




Corridas e Validação de Entradas

Parte II: Vulnerabilidades



Segurança de Software

2019
Nuno Santos

Aula passada

- Estudámos uma classe de bugs muito comum e muito séria chamada **buffer overflow**

2 SS - Nuno Santos 2019



Onde estamos

- ▶ **Parte I: Enquadramento e protecção (2 aulas)**
 - ▶ Conceitos de segurança de software, mecanismos básicos de segurança
- ▶ **Parte II: Vulnerabilidades (3 aulas)**
 - ▶ Buffer overflows, corridas e validação de entradas, vulnerabilidades na web e em bases de dados
- ▶ **Parte III: Técnicas de protecção (4 aulas)**
 - ▶ Auditoria e teste de software, análise estática de código, protecção dinâmica, validação e codificação
- ▶ **Parte IV: Tópicos avançados (1 aula)**
 - ▶ Trusted computing

▶ 3

SS - Nuno Santos

2019



Plano para esta aula

- ▶ Vulnerabilidades devido a corridas
- ▶ Vulnerabilidades devido a validação de entradas

▶ 4

SS - Nuno Santos

2019

Vulnerabilidades devido a corridas



Corridas são grande fonte de vulnerabilidades

Search Results (Kerine Search)

There are **162** matching records.
Displaying matches **1** through **20**.

Search Parameters:

- **Keyword (text search):** race
- **Search Type:** Search Last 3 Years
- **Contains Software Flaws (CVE)**

1 2 3 4 5 6 7 8 9 > >>

CVE-2015-3247

Summary: Race condition in the worker_update_monitors_config function in SPICE 0.12.4 allows a remote authenticated guest user to cause a denial of service (heap-based memory corruption and QEMU-KVM crash) or possibly execute arbitrary code on the host via unspecified vectors.

Published: 9/8/2015 11:59:02 AM

CVSS Severity: 6.9 MEDIUM

CVE-2015-5189

Summary: Race condition in pcsd in PCS 0.9.139 and earlier uses a global variable to validate usernames, which allows remote authenticated users to gain privileges by sending a command that is checked for security after another user is authenticated.

Published: 9/3/2015 10:59:02 AM

CVSS Severity: 4.9 MEDIUM

CVE-2015-3212

Summary: Race condition in net/socket.c in the Linux kernel before 4.1.2 allows local users to cause a denial of service (list corruption and panic) via a rapid series of system calls related to sockets, as demonstrated by setsockopt calls.

Published: 8/31/2015 6:59:06 AM

CVSS Severity: 4.9 MEDIUM



Exemplo – Falha de energia afecta 50M pessoas

A number of factors and failings came together to make the August 14th northeastern blackout the worst outage in North American history.

One of them was buried in a massive piece of software (...) running on an energy management computer in Ohio. (...)

The alarm system failed at the worst possible time: (...) at the critical moment of the blackout's earliest events. (...)

(...) the bug was unmasked as a particularly subtle incarnation of a common programming error called a “race condition”

There was a couple of processes that (...) were both able to get write access to a data structure at the same time. (...) that corruption lead to the alarm event application getting into an infinite loop

Kevin Poulsen, Tracking the blackout bug, Security Focus, April 8, 2004

http://www.thestar.com/photos/2013/08/13/blackout_2003_the_day_in_photos.html



O que são corridas (race conditions)

- ▶ Ocorre uma violação da assumção de atomicidade
 - ▶ Durante uma janela de oportunidade
 - ▶ Quando duas entidades acedem concorrentemente ao mesmo objecto
- ▶ A vulnerabilidade acontece sempre devido a um problema de concorrência / falta de sincronização adequada
 - ▶ Entre um processo malicioso e um processo alvo (vítima), ou
 - ▶ Entre vários processos alvo (vítimas)
- ▶ O atacante procura quebrar a assumção de atomicidade durante a janela de vulnerabilidade



Exemplo de um programa vulnerável

- ▶ Exemplo: serviço que produz números sequenciais únicos
 - ▶ Chamado por vários processos concorrentemente
 - ▶ Vulnerabilidade permite retornar o mesmo número múltiplas vezes

```
int count = 0; // shared
```

```
int getticket() {  
    count++;  
    return count;  
}
```

Assumpção de atomicidade?
Janela de vulnerabilidade

▶ 9

SS - Nuno Santos

2019



Algumas características das corridas

- ▶ Fontes de corridas
 - ▶ Dados partilhados: ficheiros e memória
 - ▶ Rotinas preemptivas (signal handlers)
 - ▶ Programas multi-threaded (processos que podem ter vários fluxos de execução concorrentes)
- ▶ Vulnerabilidades em sobretudo três tipos de corridas
 - ▶ TOCTOU
 - ▶ Ficheiros temporários
 - ▶ Concorrência

▶ 10

SS - Nuno Santos

2019



TOCTOU: Time-of-check to time of use

aka [TOCTTOU](#)
also [symlink attack](#)

► Caso típico:

- Um programa configurado com `setuid root` (por exemplo um editor de texto) é pedido para escrever num ficheiro cujo dono é o utilizador que está a correr o programa
- Root pode escrever em qualquer ficheiro, pelo que o programa tem que verificar se o utilizador actual tem o direito de escrever nesse ficheiro

```
janela vulnerab { if(!access(file, W_OK)) { // 0 if the user has write privilege
                  f = fopen(file, "wb+");
                  write_to_file(f);
                } else {
                  fprintf(stderr, "Permission denied\n");
                }
            }
```

Correr até sucesso sucesso:
\$ touch dummy
\$ ln -s dummy pointer
\$ program pointer &
\$ rm pointer;\n
ln -s /etc/passwd pointer

► 11

SS - Nuno Santos

2019



Ficheiros temporários

- Mesmo problema que os TOCTOU mais o facto dos ficheiros serem colocados num directório partilhado, local em que um atacante também pode escrever

- `/tmp`, `/var/tmp`

► Ataque típico:

- Um programa privilegiado verifica que não existe um ficheiro X no `/tmp`
- Atacante corre para correr um link de nome X para um determinado ficheiro com permissões restritas, por exemplo `/etc/passwd`
- O programa privilegiado tenta criar X, mas na realidade abre o ficheiro escolhido pelo atacante, fazendo algo indejável...

► 12

SS - Nuno Santos

2019



Concorrência

- ▶ Nos casos anteriores, a corrida é criada pelo atacante com intenção maliciosa
- ▶ Mas é possível que essa corrida seja devido ao próprio programa que corre em vários processos ou threads e acedem a objectos partilhados
- ▶ Vulnerabilidade surge se existem problemas de sincronização entre os processos ou threads

▶ 13

SS - Nuno Santos

2019



Exemplo de concorrência: Java servlet

- ▶ O código Java seguinte corre num servidor e é chamado por várias threads / processos ao mesmo tempo
- ▶ Pode acontecer que um servlet (este código) dê a dois utilizadores o mesmo valor

```
public class Counter extends HttpServlet {  
    int count = 0;    //shared!!  
    public void doGet(..., HttpServletResponse out) throws...  
    {  
        PrintWriter p = out.getWriter();  
        count++;  
        p.println(count + " hits so far!");  
    }  
}
```

▶ 14

SS - Nuno Santos

2019

Vulnerabilidades de validação de entradas

15

SS - Nuno Santos

2019

Validação de entradas

- Os programas devem restringir os valores de entrada que podem ser introduzidos pelos utilizadores



► 16

SS - Nuno Santos

2019



Exemplo de um ataque real

- ▶ SandWorm / CVE-2014-4114
- ▶ “On Tuesday, October 14, 2014, iSIGHT Partners – in close collaboration with Microsoft – announced the discovery of a zero-day vulnerability impacting all supported versions of Microsoft Windows and Windows Server 2008 and 2012.”
- ▶ “Exploitation of this vulnerability was discovered in the wild in connection with a cyber-espionage campaign that iSIGHT Partners attributes to Russia.”
- ▶ “the vulnerability exists in PACKAGER.DLL, which is a part of Windows Object Linking and Embedding (OLE) property. By using a crafted PowerPoint document [an input!], an .INF file in embedded OLE object can be copied from a remote SMB share folder and installed on the system. Attackers can exploit this logic defect to execute another malware, downloaded via the same means.”



Entradas: argumentos para os programas

- ▶ Exemplo:
`my_filter file.txt a_string -o caps`
- ▶ Um atacante pode passar argumentos malformados como parâmetro de entrada para o programa (nomes grandes → buffer overflow)
- ▶ Se o programa não validar correctamente os parâmetros, é especialmente grave, sobretudo para programas privilegiados (root)



Outros vectores de entrada para os programas

- ▶ **Variáveis de ambiente**

Exemplo: PATH=/bin:/home/alice/bin

- ▶ **Bibliotecas**

- ▶ Problema no windows: injeção automática de bibliotecas (que podem conter código malicioso)

- ▶ **Injeção de caracteres especiais**

- ▶ Delimitadores, caracteres NUL, separadores de comandos



Conclusões

- ▶ Corridas são violações das expectativas do programador sobre a atomicidade de um programa durante uma janela de vulnerabilidade
- ▶ Os principais objectivos de explorar essas vulnerabilidades são escalar privilégios e as mais comuns são: TOCTOU, ficheiros temporários, e concorrência
- ▶ Outra classe de vulnerabilidades é deficiente validação de entradas, que permite ao atacante injectar dados que resultem num desvio do comportamento normal do programa



Referências e próxima aula

► Bibliografia

- [Correia17] Capítulos 6 e 7

► Próxima aula

- Vulnerabilidades na Web e em bases de dados