

**Universidade Unicesumar**

# **Phelipe Kalinke Barreto**

**Desafio profissional parte 1 e 2**

**Curitiba,**

**2023**

# SUMÁRIO

<b>1- INTRODUÇÃO</b>	<b>3</b>
1.1.- OBJETIVOS	3
1.2.- METODOLOGIA	3
<b>2 - AMBIENTE VIRTUAL</b>	<b>4</b>
2.1 - INSTALAÇÃO E CONFIGURAÇÃO DO AMBIENTE VIRTUAL	4
2.2 - INSTALAÇÃO E CONFIGURAÇÃO DO LINUX NA MÁQUINA VIRTUAL	4
2.3 - INSTALAÇÃO E CONFIGURAÇÃO DO WEBGOAT	4
<b>3 - DESCRIÇÃO E FUNCIONALIDADES DO WEBGOAT:</b>	<b>5</b>
3.2 - COMO ACESSAR E NAVEGAR NO WEBGOAT	5
3.3 - PRÁTICAS COMUNS DE SEGURANÇA EM APLICAÇÕES WEB	5
<b>4 - IDENTIFICAÇÃO DE VULNERABILIDADES COMUNS EM APLICAÇÕES WEB -</b>	<b>6</b>
<b>5 - MELHORES PRÁTICAS PARA MITIGAÇÃO DE FRAGILIDADES EM APLICAÇÕES WEB</b>	<b>7</b>
<b>6 – CONCLUSÃO</b>	<b>8</b>
<b>7- TAREFA WEBGOAT</b>	<b>9</b>
<b>8- PROJETO DE DESENVOLVIMENTO</b>	<b>10</b>
<b>9- INJEÇÃO SQL</b>	<b>11</b>
<b>10- CONCLUSÃO DA PARTE 2</b>	<b>12</b>
<b>11- REFERÊNCIAS</b>	<b>13</b>

**1-Introdução:** Nesta seção, introduziremos o trabalho prático, fornecendo o contexto e a justificativa para a realização do mesmo. Também definiremos os objetivos e a metodologia utilizada. Contexto e justificativa do trabalho prático: Este trabalho prático foi realizado no contexto da disciplina de cibersegurança, com o objetivo de adquirir experiência prática no campo das vulnerabilidades em aplicações web, com ênfase em SQL Injection. A justificativa para essa atividade está no fato de que a cibersegurança é uma área em constante evolução e é fundamental que os profissionais estejam preparados para identificar e mitigar vulnerabilidades em sistemas e aplicações.

**1.1 - Objetivos:** O objetivo principal do trabalho prático foi criar, configurar e utilizar um ambiente virtual com o sistema operacional Kali Linux para explorar vulnerabilidades em um ambiente controlado. Mais especificamente, focamos no estudo da vulnerabilidade SQL Injection. O objetivo foi adquirir conhecimento prático sobre como identificar, explorar e mitigar essa vulnerabilidade em aplicações web.

**1.2 - Metodologia:** A metodologia utilizada envolveu a criação e configuração de um ambiente virtual com o sistema operacional Kali Linux. Para isso, utilizamos o software VirtualBox-64bits e o SO indicado nas instruções. Além disso, fizemos o download do Kali Linux por meio da ferramenta qBitTorrent. Para o estudo das vulnerabilidades, escolhemos o WebGoat como ferramenta de estudo, instalando-o e configurando-o dentro do ambiente virtual.

## **2 - Ambiente Virtual:**

**2.1 - Instalação e configuração do ambiente virtual:** A instalação e configuração do ambiente virtual foram realizadas seguindo as instruções fornecidas. Não enfrentamos dificuldades significativas nessa etapa.

**2.2 - Instalação e Configuração do Linux na Máquina Virtual:** Na instalação do Linux, encontramos algumas dificuldades. Em uma primeira tentativa, instalamos uma versão sem interface gráfica, o que nos levou a criar um novo ambiente virtual com interface gráfica. Também tivemos dificuldades na escolha da partição de disco, o que nos levou a criar um novo ambiente virtual e ajustar as configurações de instalação até obtermos sucesso. Porém, a instalação do JRE foi tranquila e seguimos as instruções fornecidas sem problemas.

**2.3 - Instalação e Configuração do WebGoat:** Enfrentamos dificuldades na instalação do WebGoat. O link fornecido inicialmente não funcionou corretamente, então procuramos por uma alternativa funcional, encontrando um link válido no endereço <https://github.com/WebGoat/WebGoat/releases/download/v2023.4/webgoat2023.4.jar>. Após utilizar esse link, a instalação e configuração foram bem-sucedidas. Vale ressaltar que houve uma alteração no comando fornecido nas instruções, substituindo "Java -jar webgoat-container-8.1.0-war-exec.jar" por "Java -jar webgoat2023.4.jar".

**3 - Descrição e Funcionalidades do WebGoat:** O **WebGoat** é um projeto de aplicativo web intencionalmente inseguro, desenvolvido para fins educacionais e de treinamento em segurança da web. Ele permite que desenvolvedores e profissionais de segurança explorem vulnerabilidades comuns encontradas em aplicações web. O WebGoat possui uma variedade de lições e exercícios que demonstram várias vulnerabilidades, incluindo SQL Injection, Crosssite Scripting (XSS) e ataques de força bruta.

**3.1 - Como acessar e navegar no webgoat:** O HackDev é um projeto de plataforma online propositadamente vulnerável, elaborado para propósitos didáticos e de capacitação em cibersegurança. Ele proporciona aos programadores e especialistas em segurança a oportunidade de investigar falhas comuns identificadas em aplicações web. O HackDev apresenta uma ampla gama de tutoriais e desafios que ilustram diversas vulnerabilidades, tais como Injeção de Código SQL, Cross-site Scripting (XSS) e ataques de tentativa e erro.

**3.2 - Práticas Comuns de Segurança em Aplicações Web:** Princípios Fundamentais das Aplicações Seguras: A garantia da segurança nas aplicações web é um elemento essencial na proteção de informações, usuários e sistemas contra ameaças e fragilidades. Existem alguns princípios básicos que são indispensáveis para assegurar a segurança nessas aplicações. A autenticação, por exemplo, é o processo de verificar a identidade do utilizador antes de lhe conceder acesso a recursos protegidos. Já a autorização regula as permissões e privilégios dos utilizadores autenticados. É crucial implementar métodos seguros de autenticação e autorização, como senhas robustas, autenticação de dois fatores e uma gestão adequada de sessões. Outro conceito de extrema importância é a proteção contra ataques de injeção, tais como a injeção de SQL e de código. É recomendável o uso de consultas parametrizadas ou declarações preparadas, evitando assim a concatenação direta de dados de entrada em consultas ou comandos. Adicionalmente, é necessário proteger-se contra ataques de Cross-Site Scripting (XSS) e Cross-Site Request Forgery (CSRF), através da aplicação de filtros de entrada, escapamento de saída e de tokens CSRF. A gestão de erros e exceções

também desempenha um papel importante na segurança. É imprescindível evitar a exposição de informações detalhadas sobre erros ao utilizador final, implementando um tratamento adequado dos mesmos e registando-os de forma segura. Manter o software atualizado, aplicar patches de segurança e utilizar criptografia para proteger a comunicação e dados sensíveis são práticas essenciais. Além disso, é recomendável realizar testes de segurança, como testes de penetração e varreduras de vulnerabilidades, com o objetivo de identificar possíveis brechas e corrigi-las antes que sejam exploradas.

**4 - Identificação de Vulnerabilidades Comuns em Aplicações Web:** A detecção de fragilidades comuns em aplicações web é de suma importância para assegurar a segurança dos sistemas. Dentre as fragilidades mais comumente encontradas, destacam-se: injeção de SQL, em que dados não confiáveis são inseridos incorretamente em consultas SQL; Scripting entre Sites (XSS), que possibilita a inserção de scripts maliciosos em páginas web; Falsificação de Solicitação entre Sites (CSRF), um tipo de ataque que engana o navegador do utilizador para executar ações indesejadas em um site; exposição de dados confidenciais, quando informações sensíveis são armazenadas ou transmitidas de forma insegura; autenticação fraca, envolvendo senhas de baixa segurança ou autenticação não confiável; gestão inadequada de sessões, que pode resultar em ataques de sequestro de sessão; falta de validação de entrada, permitindo a execução de códigos maliciosos; configuração inadequada do servidor, expondo informações sensíveis; inclusão de arquivos não confiáveis, abrindo espaço para a inserção de códigos maliciosos; e falta de controlo de acesso, permitindo acesso não autorizado a informações ou funcionalidades restritas.

**5 - Melhores Práticas para Mitigação de Fragilidades em Aplicações Web:** Para mitigar fragilidades em aplicações web, é fundamental adotar melhores práticas de segurança. Entre as recomendações mais relevantes estão: manter o software atualizado, incluindo sistema operativo, servidores web, frameworks e bibliotecas utilizadas; aplicar o princípio do "privilégio mínimo" para restringir o acesso a recursos sensíveis; validar e filtrar com rigor a entrada de dados para evitar injeção de código malicioso; utilizar criptografia para proteger informações confidenciais e garantir a segurança da comunicação; implementar autenticação segura, com senhas robustas e autenticação em dois fatores; aplicar um controle de acesso granular para que cada utilizador tenha acesso apenas ao necessário; gerir corretamente as sessões, utilizando tokens de sessão seguros e encerrando-as de forma apropriada; realizar testes de segurança regulares, como testes de penetração e varreduras de fragilidades; proteger-se contra ataques de CSRF com mecanismos de proteção; e manter registos de atividades e monitorizar a aplicação para detetar tentativas de intrusão e comportamentos suspeitos. Estas melhores práticas contribuem para fortalecer a segurança das aplicações web e reduzir o risco de exploração de fragilidades

**6 - Conclusão:** Neste trabalho prático, foi adquirida uma valiosa experiência prática no campo das vulnerabilidades em aplicações web, com foco na vulnerabilidade SQL Injection. A criação e configuração do ambiente virtual com o sistema operacional Kali Linux e a utilização do WebGoat como ferramenta de estudo foram fundamentais para explorar e compreender as vulnerabilidades comuns encontradas nesse tipo de aplicação. Durante o processo de instalação e configuração do ambiente virtual, foram encontrados alguns desafios. No entanto, com perseverança e dedicação, foram superados e os objetivos foram alcançados com sucesso. A instalação do WebGoat também apresentou suas próprias dificuldades, mas alternativas funcionais foram buscadas e a etapa foi concluída com êxito. Ao longo do trabalho, foram explorados os conceitos básicos de segurança em aplicações web, compreendendo a importância da autenticação, autorização, criptografia, gerenciamento de sessões e validação de entrada. Foram identificadas as vulnerabilidades comuns, como SQL Injection, Cross-Site Scripting (XSS), CrossSite Request Forgery (CSRF) e outras, e foram aprendidas as boas práticas para mitigar essas vulnerabilidades. Foi reconhecida a importância de manter o software atualizado, aplicar o princípio do "privilegio mínimo" para restringir o acesso a recursos sensíveis, validar e filtrar rigorosamente a entrada de dados, utilizar criptografia para proteger informações confidenciais e garantir a segurança da comunicação, implementar autenticação segura com senhas robustas e autenticação em dois fatores, aplicar um controle de acesso granular, gerir corretamente as sessões e realizar testes de segurança regulares. Este trabalho prático proporcionou uma base sólida para compreender as vulnerabilidades em aplicações web e as práticas de segurança necessárias para mitigá-las. O aprendizado adquirido nessa experiência contribuirá para enfrentar os desafios da cibersegurança e para aprimorar a proteção de sistemas e aplicações contra ameaças e fragilidades



## PARTE 2

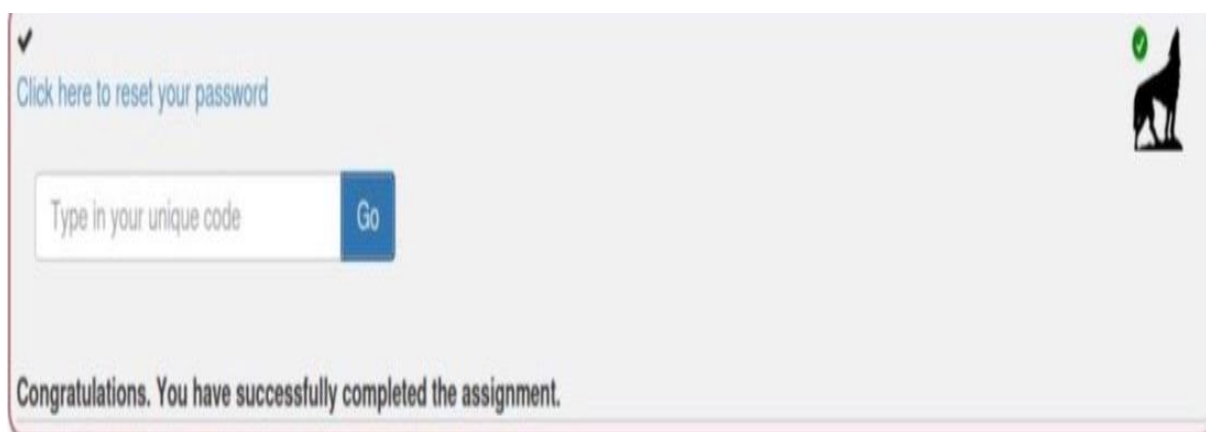
### 7- TAREFA WEBGOAT:

1-Utilizando uma caixa de correio eletrônico exclusiva do WebGoat, fiz o registro de nome para um endereço de e-mail fictício e foi enviado um código exclusivo, que, nesse caso, foi composto pelo nome de usuário invertido.



The screenshot shows a WebGoat interface for email registration. At the top left is a checkmark icon. At the top right is a wolf head icon with a green checkmark. Below the wolf icon is a text input field containing '@ Phelipekb@hotmail.com'. Below this is a blue button labeled 'Send e-mail'. Further down is another text input field labeled 'Type in your unique code' followed by a blue button labeled 'Go'. At the bottom, a message reads: 'Congratulations. You have successfully completed the assignment.'

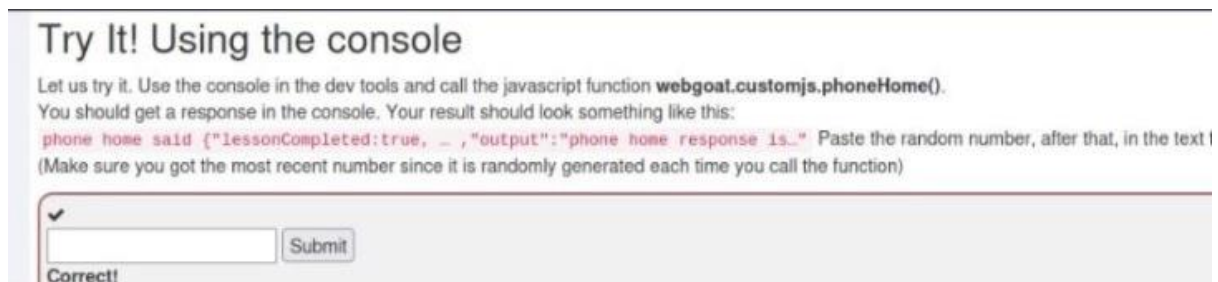
2- O objetivo principal dessa atividade abaixo é ilustrar o funcionamento de links falsos de redefinição de senha. O hyperlink presente na tarefa redirecionava para uma página de alteração de senha, mas, ao realizar a alteração, não ocorria nada além de expor a senha na barra de pesquisa.



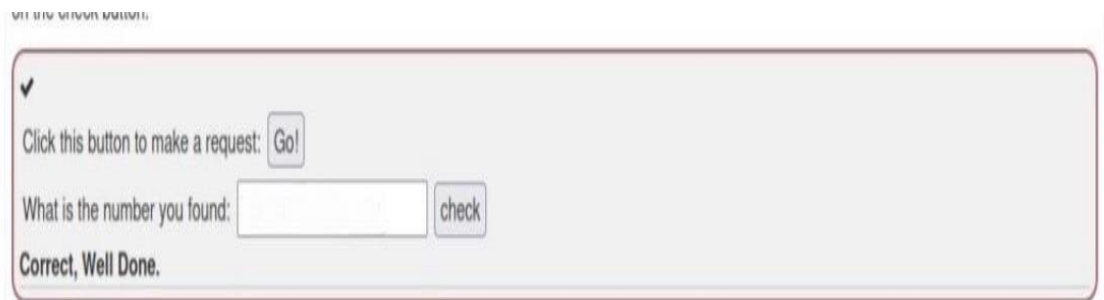
The screenshot shows a WebGoat interface for password reset. At the top left is a checkmark icon. At the top right is a wolf head icon with a green checkmark. Below the wolf icon is a blue hyperlink that reads 'Click here to reset your password'. Below this is a text input field labeled 'Type in your unique code' followed by a blue button labeled 'Go'. At the bottom, a message reads: 'Congratulations. You have successfully completed the assignment.'

## 8- Projeto de desenvolvedor:

1- Ao executar o comando JavaScript 'webgoat.customjs.phonehome' no console, conseguimos obter com sucesso o número de telefone fictício, que era o propósito principal dessa atividade.



2- Utilizando a funcionalidade de inspeção do console, com o propósito de obter o número aleatório, foi preciso clicar no botão 'GO' e examinar as solicitações na seção 'NETWORK' da inspeção. Dessa forma, conseguimos com sucesso capturar o número mencionado na imagem.



## 9- INJEÇÃO SQL:

- 1- O objetivo da atividade era obter uma informação específica da tabela de funcionários, e para isso foi utilizado o comando "SELECT \* FROM employees".

It is your turn!

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

✓

SQL query

Submit

You have succeeded!

- 2- propósito da atividade era identificar a sintaxe correta da solicitação, a fim de obter informações específicas do banco de dados.

### Try It! String SQL injection

The query in the code builds a dynamic query as seen in the previous example. The query is built by concatenating strings making it susceptible to String SQL injection:

```
"SELECT * FROM user_data WHERE first_name = 'John' AND last_name = '' + lastName + ''";
```

Try using the form below to retrieve all the users from the users table. You should not need to know any specific user name to get the complete list.

✓

SELECT \* FROM user\_data WHERE first\_name = 'John' AND last\_name = 'Smith' or 1 = 1 Get Account Info

You have succeeded:

USERID, FIRST\_NAME, LAST\_NAME, CC\_NUMBER, CC\_TYPE, COOKIE, LOGIN\_COUNT,

101	Joe	Snow	987654321	VISA	,	0
101	Joe	Snow	2234200065411	MC	,	0
102	John	Smith	2435600002222	MC	,	0
102	John	Smith	4352209902222	AMEX	,	0
103	Jane	Plane	123456789	MC	,	0
103	Jane	Plane	333498703333	AMEX	,	0
10312	Jolly	Hershey	176896789	MC	,	0
10312	Jolly	Hershey	33300003333	AMEX	,	0
10323	Grumpy	youaretheweakestlink	673834489	MC	,	0
10323	Grumpy	youaretheweakestlink	33413003333	AMEX	,	0
15603	Peter	Sand	123609789	MC	,	0
15603	Peter	Sand	338893453333	AMEX	,	0
15613	Joseph	Something	33843453533	AMEX	,	0
15837	Chaos	Monkey	32849386533	CM	,	0

- 3- O objetivo da atividade proposta consistia em obter os dados de uma tabela por meio do comando especificado, utilizando os valores substituídos de "login Count"

✓

Employee Name:

Authentication TAN:

Get department

You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!

USERID FIRST\_NAME LAST\_NAME DEPARTMENT SALARY AUTH\_TAN PHONE COLLUMPHONE

- 4- O objetivo era remover a tabela access\_log, e para isso foi utilizado o comando ';' drop table access\_log'. (FUNÇÃO DE BANCO DE DADOS).

It is your turn!

Now you are the top earner in your company. But do you see that? There seems to be a `access_log` table, where all your actions have been logged to! Better go and delete it completely before anyone notices.

✓

Action contains:

Search logs

Success! You successfully deleted the access\_log table and that way compromised the availability of the data.

**10 - Conclusão:** Resumo dos Resultados e Considerações Finais da Prática: Após todo o procedimento de instalação e configuração, com ênfase na execução de algumas tarefas relacionadas à vulnerabilidade de Injeção de SQL, foram realizadas cinco atividades de injeção no ambiente virtual usando o WebGoat. Essas atividades incluíam a extração de informações das tabelas existentes, alteração de dados, concessão e remoção de permissões para usuários não autorizados no caso de concessão, e para usuários que já possuíam as permissões no caso de remoção dessas permissões.

Limitações do Trabalho e Sugestões para Trabalhos Futuros: Como mencionado durante o desenvolvimento do documento, enfrentamos dificuldades na instalação do sistema operacional e na configuração do mesmo, além da dificuldade em baixar e acessar o WebGoat, seja por ter que procurar um novo link funcional ou ao executar comandos no terminal. A atividade de Proxies HTTP mencionada na introdução apresentou várias complicações, tanto na máquina virtual quanto na máquina física, com superaquecimento acima do normal e travamento de tela. Devido a esses problemas, não foi possível realizar essa atividade. Quanto à atividade de Criação de Nova Lição, também mencionada na introdução, ela envolve um nível de complexidade elevado, além das habilidades atualmente possuídas pelo autor do trabalho. Levando isso em consideração, optou-se por não realizar essa atividade.

## 11- REFERÊNCIAS

OWASP. SQL Injection. Disponível em: [https://owasp.org/www-community/attacks/SQL\\_Injection](https://owasp.org/www-community/attacks/SQL_Injection). Acesso em: 19 de maio de 2023.

OWASP. Cross-Site Scripting (XSS). Disponível em: <https://owasp.org/www-community/attacks/xss/>. Acesso em: 19 de maio de 2023.

<https://www.prppg.ufpr.br/siga/visitante/trabalhoConclusaoWS?idpessoal=549>

01&idprograma=40001016034P5&anobase=2019&idtc=1442

[file:///C:/Users/User/Downloads/17348-853-13940-1-10-20210928%20\(1\).pdf](file:///C:/Users/User/Downloads/17348-853-13940-1-10-20210928%20(1).pdf)

<https://owasp.org/www-community/attacks/xss/>"><https://owasp.org/www-community/attacks/xss/> <https://owasp.org/www-community/attacks/csrf>"><https://owasp.org/www-community/attacks/csrf>

<https://owasp.org/www-community/> <https://owasp.org/www-project-top-ten/>

<https://docs.microsoft.com/en-us/sql/relational->