DOI: 10.5748/9788599693148-15CONTECSI/PS-5784

# DETECTION OF DATA VULNERABILITIES IN WEB USING SQL INJECTION

Alexandre Bonadiman Angeli, ORCID 0000-0002-7992-3195, (Faculdade Estácio de Sá de Vitória, Espírito Santo – Brasil) - alexandre.angeli@outlook.com
Wellington Sousa Aguiar, ORCID 0000-0003-0677-5782, (Centro Universitário Estácio do Ceará, Ceará – Brasil) - wellington@tecsist.com
José Mário Bezerril Fontenelle, ORCID 0000-0001-6828-0123, (Centro Universitário Estácio do Ceará, Ceará – Brasil) - jomabefon@gmail.com
Francisco Antônio de Araujo e Souza, ORCID 0000-0002-5555-0419, (Centro Universitário Estácio do Ceará, Ceará – Brasil) - prof.faraujo@gmail.com

Modern life determines the necessity to insert personal and financial information in many websites, corporate systems and social networks, without the proper safety of these data. This study has as objective to present the use of technological tools to evaluate the data vulnerability in web systems and to alert the institutions about the risks to which they are exposed. The research method used was the bibliographical, constituted of book researches, scientific articles, theses and dissertations. Field research, conducting tests with 50 (fifty) financial institutions from different Brazilian states, of which 10 are for each selected market segment and applied research, when the research generates immediate and useful results for the community. Through the results obtained, we can verify the importance of carrying out constant maintenance and tests in the systems to guarantee data security. Of the fifty (50) institutions surveyed, 22% had SQL Injection failures, that being, it was possible to access, view and extract confidential data from these companies, such as logins, passwords, registrations, names, telephones, e-mails and so on. The educational segment presented the highest vulnerability index, 40% of educational institutions, 30% of city halls and colleges, 10% of Health Plan Operators, but the surveyed banking institutions were all protected. We can conclude that there are still many flaws in the digital security of companies, allowing the attack by malicious people. In order to mitigate such risks, a support material was provided to correct the encountered flaws.

Keywords: SQL Injection, SQL MAP, Database.

# DETECÇÃO DE VUNELRABILIDADES DE DADOS NA WEB USANDO SQL INJECTION

A vida moderna determina a necessidade de inserir informações pessoais e financeiras em vários sites da web, sistemas corporativos e redes sociais, sem a devida garantia de segurança destes dados. Este estudo tem por objetivo apresentar o uso de ferramentas tecnológicas para avaliar a vulnerabilidade de dados em sistemas web e alertar as instituições sobre os riscos a que estão expostos. O método de pesquisa utilizado foi o bibliográfico, constituído por pesquisas em livros, artigos científicos, teses e dissertações. Pesquisa de campo, realizando testes com 50 (cinquenta) instituições de diferentes estados brasileiros, sendo 10 para cada segmento de mercado selecionado e pesquisa aplicada, quando a pesquisa gera resultados imediatos e úteis para a comunidade. Através dos resultados obtidos, podemos verificar a importância da realização de manutenções e testes constantes nos sistemas para garantir a segurança dos dados. Das 50 (cinquenta) instituições pesquisadas, 22% possuíam falhas de SQL Injection, ou seja, foi possível acessar, visualizar e extrair dados confidenciais destas empresas, como logins, senhas, matrículas, nomes, telefones, e-mails e etc. O segmento educacional apresentou o maior

índice de vulnerabilidade, 40% das instituições de ensino, 30% das prefeituras e faculdades, 10% das Operadoras de Planos de Saúde, mas as instituições bancárias pesquisadas estavam todas protegidas. Podemos concluir que ainda existem muitas falhas na segurança digital das empresas, possibilitando o ataque por pessoas maliciosas. Visando mitigar tais riscos, foi fornecido um material de apoio para correção das falhas encontradas.

Palavras-chaves: SQL Injection, SQL MAP, Banco de dados.

## INTRODUÇÃO

Estamos em um momento conhecido como a "era da informação", tudo está conectado, realizamos transações bancárias em nossos telefones, pagamos conta de luz, água, cartão de crédito sem a necessidade de sair de casa, possibilitando buscar dados como: número do documento do carro, conta de telefone e etc. Com todas estas facilidades o número de informações que estão sendo lançadas na rede mundial é gigantesco, crescendo a cada dia, gerando assim, um volume incalculável.

De acordo com Taurion (2013):

> "[...] geramos *petabytes* de dados a cada dia. E estima-se que este volume dobre a cada dezoito meses. Variedade também, pois estes dados vêm de sistemas estruturados (hoje já são minoria) e não estruturados (a imensa maioria), gerados por e-mails, mídias sociais (Facebook, Twitter, YouTube, e outros, documentos eletrônicos, apresentações estilo Powerpoint, mensagens instantâneas, sensores, etiquetas RFID, câmeras de vídeo etc. [...]"

Todas essas informações facilitam a vida das pessoas, podemos gerenciar melhor o tempo e aproveitá-lo com melhor qualidade, porém essa quantidade toda "rodando" pela internet, se torna algo muito sensível, já que tudo está na internet, desde um simples comentário na rede social até nossas transações bancárias, CPF, telefone, endereço, informações estas que em mãos erradas, podem se tornar algo muito perigoso e prejudicial. Portanto, todo sistema que retém dados pessoais tem por obrigação manter um nível de proteção para que os mesmos não sejam capturados por pessoas não autorizadas, garantindo assim, um dos pilares da segurança da informação que é a integridade.

Sabendo que hoje essas informações estão na internet, abriu-se o leque para um novo método de fraude, pessoas com conhecimentos elevados em computação, normalmente entusiastas da área de segurança, utilizam-se de seus conhecimentos para buscar uma maneira de capturar os dados, podendo ser senhas para uso em bancos, redes sociais, endereço ou qualquer dado que possa vir a ser utilizado para benefício próprio, essas pessoas são conhecidas como crackers.

De acordo com Galvão (2015):

> "A palavra cracker vem do termo *cracking*, que significa quebra. Os crackers são indivíduos que praticam a quebra de segurança de um sistema, ou seja, cometem delitos ou crimes digitais. Como os crackers cometem atos ilegais, são vistos como criminosos[...]"

Instituições sofrem milhares de ataques diariamente, e se as mesmas não se atentarem em pequenos detalhes, podem comprometer sua segurança. Infelizmente não podemos garantir que um sistema é 100% seguro, devido a isso, constantemente são divulgadas

notícias de grandes empresas que sofreram com ataques *crackers* e tiveram seus dados vazados. A exemplo, podemos mencionar o caso que foi publicado pelo site www.exame.abril.com.br, onde cita o ocorrido com a empresa Adobe:

> "O ataque aos servidores da Adobe em 2013 resultou no vazamento de dados de pelo menos 38 milhões de usuários de programas da companhia. Entre as informações, estavam nomes de usuários, senhas e, em alguns casos, números de cartões de crédito[...]"

Essa pesquisa tem por objetivo apresentar uma técnica chamada SQL Injection, utilizando a ferramenta SQLMAP no sistema operacional Kali Linux, buscando entender as vulnerabilidades e como solucioná-las, para que empresas possam aplicar as correções necessárias para estas falhas de segurança, mitigando as chances de possíveis invasões.

Este estudo se faz necessário e relevante porque se aplicado uma técnica de injeção de comandos SQL num sistema vulnerável a esta técnica, tal aplicação permitirá que o atacante tenha acesso ao banco de dados da vítima, podendo visualizar as informações que estão contidos na base, realizar a extração, alteração ou até mesmo exclusão dos dados. Em muitos casos, podendo causar grandes prejuízos às instituições, além do perigo em se ter os dados, muitas vezes pessoais, expostos na internet para que qualquer pessoa tenha acesso privilegiado. Os testes foram realizados em sites de segmentos variados, como prefeituras, planos de saúde, escolas de ensino fundamental, médio e técnico, bancos e faculdades. A seleção de tais segmentos teve por critério o alto volume de informações que estão retidos nestes setores, como de clientes, funcionários, professores, alunos e fornecedores.

## REFERENCIAL TEÓRICO

### BANCO DE DADOS

Banco de dados é um conjunto de arquivos que estão relacionados entre si, podendo ser dados de pessoas, lugares, empresas ou qualquer outra coisa. É uma coligação organizada dos dados que, quando relacionados, criam um sentido, gerando assim uma informação, com o objetivo de dar mais eficiência durante uma pesquisa ou estudo. Atualmente, bancos de dados são considerados grandes ativos de uma empresa, ou seja, tem característica de extrema importância para instituições e até para algumas pessoas, se tornando então, uma peça fundamental de qualquer sistema.

De acordo com Puga (2014):

> "Um banco de dados é uma coleção de dados armazenados e organizados de modo a atender as necessidades integradas dos seus usuários. Possibilita a consulta e a manipulação dos dados, podendo ser manual ou computadorizado[...]"

Os bancos de dados são operados pelos Sistemas Gerenciadores de Banco de Dados (SGBD) que são responsáveis pelo gerenciamento da base, como por exemplo o controle de acesso, manipulação e organização dos dados, desta forma podemos definir SGBD como uma biblioteca de softwares para o gerenciamento de banco de dados. Antes da utilização dos SGBD's a forma de armazenamento era através de arquivos, o que implicava em vários problemas, principalmente relacionado a redundância de dados, causando assim, conflitos entre as informações.

### SQL

O nome SQL é derivado de *Structure Query Language* (Linguagem Estruturada de Consulta), sendo projetada na IBM Research como uma interface para um sistema de banco de dados relacional chamado SISTEMA R. Desde então, a SQL se tornou a linguagem padrão para os SGBD's comerciais.

SQL é uma linguagem de pesquisa declarativa, utilizada em banco de dados, na qual muitas de suas características são inspiradas na álgebra relacional, com ela podemos realizar consultas, inserir dados ou realizar exclusões, além de muitas outras funções que podemos executar num banco.

A linguagem é dividida em subconjuntos de acordo com as operações que queremos executar sobre o banco de dados, essas subdivisões são definidas como:

**DML – Data Manipulation Language ou Linguagem de Manipulação de Dados**

Este é o primeiro grupo, DML. Esse subconjunto da linguagem SQL é utilizado para realizar as operações de inclusão, alterações e exclusões de dados que estão presentes em um registro, utilizando os comandos INSERT (inclusão), UPDATE (alteração) e DELETE (exclusão), são comandos que interagem diretamente com os dados dentro da tabela.

Exemplos de comandos DML:
Sintaxe: INSERT INTO nome_tabela VALUES (lista_dados);
INSERT INTO ALUNOS (nome, telefone) VALUES ("José da Silva", "(11) 99999-9999");

Sintaxe: UPDATE nome_tabela SET CAMPO = "novo_valor" WHERE CONDIÇÃO;
UPDATE ALUNOS SET nome = "José da Silva" WHERE telefone "(21) 99999-9999";

Sintaxe: DELETE FROM nome_tabela WHERE condição
DELETE FROM ALUNOS WHERE nome = "José da Silva";

**DDL – Data Definition Language ou Linguagem de Definição de Dados**

Segundo grupo é a DDL, apesar do nome esses comandos não interagem com os dados e sim com os objetivos do banco, neste grupo são utilizados os comandos CREATE (criar), ALTER (alterar) e DROP (exclusão de tabela), vale ressaltar que os comandos DROP e DELETE mesmo possuindo a mesma intenção que é a de exclusão, existe diferenças entre eles, o comando DROP remove a tabela do banco de dados, excluindo todas as linhas, privilégios e índices, e o DELETE remove somente a linha especificada junto ao comando de uma tabela.

Exemplos de comandos DDL:
Sintaxe: ALTER TABLE nome_tabela ADD nome_coluna tipo_do_dado;
ALTER TABLE ALUNOS ADD cidade varchar(50);
Sintaxe: CREATE TABLE nome_tabela; CREATE TABLE escola;
Sintaxe: DROP TABLE "nome_tabela"; DROP TABLE "escola";

**DQL – Data Query Language ou Linguagem de Consulta de Dados**

Este terceiro grupo contém somente um único comando, SELECT (selecionar), a função utilizada para buscar um determinado dado ou um conjunto de dados de dentro de uma base. Em algumas literaturas, o comando SELECT é inserido junto aos comandos do

conjunto DML, enquanto em outros livros, é criado um conjunto à parte (DQL) para este comando.

Exemplos de comandos DQL:
Sintaxe: SELECT coluna, coluna ..., coluna FROM TABELA;
SELECT nome, telefone FROM ALUNOS;

**KALI LINUX**

Kali Linux é uma distribuição GNU/Linux baseado no Debian, ele é o sucesso do antigo Back Track. O novo sistema apresentou várias melhorias, além de possuir mais aplicativos para Pentest. O Kali tem por principal objetivo a auditoria e segurança de computadores em geral. Desenvolvido pela Offensive Security Ltd desde 21 de janeiro de 2016.

Vários softwares conhecimentos já trazem instalados no sistema operacional, incluindo o Nmap (escâner de portas), Wireshark (analisador de tráfego), John Ripper (Crackeador de senhas), Aircrack-ng (programa para testes em segurança de redes wireless) e SQLMAP (software apresentado no artigo para análise de SQL Injection), além de muitas outras ferramentas para todos os segmentos de segurança computacional, atualmente contando com um arsenal de mais de 300 (trezentas) ferramentas.

**SQL INJECTION**

De acordo com CLARKE (2012):

> "A injeção de SQL é uma das vulnerabilidades mais devastadoras que afetam os negócios, pois pode levar à exposição de todas as informações confidenciais armazenadas em uma aplicação de banco de dados, incluindo informações úteis, como login de usuário, senhas, nomes, endereços, telefone e detalhes do cartão de crédito[...]"

O SQL Injection é o nome que se dá quando ocorre falha na programação de um sistema, podendo ser web ou desktop (local), que possibilita por meio de uma inserção de dados, manipular as consultas do banco de dados. Esse processo de inserir comandos para manipular a base é chamado de injeção, por isso o nome da técnica SQL Injection ou Injeção de SQL.

Essa técnica é um modelo de ataque baseado na manipulação dos códigos SQL sem a necessidade de ser um usuário cadastrado do banco, permitindo realizar consultas, alterações, inclusões e até mesmo exclusões de informações presentes na base de dados.

De acordo com o Lacking Faces:

> "pense em SQL Injection como uma simples falha lógica, é simplesmente uma falha que, por deixar aberta a interpretações, ocorrem manipulações indesejáveis".

Os softwares estão cada vez mais estão sendo disponibilizados na Web, a maioria são aplicações dinâmicas, em que o acesso a base de dados se torna necessário. Muitas vezes, por falta de conhecimento do desenvolvedor o sistema se torna um alvo fácil de ataques. O exemplo abaixo demonstra uma simples conexão via SQL entre a aplicação e a base de dados.

SELECT * FROM usuarios WHERE login = 'Maria' AND senha = '123456'

Nesse modelo apresentado, retorna uma confirmação se o usuário “Maria” possui a senha ‘123456’. Se essa informação for verdadeira o DB (DataBase) irá localizar essa informação no banco e irá retornar à coleção de dados.

Devido a falha no desenvolvimento, não existe tratamento para inserção de caracteres especiais, desta forma, permitindo a entrada de dados como por exemplo “Mar’ia”, essa entrada, irá retornar um erro na base, pois não irá conseguir interpretar o comando que foi solicitado, retornando um erro.

Como a entrada foi inválida, também poderia ser válida, e a execução causar um transtorno enorme, como por exemplo:

SELECT * FROM usuarios WHERE login = '123' AND senha = ' ' or '1' = '1'

É observado que independente do login e senha digitados a condição sempre irá retornar verdadeiro (‘1’ = ‘1’), permitindo assim, o acesso ao sistema mesmo sem conhecimento de usuário e senha.

**SQLMAP**

SQLMAP é uma ferramenta Open Source desenvolvida em Python (Linguagem de programação) para realizar testes de penetração, foi desenvolvido por Bernardo Damele e Miroslav Stampar, utilizado para automatizar o processo de detecção e exploração de falhas de vulnerabilidade de SQL Injection, muito utilizado por profissionais na área de segurança para verificação de seus sistemas ou de seus clientes.

A ferramenta SQLMAP quando executada com sucesso, permite assumir total controle dos bancos de dados, possuindo um motor de detecção que tem por função empregar as mais atuais e devastadoras técnicas de injeção de SQL.

A tabela abaixo mostra as características do SQLMAP. Todas estas informações citadas abaixo estão presentes no site oficial da ferramenta.

| |
|---|
| Suporte total para sistemas de gerenciamento de **banco de** dados **MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, HSQLDB e Informix** . |
| Suporte total para seis técnicas de injeção de SQL: **blindado baseado em booleano, baseado em tempo, baseado em erro, baseado em consulta UNION, consultas empilhadas e fora de banda** . |
| Suporte para **se conectar diretamente ao banco de dados** sem passar por uma injeção SQL, fornecendo credenciais DBMS, endereço IP, porta e nome do banco de dados. |
| Suporte para enumerar **usuários, hashes de senha, privilégios, funções, bancos de dados, tabelas e colunas** . |
| Reconhecimento automático de formatos de hash de senha e suporte para **cracking usando um ataque baseado em dicionário** . |
| Suporte para **despejar** completamente as **tabelas de banco de dados** , um intervalo de entradas ou colunas específicas, conforme a escolha do usuário. O usuário também pode optar por despejar apenas um intervalo de caracteres da entrada de cada coluna. |
| Suporte para **procurar nomes específicos de banco de dados, tabelas específicas em todos os bancos de dados ou colunas específicas em todas as tabelas de bancos de dados** . Isso é útil, por exemplo, para identificar tabelas contendo credenciais de aplicativos personalizadas, onde nomes de colunas relevantes contêm uma string como nome e passagem. |

| |
|---|
| Suporte para **baixar e carregar qualquer arquivo** do sistema de arquivos subjacente ao servidor de banco de dados quando o software de banco de dados é MySQL, PostgreSQL ou Microsoft SQL Server. |
| Suporte para **executar comandos arbitrários e recuperar sua saída padrão** no sistema operacional subjacente do servidor de banco de dados quando o software de banco de dados é MySQL, PostgreSQL ou Microsoft SQL Server. |
| Suporte para **estabelecer uma conexão TCP com estado fora da banda entre a máquina atacante e o servidor de banco de dados** subjacente ao sistema operacional. Este canal pode ser um prompt de comando interativo, uma sessão Meterpreter ou uma sessão de interface gráfica do usuário (VNC) conforme a escolha do usuário. |
| Suporte para a **escalação de privilégios de usuário do processo de banco** de **dados** através do getsystemcomando Meterreter do Metasploit . |

**Tabela 2:** Características do SQLMAP
**Fonte:** Autoria própria

## METODOLOGIA

Esta pesquisa de campo utilizou várias abordagens, bibliográfica, quantitativa e aplicada. O problema estudado com base na bibliografia disponível nos conduziu a aplicação de uma técnica que gera resultados aplicáveis e valores estatísticos. A utilização desta abordagem é adequada quando a pesquisa visa gerar benefícios concretos para a comunidade em que estamos inseridos.

A pesquisa aplicada é motivada pela necessidade de resolver problemas concretos, mais imediatos ou não (VERGARA, 2007).

Existem outros meios de acesso ao saber que dispensam o uso de processos científicos, embora sejam válidos. Dois desses meios, aliás muito recomendáveis, são a consulta bibliográfica e a consulta documental, que se caracterizam por dirimir pequenas dúvidas, recorrendo a bibliografia específica ou a documentos, respectivamente (CERVO, 2007).

Para elaboração desta pesquisa foi utilizado o Sistema Operacional Kali Linux, pois o mesmo acompanha ferramentas específicas para o Pentest, sendo um dos mais utilizados para realização de testes de vulnerabilidade em sistemas web, desktop, redes wireless e vários outros ativos que possam sofrer ataques.

No Kali Linux contamos com várias ferramentas para o desenvolvimento desta pesquisa, foi utilizada a ferramenta SQLMAP, software que automatiza o processo de verificação da falha de SQL Injection.

Para compreensão da metodologia aplicada, será demonstrado o modelo de coleta e análise de falha proposta em uma das instituições selecionadas, um Colégio. Utilizando a ferramenta do Google foi possível buscar informações de colégios que poderiam sofrer com uma injeção de comandos SQL em suas páginas, para isso foi realizado uma busca avançada baseada em parâmetros que o próprio Google fornece para buscas e com isso foi possível levantar suspeitas de possíveis vulnerabilidades. A sintaxe utilizada na busca, foi a seguinte:

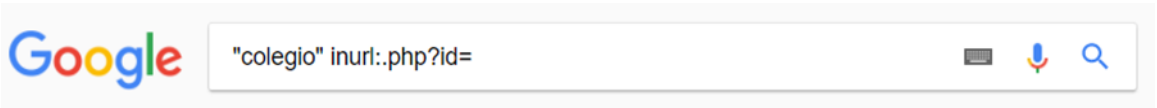

**Figura 1:** Parâmetros utilizados para retorno de uma busca avançada.
**Fonte:** Pesquisa Google.

Através desta pesquisa, o Google retorna sites que suam URL que recebem valores, sendo este um indício de possível falha na segurança.

Ao utilizar " " (aspas) o Google retorna uma palavra específica ou um conjunto de palavras, como apresentado no exemplo, ao utilizar "colégio", a ferramenta está retornando todos os sites que possuem a palavra exata "colégio", este foi o primeiro ponto da pesquisa. Para complementá-la, foi utilizado o parâmetro "inurl", filtrando mais ainda a pesquisa já que ao aplicar este código o Google retorna uma string específica que esteja contida dentro da URL, não mais em toda a página, somente no conteúdo onde foi definido o site.

Aplicando então o texto: ".php?id=" foi possível obter sites que tinham uma solicitação de receber o valor de um "id", este nome, normalmente sendo aplicado a banco de dados, foi utilizado o termo "id" pois é comum que o mesmo esteja contido por padrão em todas as bases. Aplicando esta consulta, obtivemos o seguinte resultado conforme Figura 2, abaixo:

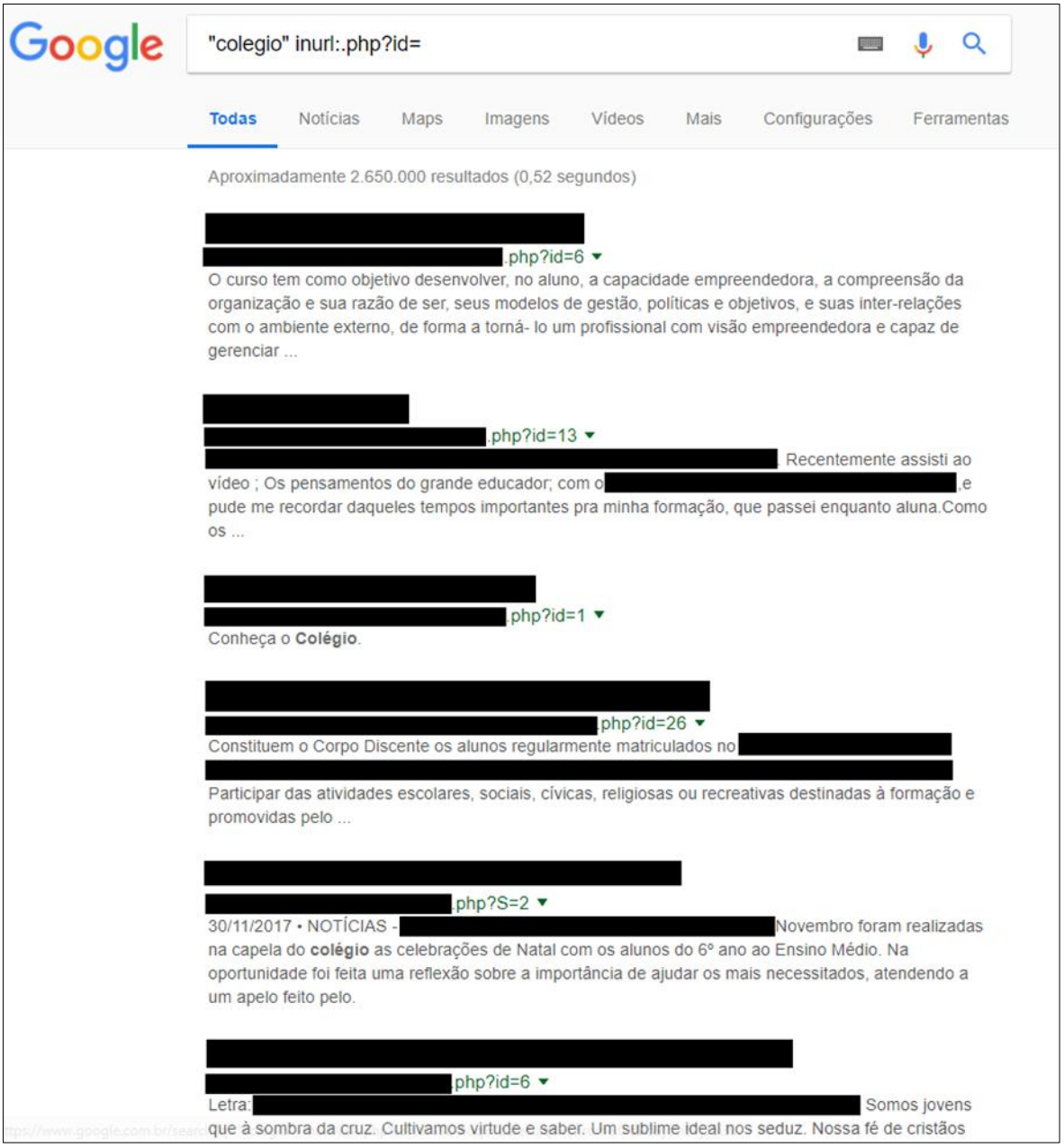


**Figura 2:** Lista completa da pesquisa retornada pelo Google.
**Fonte:** Google.

Por questão de privacidade, os nomes dos colégios foram ocultados, porém, pode-se observar que as URL's de todos possuem algo em comum, que é o retorno exatamente como solicitado através dos parâmetros de pesquisa, em todos está definido uma injeção de valor diretamente na URL, sendo este um indício de vulnerabilidade de SQL Injection. Nesta primeira ação separamos o segmento de escolas que seriam analisados.

Para essa pesquisa foram selecionadas 50 (cinquenta) instituições de diferentes segmentos de mercado para que fosse possível realizar uma pesquisa de campo, analisando os sistemas e verificando se os mesmos estavam vulneráveis. Em todos os sites que foram identificadas falhas de segurança, estas instituições foram informadas sobre a falha, encaminhamos também um documento com possíveis soluções a serem aplicadas, visando assim, resolver o problema. Para os casos graves que permitiram a conclusão dos testes,

chegando até a parte final que é a extração das informações, foi enviado um contrato de Pentest, com todos os termos de liberação para os testes sem que seus sistemas sejam afetados, garantindo que o ataque não os debilitaria, ou seja, mantivemos os pilares da segurança da informação, Confidencialidade, Disponibilidade e Integridade.

## LEVANTAMENTOS DOS DADOS E RESULTADOS

Para o conhecimento do uso do SQLMAP, foi documentado todo o processo durante a execução da ação, desde o acesso até a exibição dos dados da base de um dos colégios analisados, para segurança e confidencialidade, todas as informações referentes ao colégio foram apagadas, porém, todos os comandos estão visíveis para compreensão e aprendizado. Segue abaixo todos os passos para o levantamento dos dados:

**1º Passo:** Para realizar a execução do SQLMAP é necessário utilizar um parâmetro GET que no caso é o que o Google retornou nas pesquisas realizadas como por exemplo: (www.site.com/index.php?id=1‘), sabendo disso, é utilizado um site que tenha a URL com a informação que é preciso para a execução da ferramenta, como demonstrado na figura abaixo:

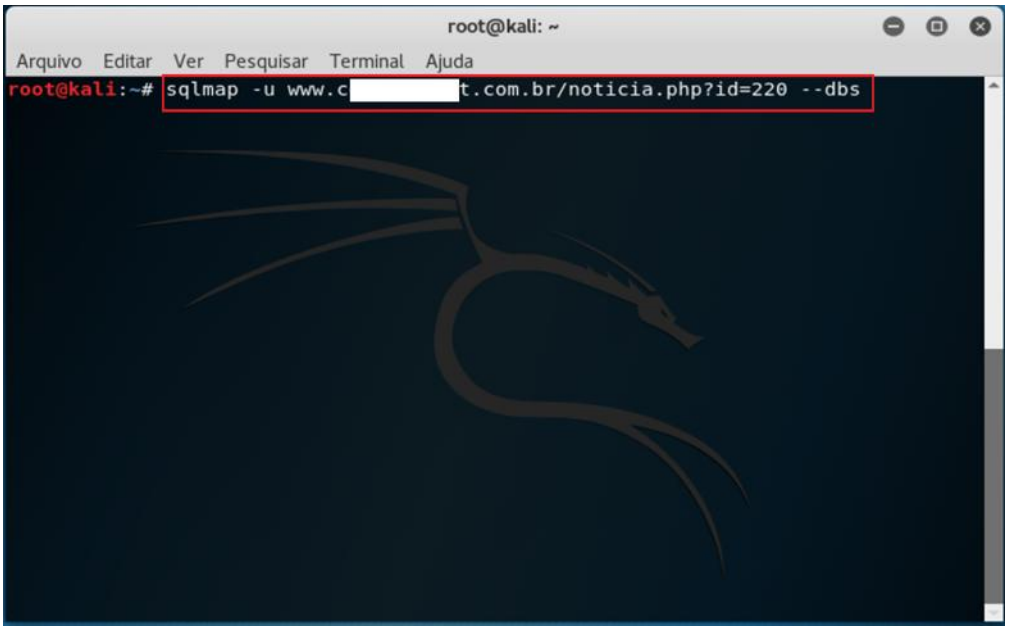


**Figura 3:** Primeiro passo na execução do SQLMAP.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

### Parâmetros utilizados:

**Sqlmap:** Chama a ferramenta que será utilizada para a verificação da vulnerabilidade;
**-u:** Parâmetro para especificar a URL (Site) que será utilizado na análise;
**--dbs:** Lista os bancos de dados do site.

Após a execução deste comando, o SQLMAP iniciará a análise, retornando se o site possui ou não uma brecha para a injeção de SQL, a imagem abaixo aponta que a URL analisada possui uma falha na segurança.

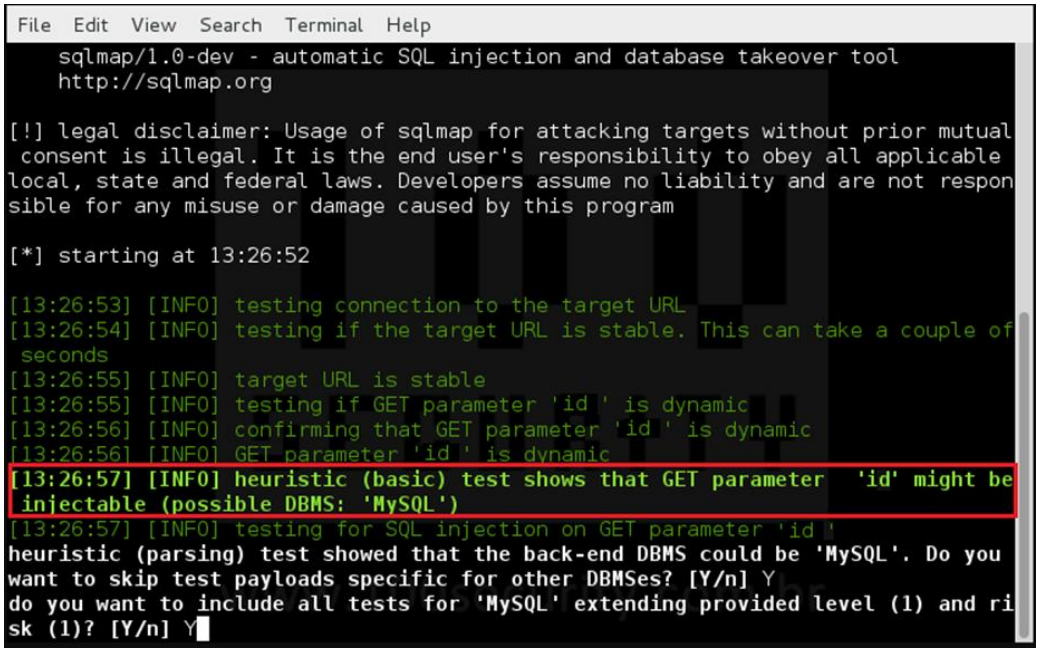


**Figura 4:** Informação de parâmetro GET vulnerável a SQL Injection.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

Essa mensagem está informando que é possível injetar comandos SQL através do parâmetro "id", desta forma podemos acessar o banco de dados. Ao final da execução do comando "—dbs" é apresentado as bases que foram encontradas, conforme Figura 5.

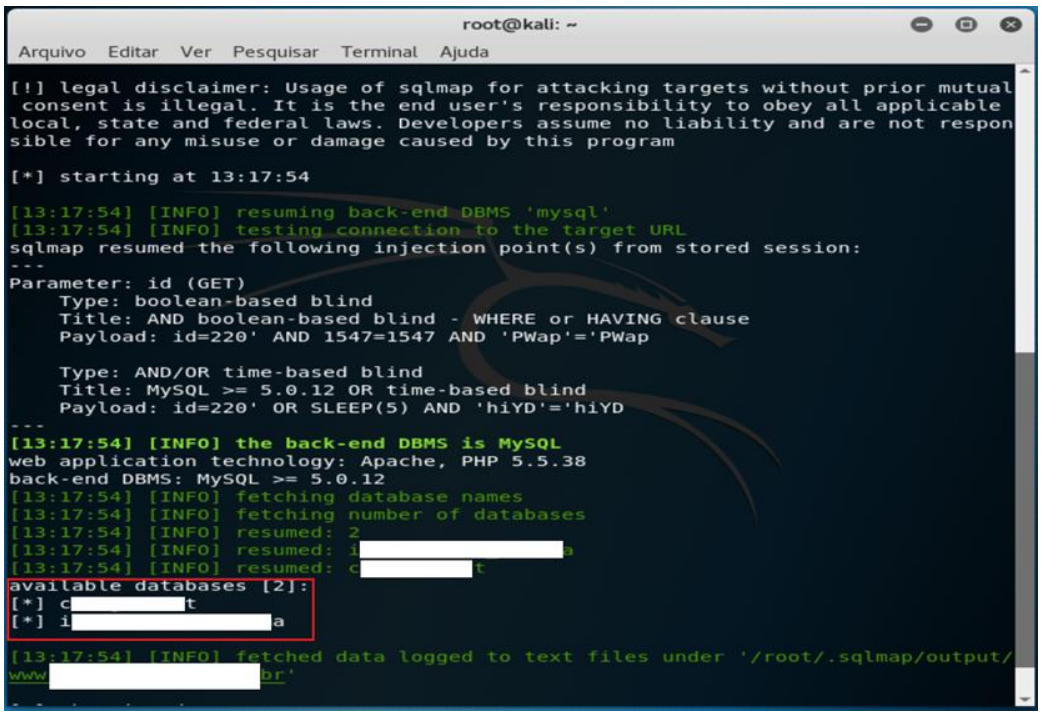


**Figura 5:** Retorno com a informação das bases que estão contidas no site.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

As bases foram ocultadas por motivos de confidencialidade da instituição, contudo, é possível observar que SQLMAP retornou "available databases [2]", informando os bancos que foram encontrados.

Sabendo quais bancos constam neste sistema, podemos prosseguir utilizando a ferramenta para ter visão detalhada do conteúdo deste banco, como tabelas, colunas e linhas com suas informações.

**2º Passo:** Próximo comando a ser executado exibe a tabela do banco selecionado, para isso, é utilizado o comando conforme apresentado na Figura 6:

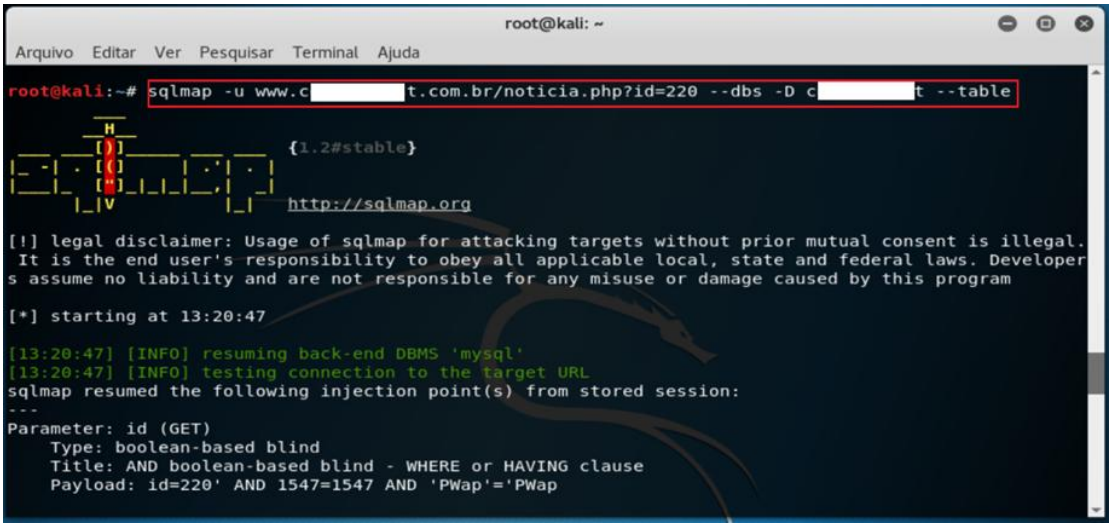


**Figura 6:** Segundo passo na execução do SQLMAP.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

**Parâmetros utilizados:**

**-D:** Seleciona o banco de dados a qual quer analisar, a sintaxe para isso é: -D "nome_banco";

**--table:** Lista todas as tabelas do banco de dados que foi selecionado. Após a execução deste comando, serão exibidas todas as tabelas que compõem o banco de dados que foi selecionado, conforme a Figura 7.

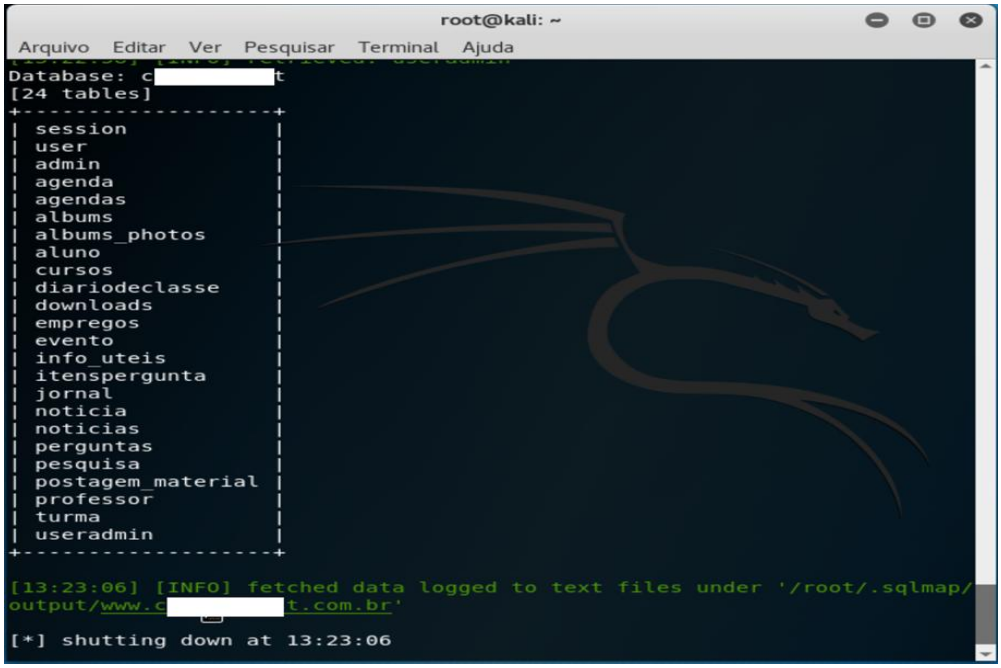

**Figura 7:** Extração das tabelas do banco de dados selecionado.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

A imagem acima (Figura 7), mostra como é exibido no terminal do Kali os nomes das tabelas do banco de dados, como no exemplo citado, recebe o retorno de 24 tabelas contendo informações de todo o sistema da instituição, como dados referentes aos professores, usuários, alunos e turmas.

Neste teste, foi utilizado a tabela de professores para realizar um acesso e a tentativa de extração dos dados relacionado aos mesmos, simulando desta forma, a maneira em que se ocorre o vazamento.

**3º Passo:** Acessando a tabela do banco de dados, será extraído do banco as colunas existentes na tabela "professor" com todos os dados que estão contidos no mesmo. Para este processo foi executado o seguinte parâmetro:

**Parâmetros utilizados:**
**-T:** Seleciona a tabela a qual deseja ter acesso;
**--columns:** Lista todas as colunas referentes a tabela que foi selecionada.

A Figura 8 apresenta o resultado desta execução.

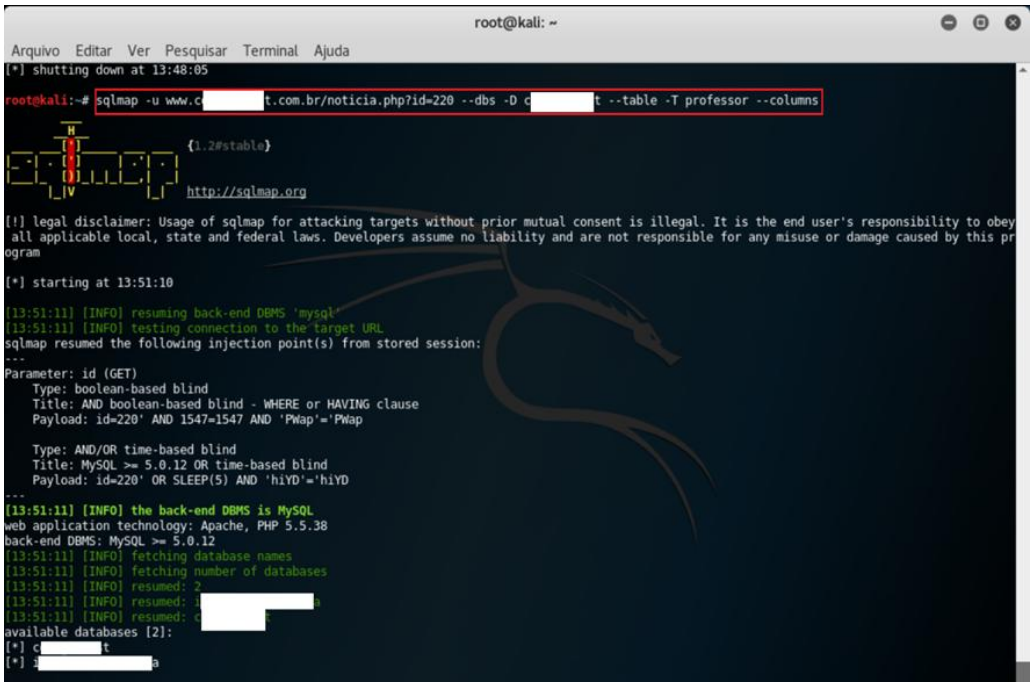


**Figura 8:** Terceiro passo na execução do SQLMAP.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

Realizando este procedimento, o SQLMAP irá retornar todas as colunas que estão na tabela "professor", conforme apresentado na Figura 9:

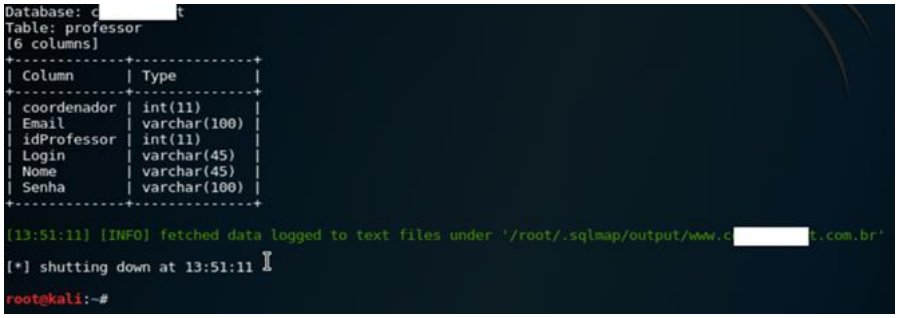


**Figura 9:** Exibição das colunas da tabela selecionada.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

Como resultado, foram extraídas 6 (seis) colunas que compõem a tabela dos professores, contendo informações sigilosas de acesso e contato, como: "coordenador", "Email", "idProfessor", "Login", "Nome" e "Senha".

Da mesma forma que podemos acessar os dados da tabela, podemos finalizar listando e apresentando os dados de cada coluna específica.

**4º Passo:** Neste passo iremos realizar o DUMP, ou seja, extrair todas as informações cadastradas nas colunas da tabela "professor", neste exemplo, serão extraídos especificamente os dados de: Nome, Login, Senha e Email, sendo extremamente perigoso se cair em mãos de pessoas maliciosas, permitindo que tenham acesso privilegiado ao sistema do colégio, podendo realizar ações que venham a ser prejudiciais aos alunos ou funcionários da instituição, com possibilidade de prejuízos econômicos para o colégio. A Figura 10 nos mostra a execução desse comando.

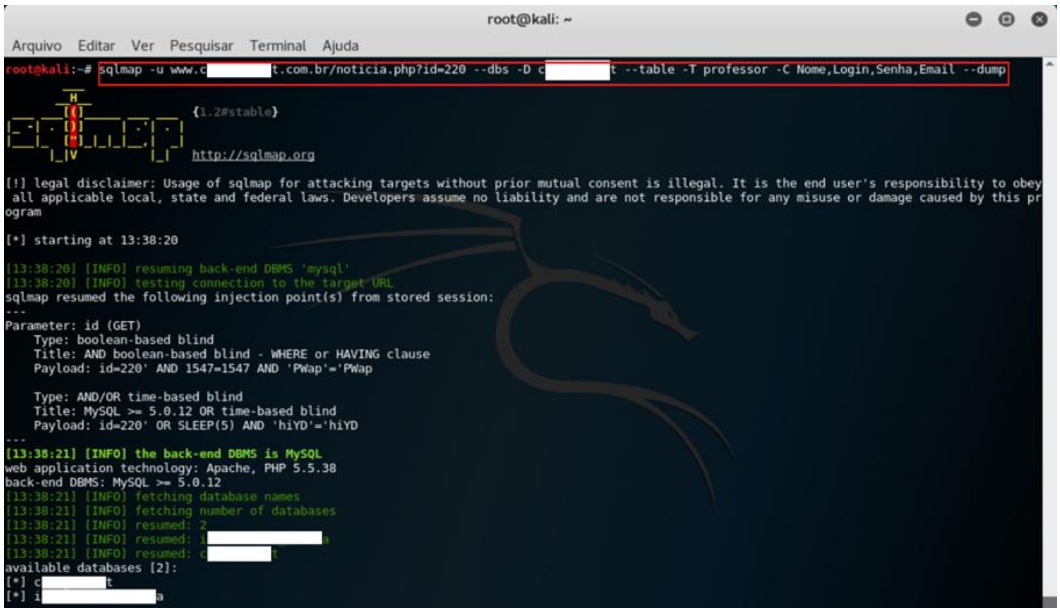


**Figura 10:** Quarto passo na execução do SQLMAP.
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

## Parâmetros utilizados:

**-C:** Define as colunas que serão avaliadas;
**Nome,Login,Senha,Email:** Colunas a qual deseja extrair os dados;
**--dump:** Extrai as informações das colunas selecionadas no parâmetro anterior.

Executando este comando o SQLMAP irá processar todos os dados que estão contidos nas colunas selecionadas, conforme Figura 11.

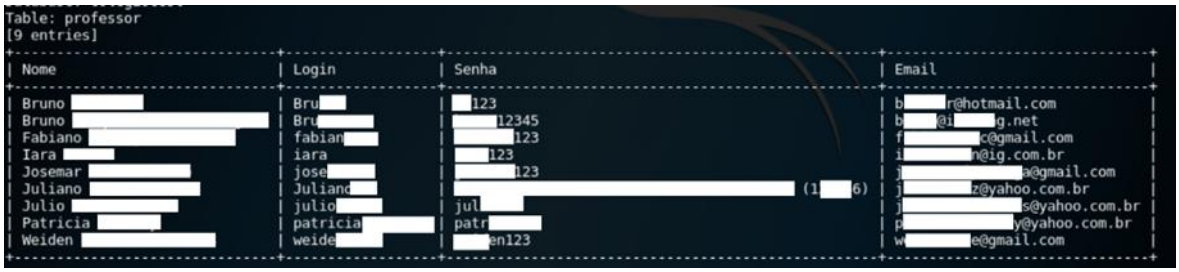


**Figura 11:** Informações contidas na coluna "professor".
**Fonte:** Terminal Kali Linux – Ferramenta SQLMAP.

Munido dessas informações que foram extraídas através da técnica de SQL Injection, podemos realizar procedimentos dentro do sistema, além do vazamento de senhas, e-mail, nome de cada professor da instituição, apresentando risco não somente ao software escolar, mas para vários outros locais, já que muitas pessoas utilizam a mesma senha para vários outros acessos como, redes sociais, e-mail e etc.

Outra análise que deve ser observada deste resultado que foi apresentado, é a quantidade de senhas consideradas "fracas" pela segurança da informação, senhas que, facilmente poderiam ser burladas utilizando-se uma outra técnica conhecida como Brute Force (Força bruta).

A Tabela 1 mostra os resultados obtidos das análises realizadas, todas as informações referentes a identificação das instituições foram suprimidas por motivos éticos e de confidencialidade.

| SEGMENTOS | ANALISADOS | VULNERÁVEIS | % |
|---|---|---|---|
| PREFEITURAS | 10 | 3 | 30,0% |
| PLANOS DE SAÚDE | 10 | 1 | 10,0% |
| ESCOLAS | 10 | 4 | 40,0% |
| FACULDADES | 10 | 3 | 30,0% |
| BANCOS | 10 | 0 | 0,0% |
| TOTAL | 50 | 11 | 22,0% |

**Tabela 1:** Dados quantitativos das empresas pesquisadas por segmento.
**Fonte:** Autoria própria.

O Gráfico 1 apresenta o impacto de vulverabilidade por segmento de mercado pesquisado.

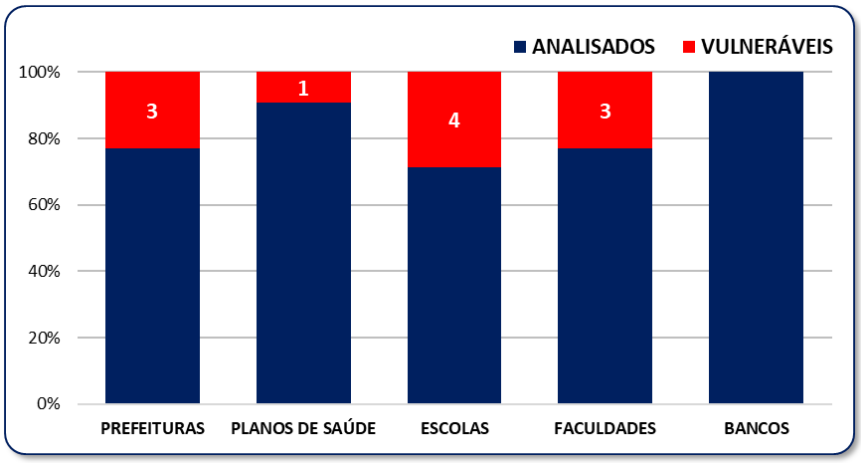


**Gráfico 1:** impacto por Segmento.
**Fonte:** Autoria própria.

Analisando a tabela e o gráfico apresentados, podemos observar que o quantitativo de instituições vulneráveis é grande, pois se inferirmos que 22% das instituições na web contém falhas, teremos um resultado imenso de dados vazados na internet.

Observando o gráfico, podemos ver que o cuidado das instituições bancárias com o tratamento de SQL Injection está extremamente satisfatório, não sendo possível acessar as informações de nenhuma das 10 instituições privadas, estatal ou federal deste segmento. Também vale ressaltar que nem mesmo parâmetros GET's foram encontrados nos sites.

Um ponto de atenção que deve ser citado, é a quantidade de falhas encontradas nas escolas de ensino fundamental, médio e técnico, pois 40% das instituições estavam vulneráveis, possuindo conteúdos pessoais de seus alunos, dentre outras informações sigilosas. Em contato com estes colégios, a maioria informou que não possui um setor de segurança da informação, desta forma concluímos que tais sistemas são desenvolvidos somente para atender a instituição sem a devida preocupação se suas informações estão ou não protegidas.

## CONCLUSÕES

Diante da grande quantidade de informações que estão atualmente "rodando" na internet, é necessário o máximo de cuidado por parte das organizações ao tratar com dados pessoais de seus colaboradores, clientes e fornecedores, acreditamos que a proteção nunca será 100% garantida, porém, todas as empresas devem procurar mitigar qualquer meio de vazamento.

Foram realizados os testes em 50 (cinquenta) sistemas, em 11 (onze) dos testes, foram obtidos sucesso na análise de vulnerabilidade por SQL Injection, outros 39 (trinta e nove) somente apresentaram a possibilidade de injeção de valor através da URL, contudo, possuindo tratamento para evitar este tipo de ataque. Com isso foi possível observar a grande quantidade de sites que possuem falhas na segurança ao considerar esta amostragem, ou seja, podemos considerar o status da segurança geral da informação como crítico, pois os dados apresentados foram resultantes de uma única técnica, sendo que existe centenas de outras técnicas que podem inclusive fazer combinações entre si, ou seja, aumentaria provavelmente este percentual de sistemas vulneráveis à invasão.

Outro fator crítico identificado durante a pesquisa de campo, foi a presença de senhas fracas, como “123456”, “nome123” ou senhas que podem ser descobertas através de uma engenharia social, como data de aniversário ou nome do filho e etc. Segundo o Blog da McAffee, a dica para se criar uma senha difícil, considerada forte de ser descoberta é combinar sempre letras, números e símbolos para criar uma senha que dificulta outros métodos de roubo de senhas.

Para auxiliar as essas empresas que foram detectadas falhas na segurança, foi encaminhado um documento contendo informações de como prevenir e corrigir a falha, quando se tratar de SQL Injection. Este documento foi muito bem recebido por todas as empresas, até mesmo por aquelas que não possuíam a vulnerabilidade, que se interessaram em adquirí-lo para uma possível verificação.

Através do contato feito junto às instituições e informando a possibilidade de falha, “abriram-se os olhos” do risco em se ter dados disponibilizados na pela rede mundial. As escolas contatadas foram de estados diferentes: Rio de Janeiro, Espírito Santo, Minas Gerais, Ceará, Pará, Rio Grande do Sul e São Paulo.

Este artigo também tem por objetivo mostrar para as instituições que todas podem ser alvos, devemos eliminar o pensamento de que “isso não acontece aqui”, normalmente as empresas somente procuram por profissionais da área de segurança após o registro de algum tipo de ataque, sofrendo perdas ou prejuízos econômicos, buscando então, aplicar uma cultura de prevenção.

## RECOMENDAÇÕES

Visando a redução das chances de um processo de invasão com injeção SQL nos sistemas WEB, segue abaixo algumas recomendações.

**Primeiro:** O Firefox disponibiliza um plugin para suporte a SQL Injection, chamado SQL INJECTION ME, este plugin, através do submit dos formulários acusa se há ou não abertura para injeção de SQL no mesmo.

**Segundo:** Outro método que pode ser utilizado, é a aplicação de expressões regulares para limpar as variáveis que são enviadas para o sistema. No exemplo abaixo, segue um modelo de código que limpa a variável de login de um post:

```
$_POST['login'] = preg_replace('/[^[:alpha:]_]/', '',$_POST['login']);
```

**Terceiro:** De acordo com a OWASP (*Open Web Application Security Project*) as melhores práticas para se prevenir de um ataque de SQL Injection são:

- Parametrização das consultas;
- Usar "stored procedures";
- Escapar toda entrada fornecida pelo usuário;
- Limitar privilégios aos acessos.

**Quarto:** Criptografar senhas ou dados que não devem estar em texto claro dentro do DB. Este item não protege contra o SQL Injection, porém, ajuda a manter os dados ilegíveis mesmo sofrendo o ataque, dificultando assim, o uso das informações para ações maliciosas.

## REFERÊNCIAS

BUKOWSKI, Marcelo. Injeção de Sql em aplicações Web Causas e Prevenções. Porto Alegre. Universidade Federal do Rio Grande Do Sul. 2009.

CERVO AL, Amado L, BERVIAN PA, SILVA R. Metodologia Científica. 6ª edição. São Paulo: Pearson Prentice Hall; 2007.

ClydeBank Technology. SQL: QuickStart Guide - The Simplified Beginner's Guide To SQL (SQL, SQL Server, Structured Query Language). Ed. 1. LydeBank Media LLC, 2015.

ELMASRI, Ramez; NAVATHE, Shamkant B. Fundamentals of Database Systems. Ed. 7. Pearson: EUA, 2015.

ERICKSON, Jon. Hacking: The Art of Exploitation. Ed. 2. Nostarch: San Francisco, 2008.

GALVÃO, Michele C. Fundamentos em Segurança da Informação. Pearson: São Paulo, 2015.

HALL, Gary; WATSON, Erin. Hacking: Computer Hacking, Security Testing,Penetration Testing, and Basic SecurDec 28. CreateSpace Independent Publishing Platform, 2016.

HERTZOG, Raphael; O'GORMAN, Jim. Mastering the Penetration Testing DistributionJun. Ed. 1. OffSec Press: Cornelius, 2017.

KROENKE, David M; AUER, David J. Database Concepts. Ed. 7. Pearson: Harlow, 2015.

MAGALDI Heitor, LIMA Patrícia. SQL Injection, entenda o que é, aprenda a evita-lo. 12ª edição, Juiz de Fora – MG. Faculdade Metodista Granbery. 2010.

NORMAN, Alan T. Hacking: Computer Hacking Beginners Guide How to Hack Wireless Network, Basic Security and Penetration Testing, Kali Linux, Your First Hack. CreateSpace Independent Publishing Platform, 2016.

PUGA, Sandra; FRANÇA, Edson; GOYSA, Milton. Banco de Dados - Implementação em SQL, PL/SQL e Oracle 11g. Pearson: São Paulo, 2014.

TAURION, César. Cloud Computing – Computação em nuvem: transformando o mundo da tecnologia da informação. Ed. Brasport: Rio de Janeiro, 2009.

TAURION, César. Big Data. Ed. Brasport: Rio de Janeiro, 2013.

VELU, Vijay K. Mastering Kali Linux for Advanced Penetration Testing - Second Edition: Secure your network with Kali Linux - the ultimate white hat hackers. Ed. 2. Birmingham, 2016.

VERGARA SC. Projetos e relatórios de pesquisa em administração. 7ª edição. São Paulo: Atlas; 2007.