Google *Hacking* para Ataques *SQL Injection*

Amaury Walbert de Carvalho[1], Antonio Pires de Castro Júnior[2]

[1]Faculdade de Tecnologia SENAI de Desenvolvimento Gerencial (FATESG)
Rua 227-A, nº 95, St. Leste Universitário – CEP 74610-155 – Goiânia – GO – Brazil

{amaurywalbert,apcastrojr}@gmail.com

Abstract. *The objective of this paper is to present a study on the Google Hacking and SQL Injection, alerting administrators of web systems and information security professionals about the risks of combining these techniques for detecting faults and vulnerability exploits. A case study using the tools Google Dorks and SqlMap is described to show the ease with which an attacker can gain access to sensitive information of an organization by detecting and exploiting faults in web systems. Finally, some measures are introduced to minimize the problems caused by this type of attack..*

Resumo. *O objetivo deste trabalho é apresentar um estudo sobre o Google Hacking e SQL Injection, alertando os administradores de sistemas web e profissionais de segurança da informação sobre os riscos da combinação dessas técnicas de detecção de falhas e exploração de vulnerabilidades. Um estudo de caso usando as ferramentas Google Dorks e SqlMap é descrito para mostrar a facilidade com que um atacante consegue ter acesso a informações sensíveis de uma organização através da detecção e exploração de falhas em sistemas web. Ao final, algumas medidas são apresentadas para minimizar os problemas causados por esse tipo de ataque.*

1. Introdução

A segurança da informação tem se tornado assunto cada vez mais preocupante entre os administradores de sistemas web. A proximidade e facilidade de interação que esses sistemas oferecem aos clientes e parceiros de muitas organizações também podem ser o elo mais frágil caso as medidas de segurança necessárias não sejam implantadas. Princípios de confidencialidade, integridade e disponibilidade devem ser abordados com muita cautela por qualquer organização que trata a informação como um ativo valioso, pois um ataque bem sucedido explorando falhas e ferindo esses princípios podem causar prejuízos imensuráveis.

A Internet permite que diversos tipos de negócios aconteçam através de sistemas web, e as ferramentas de busca são aliadas no processo de divulgação dos produtos e serviços de uma organização. Essas mesmas ferramentas podem ser usadas por atacantes para descobrir falhas de segurança nos sistemas. Existem diversas técnicas usadas para esse propósito, uma delas é o Google *Hacking*.

Exploradas de maneira correta, as falhas de segurança em um sistema web permitem que o atacante tenha acesso irrestrito à informações sensíveis de uma organização. De acordo com um levantamento feito pela OWASP [1], o maior número

de incidentes de segurança em sistemas web no ano de 2013 foi através de injeção de códigos *SQL* em sistemas vulneráveis. Este método de exploração é conhecido como *SQL Injection*.

O objetivo deste trabalho é chamar a atenção dos profissionais de segurança da informação e dos administradores de sistemas para as ameaças existentes. Atacantes estão combinando ferramentas do mecanismo de buscas do Google para identificar alvos, e explorando essas vulnerabilidades com ferramentas que fazem injeção de códigos SQL de forma automatizada para ter acesso a informações sensíveis.

Na próxima seção, o leitor encontra um levantamento sobre a facilidade com que as informações de uma organização podem ser visualizadas através do motor de buscas do Google, e a necessidade de proteção de sistemas informacionais. As seções 3 e 4 trazem um estudo sobre as técnicas de Google Hacking e SQL Injection, respectivamente. A seção 5 descreve um estudo de caso usando as ferramentas Google Dorks e SqlMap para detecção de um alvo vulnerável, exploração e acesso a informações sensíveis de uma organização.

2. Levantamento Literário

2.1. Segurança da Informação

Já há alguns anos a informação movimenta negócios pelo mundo todo e muitas organizações dependem exclusivamente da manipulação de informações para a geração de receita, e por consequência, manter sua existência e lucratividade. Para PMG Solutions [2] a informação se tornou uma mercadoria valiosa e é difícil observar qualquer processo de negócio que não trabalhe com informações.

Castells [3], com o intuito de identificar o período em que vivemos e mostrar os procedimentos utilizados na manipulação da informação, faz um paralelo entre os termos “sociedade da informação” e “sociedade informacional”. Para o referido autor, “sociedade da informação” é usado para ilustrar a importância da informação e da comunicação em qualquer sociedade. Entretanto, “sociedade informacional” refere-se ao modo como a geração, o processamento e a disseminação da informação ocorre em consonância com uma organização social e se tornam fontes fundamentais de poder e produtividade, graças às tecnologias que surgem.

Assim como qualquer bem material que possui valor e gera cobiça, a informação deve ser protegida de forma que seu acesso seja restrito. De acordo com o ramo de atuação de determinada pessoa ou organização, ter informações divulgadas de maneira incorreta pode acarretar um prejuízo financeiro e até mesmo levar à falência. Empresas do ramo financeiro e de tecnologias da informação e comunicação geralmente se preocupam mais com a segurança das informações que manipulam e, à medida que esse assunto começa a ser tratado de maneira adequada dentro das organizações, mecanismos, métodos e técnicas começam a aparecer para garantir a segurança da informação. Porém, é possível observar que ainda há pequenas organizações que não estão preocupadas ou desconhecem as vulnerabilidades existentes.

A Era da Informação, como está sendo chamada a fase do desenvolvimento humano que estamos vivendo, tem relação com o aumento exponencial do uso da

Internet. Essa tecnologia permite que informações sejam trocadas a uma velocidade surpreendente, facilitando e até mesmo criando novas formas de negócio. Instituições públicas e privadas fazem uso da Internet para divulgar sua marca e seus produtos, assim como facilitar a interação com seus parceiros e clientes, através de sistemas web. Vários tipos de dispositivos estão conectados à rede mundial de computadores, trocando informações e compartilhando recursos, trazendo mobilidade e aumentando ainda mais o fluxo de informações que trafegam pela rede.

Como uma boa parte das informações sensíveis de uma organização estão sendo manipuladas por sistemas computacionais, indivíduos mal-intencionados podem se aproveitar de falhas nos sistemas para ter acesso a essas informações. Empresas são testadas a todo momento sobre aspectos relacionados à segurança da informação e os noticiários constantemente mostram casos de incidentes, geralmente provocados por *hackers*, que vão desde acessos indevidos a sistemas web até alterações em sistemas de banco de dados, com perdas de informações. O fato é que as brechas de segurança existem, sejam em sistemas automatizados ou provocadas por fatores humanos, e devem ser consideradas e tratadas.

Para Ferreira [4] existem três principais atributos de qualidade que a informação deve ser submetida: integridade, confidencialidade e disponibilidade. Esses atributos formam os pilares da segurança da informação. Horton & Muage *apud* Alexandria [5] caracterizam esses pilares como Modelo CIA (*Confidentiality, Integrity and Availability*) e ressaltam a necessidade das empresas entenderem e avaliarem cada item do modelo e ainda destacam a importância da aplicação de efetivos métodos para proteger a informação usando pessoas, processos e tecnologias.

Malandrin [6] defende a ideia de que uma organização deve adotar um Sistema de Gestão da Segurança da Informação (SGSI) e relaciona o Modelo CIA como propriedades essenciais sobre a informação da organização. Para este autor, essas propriedades são assim definidas:

- Confidencialidade: a propriedade que a informação tem de que não será divulgada a entidades não autorizadas. Essa propriedade pode se referir ao conteúdo da informação ou a própria existência da informação. Controles de acesso ajudam a manutenção de confidencialidade da informação.
- Integridade: a propriedade que a informação tem de não ser alterada por entidades não autorizadas. Essa propriedade também se refere tanto quanto ao conteúdo da informação como a sua origem. Controles existem para permitir a prevenção contra ataques a integridade ou a detecção de ataques que já aconteceram.
- Disponibilidade: a propriedade que a informação tem de que estará disponível para as entidades autorizadas. A indisponibilidade de uma informação é quase tão ruim quando a inexistência da informação.

Para Horton & Muage apud Alexandria [5], a confidencialidade garante que uma empresa estabeleça vantagem competitiva na produção, no tempo de comercialização e até mesmo na confiança do cliente. Informações que são divulgadas sem a anuência da empresa passa uma impressão de falta de compromisso com o cliente e ainda permite

que concorrentes possam avaliar os dados de determinado produto, como materiais, métodos e processos de fabricação. Todos esses fatores causam prejuízos financeiros.

Em situações onde a confidencialidade não é fator crítico para a segurança da informação, como em alguns sistemas web de instituições públicas, onde o acesso à informação deve ser permitido e garantido, sem que isso cause prejuízo para a instituição, ainda assim, a integridade e a disponibilidade desses sistemas devem ser preservadas para o bom funcionamento e cumprimento dos objetivos da instituição [5].

Com o aumento considerável da preocupação com a segurança da informação, normas e padrões foram estabelecidos para que os interessados possam realizar procedimentos que diminuam os riscos de ataques contra seus sistemas, entre eles os sistemas web, um dos principais.

No Brasil, existe a NBR ISO/IEC 27002, um código de prática para a segurança da informação baseado em normas internacionais e publicado pela ABNT (Associação Brasileira de Normas Técnicas). Essa norma é baseada nas melhores práticas de segurança da informação reconhecidas mundialmente e por sua constante revisão e atualização, deveria ser utilizada por todas as empresas. Para Franciscatto [7] a adoção da ISO/IEC 27002 garante, além da segurança da informação, uma correta comunicação entre fabricante e cliente, a proteção do consumidor e a eliminação de barreiras técnicas e comerciais.

Inicialmente publicada em 2000 como ISO 17799, a norma nacional era baseada em normativas britânicas. Em 2005, a ISO/IEC 27002 foi revisada pela ABNT e passou a contar com 133 controles de segurança distribuídos em 11 seções. A referida norma ainda conta com 15 capítulos e 39 categorias principais de segurança [5].

A Tabela 1 mostra a estrutura da ISO/IEC 27002:

Capítulo	Título	Nro Sub-Capítulos
5	Política de Segurança da Informação	1
6	Organizando a Segurança da Informação	2
7	Gestão de Ativos	2
8	Segurança em Recursos Humanos	3
9	Segurança Física e do Ambiente	2
10	Gestão de Operações e Comunicações	10
11	Controle de Acesso	7
12	Aquisição, Desenvolvimento e Manutenção de SI	6
13	Gestão de Incidentes de SI	2
14	Gestão da Continuidade do Negócio	1

15	Conformidade	3

Tabela 1: ISO 27002 - Estrutura [7]

2.2. Google

O Google *Search* é uma das ferramentas de busca de páginas web mais usadas no mundo. Não é difícil encontrar usuários que se tornaram dependentes desse mecanismo de buscas que surgiu com o intuito de facilitar o acesso às páginas disponíveis na internet. Com a popularização da web, milhares de páginas começaram a surgir e esse número continua aumentando de maneira exponencial, formando uma imensa biblioteca virtual. A ideia por traz do Google *Search* é criar um índice com o maior número de informações possíveis e permitir que os usuários possam localizar páginas relacionadas ao conteúdo procurado [8].

A ferramenta de busca do Google, que em 2014 completa 16 anos, é a página mais acessada em vários países do mundo [9]. No Brasil, o volume de sua participação nas buscas efetuadas fica acima de 90%, consolidando sua autoridade e eficácia na realização de pesquisas [10].

Organizando o conteúdo da internet, Google Inc. se tornou uma das empresas mais valiosas do mundo e, seguindo a filosofia de facilitar o acesso à informação, tem se aventurado e despontado também em vários outros ramos da tecnologia.

2.2.1. A Ferramenta de Busca

O sucesso do Google começou com seu inovador mecanismo de pesquisas de páginas web. Até o seu surgimento, os sites de busca existentes eram limitados e ineficientes. Os fundadores do Google, Larry Page e Sergey Brin, desenvolveram o *PageRank*, que classifica as páginas de acordo com a sua importância, baseando-se na ligação existente entre as páginas [8]. O algoritmo do *PageRank* examina a consulta e usa mais de 200 sinais para decidir quais dos milhões de páginas e conteúdos são as respostas mais relevantes para a consulta, como por exemplo, o grau de atualidade do conteúdo de um site, o número de outros sites que contêm *links* para um site específico e a autoridade desses *links*, palavras na página da web, sinônimos para as palavras-chave de pesquisa, verificação ortográfica, etc. O Google refina esses algoritmos de classificação com mais de 500 aprimoramentos por ano [11].

> A classificação das páginas (PageRank) confia na natureza excepcionalmente democrática da Web, usando sua vasta estrutura de links como um indicador do valor de uma página individual. Essencialmente, o Google interpreta um link da página A para a página B como um voto da página A para a página B. Mas o Google olha além do volume de votos, ou links, que uma página recebe; analisa também a página que dá o voto. Os votos dados por páginas "importantes" pesam mais e ajudam a tornar outras páginas "importantes" [12].

Para que a pesquisa possa ser feita, primeiro o Google precisa localizar, organizar e armazenar informações sobre as páginas e indexá-las a um banco de dados, que será consultado em cada pesquisa realizada em sua ferramenta de busca. A partir de então, o resultado da pesquisa é classificado com o *PageRank* e exibido para o usuário.

Para localizar as informações o Google *Search* usa um conjunto de algoritmos conhecidos como *crawlers*, ou rastreadores da web. O *crawler* mais conhecido é chamado de "Googlebot". Ele é responsável por verificar os *links* de cada página e segui-los, sempre enviando informações ao servidor do Google. Assim como um usuário navega na internet através dos *hiperlinks*, o Googlebot consegue percorrer todas as ligações entre as páginas e enviar para o banco de dados do Google todas as informações contidas em cada uma das páginas que visita [13].

O rastreamento das páginas começa com um conjunto de páginas rastreadas anteriormente e deve ser constante, garantindo que todas as referências encontradas estejam atualizadas e, por isso, o robô de buscas do Google não para. Sites com muitas referências são indexados primeiro e aqueles com poucas referências podem demorar semanas ou meses para aparecerem nos resultados das pesquisas. Mesmo sabendo que uma grande maioria dos usuários da internet usa o seu mecanismo de busca para localizar páginas web, o Google não aceita dinheiro para aumentar a frequência de acesso a um site ou manipular o resultado das pesquisas. A ordem de apresentação dos resultados é feita de acordo com a classificação realizada pelo *PageRank* e por experiências anteriores dos próprios usuários [13].

Assim como o índice de um livro, o Google também consegue catalogar as páginas que visita, e assim mantém referências dos termos encontrados nelas. Quando um usuário utiliza a ferramenta de busca para localizar uma palavra, por exemplo, o Google percorre seu índice e mostra todas as referências encontradas, classificadas através do *PageRank*. Segundo informações divulgadas pela própria empresa, o índice da ferramenta de buscas do Google ultrapassa os 100.000.000 de gigabytes e exigiu mais de um milhão de horas de computação para ser construído [13].

Sempre tentando aprimorar os resultados e exibi-los de acordo com o contexto em que o usuário está inserido, o Google vem desenvolvendo novos recursos para facilitar a pesquisa. Várias dicas e truques podem ser encontradas no site da ferramenta de buscas [14].

2.2.2. A Pesquisa

A interface web da ferramenta de busca permite que a pesquisa possa ser realizada de modo muito simples. Um palavra, um termo ou uma frase pode ser passada como parâmetro para a ferramenta de busca, que realiza todo o processo descrito no tópico anterior até devolver apenas resultados relevantes. É comum que os usuários encontrem o que precisam apenas com um consulta simples, graças ao *PageRank*, muito embora a quantidade de resultados possa ser na ordem de milhares ou milhões. Porém, caso o usuário esteja buscando por algo muito específico, ele pode fazer uso dos operadores da ferramenta de busca para refinar sua pesquisa até o resultado ser preciso.

Em um consulta simples, esses operadores da ferramenta de busca trabalham de forma transparente e sem a intervenção do usuário. Nesse caso alguns termos da pesquisa podem ser adicionados ou ignorados pela ferramenta de busca. Conhecendo o mecanismo de busca e seus operadores, é possível então realizar consultas complexas, porém muito mais refinadas e objetivas. Os principais operadores da ferramenta de busca são os operadores *booleanos* (*AND*, *OR* e *NOT*). Praticamente todos os motores de busca existentes na web hoje fazem uso desses operadores. Conhece-los fará com o

que usuário passe a interagir de forma direta com a ferramenta de busca e possa realizar consultas de modo bem particular. Combinados com caracteres especiais da ferramenta, a pesquisa se torna muito mais específica. Porém, o uso inadequado destes operadores podem alterar drasticamente os resultados que são retornados [15]. Para conhecer os operadores e saber como usá-los, leia Long [15].

3. Google Hacking

Apesar da interface simples, a página inicial do Google (www.google.com.br) esconde uma poderosa ferramenta de busca, capaz de localizar diversos tipos de conteúdo indexados em sua base de dados. Usuários iniciantes são conduzidos por mensagens de auxílio nos resultados de suas pesquisas. Usuários experientes têm a opção de uma interface um pouco mais complexa (http://www.google.com.br/advanced_search), carregada de detalhes mas que promovem uma pesquisa muito mais refinada. Essa interface faz uso dos operadores da ferramenta de busca. Com um pouco de paciência e muita prática, os usuários podem personalizar suas pesquisas através dos operadores até mesmo no campo de pesquisa da página inicial, restringindo os resultados de forma que estes sejam precisos [15].

As novas tecnologias surgem para facilitar a vida do ser humano, porém nem todos usam as ferramentas da maneira adequada. Usuários mal-intencionados podem usar um motor de busca para localizar informações que podem ser usadas para prejudicar outras pessoas. Toledo [16] mostra que existe uma quantidade expressiva de informações sensíveis que podem ser acessadas através de uma simples pesquisa no Google. O ato de explorar falhas em sistemas web ou localizar informações confidenciais usando a ferramenta de pesquisa do Google é chamado de Google *Hacking*.

McGuffe [17] define Google *Hacking* como um conjunto de ferramentas e técnicas que podem ser combinadas ou usadas de forma isoladas para descobrir vulnerabilidades e problemas de segurança na Internet. Assim como os operadores avançados do Google permitem que usuários possam melhorar suas pesquisas e conseguir localizar um determinado conteúdo de forma rápida e precisa, usuários mal-intencionados podem usar os mesmos artifícios para identificar possíveis alvos para ataques.

Profissionais de segurança da informação e usuários avançados dos sistemas web tentam identificar os tipos de consultas que podem resultar em busca por informações sensíveis. Exitem diversos catálogos disponíveis na internet que descrevem as principais técnicas usadas pelos praticantes do Google *Hacking*. A intenção é divulgar o máximo possível essas técnicas para que todos os responsáveis por administrar sistemas web possam conhecê-las e tentar evitar que os seus sistemas sejam explorados. A coletânea mais conhecida talvez seja o GHDB [18]. Criado originalmente por Johnny Long, autor do livro *Google Hacking for Penetration Testers* [15], o GHDB foi incorporado ao *Exploit Database*, e as técnicas de consulta usadas no Google *Hacking* passaram a ser conhecidas também como Google *Dorks* [18]. A partir de então os diversos softwares que fazem testes de intrusão podem usar o GHDB para aprimorar os testes e verificar se os sistemas estão vulneráveis a estes tipos de consultas.

De acordo com o site Acunetix [19] os *dorks* disponíveis no GHDB podem identificar as seguintes tipos de dados:

- Avisos e vulnerabilidades do servidor;
- Mensagens de erro que contêm muita informação;
- Arquivos contendo senhas;
- Diretórios sensíveis;
- Páginas que contêm portais de *logon*;
- Páginas que permitem acesso à dispositivos de rede ou dados como *logs* de *firewall*.

Os Google *Dorks* podem ser classificados em 14 categorias [18], o que facilita até mesmo a localização de um determinado *dork* dentro do GHDB. A seguir é mostrado apenas alguns exemplos de pesquisas que podem retornar dados sensíveis:

1. **Sistemas Vulneráveis =** `inurl:"php?id="` – Essa consulta retorna sites que podem ser vulneráveis a ataques de SQL Injection.
2. **Dispositivos de Rede =** `inurl:IndexFrame.shtml "Axis Video Server"` – Nesse caso é possível identificar sites que permitem acesso à câmeras de segurança.
3. **Diretórios Sensíveis =** `intitle:"index of"` – Retorna sites que permitem navegar pelos diretórios, revelando várias informações sobre o servidor.
4. **Mensagens de Erro =** `"Warning: mysql_connect(): Access denied for user: '*@*" "on line" -help -forum` – Essa consulta retorna sites que permitem a visualização do nome de usuário do banco de dados MySQL.
5. **Portais de *Login* =** `site:login.*.*` – Portais de Login podem ser facilmente identificados com essa consulta.
6. **Arquivos Contendo Senhas =** `intitle:"Index of" ".htpasswd" htpasswd.bak` – Essa consulta retorna sites que permitem navegar pelo diretório que contém o arquivo de senhas do sistema web.

A maioria dos Google *Dorks* disponíveis no GHDB já não funcionam mais. Atualizações dos sistemas web e correções de falhas realizadas pelos seus administradores evitam que as informações sejam descobertas por esse tipo de técnica. Ao longo do tempo o próprio Google tem bloqueado algumas consultas por considerá-las maliciosas. Porém, um conhecimento avançado do mecanismo de busca e um pouco de paciência podem render aos usuários novos *dorks*. Combinando os operadores lógicos, operadores avançados e tendo um alvo a ser direcionado, é possível que novas consultas possam resultar em dados sensíveis que ainda não tinham sido descobertos.

3.1. Google Caching

Os robôs de pesquisa do Google vasculham a internet para localizar o maior número de páginas e indexá-las à sua base de dados. Quando um usuário faz uma consulta, o motor de busca realiza uma pesquisa em sua base de dados e então retorna os resultados para esse usuário. Para tornar esse procedimento eficaz, o Google faz uma cópia de cada página que ele encontra, armazenando todas em seu *cache*. Ao visualizar o resultado de uma pesquisa, o usuário tem a opção de acessar diretamente o endereço real da página ou então pode optar por visualizar apenas uma versão da página que esteja no *cache* do Google [17].

Para Toledo [16], o uso de *cache* pode fazer com que um atacante recolha informações que estão disponíveis nas páginas sem mesmo fazer o acesso direto a elas. Neste caso, uma camada extra de segurança para o atacante é adicionada, uma vez que não é necessário estabelecer uma conexão direta com o servidor web de destino, e por isso, não ser rastreado.

Long [15] alerta também sobre o uso do recurso de tradução oferecido pela ferramenta de busca. É possível usar o Google como um servidor *proxy* transparente através desse serviço de tradução. Quando uma página que está em outro idioma é exibida no resultado da pesquisa, um *link* para tradução da página é exibido ao lado do resultado. Ao clicar no *link*, o usuário é levado até uma cópia traduzida da página, hospedada nos servidores do Google. Com esse recurso, o usuário também pode visualizar o conteúdo de uma página, sem estabelecer uma conexão direta com o servidor da mesma. Todo o processo será realizado pelo Google, que funcionará como um intermediário entre o usuário e a página.

Outro fator importante do uso do cache de páginas do Google é que informações sensíveis que foram disponibilizadas em sistemas web ainda podem estar no *cache* mesmo que os administradores tenham corrigido as falhas que outrora permitiam o acesso a esses dados. Nesse caso é difícil mensurar o tamanho do estrago que pode ter sido causado enquanto esses dados ficaram expostos nos sistemas do alvo, uma vez que eles ainda podem estar no servidores do Google.

3.2. Como Evitar o Google Hacking?

É praticamente impossível evitar o Google *Hacking*. Diariamente surgem novos dorks que exploram falhas de segurança e podem ser usados para levantar informações confidenciais ou dados sensíveis na internet. Billing [20] faz uma síntese sobre as técnicas que podem minimizar os efeitos desse tipo de ataque. Segundo os autores, os profissionais de segurança devem realizar os mesmos procedimentos necessários no tratamento de outras questões de segurança da informação. Nesse sentido, estabelecer uma política de segurança sólida, no que diz respeito aos tipos de informações que podem ser disponibilizadas na web é um fator fundamental.

Outro ponto interessante é usar as mesmas técnicas do Google *Hacking* para explorar os seus próprios sistemas. Estar atentos aos novos *dorks* e testá-los contra os seus sites é uma maneira de verificar quais são as falhas que podem ser exploradas e com isso conseguir corrigir antes de um possível ataque efetivo. Caso as informações sensíveis sejam disponibilizadas por funcionários das empresas, cursos e treinamento

podem ser aplicados para que esses funcionários conheçam os riscos de uma informação veiculada na internet de maneira imprópria, Logo, as políticas de segurança devem ser aplicadas também à esses tipos de usuários.

A partir do conhecimento das técnicas de Google *Hacking*, os administradores podem criar ferramentas automatizadas para testar continuamente seus sistemas. Existem também soluções prontas que fazem esse trabalho. Os *Scanners* de Vulnerabilidades Web já estão fazendo uso do GHDB e também de outras bibliotecas, para tentar identificar sistemas que podem estar divulgando informações sensíveis e passíveis de serem exploradas por técnicas de Google *Hacking*.

4. *SQL Injection*

De acordo com um levantamento feito pela OWASP [1], o *Structured Query Language Injection*, ou *SQL Injection*, foi a técnica mais utilizada para ataques a aplicações web em 2013. Tentando explorar variáveis usadas pelas aplicações para acessar o banco de dados, o atacante pode conseguir inserir comandos *SQL* maliciosos e receber como retorno informações sensíveis diretamente do banco de dados, ou, em alguns casos, inserir comados do próprio sistema operacional. Esse tipo de ataque acontece principalmente em sites que disponibilizam formulários ou portais de *login*, onde o usuário deve interagir com o banco de dados através da aplicação web. Nos casos onde não há tratamento dos dados inseridos, é possível que o ataque aconteça.

> Um ataque bem sucedido permite que o atacante possa ler, atualizar, excluir, inserir e executar outros comandos *SQL* no banco de dados. O atacante pode ir mais longe e executar comandos do sistema operacional e lançar outros tipos de ataques no servidor [21].

Um exemplo simples pode ser descrito a partir de uma página de *login*. Um usuário deve passar valores referentes ao *login* e senha para a aplicação web. A partir de comandos previamente definidos, a aplicação se comunica com o banco de dados para consultar os valores passados pelo usuário. A consulta é formada por comandos *SQL* combinados com as variáveis da aplicação, contendo os valores passados pelo usuário. Caso o ambiente descrito não tenha nenhum mecanismo para verificar os tipos de dados inseridos, outros comandos *SQL* podem ser inseridos nos campos de *login* e senha, e assim modificar a consulta ao banco de dados. Tudo o que o atacante precisa saber é um pouco dos comandos *SQL* e intuição para tentar adivinhar nomes usados pelos administradores do banco de dados para as tabelas e campos [19].

Na prática, um ataque de *SQL Injection* funcionaria da seguinte maneira:

```
SELECT produto FROM tabela WHERE nomeproduto = 'usuário insere nome do produto';
```

O comando acima irá retornar o campo produto de uma tabela onde os valores inseridos pelo usuário no campo correspondente ao nome do produto forem iguais aos valores mantidos no banco de dados. Os valores inseridos pelo usuário na aplicação web será usado para completar o comando *SQL*. Se um atacante insere os seguintes valores:

```
teste' OR 'x' = 'x
```

a sintaxe da consulta passar a ser a seguinte:

```
SELECT produto FROM tabela WHERE nomeproduto = 'teste' OR 'x' = 'x'
```

Com a injeção de caracteres que alteram o código *SQL*, a consulta irá retornar valores do campo produto onde o nome do produto for teste, ou onde x for igual a x. Como x é igual a x em qualquer situação, o teste lógico realizado pelo operador *OR* irá retornar um resultado verdadeiro e todos os valores do campo produto serão listados.

Caso um comando *SELECT* seja executado em uma loja online vulnerável a ataques de *SQL Injection*, é possível que um atacante consiga visualizar informações de clientes, como endereço, CPF, e até mesmo o número do cartão de crédito. Para Sadeghian [21], a flexibilidade da linguagem *SQL* permite que essa técnica de ataque possa ser usada com muita frequência nos dias atuais e por isso lojas online devem tomar cuidado com seus sistemas web, pois um ataque bem sucedido pode levar a empresa à falência. Já para OWASP [22], esse tipo de ataque ocorre porque em caso de sucesso, o atacante pode obter acesso completo ao banco de dados de um sistema, tornado-se um alvo atrativo. O autor também destaca que existem significativas vulnerabilidades *SQL*, o que não pode ser considerado aceitável, visto que é simples evitar que tais vulnerabilidades existam.

4.1. Categorias dos Ataques de SQL Injection

No trabalho publicado por Kumar [23], os autores classificam os ataques de *SQL Injection* em quatro categorias:

- Manipulação *SQL*: É o processo de modificar as instruções *SQL* usando várias operações como `UNION`. Outra maneira seria alterar a cláusula `WHERE` da instrução *SQL* para obter resultados diferentes.
- Injeção de Código: é o processo de inserção de uma nova instrução *SQL*.
- Injeção de Chamadas de Função: é o processo de inserção de várias chamadas de funções de banco de dados em uma instrução *SQL* vulnerável. Estas chamadas de funções podem fazer uma chamada de sistema operacional ou manipular os dados no banco de dados.
- Estouro de *Buffer*: é causado pelo uso de injeção de chamadas de funções. Para a maioria dos bancos de dados *open source* e comerciais, correções estão disponíveis. Este tipo de ataque é possível quando o servidor não está atualizado com as correções necessárias.

Já para Sadeghian [21], os ataques de *SQL Injection* podem ser divididos em três categorias:

1. *Inband*: A informação irá ser extraída a partir do mesmo canal utilizado para o ataque, que é o método mais simples. Por exemplo, a lista de usuários será exibida na página atual.
2. *Out-of-band*: a informação extraída é enviada de volta para o atacante usando um outro canal, tais como e-mail.
3. Inferencial: Também conhecido como *Blind Injection*, os dados são enviados de volta diretamente para o atacante. No entanto, o atacante pode reconstruir

os dados para tentar outros ataques e observar o comportamento da aplicação web.

4.2. Ferramentas para Ataques de SQL Injection

4.2.1. SqlMap

O SqlMap (http://sqlmap.sourceforge.net/) é uma poderosa ferramenta em linha de comando e de código aberto para a verificação de vulnerabilidades à ataques de *SQL Injection* em sistemas web. Com essa ferramenta é possível automatizar o processo de detecção e exploração dessas vulnerabilidades pois ela possui um mecanismo de detecção com várias funções e oferece suporte aos principais Sistemas Gerenciadores de Bancos de Dados (SGBDs). Tais características fazem do SqlMap uma ferramenta essencial para que profissionais de segurança da informação possam realizar testes de intrusão [24].

Uma vez que as vulnerabilidades foram identificadas, a ferramenta permite que o usuário consiga recuperar várias informações do banco de dados, tais como sessões, nomes de usuário e *hashes* de senha, tabelas, colunas, e toda a estrutura do banco de dados. A ferramenta ainda consegue identificar padrões dos *hashes* de senhas e oferece suporte para quebrá-los, usando ataque de força bruta baseado em dicionário [24].

De acordo com o trabalho publicado por Ciampa et. al [25], o SqlMap implementa três técnicas diferentes para explorar vulnerabilidades de injeção SQL:

- Inferencial: a ferramenta anexa, para um determinado parâmetro no pedido HTTP, uma instrução *SQL* `SELECT` sintaticamente válida. Para cada resposta HTTP, a ferramenta determina o valor da declaração de saída, analisando-o caractere por caractere, sendo possível avaliar o comportamento do sistema web.
- Consulta *UNION* (inband): a ferramenta anexa, para o parâmetro de destino no pedido HTTP, uma instrução *SQL* sintaticamente válida começando com um "`UNION ALL SELECT`".
- Suporte a Consultas Batched (stacted): a ferramenta testa se a aplicação web oferece suporte a consultas empilhadas, ou seja, se o SGBD aceita uma sequência de instruções *SQL* separadas por um " `;` ".

4.2.2. Google *Dorks*

Quando não há tratamentos dos dados de entrada para um banco de dados *SQL* através de um sistema web, é possível que haja a possibilidade de injetar códigos maliciosos. Caso o sistema web não tenha nenhum mecanismo de verificação dos dados de entrada e a consulta seja executada no banco de dados, isso pode dar origem a um novo problema, que são as mensagens de erro. Essas mensagens geradas pelo banco de dados podem conter informações sensíveis sobre o SGBD. Nesse caso, além do tratamento dos dados de entrada, o administrador do banco de dados também deve se preocupar com as mensagem exibidas no retorno de uma consulta. Dependendo da mensagem exibida, o atacante consegue identificar se aquele sistema está vulnerável ou não a ataques de *SQL Injection*.

Uma maneira de identificar sites que possuem variáveis dinâmicas, ou seja, variáveis que recebem os valores passados pelo usuário para formar a consulta a um banco de dados, é utilizar alguns operadores avançados do Google. Qualquer usuário pode montar um Google *Dork* para esse tipo de pesquisa. Vários sites na internet disponibilizam listas de Google *Dorks* que identificam variáveis dinâmicas, o que facilita a busca por um alvo. Depois de realizar a pesquisa, o atacante terá uma lista de sites específicos para começar a verificar quais deles retornam mensagens que podem ser usadas para um ataque de *SQL Injection*.

4.3. Evitando os Ataques

Propor uma solução única para os ataques de *SQL Injection* é muito difícil por causa da flexibilidade da linguagem *SQL*. Sadeghian [21] propõem dois modelos que devem ser usados em paralelo: Consultas Parametrizadas e Configuração Inteligente do Sistema Gerenciador de Banco de Dados.

As Consultas Parametrizadas são usadas pelo desenvolvedor no momento da codificação. Espaços devem ser reservados nas consultas *SQL* para receberem os valores das variáveis. As instruções *SQL* são executadas sem as variáveis e depois com elas. Assim, mesmo que um atacante passe comandos *SQL*, estes serão tratados apenas como strings normais.

> Os procedimentos armazenados têm o mesmo efeito que o uso de instruções preparadas quando implementada de forma segura*. Eles exigem que o desenvolvedor defina o código SQL em primeiro lugar, e, em seguida, passe os parâmetros. A diferença entre as declarações preparadas e procedimentos armazenados é que o código SQL para um procedimento armazenado é definido e armazenado no próprio banco de dados e, em seguida, chamado a partir da aplicação. Ambas as técnicas têm a mesma eficácia na prevenção de injeção SQL então sua organização deve escolher qual abordagem faz mais sentido para ela [22].

A Configuração Inteligente do SGBD, sugere que sejam criados usuários diferentes e com privilégios restritos para cada um. O administrador pode conceder privilégios para certas páginas de apenas consulta. Nesse caso um atacante apenas conseguirá usar o comando *SELECT*, excluindo a possibilidade de alteração do banco de dados, o que não isenta a possibilidade de visualização do conteúdo armazenado.

Outro método empregado para diminuir os problemas com *SQL Injectio*n é tratar os valores de entrada para que as variáveis do sistema web não considere valores que representem um código *SQL* malicioso. Controlar o tamanho de uma determinada variável e eliminar caracteres especiais usados pelo banco de dados são exemplos de como tentar tratar esses valores.

> Esta técnica funciona assim. Cada SGBD suporta um ou mais esquemas de caracteres de escape específicos para certos tipos de consultas. Se você usar os caracteres de escape adequados para toda a entrada fornecida pelo usuário, o DBMS não interpretará essa entrada como código SQL escrito pelo desenvolvedor, evitando assim eventuais vulnerabilidades de injeção SQL [22].

Ataque de *SQL Injection* têm impacto direto na integridade e confidencialidade da informação, causando enormes prejuízos para o negócio de qualquer instituição. O

ideal é que os administradores de banco de dados e os desenvolvedores dos sistemas web consigam combinar os métodos e técnicas de prevenção para que os riscos de um ataque bem sucedido sejam minimizados.

5. Estudo de Caso

Este estudo de caso foi desenvolvido com o propósito de mostrar ao leitor como é simples o uso das técnicas de Google *Hacking* para detectar um alvo de ataque e conseguir acesso a banco de dados através de falhas de segurança em sistemas web com variáveis dinâmicas que permitem o uso de *SQL Injection*.

Primeiramente definimos um escopo para o teste de detecção de sistemas web vulneráveis a *SQL Injection*. Restringimos a consulta utilizando os operadores avançados do Google para formar um *dork* específico para este estudo de caso. Escolhemos pesquisar apenas sistemas web com variáveis dinâmicas que estão no domínio *edu.br*. O Goole *Dork* ficou assim:

```
inurl:"php?id=" site:edu.br
```

O operador *inrul* irá restringir os resultados à apenas sites que contenham em sua url o valor "*php?id=*", nesse caso, sistemas que usam variáveis dinâmicas passadas como parâmetro para formar a consulta que será executa no banco de dados. O operador *site* irá retornar resultados apenas no domínio "*edu.br*".

A Figura 1 mostra os resultados obtidos com o Google *Dork* no mecanismo de buscas do Google.

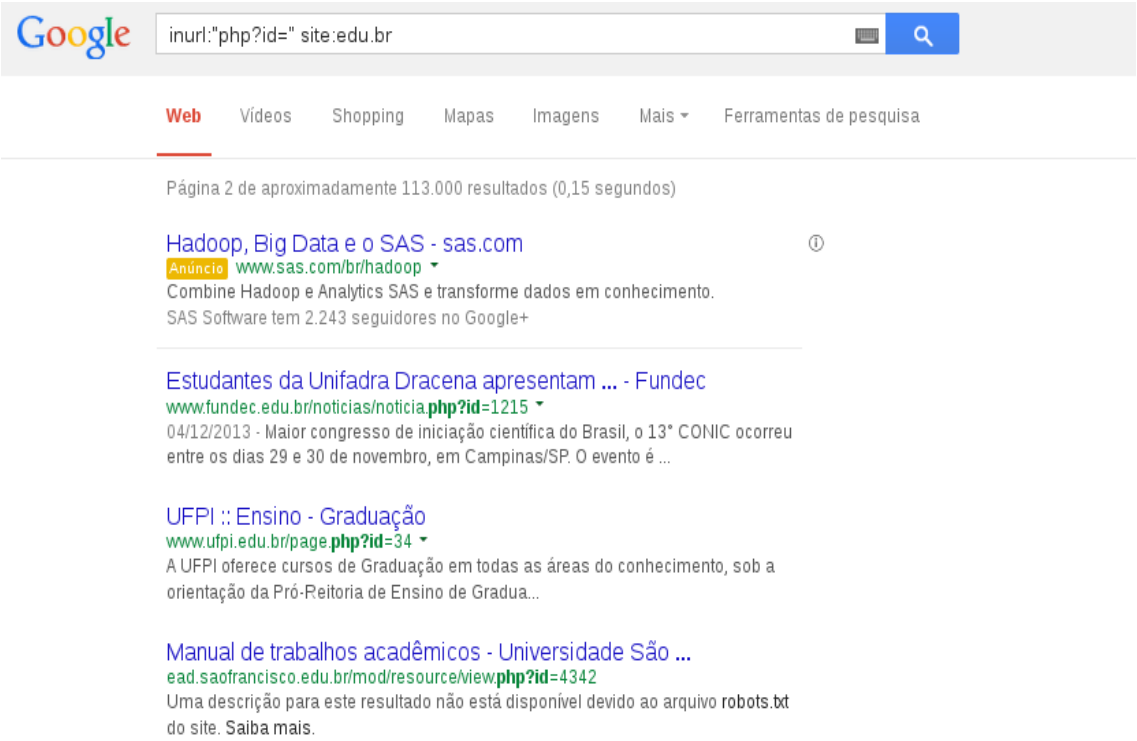

Figura 1: Pesquisa Usando o Google *Dork*

A pesquisa retornou cerca de 113.000 páginas. O próximo passo é identificar uma página que seja vulnerável a ataques de *SQL Injection*. Para isso, digitamos um apóstrofo ao final do endereço da página, que é onde a variável do sistema recebe os parâmetros necessários para a consulta ao banco de dados. Caso o sistema web interprete o apóstrofo, uma mensagem de erro de sintaxe *SQL* é mostrada, apontando que o sistema web não faz o tratamento adequado do conteúdo das variáveis que são passadas ao SGBD. Caso o sistema não seja vulnerável, o apóstrofo será ignorado e a página será exibida sem nenhum problema. A Figura 2 mostra um exemplo de um sistema web vulnerável.

Figura 2: Mensagem de Erro do SGBD

Após identificar o alvo, o atacante pode usar a ferramenta SqlMap para automatizar o processo de injeção de código *SQL* e ter acesso ao banco de dados. A ferramenta utiliza vários métodos de *SQL Injection* para explorar a vulnerabilidade existente. O comando usado para iniciar o processo de exploração da falha é mostrado a seguir:

```
sqlmap -u http://siteexemplo.edu.br/popUpVisualizar.php?id=49557 --dbs --random-agent –ignore-proxy
```

As opções do comando servem para indicar a url a ser explorada (-u), identificar o SGBD e os bancos de dados existentes (--dbs), usar cabeçalhos de requisições HTTP de diferentes navegadores de forma aleatória (--random-agent) e ignorar conexões com proxy (--ignore-proxy). A Figura 3 mostra o resultado desse comando com a página vulnerável identificada anteriormente.

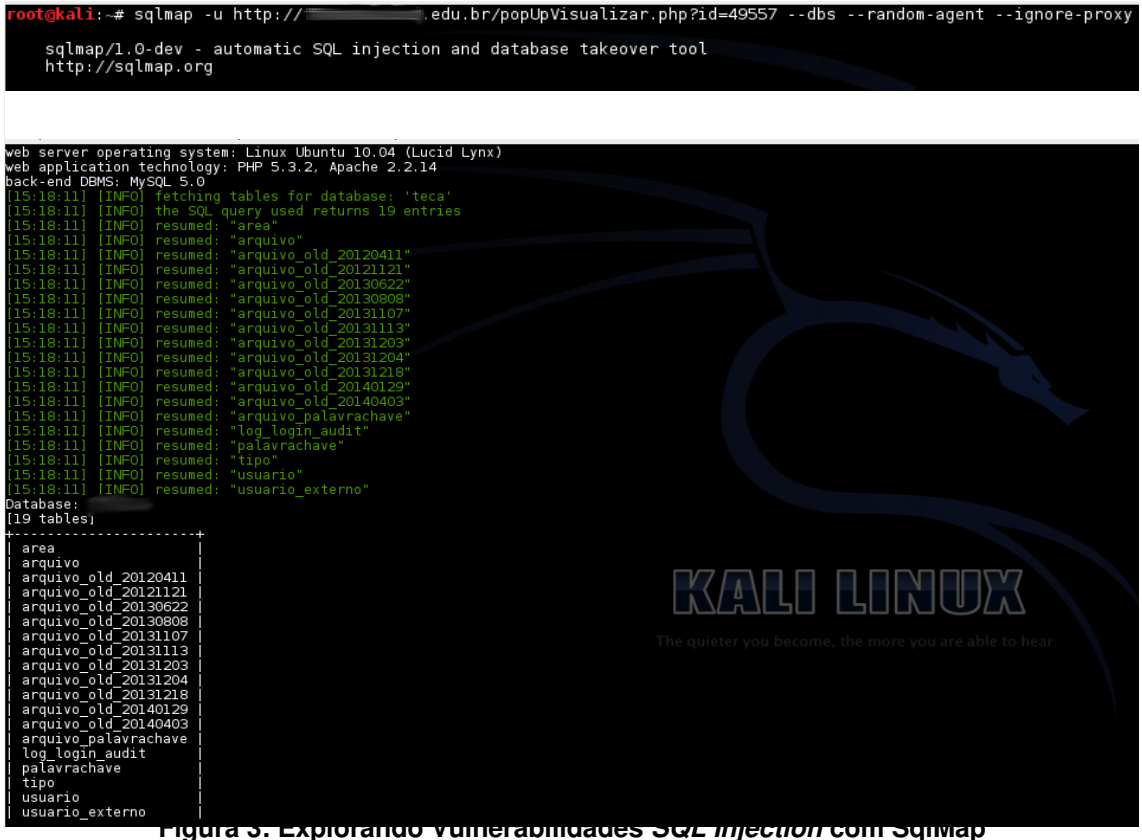

Figura 3: Explorando Vulnerabilidades *SQL Injection* com SqlMap

O procedimento de análise da estrutura de um banco de dados pode prosseguir para as colunas de uma tabela e até mesmo para os valores armazenados nessas colunas. A ferramenta SqlMap é completa nesse sentido. Além de permitir que seja feito uma cópia da estrutura do banco de dados e do próprio conteúdo do banco, a ferramenta ainda consegue identificar colunas que armazenam *hashes* de senhas, possuindo a opção de tentar descobrir a senha por ataque de força bruta a partir de um dicionário de senhas da própria ferramenta ou por valores e dicionários passados pelo atacante. O comando usado para esse tipo de procedimento é mostrado a seguir:

```
sqlmap -u http://siteexemplo.edu.br/popUpVisualizar.php?id=49557 -D database_exemplo -T usuario --columns --dump
```

As opções do comando indicam uma tabela a ser explorada (-T), a estrutura de colunas (*--column*) e o despejo (*--dump*), ou cópia, da estrutura e do conteúdo da tabela para o dispositivo do atacante. A Figura 5 mostra o resultado do comando anterior e a Figura 6 mostra a ferramenta SqlMap descobrindo as senhas a partir do *hash*.

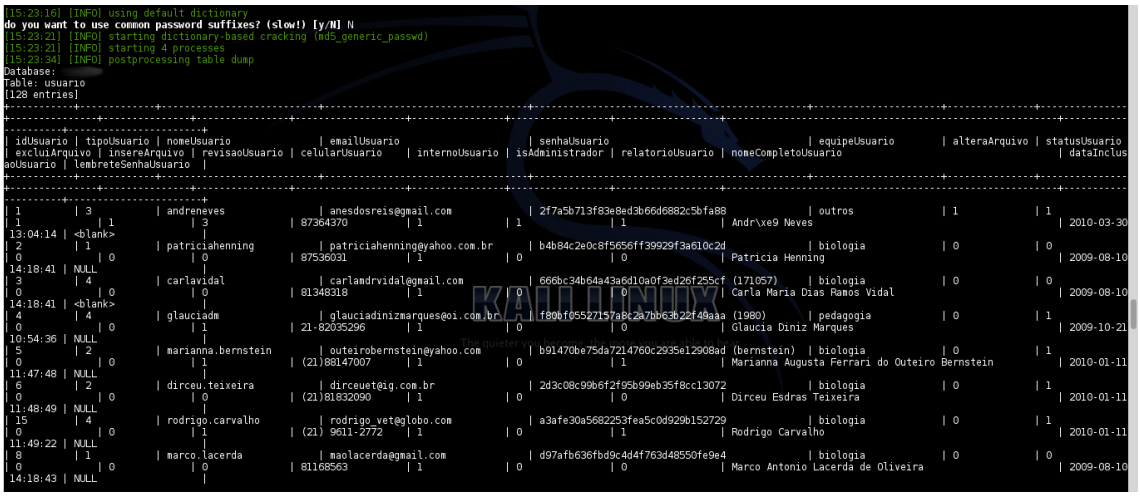

Figura 5: *Dump* de Uma Tabela do Alvo

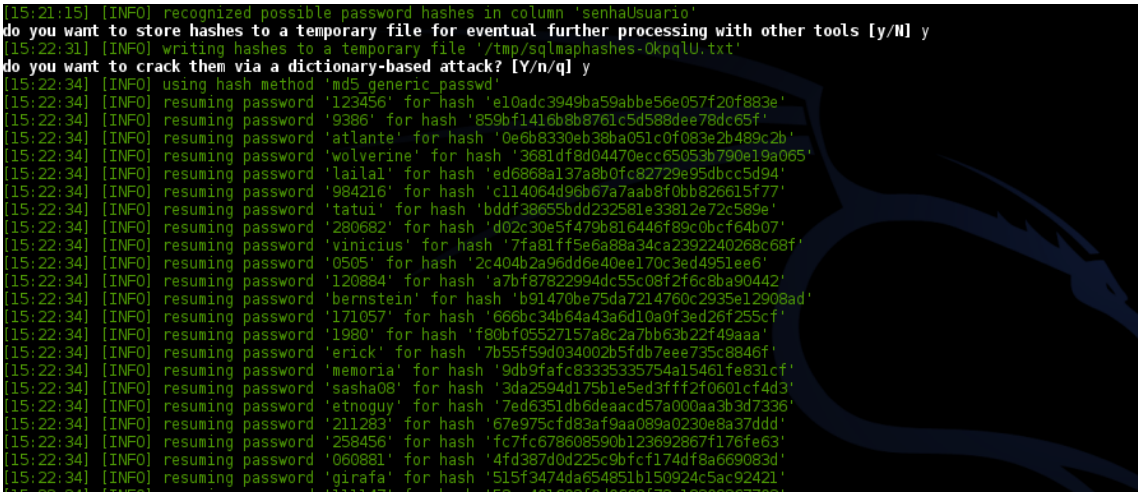

Figura 6: Descobertas de Senhas a Partir do *Hash* pela Ferramenta SqlMap

6. Conclusão

Este artigo mostra como um atacante consegue extrair dados sensíveis de um sistema web aplicando técnicas relativamente simples, e usando ferramentas de fácil acesso. Muitas organizações dependem dos seus sistemas web para a manipulação de informações valiosas e nem todas conseguem tratar a segurança da informação de maneira correta. Atacantes estão a todo momento buscando novas formas de ferir os princípios de integridade, disponibilidade e confidencialidade da informação e cabe aos profissionais de segurança e administradores de sistemas estarem atualizados quanto às vulnerabilidades existentes.

Normas específicas, como a ISO/IEC 27002, estão disponíveis para auxiliar no processo de elaboração e manutenção de uma política que mantenha os pilares da segurança da informação. Avaliar o tipo de informação que pode ser veiculada em sistemas web também faz parte do processo de segurança, porém, no caso de ataques utilizando as técnicas descritas nesse trabalho, outros mecanismos de defesa devem ser adotados e colocados em prática.

Ataques de *SQL Injection* acontecem por negligência dos profissionais responsáveis pela implantação e manutenção do sistema ou por falhas nos softwares de terceiros adotados pela organização. De qualquer modo, a passagem de parâmetros em um sistema web deve ser testada a fim de detectar códigos inseridos pelo usuário, que podem comprometer o banco de dados. Da mesma forma, a saída gerada pelo banco de dados pode conter informações sensíveis e também deve ser verificada. A construção de um sistema web pautado por tipos de consultas parametrizadas e a configuração inteligente de um SGBD são recomendadas.

No caso do Google *Hacking*, a recomendação é que os responsáveis pelo sistema web analise periodicamente os *dorks* existentes e aplique a técnica em seu próprio sistema para verificar que tipos de informações podem ser extraídas. Vários softwares de testes de vulnerabilidade já estão adotando os bancos de dados de *dorks* disponíveis na Internet para realizar uma verificação completa e identificar quais sistemas estão divulgando informações sensíveis e quais deles podem ser explorados pelas técnicas de SQL *Injection*.

6.1. Trabalhos Futuros

Existem inúmeras possibilidades de uso das técnicas de ataques referenciadas neste trabalho. Aqui apresentamos um cenário específico para sistemas web de ambientes educacionais. Sobre essa perspectiva, há a intenção, em trabalhos futuros, de se realizar um levantamento das vulnerabilidades existentes em redes sociais usando as mesmas técnicas: Google *Hacking* e SQL *Injection*.

Uma proposta de trabalho interessante é analisar até onde o Google *Hacking* pode contribuir para que ataques de injeção de códigos sejam considerados um dos mais utilizados e o mais crítico.

O cresceste uso de ferramentas que criam sites a partir de códigos prontos é um atrativo para pessoas que precisam de um site mas não possuem conhecimento necessário para o desenvolvimento e nem estão dispostas a pagar um profissional. Entretanto, devemos estar atentos ao nível de segurança que essas ferramentas

proporcionam. Este tema também deve ser explorado e um estudo divulgado para informar esse perfil de usuários desses sistemas web.

7. Referências Bibliográficas

[1] Top 10 2013 - OWASP. Open Web Application Security Project. Disponível em: <https://www.owasp.org/index.php/Top_10_2013-Top_10> Acesso em: 10 Mai. 2014.

[2] ISO/IEC 27002: Fundamentos. PMG Solutions. Manual de Treinamento. 2011. Disponível em: <http://www.pmgeducation.com.br/demo/e-book_ISO_IEC_27002_Online_DEMO.pdf > Acesso em: 20 Mai. 2014.

[3] CASTELLS, M. A Sociedade em Rede - A Era da Informação: Economia, Sociedade e Cultura. Volume 1. 8a. ed. São Paulo, S.P.: Paz e Terra, 2005.

[4] FERREIRA, Fernando Nicolau Freitas. Segurança Da Informação. 1a. ed. 176p. Editora Ciência Moderna, 2003.

[5] ALEXANDRIA, João Carlos Soares de. GESTÃO DA SEGURANÇA DA INFORMAÇÃO – Uma Proposta Para Potencializar a Efetividade da Segurança da Informação em Ambiente de Pesquisa Científica. Tese (Doutorado) Instituto de Pesquisas Energéticas e Nucleares. Universidade de São Paulo, São Paulo, 2009.

[6] MALANDRIN, L. J. A. A. Modelo de Suporte a Políticas e Gestão de Riscos de Segurança Voltado à Terceirização de TIC, Computação em Nuvem e Mobilidade. Dissertação (Mestrado) - Departamento de Engenharia de Computação e Sistemas Digitais. Escola Politécnica da Universidade de São Paulo. São Paulo, 2013.

[7] FRANCISCATTO, Roberto. Notas de Aula: Aula 2 - ISO 27002. Especialização em Gestão de TI. Universidade Federal de Santa Maria, 2013. Disponível em: <http://www.cafw.ufsm.br/~roberto/trabalhos/aulas/Aula%202%20-%20ISO-IEC %2027002.pdf> Acesso em: 28 Abr. 2014.

[8] Criação do site de buscas Google muda a história da internet. Designer Zone. Disponível em: <http://www.designerzone.com.br/ultimas-noticias/criacao-do-site-de-busca-google-muda-a-historia-da-internet> Acesso em: 30 Abr. 2014.

[9] Google completa 15 anos. Relembre a história em fotos. Olhar Digital. Disponível em: <http://olhardigital.uol.com.br/noticia/37874/37874> Acesso em: 05 Mai. 2014.

[10] Google lidera buscas no Brasil, mas Bing é mais relevante, diz estudo. Olhar Digital. Disponível em: <http://olhardigital.uol.com.br/noticia/38245/38245> Acesso em: 20 Mai. 2014.

[11] Pesquisa em Frações de Segundos. Google Inc. Disponível em: <http://static.googleusercontent.com/media/www.google.com/pt-BR//intl/pt-BR/insidesearch/howsearchworks/assets/searchInfographic.pdf> Acesso em 15 Mai. 2014.

[12] Rastreamento e indexão - Por dentro da pesquisa. Google Inc. Disponível em: <http://www.google.com/intl/pt-BR/insidesearch/howsearchworks/crawling-indexing.html> Acesso em: 20 Mai. 2014.

[13] Razões para se usar o Google. Google Inc. Disponível em: <http://www.google.com.br/why_use.html> Acesso em: 14 Mai. 2014.

[14] Dicas e truques de pesquisa - Por dentro da pesquisa. Google Inc. Disponível em: <http://www.google.com/intl/pt-BR/insidesearch/tipstricks/all.html> Acesso em: 15. Mai. 2014.

[15] Long, J. Google Hacking for Penetration Testers, Volume 2. Syngress Publishing - Elsevier, 2008.

[16] Toledo, A. S. d. O. and Moraes, S. H. d. Google hacking. Pós em Revista, (6):358–367, 2012.

[17] McGuffee, J. W. and Hanebutte, N. Google hacking as a general education tool. J. Comput. Sci. Coll., 28(4):81–85, 2013.

[18] Google Hacking Database, GHDB, Google Dorks. Exploit Database. Disponível em: <http://www.exploit-db.com/google-dorks/> Acesso em: 29 Abr. 2014.

[19] What is Google Hacking? Acunetix. Disponível em: <https://www.acunetix.com/websitesecurity/google-hacking/> Acesso em: 29 Abr. 2014.

[20] Billig, J., Danilchenko, Y., and Frank, C. E. Evaluation of google hacking. In Proceedings of the 5th Annual Conference on Information Security Curriculum Development, InfoSecCD'08, pages 27–32, New York, NY, USA. ACM, 2008.

[21] Sadeghian, A., Zamani, M., and Ibrahim, S. Sql injection is still alive: A study on sql injection signature evasion techniques. In Informatics and Creative Multimedia (ICICM), 2013 International Conference on, pages 265–268, 2013a.

[22] SQL Injection - OWASP. Open Web Application Security Project. Disponível em: <https://www.owasp.org/index.php/SQL_Injection> Acesso em: 10 Mai. 2014.

[23] Kumar, P. and Pateriya, R. A survey on sql injection attacks, detection and prevention techniques. In Computing Communication Networking Technologies (ICCCNT), 2012 Third International Conference on, pages 1–5, 2012.

[24] Sqlmap: Automatic SQL injection and database takeover tool. Sqlmap. Disponível em: <http://sqlmap.org/> acesso em: 10 Mai. 2014.

[25] Ciampa, A., Visaggio, C. A., and Di Penta, M. A heuristic-based approach for detecting sql-injection vulnerabilities in web applications. In Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems, SESS'10, pages 43–49, New York, NY, USA. ACM, 2010.