corretivas e preventivas, com suporte constante dos projetos disponibilizados pelo OWASP.
  - O objetivo primário é educar programadores, designers, arquitetos e organizações acerca das consequências das 10 vulnerabilidades mais comuns em Aplicações Web.

As vulnerabilidades da OWASP TOP 10

- O OWASP TOP 10 é uma lista de vulnerabilidades mais comuns e para garantir a eficácia no uso de suas recomendações, deve ser entendido exatamente dentro deste conceito
  - Não é e nem tem a intenção de ser uma lista exaustiva das vulnerabilidades que podem ocorrer em uma Aplicação Web
  - Deve ser utilizada como uma referência básica para adequação do nível de segurança em Aplicações Web e posterior desenvolvimento de um programa de proteção amplo

OWASP: Conceito de Risco em Aplicações



As Vulnerabilidades da OWASP TOP 10

- A1 - Injeção
- A2 - Quebra de Autenticação
- A3 – Exposição de Dados Sensíveis
- A4 – Entidades Externas de XML
- A5 – Quebra de Controle de Acesso
- A6 – Configuração de Segurança Incorretas
- A7 – Cross-Site Scripting (XSS)
- A8 – Desserialização Insegura
- A9 - Utilização de Componentes Vulneráveis
- A10 – Registro e Monitoramento Insuficientes

A OWASP TOP 10 – 2013 *vs* 2017

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	∪	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	∪	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

Resumo

- O OWASP TOP 10 é uma lista de vulnerabilidades mais comuns e para garantir a eficácia no uso de suas recomendações, deve ser entendido exatamente dentro deste conceito
  - Não é e nem tem a intenção de ser uma lista exaustiva das vulnerabilidades que podem ocorrer em uma Aplicação Web
  - Deve ser utilizada como uma referência básica para adequação do nível de segurança em Aplicações Web e posterior desenvolvimento de um programa de proteção amplo