Curso de Sistemas de Informação
Universidade Estadual de Mato Grosso do Sul

ReadFunSec: Aplicação para análise de funções de segurança do arquivo de configuração do PHP

Raisson Fernando Martins Dias Vieira

Prof. Esp. Alcione Ferreira (Orientador)
Prof. MSc. André Chastel (Co-orientador)

Dourados - MS
2018

Curso de Sistemas de Informação
Universidade Estadual de Mato Grosso do Sul

ReadFunSec: Aplicação para análise de funções de segurança do arquivo de configuração do PHP

Raisson Fernando Martins Dias Vieira

Novembro de 2018

Banca Examinadora:

Prof. Esp. Alcione Ferreira (Orientador)
Área de Computação - UEMS

Prof. MSc. André Chastel (Co-orientador)
Área de Computação - UEMS

Prof. Dr. Ricardo Luís Lachi
Área de Computação - UEMS

Prof. MSc. Jéssica Bassani de Oliveira
Área de Computação - UEMS

ReadFunSec: Aplicação para análise de funções de segurança do arquivo de configuração do PHP

Raisson Fernando Martins Dias Vieira

Este exemplar corresponde à redação final da monografia da disciplina Projeto Final de Curso devidamente corrigida e defendida por Raisson Fernando Martins Dias Vieira e aprovada pela Banca Examinadora, como parte dos requisitos para a obtenção do título de Bacharel em Sistemas de Informação.

Dourados, 24 de novembro de 2018.

Prof. Esp. Alcione Ferreira (Orientador)

Prof. MSc. André Chastel (Co-orientador)

Dedico este trabalho a toda a minha família, minha esposa e amigos por tudo que fizeram por mim.
Aos meus professores e colegas de universidade que estiveram comigo nessa longa jornada, onde estivemos juntos por muitos momentos em grandes desafios em busca de conhecimento e sempre adquirindo tanto experiência no sentido acadêmico quanto virtudes no sentido pessoal.

"A única maneira de fazer um bom trabalho é amando o que você faz. Se você ainda não encontrou, continue procurando. Não se desespere. Assim como no amor, você saberá quando tiver encontrado"

Steve Jobs

Agradecimentos

Agradeço primeiramente a Deus por todas as oportunidades que colocou em meu caminho.

Aos meus pais Luiz e Osana, por tudo que fizeram por mim. Pela dedicação e apoio dado para que eu conseguisse todos os meus objetivos.

Aos meus avós Oracília, Norvino, Tereza e João pelo auxílio que deram aos meus pais em minha educação e pelo amor dado em toda minha vida.

A minha esposa Karita pelo incentivo, e pela paciência nos momentos em que estive ausente me dedicando aos estudos.

Aos meus irmãos Rian, Felipe e Falquier pela ajuda nos dias em que precisei para honrar meus compromissos.

Aos colegas de faculdade pelos bons momentos de concentração, aplicação aos estudos, descontração e alegria durante os tempos de universidade.

Ao meu orientador Prof. Alcione Ferreira, ao co-orientador Prof. Msc. André Chastel pelo empenho e disponibilidade em conduzir este trabalho, proporcionando a mim uma grande experiência e acúmulo de conhecimento.

A todo o corpo docente do curso de Sistemas de Informação da UEMS pela dedicação em transmitir conhecimento a mim e aos demais alunos, ao respeito e atenção durante cada minuto de aula que estivemos presente nesses anos de formação acadêmica.

Ao meu colega de trabalho Jesse Dos Santos pela ajuda em relação a flexibilidade de horários para que eu pudesse estar presente na sala de aula e pela paciência nos momentos em que priorizei os estudos e precisei me ausentar no trabalho.

A cada servidor da UEMS pela dedicação ao trabalho, fazendo com que esta instituição se mantenha um lugar agradável, um ambiente calmo, aconchegante e confortável onde tive o prazer não só estudar mais também passar por momentos inesquecíveis e que levarei para toda a vida.

Obrigado a todos, pelo apoio e companheirismo.

Resumo

A internet revolucionou o mundo comercial, tanto para compra, venda e transações em geral. A maneira que o mercado evolui tecnologicamente, obriga desde pequenas a grandes empresas a se informatizarem. Muitas vezes isso pode trazer riscos, pois, com um investimento relativamente pequeno as empresas conseguem adquirir um *home page*, porém as vezes a falta de experiência e conhecimento de programadores fazem com que essas empresas se tornem vulneráveis diante de ataques virtuais. A fragilidade de seus sistemas acabam se tornando porta de entrada a atacantes, ou seja, usuários mal intencionados. Segurança de servidores é um assunto muito importante quando se fala em segurança da informação, mas muitas vezes desconhecido por atuantes da área. Esse trabalho pretende tratar sobre esse assunto, tendo o arquivo de configuração do PHP (Hypertext Preprocessor) e suas funções de segurança como objeto de estudo, e a criação de uma ferramenta de análise de código que faz a leitura deste arquivo de configuração e verifica se as funções de segurança podem acarretar em vulnerabilidades no sistema.

Palavra-chave: Programação, Segurança, PHP.

Sumário

Agradecimentos **xi**

Resumo **xiii**

1 Introdução **1**
1.1 Objetivo 2
1.2 Motivação 2
1.3 Organização do trabalho 3

2 Conteúdo Bibliográfico **5**
2.1 PHP 5
2.2 Ferramentas de Análise de Código 6
2.3 Ataques e Vulnerabilidades 6
2.3.1 Footprints e Engenharia Social 7
2.3.2 Enumeração 7
2.3.3 Ataques Remotos 7
2.3.4 DoS e DDoS 9
2.3.5 Cookies 10
2.3.6 Buffer Overflow 10
2.3.7 Sql Injection 10
2.3.8 LFI 11
2.3.9 Ataque a Segurança em Servidores 11
2.3.10 Buscando Exploits na Internet 12
2.4 Ferramentas de Segurança 13
2.4.1 Firewall e Proxy 13
2.4.2 Sistemas de Detecção de Intrusão 13
2.5 O Arquivo de Configuração do PHP 14

3 ReadFunSec **15**
3.1 Preparando o Ambiente 15
3.2 ReadFunSec e seu funcionamento 19

4 Conclusão **23**

Lista de Siglas

C - Linguagem de programação estruturada criada em 1972 por Dennis Ritchie.

CGI - Common Gateway Interface.

CVE - Common Vulnerabilities and Exposures - Vulnerabilidades e Exposições Comuns.

FMI - Fundo Monetário Internacional.

HTML - Hypertext Markup Language - Que no português significa linguagem de marcação de texto, linguagem usada para criação de páginas Web.

HTTP - Hypertext Transfer Protocol - Protocolo de Transferência de Hypertexto.

IDS - Intrusion Detection System

JSON - JavaScript Object Notation - Arquivo de Texto com Formato Independente de Linguagens de Programação.

NVD - National Vulnerability Database - Banco de Dados Nacional de Vulnerabilidades.

PHP - Hypertext Preprocessor - Linguagem de programação voltada para Web.

SCAP - Security Content Automation Protocol - Protocolo de Automação de Conteúdo de Segurança.

SSH - Secure Shell - Protocolo de Rede Criptográfico.

SSL - Secure Sockets Layers - Protoloco de Segurança de Telecomunicações Via Internet.

TI - Tecnologia da Informação.

UID - User Identification - Identificação do usuário.

Lista de Figuras

3.1 instalação do gerenciador de pacotes para Windows. 16
3.2 instalação do NodeJS para Windows. 16
3.3 verificando versão do NodeJS e NPM no Windows. 16
3.4 acessando NodeJS na versão 10.x para Linux (Ubuntu e Debian). 17
3.5 instalação do NodeJS para Linux (Ubuntu e Debian). 17
3.6 verificando versão do NodeJS e NPM no Linux. 17
3.7 página principal do ReadFunSec. 19
3.8 relatório gerado com as funções encontradas e sua respectiva linha. 20
3.9 acesso do cliente ao servidor através de funções middleware. 21

Capítulo 1

Introdução

A quantidade de aplicações **web!** vem crescendo conforme o computador vem se tornando uma necessidade básica, tanto para uso doméstico quanto para uso empresarial.

De acordo com Medeiros (2012), o PHP é a linguagem de programação mais utilizada para desenvolvimento *web*, e a facilidade de se usar a mesma faz com que os programadores desenvolvam aplicações inseguras devido à inexperiência e falta de conhecimento no assunto.

Segundo Afonso (2011), empresas não dão a devida importância quando se fala de segurança e o baixo investimento nesse quesito, fez com que chegasse ao estado problemático atual, no qual muitas empresas e até mesmo órgãos públicos acabam sendo vítimas de ataques e podem ter seus dados roubados, alterados ou destruídos.

Um servidor de aplicação *web* hospeda o código, escrito em PHP, que é executado no lado do servidor, de um programa ou sistema voltado para uso na *internet*. Esse servidor é quem fica responsável pelas regras de negócio, segurança do sistema e armazenamento dos dados. Outros serviços também disponíveis em servidores de aplicação são: tolerância à falhas, balanceamento de cargas, gerenciamento de componentes, gerenciamento de transações, console de gerenciamento e segurança (Pereira, 2013).

De acordo com Cert.Br (2012), para que um sistema computacional seja considerado seguro é necessário que atenda a três requisitos básicos: confidencialidade, integridade e disponibilidade. É preciso que seja considerado alto o nível de segurança em que esse sistema seja desenvolvido, já que na maioria das vezes esses sistemas são encarregados de manipular dados extremamente confidenciais, como senhas, números de cartões de créditos entre outros dados pessoais.

Grandes organizações como FMI, Lockheed Martin, Google, Sony, entre outras, sofreram algum tipo de ataque cibernéticos, o que gerou discussões em relação a importância de segurança em TI (Cortez and Kubota, 2012).

1.1 Objetivo

A sociedade de hoje evolui a cada dia quando o assunto é tecnologia. O avanço da *internet* e de aparelhos conectados cresce a cada dia, deixando cada vez mais, portas abertas para cibercrimes.

Neste trabalho foi desenvolvido uma ferramenta de análise de código, que possibilita fazer uma análise do arquivo de configuração do PHP, encontrar funções de segurança com valores considerados como vulneráveis ao sistema, permitindo que essas funções sejam analisadas e corrigidas com mais eficiência e rapidez.

1.2 Motivação

A falta de conhecimento da linguagem de programação utilizada ou de controle durante o desenvolvimento de *softwares* podem ocasionar em inserção de vulnerabilidades em sistemas computacionais (Afonso, 2011).

Nos últimos tempos, a sociedade mundial acompanha a maneira com que ataques virtuais estão se desenvolvendo. Empresas de todos os ramos, usuários domésticos, instituições bancárias e ate mesmos órgãos públicos são atacados por *hackers* com diversas intenções. Uma série de ataques cibernéticos em grandes instituições como bancos e órgãos governamentais atraíram a atenção para o assunto (Cortez and Kubota, 2012).

São vários os motivos que levam *hackers* a desferir ataques virtuais, invasões com objetivos criminosos ou simples diversão são alguns deles além da busca por demonstração de poder, prestígio e motivações financeiras, ideológicas e comerciais (Cert.Br, 2012).

A importância da segurança da informação e a necessidade de ter aplicações seguras motivou a desenvolver uma ferramenta capaz de automatizar a análise do arquivo de configuração do PHP para que os profissionais responsáveis em implantar o sistema possam analisar as funções de segurança e verificar se estão devidamente ajustadas de acordo com suas necessidades. É importante que esta análise seja feita de maneira criteriosa, pois caso uma falha passe despercebida, ela pode gerar uma vulnerabilidade no sistema.

1.3 Organização do trabalho

No **capítulo 2** é mostrado alguns conceitos importantes sobre a linguagem estudada, o PHP, a utilização de ferramentas de análise de código e sua eficiência, apresentado alguns ataques voltados para aplicações *web*, apresentado o problema e a importância da segurança em servidores de aplicação e abordado mais sobre o arquivo de configuração do PHP.

No **capítulo 3** é apresentado a aplicação desenvolvida e mostrado as tecnologias utilizadas. Também é mostrado os resultados obtidos com a análise de código no arquivo de configuração do PHP.

No **capítulo 4** é feito a conclusão do trabalho, onde é falado sobre a importância da segurança em ambiente *web* devido aos ataques em que um servidor de aplicação está exposto, e o resultado obtido com o ReadFunSec.

Capítulo 2

Conteúdo Bibliográfico

Neste capítulo serão abordados os conceitos das áreas da informática que foram bases da pesquisa para a construção deste trabalho.

2.1 PHP

O PHP é uma linguagem de programação de ampla utilização (Mehdi et al., 2016). O seu uso é facilitado por ser uma linguagem *open source*, ou código fonte aberto, e por permitir a integração com outras linguagens de programação como HTML e JavaScript.

De acordo com o Mehdi et al. (2016), a grande maioria dos sistemas operacionais, como Linux, Microsoft Windows e Mac OS proporcionam a utilização do PHP, assim como os servidores *webs* atuais.

O PHP, que é uma linguagem interpretada no servidor e foi concebida para ser utilizada exclusivamete na *web*. Segundo Sica (2006), a linguagem tem o objetivo de construir páginas pessoais com mais funcionalidades que o HTML.

A primeira versão do PHP foi escrita em Perl para funcionar como CGI (*Commom Gateway Interface*), que é o método usado para permitir a interação entre o servidor e os clientes. Devido a propagação dessa linguagem, foi disponibilizado uma parte da documentação pelo autor, Rasmus Lerdof, que deu origem ao PHP v.01 e, a partir daí, surgiram novos colaboradores que contribuiram para a evolução do PHP (Sica, 2006).

2.2 Ferramentas de Análise de Código

Programas de análise de códigos são ferramentas de apoio a programadores que visam verificar o código desenvolvido e gerar relatórios de erros, tanto sobre de falhas de programação quanto de testes de segurança. De acordo com Teixeira (2007), num ambiente de programação, a análise de código é um mecanismo de apoio ao desenvolvedor na detecção de erros e falhas. Essas ferramentas são desenvolvidas com objetivos específicos.

Medeiros (2012) desenvolveu uma ferramenta com a qual é possível analisar códigos em PHP, verificando se os mesmos permitem a injeção de códigos. O PHPSecIn auxilia na verificação de falhas de programação que possibilite injeção de SQL, XSS e PHP. A ferramenta faz a leitura do código PHP, encontra os métodos *$_GET* e *$_POST* dentro do código, e verifica se estão sendo feitas tratativas para evitar a injeção de código. Após a verificação é gerado um relatório com os trechos onde não está sendo feita nenhuma tratativa, ou seja, é possível realizar a injeção de código.

Teixeira (2007) criou um programa que verifica a vulnerabilidade de segurança em códigos desenvolvido na linguagem C. Esta ferramenta, designada Mute, gera um novo relatório através da agregação de resultados de diversas ferramentas, mas potencialmente detecta um maior número de vulnerabilidade, mantendo os falsos positivos dentro de valores aceitáveis.

Essas ferramentas, no geral, fazem a leitura do código e procuram trechos específicos, que necessitam de tratamentos especiais e verificam se os mesmos estão sendo tratados. No final da análise, a ferramenta gera um relatório apontando todas vulnerabilidades encontradas e onde estão localizadas, possibilitando ao programador que verifique seu código e corrija falhas que deixam seu programa vulnerável a ataques e erros.

2.3 Ataques e Vulnerabilidades

Apresentaremos agora alguns ataques voltados para o ambiente *web*, nos quais o principal alvo desses ataques é o servidor de aplicação. Esses ataques são realizados por *hackers* que através de computadores clientes de uma rede, exploram as vulnerabilidades e invadem os sistemas para roubar dados, indisponibilizar serviços entre outros crimes virtuais. Essas vulnerabilidades podem ser causadas por falhas de programação, de configuração do ambiente e até mesmo da infraestrutura e segurança da rede.

2.3.1 Footprints e Engenharia Social

Reconhecimento ou Footprints é o método utilizado para obter informações de um alvo usando táticas seguras sem riscos de deteção, como pesquisas na *internet* ou visitas em websites alvos com objetivos de levantar informações interessantes ao atacante para que, a partir daí, seja definido as técnicas e ferramentas para o ataque. As técnicas de reconhecimento podem ser classificadas como ativas ou passivas. Reconhecimento passivo consiste em coletar informações relativas ao alvo utilizando ferramentas públicas disponíveis, como a *internet*. Já o reconhecimento ativo é aquele onde se tem um contato direto entre o atacante e o alvo, como engenharia social, entrevistas ou contatos telefônicos (Givaroto and Santos, 2013).

Givaroto and Santos (2013) também descreve Engenharia Social como uma das técnicas mais utilizadas para o levantamento de informações, onde o atacante explora o fator humano. A credulidade, a inocência, a confiança ou a curiosidade são fatores que levam os usuários de sistemas computacionais a compartilharem informações sigilosas com desconhecidos que, muitas vezes, utilizam-se técnicas e artifícios para enganar funcionários de empresas, integrantes de redes sociais e outras pessoas que se tornam alvos de futuros ciberataques.

2.3.2 Enumeração

A enumeração tem como objetivo mapear a rede e descobrir serviços e versões que são executados no sistema alvo, facilitando a posterior pesquisa de vulnerabilidades e *exploits* específicos. Nesta etapa o atacante levanta informações como blocos de IP, serviços como HTTP, SSH, SSL, Mysql, nomes de domínios, versão do sistema operacional, protocolos de comunicação, mecanismos de autenticação, etc (Erickson, 1977).

2.3.3 Ataques Remotos

Esse tipo de ataque é realizado através de um computador que está distante do alvo e está conectado por uma rede local ou pela *internet*. Os riscos relacionados a um ataque remoto podem ser muito graves. Depois de conectado remotamente ao alvo é feito uma escalação de privilégio e assim, é possível apagar dados, desativar serviços, criar e excluir usuários (Tâmega, 2003).

É importante que todas as medidas de proteção contra ataques remotos sejam tomadas. Esses ataques se baseiam em informações referente a conexão e serviços do alvo que foram coletadas pelo atacante. Certas informações são primordiais para esses ataques como:

- Configuração de rede;
- Fraquezas a serem exploradas;
- Usuários pertencentes a rede;
- Origem da conectividade da rede.

Essas informações são coletadas através de consulta remota nos servidores aos quais se deseja atacar. Algumas informações tornam os ataques mais eficientes como saída de conectividade desse servidor, nome das máquinas conectadas, o sistema operacional instalado e nome do administrador da rede.

Após coletadas essas informações, o atacante precisa analisar os resultados obtidos com o objetivo de encontra vulnerabilidades neste sistema alvo. Uma maneira de encontrar essas vulnerabilidades é pesquisando na *internet* em *sites* específicos que disponibilizam informações sobre falhas de segurança em *softwares* e sistemas.

Depois de verificada a vulnerabilidade, é possível criar programas maliciosos ou até mesmo utilizar ferramentas como Nmap ou Metasploit para realizar a invasão e conseguir o acesso ao servidor.

O Nmap é uma ferramenta de mapeamento utilizada para encontrar computadores em uma rede. Com ele é possível descobrir os nomes e endereços *IP* dos computadores de uma rede e os sistemas e serviçoes que estão executando. Este processo é feito através de um *scan port* realizado na rede (Lyon, 1997).

O Metasploit é um *software* utilizado para analisar vulnerabilidades de segurança. Essa ferramenta é utilizada em testes de invasão. Com ela é possível invadir computadores explorando as falhas de segurança contidos no sistema operacional, banco de dados e aplicações em geral (Givaroto and Santos, 2013).

Tanto Nmap quanto Metasploit são feramentas utilizadas em sistema operacional *Linux* voltadas para testes de invasão e possuem algoritmos nativos para análise de redes e computadores.

2.3.4 DoS e DDoS

Tâmega (2003) define DoS como um ataque de negação de serviço onde um usuário ou servidor é impedido de utilizar um serviço que normalmente é disponibilizado para a rede.

Esse ataque consiste em realizar um grande número de requisições, mais do que o servidor está preparado para suportar, dessa forma o servidor encerra sua operação, seja travando ou reiniciando, devido a alta demanda. Como exemplo um servidor *web* recebendo um grande número de requisições e não seja capaz de processar todas. Esse servidor encerrará o serviço e reiniciará aplicação, deixando o site temporariamente indisponível.

Segundo Oliveira (2003), ataques DDoS são semelhantes aos DoS, porém, o primeiro tem como origem vários computadores, podendo chegar a milhares de máquinas disparando ataques DoS para um ou mais servidores. Para isso, o atacante utiliza-se de agentes, ou computadores zumbis que disparam requisições contra um alvo selecionado de acordo com a vulnerabilidade. Este ataque se transforma em um DoS de grande escala.

2.3.5 Cookies

Cookies são descrito por Tâmega (2003) como arquivos de textos enviados ao navegador que visitam as páginas no servidor e são armazenados temporariamente em modo local no computador que acessou a página *web*. Sempre que a página é acessada novamente esses arquivos são ativados. É através dos *cookies* que o servidor é informado sobre quantas vezes uma mesma página é acessada pelo mesmo navegador, mas eles também armazenam informações referente a página acessada e pode conter dados definidos pelo programador como login, senha e *e-mail*.

Falhas de segurança em navegadores podem ocasionar no roubo de *cookies* contendo informações críticas, e embora não fiquem armazenado no servidor e sim no computador cliente, as informações contidas no *cookie* podem fornecer o acesso do atacante ao servidor como se fosse o usuário que teve seus dados roubados.

2.3.6 Buffer Overflow

Buffer Overflow é uma falha que ocorre quando um processo tenta armazenar mais dados do que ele pode suportar. As informações excedentes são enviadas para um *buffer* adjacentes onde podem ser corrompidos ou sobrescrever dados inválidos. Um ataque *hacker* explora esta falha, onde o os dados extras passados pelo criminoso contém códigos maliciosos que, ao ser realocados em outro *buffer*, podem danificar arquivos e alterar o funcionamento de programas. É comum que programas escritos na linguagem de programação C tenham essa falha (Tâmega, 2003).

2.3.7 Sql Injection

De acordo com Malerba (2010), a injeção de Sql ocorre na camada do banco de dados da aplicação devido a falta de boas práticas nas filtragens de consultas em Sql. Este tipo de falha é considerado como de enorme potencialidade, pois permite ao *hacker* fazer inserções, exclusões e alterações no banco de dados da aplicação, expondo assim, dados sigilosos ou até mesmo destruindo toda a base de dados.

2.3.8 LFI

LFI (Local File Inclusion) é uma falha que permite ao invasor inserir um arquivo indevido no servidor, explorando falhas de validação e autenticação de usuários e utilizando mecanismos de inclusão dinâmicas de arquivos implementado na aplicação. A saída do conteúdo do arquivo pode levar a ataques como execução de códigos maliciosos no servidor, execução de códigos no lado do cliente, como javascript, negação de serviços e divulgação de informações confidenciais. Boas práticas de filtragens, podem evitar este tipo de ataque, como filtrando os valores de entrada de páginas *web*, evitando que o *hacker* envie como valor o caminho de diretórios contendo estes arquivos (Malerba, 2010).

2.3.9 Ataque a Segurança em Servidores

Sistemas de *softwares* que são desenvolvidos com o objetivo de fornecer infraestrutura de serviços para executar aplicações distribuídas são denominados como servidores de aplicação (Burk and Labourey, 2002).

Geralmente as aplicações em PHP são executadas em servidores de aplicação. Esses servidores fornecem serviços a computadores clientes que acessam a aplicação através da rede, seja uma rede local ou através da *internet*. Essas aplicações são responsáveis pela manipulação de grande quantidade de dados que quase sempre, são dados extremamente sigilosos, como senhas, dados pessoais e dados estratégicos de empresas e organizações (Pereira, 2013).

Terry et al. (1997) descreve como uma categoria de ameaça, ataques a servidores de *internet*, onde esses são invadidos e, com isso o intruso têm acesso a arquivos restritos armazenados no servidor, ficando possibilitado que o atacante leia, altere ou exclua esses dados.

Outra técnica de ataque é o Disfarce, nela o invasor se passa por outro usuário para ter acesso aos dados do usuário titular da conta. Geralmente esse crime acaba resultando no roubo de informações sigilosas. Se o usuário A assumir a identidade do usuário B, o usuário A terá acesso a dados e outros tipos de privilégios que o usuário B tem e é restrito a somente ele.

Outro problema comum citado por Terry et al. (1997) em relação a segurança de servidores *web* é o ataque a bibliotecas compartilhadas, já que esses sistemas possui diversas bibliotecas que são de uso coletivo no sistema, ou seja, vários usuários compartilham a mesma biblioteca.

2.3.10 Buscando Exploits na Internet

Depois de coletado todas as informações de um alvo, como versão de sistema operacional, banco de dados entre outras configurações, é possível pesquisar por falhas de segurança dessas aplicações. A exploração de um programa é a ação de induzir o computador a executar tarefas que você quer que ele faça, mesmo que o programa que esteja sendo executado seja programado para evitar tal ação. Essa exploração é feita através de falhas de segurança ou imperfeiçoes de projeto e implementações em sistemas computacionais (Erickson, 1977).

Na *internet*, alguns sites funcionam como base de dados internacionais públicas para troca de informações entre produtos sobre falhas de segurança, como CVE Details, National Vulnerabilities Database NVD, Exploit Database, etc.

A CVE (Common Vulnerabilities and Exposures) é uma dessas base de dados, onde é publicado falhas de segurança conhecidas e suas correções. Na página inicial é possível realizar as buscas por produto, fabricante ou por vulnerabilidade. A CVE usa como referências algumas plataformas como Github, Php.net, Security Focus e os próprios fornecedores de sistemas operacionais e banco de dados, como Debian.org, Postgresql.org e Technet.microsoft.com entre outros.

Já o Exploit Database é um site onde é possível buscar por ataques já descobertos para cada vulnerabilidade e fazer o download do código fonte do Exploit. No Exploit Database pode se encontrar ataques das seguintes categorias:

- Exploração remota;
- Exploração de aplicação web;
- Exploração local e escalação de privilégio;
- Negação de serviço;
- Exploit shellcode.

Cada vulnerabilidade recebe uma ID e é classificado por tipo, data de publicação, plataforma e autor. Também é possível verificar se o ataque é válido ou não. NVD (National Vunerabilities Database) é um banco de dados do governo dos Estados Unidos que é composto por dados de gerenciamento de vulnerabilidades usando o protocolo SCAP (Security Content Automation Protocol). No NVD é possível encontrar listas de verificações de segurança, falhas de *softwares* relacionadas à segurança, configurações incorretas, nomes de produtos e métricas de impacto.

2.4 Ferramentas de Segurança

Para que um sistema seja considerado seguro, é necessário que a infraestrutura e rede onde o mesmo esteja sendo executado possua medidas de segurança para garantir que essa aplicação não seja acessada indevidamente. Essas medidas de segurança são muito importantes para que os dados dos usuários sejam protegidos e as aplicacões funcionem corretamente. Existem ferramentas de *software* dedicadas a implementar essas medidas, as quais devem ser utilizadas pelos administradores da rede.

2.4.1 Firewall e Proxy

Laureano (2005) aponta o *firewall* como uma das principais ferramentas de segurança para rede. Essa ferramenta garante que determinados computadores não sejam acessados por utilizadores não autorizados. Geralmente *firewalls* são instalados entre a rede local e a *internet* e tem como objetivo isolar essas duas redes. Regras de segurança são implementadas e, em sentido restrito, examina cada pacote e determina sua origem. Pode ser implementado uma lista branca, com as condições permitidas ou a lista negra, com as condições negadas, que serão bloqueadas pelo *firewall*.

Apesar de ser uma ferramenta de extrema importância para proteção de uma rede, sua utilização isoladamente não garante a segurança. Laureano (2005) ainda explica que os *firewalls* podem ser divididos em duas grandes classes: filtro de pacotes e servidor *proxy*. O primeiro é um dos principais mecanismos desta ferramenta, onde é definido as listas brancas ou negras, que filtrará o trafego de pacotes na rede.

Já os servidores *proxies* atuam como intermediários entre o computador do usuário e a *internet*, permitindo ou não a conexão de serviços e páginas *web* em uma rede de modo indireto. Com ele, o administador da rede pode controlar quem acessa a rede da empresa e quais os serviços essas pessoas poderão utilizar, provendo segurança de acesso.

2.4.2 Sistemas de Detecção de Intrusão

Oliveira (2003) define Sistemas de Detecção de Intrusão, IDS, como sistemas inteligentes capazes não só de detectar invasões em tempo real mas também podem aplicar ações necessárias contra o ataque. IDS podem ser baseados em regras, onde são criadas de acordo com os tipos de invasões e as ações a serem tomadas, ou baseado modo adaptável onde são empregados técnicas mais avançadas, como inteligência artificial.

2.5 O Arquivo de Configuração do PHP

O php.ini, ou arquivo de configuração do PHP é o arquivo responsável por armazenar as diretivas de configuração do servidor. Ao iniciar o PHP o arquivo é lido pelo sistema. Já para as versões de módulo de servidor, o php.ini é lido quando o servidor é inicializado. Durante a execução do sistema, é possível alterar os valores do php.ini através da função *ini_set()*, que tem como parâmetros duas strings e uma string como retorno (Mehdi et al., 2016).

```
1 String ini_set (String $varname, String $newvalue)
```

A primeira string passada como parâmetro é o nome da função que o programador deseja acessar, enquanto o segundo parâmetro é o valor que o programador deseja alterar, ou seja, o novo valor setado a ser passado para a função alterada. O valor retornado será FALSE em caso de falha na alteração ou o valor anterior da função alterada em caso de sucesso da operação.

A lista de diretivas do php.ini é extensa e possui funções que podem ser usadas em modo *default* ou configuradas de acordo com o uso desejado. O modo de segurança, *safe_mode*, é uma tentativa de resolver o problema de servidores compartilhados e quando ativado, faz uma checagem de UID, ou seja, se os arquivos que o usuário pretende acessar são de acesso permitido a ele ou não, porém, o *safe_mode* foi removido a partir versão 6.0.0 do PHP (Mehdi et al., 2016).

Mehdi et al. (2016) ainda mostra exemplos de funções importantes do php.ini como *disable_functions( )*, função que permite desativar outras funções por razões de segurança, e funções do modo de segurança, como *safe_mode_allowed_env_vars( )*, que permite ao usuário editar qualquer variável ambiente do sistema, e *safe_mode_gid( )*, que quando ativada faz uma checagem mais rígida em relação ao acesso de arquivos e pastas de usuários.

Capítulo 3

ReadFunSec

ReadFunSec é uma aplicação desenvolvida em Javascript que utiliza NodeJs no lado do servidor e HTML e Javascript no lado cliente, onde também é ultilizado Bootstrap como *framework* para estilização. O acesso ao servidor por parte do cliente é feito usando funções *middlewares* através do *framework* Express.

Inicialmente a proposta inicial previa a utilização da linguagem de programação C++ para constrir uma aplicação para instalação local no computador, porém, posteriormente foi escolhido construir uma aplicação com arquitetura cliente-servidor utilizando NodeJs devido ser uma tecnologia atual onde se permite facilmente construir aplicações rápidas e escaláveis utilizando a linguagem de programação Javascript.

3.1 Preparando o Ambiente

É preciso instalar o NodeJs para executar a aplicação. Basta abrir o terminal do Linux ou Power Shell no Windows e digitar os comandos descritos nas figuras abaixo.

Na **figura 3.1** é mostrado como fazer a instalação do gerenciador de pacotes do Windows.

Na **figura 3.2** é possível ver a instalação do NodeJs pelo Powershell e na **figura 3.3** podemos confirmar a instalação verificando a versão que foi instalada.

- Windows:
  - *Set-ExecutionPolicy Bypass -Scope Process -Force;iex ((New-ObjectSystem. Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))*
  - *cinst nodejs.install*

    No mesmo terminal digite *node -v* para validar a instalação e verificar a versão do NodeJs. O comando *npm -v* valida a instalação do módulo NPM e mostra a versão.

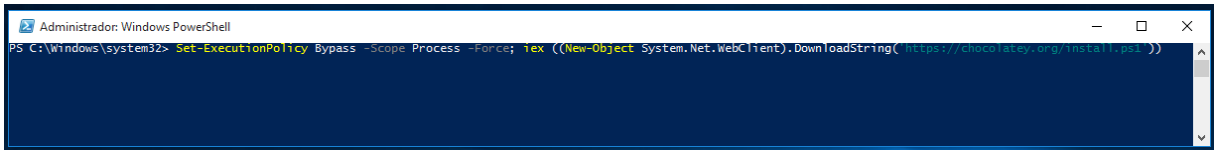


Figura 3.1: instalação do gerenciador de pacotes para Windows.

Figura 3.2: instalação do NodeJS para Windows.

Na **figura 3.4** é mostrado o download do NodeJs para Linux utilizando a distribuição *Ubuntu*.

A **figura 3.5** exibe a instalação do NodeJs enquanto a **figura 3.6** mostra as versões do NodeJs e NPM que foram instaladas no Linux.

- Linux:
  - *apt-get install curl*
  - *curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash*
  - *apt-get install -y nodejs*
  - *apt-get install nodejs-legacy*

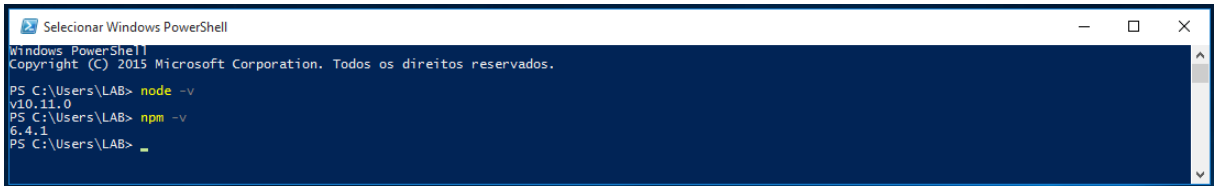


Figura 3.3: verificando versão do NodeJS e NPM no Windows.

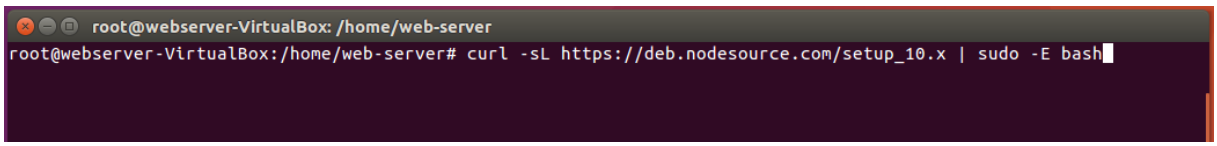


Figura 3.4: acessando NodeJS na versão 10.x para Linux (Ubuntu e Debian).

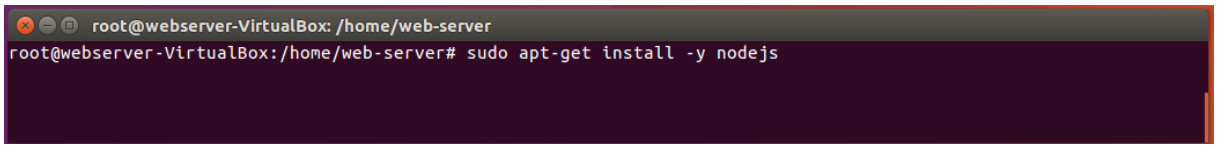


Figura 3.5: instalação do NodeJS para Linux (Ubuntu e Debian).

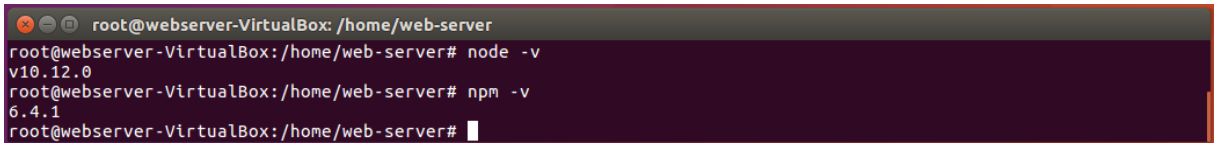


Figura 3.6: verificando versão do NodeJS e NPM no Linux.

Assim, será instalado também o Node Package Manager (NPM). Também é preciso instalar os seguintes módulos terceiros para o funcionamento do programa: *linebyline* para ler o arquivo, *fs* para gravar os dados do *array* final em arquivo JSON e *jsPDF* para gerar um arquivo PDF com o relatório das funções de segurança verificada. Os módulos *express*, *multer* e *body-parse* são usados nas funções *middlewares*, que permite ao lado cliente acessar o lado do servidor. Para instalar essas dependências, acesse a pasta *readfunsec/js* através de Windows Power Shell ou o terminal do Linux execulte o comando *npm install*. Este comando instalará todas estas dependências que estão descritos no arquivo *package.json*. Em seguida execute o comando *npm start* para iniciar o servidor Node para executar a aplicação.

Para iniciar o programa, abra o navegador (Google Chrome, Internet Explorer, Mozilla Firefox, etc.) e acesse o endereço *localhost:8081*.

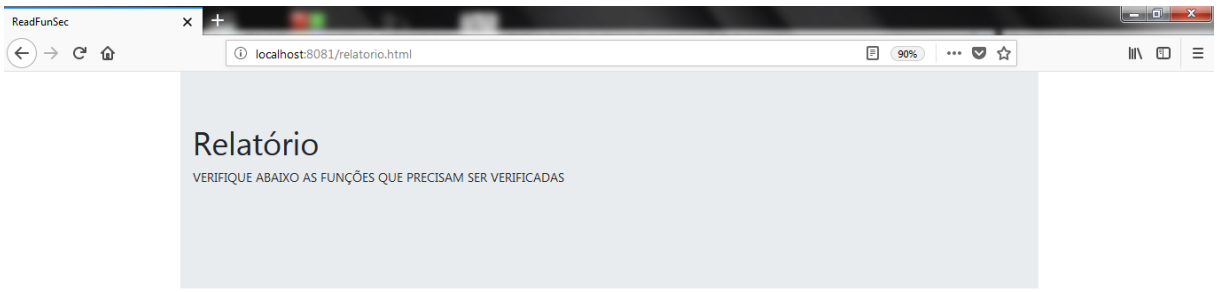
3.2 ReadFunSec e seu funcionamento

Com os conceitos de NodeJS, foi possível dividir o programa em 5 módulos principais (*verificar.js*, *regex.js*, *geradorPdf.js*, *manipulaAjax.js* e *filedrag.js*), um módulo *middleware* (*server.js*) e um módulo auxiliar (*funcoes.js*).

Na **figura 3.7** podemos ver o acesso a página inicial do ReadFunSec, onde é solicitado que carregue o arquivo que se deseja verificar. Este arquivo será enviado ao servidor através de uma requisição Ajax.

Figura 3.7: página principal do ReadFunSec.

O módulo *verificar.js* recebe o caminho do arquivo armazenado no servidor que foi passado pela página inicial, faz a leitura linha a linha do arquivo e verifica se a linha é uma função válida. Esta verificação é feita usando expressões regulares que são especificadas no módulo *regex.js*.Caso a linha seja um comentário, linha branco ou funções que não é relacionadas a segurança, a linha é descartada. As linhas que contém funções válidas relacionadas a segurança e possuem valores que geram vulnerabilidades são armazenadas em um *array* que posteriormente será convertido para o formato JSON e salvo no servidor. As funções que foram encontradas e armazenadas no JSON serão exibidas em um relatório no navegador como podemos ver na **figura 3.8**. A exibição também é feito usando requisição Ajax.



MENSAGEM	LINHA	FUNÇÃO/VALOR
ALERTA: FUNÇÃO PODE GERAR VULNERABILIDADE	Linha 188	open_basedir = /var/etc/pwd
ALERTA: FUNÇÃO PODE GERAR VULNERABILIDADE	Linha 318	disable_functions = /safe_mode/safe_mode_gir/safe_mode_protected/
ALERTA: FUNÇÃO PODE GERAR VULNERABILIDADE	Linha 323	disable_classes = class_securyt.class_path_bloq.
ALERTA: FUNÇÃO PODE GERAR VULNERABILIDADE	Linha 379	safe_mode = 0
ALERTA: FUNÇÃO PODE GERAR VULNERABILIDADE	Linha 410	safe_mode_gid = 0
ALERTA: FUNÇÃO PODE GERAR VULNERABILIDADE	Linha 484	safe_mode_exec = /bin/bash

Figura 3.8: relatório gerado com as funções encontradas e sua respectiva linha.

O módulo *regex.js* contém apenas um *array* de objetos com dois elementos: o primeiro é a função de segurança do php.ini, o segundo elemento é o valor esperado. As funções que estão neste módulo são inseridas manualmente no código fonte. Foram selecionadas todas as funções que fazem parte do modo seguro do PHP e os valores recomendados pela linguagem. Segue as funções inseridas no módulo *regex.js*:

- *safe_mode*;
- *safe_mode_gid*;
- *safe_mode_include*;
- *safe_mode_exec*;
- *safe_mode_allowed*;
- *safe_mode_protected*;
- *open_basedir*;
- *disable_functions*;
- *disable_class*.

O módulo *fileDrag.js* é responsável por pegar o nome e caminho do arquivo passado na página inicial e enviar estas informações para o módulo *funcoes.js*.

O módulo *geradorPdf.js* salva um arquivo PDF, informando as funções que apresentam vulnerabilidade.

O módulo *funcoes.js* contém as funções que são chamadas através das páginas HTMLs e acioanam as funções middlewares, que por sua vez executam as funções dos módulos Node e redirecionam para as próximas páginas do programa.

A **figura 3.9** mostra como é feito o acesso ao servidor por funções *middlewares*, que utilizam o *framework* Express para gerar essas funções. O lado do cliente faz as requisições que apenas chamam funções auxiliares, que por sua vez, chamam as funções *middlewares* responsáveis por executar as funções do lado do servidor e redirecionar para as novas páginas do lado cliente.

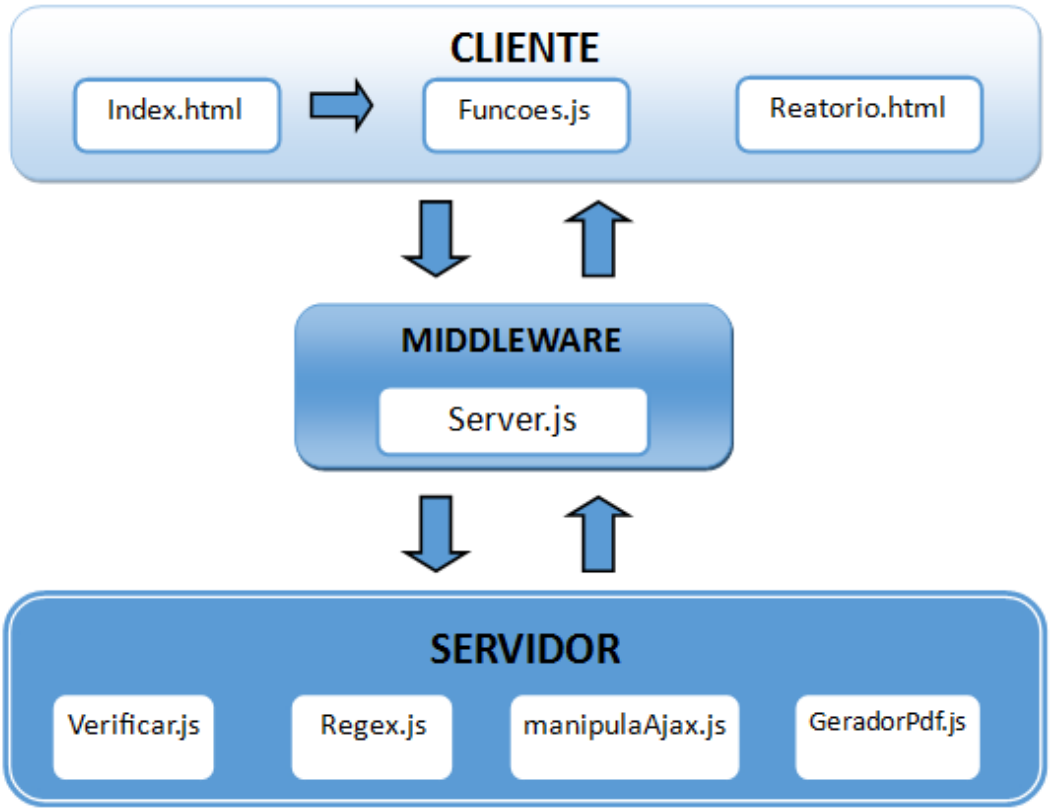


Figura 3.9: acesso do cliente ao servidor através de funções middleware.

Capítulo 4

Conclusão

Com os estudos apresentos, foi possível verificar que há um grande número de ciberataques que são feitos através de vulnerabilidades e sistemas *web*. Estas vulnerabilidades podem ser geradas tanto por falha de infraestrutura quanto falha de programação em configuração de sistemas.

Como o PHP é uma linguagem com grande utilização em desenvolvimento *web*, essa linguagem está sujeita a muitas falhas de configuração devido a falta de experiência de programadores ou dos profissionais responsáveis por implantar sistemas. A configuração de um servidor *web* sem as devidas tratativas de segurança pode acarretar em sérios problemas depois que o sistema for colocado em produção, pois falhas como LFI ou injeção de código podem afetar o funcionamento do servidor e alterar, inserir e excluir arquivos de banco de dados.

O arquivo de configuração do PHP, o php.ini, contém várias funções que se referem a segurança da aplicação, como permissão de acesso a pastas, validação de acesso a arquivos do usuário ou funções e classes desabilitadas. Devido ao grande número de linhas, como comentários ou linhas em branco, é possível que funções válidas passem despercebidas ao analisar este arquivo.

Para que a segurança da informação seja garantida em ambientes computacionais, é preciso adotar medidas como boas práticas de programação, política de segurança da informação e implantar sistemas como *firewalls*, IDS entre outras várias soluções disponíveis para proteger o servidor de aplicação e a rede de invasões.

O ReadFuncSec proporcionou uma busca detalhada neste arquivo de configuração, exibindo ao usuário apenas as funções de segurança para que sejam analisadas e corrigidas caso necessário, encontrando com facilidade estas funções dentro do arquivo. Assim, é possível analisar todos os arquivos de configuração de uma aplicação em pouco tempo.

Referências Bibliográficas

Afonso, V. M. (2011). **Um sistema para análise de detecção de ataques ao navegador web**. Master's thesis, Universidade Estadual de Campinas, Instituto de Computação, Campinas.

Burk, B. and Labourey, S. (2002). **Clustering with JBoss 3.0**. `http://www.onjava.com/pub/a/onjava/2002/07/10/jboss.html`, acessado em 16 Julho de 2016.

Cert.Br, C. d. E. R. e. T. d. I. d. S. n. B. (2012). **Cartilha de Segurança na Internet**. `http://cartilha.cert.br/creditos/`, acessado em 12 de Novembro de 2018.

Cortez, I. and Kubota, L. (2012). **Contramedidas em segurança da informação e vunerabilidade cibernética: evidência empírica de empresas brasileiras**. `http://www.scielo.br/pdf/rausp/v48n4/10.pdf`, acessado em 11 de Novembro de 2018.

Erickson, J. (1977). ***Hacking: The Art Of Exploitation***. San Francisco, EUA, 2 edition.

Givaroto, S. C. R. and Santos, G. R. (2013). ***Backtrack Linux - Auditoria e Testes de invasão em Redes de Computadores***. Rio De Janeiro: Campus, 1 edition.

Laureano, M. A. P. (2005). **Gestão de Segurança da Informação**. `http://www.mlaureano.org/aulas_material/gst/apostila_versao_20.pdf`, acessado em 11 Novembro de 2018.

Lyon, G. (1997). **NMAP.ORG**. `https://nmap.org/`, Acessado em 17 de Novembro de 2018.

Malerba, C. (2010). **Vunerabilidades e Exploits: Técnicas, Detecção e Prevenção**. Trabalho de conclusão de curso (bacharel em Ciência da Computação), Universidade Federal do Rio Grande do Sul, Porto Alegre, Brasil.

Medeiros, E. i. (2012). **PHP: detecção de falhas que possibilitem a injeção de código**. Trabalho de conclusão de curso (bacharel em Sistemas de Informação), Universidade Estadual de Mato Grosso do Sul, Dourados, Brasil.

Mehdi, A., F., B., A., D., N., L., H., M., and G., R. (2016). **Manual Do PHP**. `https://secure.php.net/manual/pt_BR/about.php`, acessado em 11 de Novembro de 2018.

Oliveira, W. (2003). ***Tecnicas para Hackers - Soluções Para Segurança***. Porto - Lisboa, 2 edition.

Pereira, M. (2013). **Segurança de Servidores Web**. Faculdade de Tecnologia Senac (curso superior em tecnologia em Redes de Computadores), Pelotas, Brasil.

Sica, C. (2006). ***PHP Orientado a Objetos - Fale a linguagem da internet***. Rio De Janeiro, 1 edition.

Tâmega, F. (2003). ***Hacker Inside - Top Secret***. Goiânia, 1 edition.

Teixeira, E. P. L. (2007). **Ferramenta de análise de código para detecção de vulnerabilidades**. Master's thesis, Universidade de Lisboa, Faculdade de Ciência, Departamento de Informática, Lisboa.

Terry, B., Anish, B. B., Eugene, S., and Carol, A. (1997). ***Segurança na Internet***. Rio De Janeiro: Campus, 1 edition.