

Mestrado em Engenharia Informática
Universidade do Minho

Engenharia de Segurança

Aula 07 TP - 23/03/2020

João Miranda - PG41845
Sandro Cruz - PG41906

6 de Abril de 2020

Conteúdo

1	Vulnerabilidade de codificação	2
1.1	Experiência 1.1 - Common Weakness Enumeration (CWE)	2
1.1.1	Contexto	2
1.1.2	Vulnerabilidades de projeto	3
1.1.3	Vulnerabilidades de codificação	5
1.1.4	Vulnerabilidades operacionais	6
1.2	Pergunta 1.1 - Common Weakness Enumeration (CWE)	6
1.2.1	Exercício 1	6
1.2.2	Exercício 2	10
1.3	Experiência 1.2 - Common Vulnerabilities and Exposures (CVE)	12
1.4	Experiência 1.3 - Common Vulnerability Scoring System (CVSS)	13
1.5	Experiência 1.4 - National Vulnerability Database (NVD)	14
1.6	Pergunta P1.2	15
1.6.1	Exercício 1	15
1.6.2	Exercício 2	16
1.7	Pergunta P1.3	16
1.7.1	Vulnerabilidades de projeto	16
1.7.2	Vulnerabilidades de codificação	17
1.7.3	Vulnerabilidades operacionais	17
1.8	Pergunta P1.4	18
1.9	Experiência 1.5 - Exploit Database	18
1.10	Experiência 1.6 - Google Hacking Database	18
1.11	Experiência 1.7 - Linguagem C	19

Capítulo 1

Vulnerabilidade de codificação

1.1 Experiência 1.1 - Common Weakness Enumeration (CWE)

1.1.1 Contexto

Relações

Existem tabelas que revelam os pontos fracos e as categorias de alto nível relacionados a esse ponto fraco. Os relacionamentos são definidos como *ChildOf*, *ParentOf*, *MemberOf* e fornecem informações sobre itens semelhantes que podem existir em níveis cada vez mais altos de abstração.

Modos de introdução

Os modos de introdução fornecem informações sobre como e quando essa fraqueza pode ser introduzida. A fase identifica um ponto no ciclo de vida em que a introdução pode ocorrer, enquanto a nota fornece um cenário típico relacionado à introdução durante a fase especificada.

Plataformas aplicáveis

As plataformas aplicáveis consistem na exploração da área para as quais a fraqueza especificada pode acontecer. Podem ser para linguagens, sistemas operativos, arquiteturas, paradigmas, tecnologias ou uma classe específica dessas plataformas. A plataforma é listada juntamente com a frequência com que a fraqueza especificada aparece nessa instância.

Consequências comuns

De forma a avaliar as consequências mais comuns, é construída uma tabela que especifica diferentes consequências individuais associadas à fraqueza. O alcance identifica a área de segurança da aplicação que é violada, enquanto o impacto

descreve o impacto técnico negativo que surge se um atacante conseguir explorar essa fraqueza. A probabilidade fornece informações sobre a probabilidade ocorrência de uma consequência específica em relação às outras.

Relacionamentos

A tabela de relacionamentos *MemberOf* demonstra categorias e visualizações adicionais do CWE que referenciam essa fraqueza como membro. Essas informações, geralmente, são úteis para entender onde uma fraqueza se encaixa no contexto de fontes de informações externas.

1.1.2 Vulnerabilidades de projeto

Com o intuito da percepção dos membros desta classe de vulnerabilidades, foi explorado um destes denominado de **Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')** em que o seu ID corresponde ao 89 (CWE-89) e o tipo é o B (Base). O tipo corresponde a uma fraqueza que ainda é maioritariamente independente de um recurso ou tecnologia, mas com detalhes suficientes para fornecer métodos específicos para detecção e prevenção. Os pontos fracos neste nível (Básico) geralmente descrevem problemas em termos das dimensões de comportamento, propriedade, tecnologia, idioma e recurso.

Descrição

O *software* constrói todo ou parte de um comando SQL usando entrada influenciada externamente de uma componente *upstream*, mas não neutraliza ou neutraliza incorretamente elementos especiais que podem modificar o comando SQL pretendido quando ele é enviado para uma componente *downstream*. A consulta SQL gerada pode fazer com que essas entradas sejam interpretadas como SQL em vez de dados comuns do utilizador. Isso pode ser utilizado para alterar a lógica da consulta para ignorar as verificações de segurança ou para inserir instruções adicionais que modificam a base de dados *backend*, possivelmente incluindo a execução de comandos do sistema.

A injeção de SQL tornou-se um problema comum em *sites* direcionados a bases de dados. A falha é facilmente detetada e explorada e, como tal, qualquer *site* ou pacote de *software* com uma base mínima de utilizadores provavelmente estará sujeito a uma tentativa de ataque desse tipo. Essa falha depende do facto do SQL não fazer distinção real entre os planos de controlo e os dados.

Relações

▼ 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 "Software Development" (CWE-699)

Nature	Type	ID	Name
MemberOf	🔴	137	Data Representation Errors

Figura 1.1: Tabelas dos pontos fracos.

Modos de introdução

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.

Figura 1.2: Modos de introdução.

Plataformas aplicáveis

Languages

Class: Language-Independent (*Undetermined Prevalence*)

Technologies

Database Server (*Undetermined Prevalence*)

Figura 1.3: Plataformas aplicáveis.

Consequências comuns

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.	

Figura 1.4: Tabela das consequências mais comuns.

1.1.3 Vulnerabilidades de codificação

Para a percepção dos membros desta classe de vulnerabilidades, foi explorado um destes denominado de **Path Equivalence: 'file name' (Internal Whitespace)** em que o seu ID corresponde ao 48 (CWE-48) e o tipo é o V (Variante). O Variante corresponde a uma fraqueza que está vinculada a um determinado tipo de produto que, geralmente, envolve uma linguagem ou uma tecnologia específica. É mais específico do que uma fraqueza de tipo B (Base). Os pontos fracos no nível da variante normalmente descrevem problemas em termos de 3 a 5 das dimensões de compormamento, propriedade, tecnologia, linguagem e recurso.

Descrição

Este problema está relacionado com um sistema de *software* que aceita a entrada do caminho na forma de espaço interno ('nome do ficheiro (SPACE)') e que, sem validação apropriada pode levar à resolução ambígua do caminho e permitir que um atacante atravesse o sistema de ficheiros para locais não desejados ou acesse ficheiros arbitrários.

Relações

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

Nature	Type	ID	Name
ChildOf	B	41	Improper Resolution of Path Equivalence

Figura 1.5: Tabelas dos pontos fracos.

Modos de introdução

Phase	Note
Implementation	

Figura 1.6: Modos de introdução.

Plataformas aplicáveis

Languages

Class: Language-Independent (*Undetermined Prevalence*)

Figura 1.7: Plataformas aplicáveis.

Consequências comuns

Scope	Impact
Confidentiality Integrity	Technical Impact: <i>Read Files or Directories; Modify Files or Directories</i>

Figura 1.8: Tabela das consequências mais comuns.

Relacionamentos

Nature	Type	ID	Name
MemberOf	C	981	SFP Secondary Cluster: Path Traversal

Figura 1.9: Tabela de relacionamentos.

1.1.4 Vulnerabilidades operacionais

Relativamente aos membros desta classe de vulnerabilidades, foi explorado o membro denominado de **Configuration** em que o seu ID de categoria corresponde ao 16 (CWE CATEGORY).

Descrição

As fraquezas nesta categoria são normalmente introduzidas durante a configuração do *software*.

Relacionamentos

Nature	Type	ID	Name
MemberOf	V	635	Weaknesses Originally Used by NVD from 2008 to 2016
MemberOf	C	933	OWASP Top Ten 2013 Category A5 - Security Misconfiguration
MemberOf	C	1032	OWASP Top Ten 2017 Category A6 - Security Misconfiguration

Figura 1.10: Tabela de relacionamentos.

1.2 Pergunta 1.1 - Common Weakness Enumeration (CWE)

1.2.1 Exercício 1

CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer

Este problema consiste no *software* executar operações num *buffer* de memória

podendo ler ou gravar num local de memória que esteja fora do limite pretendido do *buffer*. Algumas linguagens permitem o endereçamento direto dos locais da memória e não garantem automaticamente que esses locais sejam válidos para o *buffer* de memória que está a ser referenciado. Isso pode fazer com que operações de leitura ou gravação sejam executadas em locais de memória que podem estar associados a outras variáveis, estruturas de dados ou dados internos do programa. Como resultado, um atacante pode ser capaz de executar código arbitrário, alterar o fluxo de controlo pretendido, ler informações confidenciais ou causar falha no sistema.

Plataformas aplicáveis

As plataformas aplicáveis correspondem às linguagens C, C++ (frequentemente prevalentes) e a classe é o *Assembly* (prevalência indeterminada).

Consequências mais comuns

Alcance	Impacto
Integridade Confidencialidade Disponibilidade	<p>Impacto Técnico: Executar códigos ou comandos não autorizados; Modificar memória</p> <p>Se a memória acessível pelo atacante puder ser efetivamente controlada, talvez seja possível executar código arbitrário, como ocorre com um buffer overflow padrão. Se o atacante puder transcrever a memória de um ponteiro (32, 64 bits), poderá redirecionar um ponteiro de função para o seu próprio código malicioso. Mesmo o atacante podendo alterar apenas uma execução de código arbitrário de 1 byte pode ser possível. Às vezes, isto ocorre porque o mesmo problema pode ser explorado repetidamente para o mesmo efeito. Outras vezes, é porque o atacante pode transcrever dados específicos de aplicações críticas para a segurança.</p>
Confidencialidade Disponibilidade	<p>Impacto Técnico: Leitura da memória; DoS: Empancar, sair ou reiniciar; DoS: Consumo de CPU; DoS: Consumo de memória</p> <p>Atravessar o limite do acesso à memória provavelmente resultará na corrupção de memória relevante e, talvez, nas instruções, possivelmente levando a uma falha. Outros ataques que levam à falta de disponibilidade são possíveis, incluindo colocar o programa em loop infinito.</p>
Confidencialidade	<p>Impacto Técnico: Leitura de memória</p> <p>Atravessar o limite da leitura da memória pode resultar no facto do atacante poder ter acesso a informações confidenciais. Se as informações confidenciais contiverem detalhes do sistema, estes poderão ser utilizados para uma nova criação de ataques com consequências possivelmente mais graves.</p>

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

Este problema consiste no *software* não neutralizar ou neutralizar incorreta-

mente a entrada controlável pelo utilizador antes de ser colocada na saída utilizada como uma página *web* que é vinculada a outros utilizadores. Após o *script* malicioso (XSS) ser injetado, o atacante pode executar uma variedade de atividades mal-intencionadas. O atacante pode transferir informações privadas como *cookies* que podem incluir informações da sessão e da máquina da vítima; enviar solicitações maliciosas para um *site* em nome da vítima, o que pode ser especialmente perigoso para o *site* se a vítima tiver privilégios de administrador para gerenciar o *site*. Os ataques de *phishing* podem ser utilizados para emular *sites* confiáveis e induzir a vítima a digitar uma *password*, permitindo que o atacante comprometa a conta da vítima nesse *site*. Por fim, o *script* poderia explorar uma vulnerabilidade no próprio navegador da *web*, possivelmente assumindo o controlo da máquina da vítima, às vezes denominada de invasão-por-hackers. Em muitos casos, o ataque pode ser iniciado sem que a vítima esteja ciente disso. Mesmo com utilizadores cuidadosos, os atacantes costumam usar uma variedade de métodos para codificar a parte maliciosa do ataque, como codificação de URL ou Unicode, para que a solicitação pareça menos suspeita.

Plataformas aplicáveis

As plataformas aplicáveis correspondem à linguagem de classe *Linguagem-Independente* (prevalência indeterminada); tecnologia de classe *Web Based* (frequentemente prevalente).

Consequências mais comuns

Alcance	Impacto
Controlo de acesso Confidencialidade	<p>Impacto Técnico: Mecanismo de proteção de desvio; Ler dados da aplicação</p> <p>O ataque mais comum realizado com scripts entre sites envolve a divulgação de informações armazenadas nos cookies do utilizador. Normalmente, um utilizador mal-intencionado cria um script do lado do cliente que, quando analisado por navegador da web, realiza alguma atividade (como enviar todos os cookies do site para determinado endereço de e-mail). Este script será carregado e executado por cada utilizador que visitar o site. Como o site que solicita a execução do script tem acesso aos cookies em questão, o script malicioso também tem.</p>
Integridade Confidencialidade Disponibilidade	<p>Impacto Técnico: Executar códigos ou comandos não autorizados</p> <p>Em algumas circunstâncias, pode ser possível executar código arbitrário no computador da vítima quando o script entre sites é combinado com outras falhas.</p>
Confidencialidade Integridade Disponibilidade Controlo de acesso	<p>Impacto Técnico: Executar códigos ou comandos não autorizados; Mecanismo de proteção de desvio; Ler dados da aplicação</p> <p>A consequência de um ataque XSS é a mesma, independentemente de ser armazenada ou refletida. A diferença está em como a carga útil chega ao servidor. O XSS pode causar uma variedade de problemas para o utilizador final, que variam em gravidade, desde um aborrecimento até ao comprometimento total da conta. Algumas vulnerabilidades de script entre sites podem ser exploradas para manipular ou roubar cookies, criar solicitações que podem ser confundidas com as de um utilizador, comprometer informações confidenciais ou executar código malicioso nos sistemas do utilizador para vários propósitos. Outros ataques prejudiciais incluem a divulgação de ficheiros do utilizador, a instalação de programas de cavalos de Tróia, o redirecionamento do utilizador para outra página ou site, a execução de controlos Active X (Microsoft Internet Explorer) de sites que o utilizador considera confiáveis e a modificação de conteúdo.</p>

CWE-20: Improper Input Validation

O problema consiste no produto não validar ou validar incorretamente as entradas que podem afetar o fluxo de controlo ou o fluxo de dados de um programa. Quando o *software* não valida a entrada corretamente, um atacante pode criar a entrada de uma forma que não é esperada pelo restante da aplicação. Isso fará com que partes do sistema recebam uma entrada não intencional, o que pode resultar em fluxo de controlo alterado, controlo arbitrário ou execução arbitrária de código.

Plataformas aplicáveis

As plataformas aplicáveis correspondem à linguagem de classe é *Linguagem-Independente* (prevalência indeterminada).

Consequências mais comuns

Alcance	Impacto
Disponibilidade	Impacto Técnico: <i>DoS: empancar, sair ou reiniciar; DoS: consumo de CPU; DoS: consumo de memória</i> Um atacante pode fornecer valores inesperados e causar uma falha no programa ou consumo excessivo de recursos, como memória e CPU.
Confidencialidade	Impacto Técnico: Ler memória; Ler ficheiros ou caminhos Um atacante pode ler dados confidenciais se conseguir controlar as referências de recursos.
Integridade Confidencialidade Disponibilidade	Impacto Técnico: Modificar memória; Executar códigos ou comandos não autorizados Um atacante pode utilizar uma entrada maliciosa para modificar dados ou possivelmente alterar o fluxo de controlo de formas inesperadas, incluindo a execução arbitrária de códigos.

1.2.2 Exercício 2

Como o nosso grupo é o número 10 vamos investigar a CWE que está no 13º lugar do **The CWE Top 25**.

CWE-287: Improper Authentication

Este problema acontece quando um ator afirmar ter uma determinada identidade e assim o *software* não prova ou prova insuficientemente que a afirmação está correta.

Plataformas aplicáveis


As plataformas aplicáveis correspondem à linguagens de classe é *Linguagem-Independente* (prevalência indeterminada).

Consequências mais comuns

Alcance	Impacto
Integridade Confidencialidade Disponibilidade Controlo de acesso	Impacto Técnico: Ler os dados da aplicação; Obter privilégios ou assumir identidade; Executar códigos ou comandos não autorizados Esta fraqueza pode levar à exposição de recursos ou funcionalidades a atores não intencionais, possivelmente fornecendo aos atacantes informações confidenciais ou até mesmo executando código arbitrário.

Código para esta *weakness*

De seguida vai ser apresentado um código que pretende garantir que o utilizador já efetuou *login*. Caso contrário, o código executa autenticação com o nome do utilizador e a *password* fornecidos pelo utilizador. Caso seja bem sucedido, o atacante define os *cookies* do *login* sucedido e os *cookies* do utilizador para lembrar que este já efetuou *login*. Por fim, o código executa tarefas de administrador se o utilizador que fez *login* tiver nome de utilizador **Administrador** conforme registado no *cookie* do utilizador.



```
Example Language: Perl [hide code]
my $q = new CGI;

if ($q->cookie('loggedin') ne "true") {
    if (!AuthenticateUser($q->param('username'), $q->param('password'))) {
        ExitError("Error: you need to log in first");
    }
    else {
        # Set loggedin and user cookies.
        $q->cookie(
            -name => 'loggedin',
            -value => 'true'
        );

        $q->cookie(
            -name => 'user',
            -value => $q->param('username')
        );
    }
}

if ($q->cookie('user') eq "Administrador") {
    DoAdministratorTasks();
}
```

Figura 1.11: Código escrito em Perl.

Infelizmente, este código pode ser ignorado. O atacante pode definir os *cookies* independentemente, para que o código não verifique o nome do utilizador e a *password*, fazendo isso com uma solicitação HTTP contendo cabeçalhos como:

```
GET /cgi-bin/vulnerable.cgi HTTP/1.1
Cookie: user=Administrador
Cookie: loggedin=true
```

Ao ser definido o *cookie* de *loggedin* como *true*, o atacante ignora toda a verificação de autenticação. Utilizando o valor **Administrador** no *cookie* do utilizador, o atacante também pode obter privilégios para administrar o *software*.

Exemplo de CVEs

- **CVE-2009-3421:** Quando o *register_globals* está ativado no *login* no Zenas PaoBacheca Guestbook 2.1 (admin.php), é permitido que atacantes remotos ignorem a autenticação e obtenham acesso administrativo definindo o parâmetro *login_ok* como 1.
- **CVE-2009-2382:** O *admin.php* no phpMyBlockchecker 1.0.0055 permite que atacantes remotos ignorem a autenticação e obtenham acesso administrativo configurando o *cookie* PHPMYBCAdmin como *LOGGEDIN*.

- **CVE-2009-2422:** O código de exemplo para a funcionalidade da autenticação *digest* (`http_authentication.rb`) no Ruby on Rails antes da versão 2.3.3 define um bloco *authenticate_or_request_with_http_digest* que retorna nulo em vez de falso quando o utilizador não existe, o que permite que atacantes dependentes de contexto ignorem a autenticação de aplicações que são derivadas deste exemplo enviando um nome de utilizador inválido sem uma *password*.

1.3 Experiência 1.2 - Common Vulnerabilities and Exposures (CVE)

Utilizando o <https://cve.mitre.org/> conseguimos verificar que:

O detalhe da vulnerabilidade mais recente:

A CVE mais recente é a CVE-2020-11501, obtivemos esta CVE utilizando a conta do Twitter da CVE que fornece sempre os utilizações feitos a lista de novas CVEs.

The screenshot shows the details for CVE-2020-11501 on the NVD website. The entry is titled 'CVE-2020-11501' and is linked to the National Vulnerability Database (NVD). The description states: 'GnuTLS 3.6.x before 3.6.13 uses incorrect cryptography for DTLS. The earliest affected version is 3.6.3 (2018-07-16) because of an error in a 2017-10-06 commit. The DTLS client always uses 32 '\0' bytes instead of a random value, and thus contributes no randomness to a DTLS negotiation. This breaks the security guarantees of the DTLS protocol.' The references section lists three links: a GitHub commit, a GitHub issue, and a GitHub security advisory. The assigning CNA is MITRE Corporation. The date entry created is 20200403. The phase is 'Assigned (20200403)'. The votes and comments sections are empty. The proposed section is also empty. At the bottom, there is a search bar and a link to the CVE Request Web Form.

Figura 1.12: Detalhes da CVE-2020-11501.

As vulnerabilidades identificadas no Google Chrome:

Dado que existem 2166 vulnerabilidades associadas ao Google Chrome, iremos apenas apresentar as mais recentes.

Search Results	
There are 2166 CVE entries that match your search.	
Name	Description
CVE-2020-6449	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6429	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6428	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6427	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6426	Inappropriate implementation in V8 in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6425	Insufficient policy enforcement in extensions in Google Chrome prior to 80.0.3987.149 allowed an attacker who convinced a user to install a malicious extension to bypass site isolation via a crafted Chrome Extension.
CVE-2020-6424	Use after free in media in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6422	Use after free in WebGL in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6420	Insufficient policy enforcement in media in Google Chrome prior to 80.0.3987.132 allowed a remote attacker to bypass same origin policy via a crafted HTML page.
CVE-2020-6418	Type confusion in V8 in Google Chrome prior to 80.0.3987.122 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6417	Inappropriate implementation in installer in Google Chrome prior to 80.0.3987.87 allowed a local attacker to execute arbitrary code via a crafted registry entry.
CVE-2020-6416	Insufficient data validation in streams in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6415	Inappropriate implementation in JavaScript in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6414	Insufficient policy enforcement in Safe Browsing in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to bypass navigation restrictions via a crafted HTML page.
CVE-2020-6413	Inappropriate implementation in Blink in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to bypass HTML validators via a crafted HTML page.
CVE-2020-6412	Insufficient validation of untrusted input in Omnibox in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to perform domain spoofing via IDN homographs via a crafted domain name.
CVE-2020-6411	Insufficient validation of untrusted input in Omnibox in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to perform domain spoofing via IDN homographs via a crafted domain name.
CVE-2020-6410	Insufficient policy enforcement in navigation in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to confuse the user via a crafted domain name.
CVE-2020-6409	Inappropriate implementation in Omnibox in Google Chrome prior to 80.0.3987.87 allowed a remote attacker who convinced the user to enter a URI to bypass navigation restrictions via a crafted domain name.
CVE-2020-6408	Insufficient policy enforcement in CORS in Google Chrome prior to 80.0.3987.87 allowed a local attacker to obtain potentially sensitive information via a crafted HTML page.
CVE-2020-6407	Out of bounds memory access in streams in Google Chrome prior to 80.0.3987.122 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6406	Use after free in audio in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6405	Out of bounds read in SQLite in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to obtain potentially sensitive information from process memory via a crafted HTML page.
CVE-2020-6404	Inappropriate implementation in Blink in Google Chrome prior to 80.0.3987.87 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page.
CVE-2020-6403	Incorrect implementation in Omnibox in Google Chrome on iOS prior to 80.0.3987.87 allowed a remote attacker to spoof the contents of the Omnibox (URL bar) via a crafted HTML page.

Figura 1.13: Algumas vulnerabilidades associadas ao Google Chrome.

As vulnerabilidades identificadas no Facebook:

Dado que existem 72 vulnerabilidades associadas ao Facebook, iremos apenas apresentar as mais recentes.

Search Results	
There are 72 CVE entries that match your search.	
Name	Description
CVE-2019-9911	The social-networks-auto-poster-facebook-twitter-g plugin before 4.2.8 for WordPress has wp-admin/admin.php?page=--nssnap-reposter&action=edit item XSS.
CVE-2019-7411	Multiple stored cross-site scripting (XSS) in the My Themeshop Launcher plugin 1.0.8 for WordPress allow remote authenticated users to inject arbitrary web script or HTML via fields as follows: (1) Title, (2) Favicon, (3) Meta Description, (4) Subscribe Form (Name field label, Last name field label), (5) Contact Form (Name field label and Email field label), and (6) Social Links (Facebook Page URL, Twitter Page URL, Instagram Page URL, YouTube Page URL, LinkedIn Page URL, Google+ Page URL, RSS URL).
CVE-2019-3565	Legacy C++ Facebook Thrift servers (using cpp instead of cpp2) would not error upon receiving messages with containers of fields of unknown type. As a result, malicious clients could send short messages which would take a long time for the server to parse, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2019.05.06.00.
CVE-2019-3564	Go Facebook Thrift servers would not error upon receiving messages with containers of fields of unknown type. As a result, malicious clients could send short messages which would take a long time for the server to parse, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2019.02.18.00.
CVE-2019-3559	Java Facebook Thrift servers would not error upon receiving messages with containers of fields of unknown type. As a result, malicious clients could send short messages which would take a long time for the server to parse, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2019.02.18.00.
CVE-2019-3558	Python Facebook Thrift servers would not error upon receiving messages with containers of fields of unknown type. As a result, malicious clients could send short messages which would take a long time for the server to parse, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2019.02.18.00.
CVE-2019-3553	C++ Facebook Thrift servers would not error upon receiving messages declaring containers of sizes larger than the payload. As a result, malicious clients could send short messages which would result in a large memory allocation, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2020.02.03.00.
CVE-2019-3552	C++ Facebook Thrift servers (using cpp2) would not error upon receiving messages with containers of fields of unknown type. As a result, malicious clients could send short messages which would take a long time for the server to parse, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2019.02.18.00.
CVE-2019-19684	nopCommerce v4.2.0 allows privilege escalation via file upload in Presentation/NoP/Web/Admin/Areas/Controllers/PluginController.cs via Admin/FacebookAuthentication/Configure because it is possible to upload a crafted Facebook Auth plugin.
CVE-2019-15841	The Facebook-for-woocommerce plugin before 1.9.14 for WordPress has CSRF via ajax_woo_infobanner_post_click, ajax_woo_infobanner_post_xout, or ajax_fb_toggle_visibility.
CVE-2019-15840	The Facebook-for-woocommerce plugin before 1.9.14 for WordPress has CSRF.
CVE-2019-15781	The Facebook-by-webbizar plugin before 2.8.5 for WordPress has CSRF.
CVE-2019-13344	An authentication bypass vulnerability in the CRUDDLAB WP Like Button plugin through 1.6.0 for WordPress allows unauthenticated attackers to change settings. The contains() function in wp_like_button.php did not check if the current request is made by an authorized user, thus allowing any unauthenticated user to successfully update settings, as demonstrated by the wp-admin/admin.php?page=facebook-like-button&each_page_url or code_snippet parameters.
CVE-2019-11939	GoLang Facebook Thrift servers would not error upon receiving messages declaring containers of sizes larger than the payload. As a result, malicious clients could send short messages which would result in a large memory allocation, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2020.03.16.00.
CVE-2019-11938	Java Facebook Thrift servers would not error upon receiving messages declaring containers of sizes larger than the payload. As a result, malicious clients could send short messages which would result in a large memory allocation, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2019.12.09.00.
CVE-2018-6858	Cross Site Scripting (XSS) exists in PHP Scripts Mail Facebook Clone Script.
CVE-2018-6367	SQL Injection exists in Vastal I-Tech Buddy Zone Facebook Clone 2.9.9 via the /chat_im/chat_window.php request_id parameter or the /search_events.php category parameter.
CVE-2018-5978	SQL Injection exists in Facebook Style Php Ajax Chat Zechat 1.5 via the login.php User field.
CVE-2018-5214	The "Add Link to Facebook" plugin through 2.3 for WordPress has XSS via the a2fb_facebook_id parameter to wp-admin/profile.php.

Figura 1.14: Algumas vulnerabilidades associadas ao Facebook.

1.4 Experiência 1.3 - Common Vulnerability Scoring System (CVSS)

Verificamos diferentes opções para testar a forma como certas características são prioritárias.

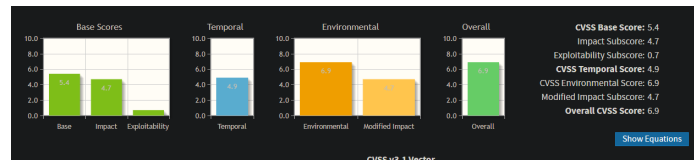


Figura 1.15: Exemplo de um dos testes.

1.5 Experiência 1.4 - National Vulnerability Database (NVD)

Utilizando o <https://nvd.nist.gov/> conseguimos verificar que:

Qual é a vulnerabilidade mais recente identificada?

Fazendo uma pesquisa descobrimos que a vulnerabilidade é a CVE-2020-10960.

Q Search Results (Refine Search)			Sort results by: Publish Date Descending	Sort
Search Parameters: <ul style="list-style-type: none"> Results Type: Overview Search Type: Search Last 3 Months 			There are 4,980 matching records. Displaying matches 1 through 20 .	
Vuln ID	Summary	CVSS Severity		
CVE-2020-10960	In MediaWiki before 1.34.1, users can add various Cascading Style Sheets (CSS) classes (which can affect what content is shown or hidden in the user interface) to arbitrary DOM nodes via HTML content within a MediaWiki page. This occurs because jquery.makeCollapsible allows applying an event handler to any Cascading Style Sheets (CSS) selector. There is no known way to exploit this for cross-site scripting (XSS). Published: April 03, 2020; 11:15:14 AM -04:00	(not available)		
CVE-2020-10689	A flaw was found in the Eclipse Che up to version 7.8.x, where it did not properly restrict access to workspace pods. An authenticated user can exploit this flaw to bypass JWT proxy and gain access to the workspace pods of another user. Successful exploitation requires knowledge of the service name and namespace of the target pod. Published: April 03, 2020; 11:15:14 AM -04:00	(not available)		

Figura 1.16: Exemplo de um dos testes.

Essa vulnerabilidade é a mesma vulnerabilidade mais recente encontrada na experiência 1.2 (CVE)? Qual poderá ser o motivo?

Não é, achamos que isto é devido a lista das CVEs ser global e a das NVDs ser só dos Estados Unidos, ou seja enquanto que uma armazena dados globais a outra apenas armazena os dados referentes as vulnerabilidades dos Estados Unidos.

As vulnerabilidades identificadas no Google Chrome:

Vuln ID	Summary	CVSS Severity
CVE-2020-6449	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page. Published: March 23, 2020, 12:15:17 PM -04:00	V3.1: 9.3 HIGH V2: 9.3 HIGH
CVE-2020-6429	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page. Published: March 23, 2020, 12:15:17 PM -04:00	V3.1: 9.3 HIGH V2: 9.3 HIGH
CVE-2020-6428	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page. Published: March 23, 2020, 12:15:17 PM -04:00	V3.1: 9.3 HIGH V2: 9.3 HIGH
CVE-2020-6427	Use after free in audio in Google Chrome prior to 80.0.3987.149 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page. Published: March 23, 2020, 12:15:17 PM -04:00	V3.1: 9.3 HIGH V2: 9.3 HIGH

Figura 1.17: Exemplo de um dos testes.

As vulnerabilidades identificadas no Facebook:

Vuln ID	Summary	CVSS Severity
CVE-2019-11939	Golang Facebook Thrift servers would not error upon receiving messages declaring containers of sizes larger than the payload. As a result, malicious clients could send short messages which would result in a large memory allocation, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2020.03.16.00. Published: March 17, 2020, 09:15:11 PM -04:00	V3.1: 7.5 HIGH V2: 9.0 MEDIUM
CVE-2020-1887	Incorrect validation of the TLS SNI hostname in osquery versions after 2.9.0 and before 4.2.0 could allow an attacker to MITM osquery traffic in the absence of a configured root chain of trust. Published: March 12, 2020, 08:15:11 PM -04:00	V3.1: 9.1 CRITICAL V2: 9.0 MEDIUM
CVE-2019-3553	C++ Facebook Thrift servers would not error upon receiving messages declaring containers of sizes larger than the payload. As a result, malicious clients could send short messages which would result in a large memory allocation, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2020.02.03.00. Published: March 10, 2020, 05:15:11 PM -04:00	V3.1: 7.5 HIGH V2: 9.0 MEDIUM
CVE-2019-11938	Java Facebook Thrift servers would not error upon receiving messages declaring containers of sizes larger than the payload. As a result, malicious clients could send short messages which would result in a large memory allocation, potentially leading to denial of service. This issue affects Facebook Thrift prior to v2020.03.16.00. Published: March 17, 2020, 09:15:11 PM -04:00	V3.1: 7.5 HIGH V2: 9.0 MEDIUM

Figura 1.18: Exemplo de um dos testes.

1.6 Pergunta P1.2

1.6.1 Exercício 1

Através da informação dada sobre o número de linhas de código, é possível calcular o número de *bugs*. Considerando que o número de linhas de código é 1000, vai ser utilizada a seguinte fórmula:

$$N_{bugs} = (N_{Linhasdecodigo}/1000) * N_{bugs1000linhas}$$

Sendo que o número de *bugs* varia entre 10 e 100 a cada 1000 linhas conclui-se que:

- **Facebook:** possui 62000000 linhas de código fonte. Logo a estimativa para 10 *bugs* por 1000 linhas é de 620000; para 100 *bugs* por 1000 linhas é de 6200000.
- **Software de automóveis:** possui 100000000 linhas de código fonte. Logo a estimativa para 10 *bugs* por 1000 linhas é de 1000000; para 100 *bugs* por 1000 linhas é de 10000000.
- **Linux 3.1:** possui 15000000 linhas de código fonte. Logo a estimativa para 10 *bugs* por 1000 linhas é de 150000; para 100 *bugs* por 1000 linhas é de 1500000.
- **Serviços Internet da Google:** possui 2000000000 linhas de código fonte. Logo a estimativa para 10 *bugs* por 1000 linhas é de 20000000; para 100 *bugs* por 1000 linhas é de 200000000.

1.6.2 Exercício 2

De uma forma geral um *bug* pode ser descrito como uma falha num programa que resulte em comportamento inesperado no funcionamento do mesmo. Como tal, é possível estabelecer uma relação direta entre o número de *bugs* e o número de linhas de código de um pacote de *software*. No entanto, a relação entre o número de *bugs* e o número de vulnerabilidades é menos clara, sendo difícil determinar, com precisão, quantos *bugs* num dado sistema resultam em vulnerabilidades bem como quais dessas vulnerabilidades são passíveis de serem exploradas.

Portanto, é complexa uma estimativa do número de vulnerabilidades no código fonte de cada projeto, visto ser algo que depende de diversos fatores como a linguagem de programação escolhida, a experiência e conhecimento do programador ou até o tipo de aplicação desenvolvida.

1.7 Pergunta P1.3

1.7.1 Vulnerabilidades de projeto

- **CVE-2001-0003:** O *Web Extender Client* (WEC) no *Microsoft Office 2000*, *Windows 2000* e *Windows me* não processa adequadamente as configurações de segurança do *Internet Explorer* para autenticação NTLM, o que permite que os atacantes obtenham credenciais NTLM e possivelmente obtenham a *password*.

Como correção, foi recomendado que os clientes com uma versão afetada dos produtos, anteriormente listados, recorressem à instalação de um *patch* que resolveria o problema.

- **CVE-2015-4495:** O leitor de pdf do *Mozilla Firefox* até à versão 39.0.3 permitia que atacantes remotos ultrapassassem a política *Same Origin Policy* e lessem ficheiros arbitrários ou ganhassem privilégios, através de vetores que envolvessem código em *Javascript* escrito pelo atacante e um *setter* ativo.

Para a correção deste problema, foi necessário que os clientes atualizassem o *browser* para uma versão superior à 39.0.3.

A correção das vulnerabilidades de projeto pode implicar uma nova estruturação de todo o sistema, o que poderia implicaria gastos dispendiosos económico e um tempo excessivo para uma organização. Por outro lado, pode implicar apenas modificar componentes específicas do sistema, o que já é bastante acessível.

1.7.2 Vulnerabilidades de codificação

- **CVE-2005-3276:** A função *sys_get_thread_area* em *process.c* no Linux 2.6 antes da versão 2.6.12.4 e 2.6.13 não faz a limpeza de uma estrutura de dados antes de copiá-la para o espaço do utilizador, o que pode permitir que um processo do utilizador obtenha informações confidenciais.

Para a resolução deste problema, teve de se fazer uma atualização do sistema Linux.

- **CVE-2016-5537:** A vulnerabilidade na componente *NetBeans* no *Oracle Fusion Middleware 8.1* permitia aos utilizadores locais afetar a confidencialidade, integridade e disponibilidade através de vetores desconhecidos. Esta vulnerabilidade tem como função uma passagem no diretório que permitia que os utilizadores locais com determinadas permissões escrevessem em arquivos arbitrários e, conseqüentemente, adquirissem privilégios numa entrada de arquivos num ficheiro zip importado como um projeto.

A *Oracle* forneceu *patches* para a correção deste problema.

Na correção de vulnerabilidades de codificação a dificuldade pode estar associada à deteção de vulnerabilidade que, por vezes, pode ser difícil. A correção de um *bug* de *software* que esteja a induzir mais uma vulnerabilidade deste tipo poderá, normalmente, passar por um ou mais *patches*. No entanto, pode ser necessário introduzir mais vulnerabilidades ou pode simplesmente não ser possível a aplicação de um *patch* em certos sistemas.

1.7.3 Vulnerabilidades operacionais

- **CVE-2011-0521:** A função *dvb_ca_ioctl* em *drivers/media/dvb/ttpci/av7110_ca.c* no *kernel* Linux anterior ao 2.6.38-rc2 não verifica sinal de um determinado campo inteiro, o que permite que os utilizadores locais causem uma DoS (corrupção de memória) ou possivelmente ter outro impacto não especificado por um valor negativo.

A resolução consiste na aplicação do *patch* para esta vulnerabilidade que está disponível no *Linux Kernel GIT Repository*.

- **CVE-2018-1999036:** Existe uma exposição de vulnerabilidade a informações confidenciais no *Jenkins SSH Agent Plugin 1.15* e suas versões anteriores em *SSHAgentStepExecution.java* que expõe a *password* da chave privada SSH a utilizadores para ler o *log* de compilação. Para resolver este problema colocou-se o *plugin* a não registar mais a chamada *ssh-add* que revelava a *password*.

A correção de vulnerabilidades operacionais passa por haver uma política rigorosa de práticas a ter quando se lida com dados confidenciais, de forma a que entidades terceiras não lhes possam nunca aceder.

1.8 Pergunta P1.4

Uma vulnerabilidade de dia zero é uma vulnerabilidade em que não existe conhecimento público prévio da vulnerabilidade, o seu conhecimento é ignorado ou não simplesmente não existe forma de prever ou proteger o sistema dessa vulnerabilidade. Uma vulnerabilidade que não é de dia zero é uma vulnerabilidade que é do conhecimento público, mas da qual não foram tomadas as medidas de forma a prevenir um ataque. A grande diferença entre as duas é os domínios em que elas são conhecidas, enquanto que a de dia zero é completamente desconhecida ou conhecida por uma pessoa ou um grupo que a mantém desconhecida do público geral, uma vulnerabilidade que não é de dia zero é uma vulnerabilidade de conhecimento público.

1.9 Experiência 1.5 - Exploit Database

O *exploit* mais recentemente verificado denomina-se de **AIDA64 Engineer 6.20.5300 - 'Report File' filename Buffer Overflow (SEH)** que tem como função explorar o *buffer Report File* ao enviar um relatório de e-mail por meio do assistente de relatório. A inserção de uma sequência excessivamente longa resulta numa falha que substitui o SEH.

O **CVE-2019-6780** é relativo ao **Wordpress Plugin Wisechat 2.6.3 - Reverse Tabnabbing** e consiste na manipulação incorreta de *links* externos de um *plugin Wisechat* que seja anterior ao 2.7. Este *exploit* tem então como função enviar o URL **http://mtk911.cf/OR/** para esse *chat* e quando alguém carregasse nesse *link* seria redirecionado para um potencial *site* de *phishing* para uma obtenção de credenciais para os utilizadores.

1.10 Experiência 1.6 - Google Hacking Database

O *dork* mais recentemente verificado denomina-se de **site:*/membersarea in:title:"login"** e é uma *query* que retorna todas as páginas que possuam *login* tal que no URL contenham **/membersarea**.

Outro exemplo de *dork* poderia ser o **site:*/signup/password.php** que explora todas as páginas cujo URL termina em **/signup/password.php**. Por fim, mais um diferente dos anteriores seria o **filetype:reg reg [HKEY_CURRENT_USER\Software\] -git**. Tem como função procurar algumas *passwords* para o *WinVNC* e *plugins* FAR para o e-mail, FTP e o *ProxyFTP*.

1.11 Experiência 1.7 - Linguagem C

```
core@core-VirtualBox:~/Desktop/ES-C$ cat codigo.c
#include <stdio.h>

void main(){
    printf("Hello World\n");
}
```

Figura 1.19: Código do ficheiro ".c" criado para a experiência

```
core@core-VirtualBox:~/Desktop/ES-C$ cat codigo.s
.file "codigo.c"
.text
.section .rodata
.LC0:
.string "Hello World"
.text
.globl main
.type main, @function
main:
.LFB0:
.cfi_startproc
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
leaq .LC0(%rip), %rdi
call puts@PLT
nop
popq %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
.LFE0:
.size main, .-main
.ident "GCC: (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0"
.section .note.GNU-stack,"",@progbits
```

Figura 1.20: Código do ficheiro ".s" criado para a experiência

```

core@core-VirtualBox:~/Desktop/ES-CS$ diff -y codigo.c codigo.s
#include <stdio.h>                                     | .file "codigo.c"
                                                        | .text
                                                        | .section .rodata
void main(){                                           | .LC0:
    printf("Hello World\n");                           | .string "Hello World"
}                                                        | .text
>                                                        | .globl main
>                                                        | .type main, @function
> main:                                                 |
> .LFB0:                                                |
> .cfi_startproc                                       |
> pushq %rbp                                           |
> .cfi_def_cfa_offset 16                               |
> .cfi_offset 0, -16                                  |
> movq %rsp, %rbp                                     |
> .cfi_def_cfa_register 6                             |
> leaq .LC0(%rip), %rdi                               |
> call puts@PLT                                       |
> nop                                                  |
> popq %rbp                                           |
> .cfi_def_cfa 7, 8                                   |
> ret                                                  |
> .cfi_endproc                                         |
> .LFE0:                                                |
> .size main, .-main                                  |
> .ident "GCC: (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0"   |
> .section .note.GNU-stack,"",@progbits

```

Figura 1.21: Comparação dos dois ficheiros