

**FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RN -
FARN**

DIOGO MAED DANTAS MAIA

SEGURANÇA DA INFORMAÇÃO APLICADA A APLICAÇÕES WEB

NATAL, AGOSTO 2011

DIOGO MAED DANTAS MAIA

SEGURANÇA DA INFORMAÇÃO APLICADA A APLICAÇÕES WEB

Faculdade Natalense para o Desenvolvimento
do RN – FARN Curso de Pós-graduação em
Desenvolvimento de Sistemas Corporativos
Trabalho de conclusão de curso.

Orientador: David Ricardo Pereira

NATAL, AGOSTO 2011

SEGURANÇA DA INFORMAÇÃO APLICADA A APLICAÇÕES WEB

DIOGO MAED DANTAS MAIA

‘Esta Dissertação foi julgada adequada para obtenção do Título de Especialista em Desenvolvimento de Sistemas Corporativos, e aprovada em sua forma final pelo Curso de Pós-Graduação em Especialização em Desenvolvimento de Sistemas Corporativos’

Prof. David Ricardo do Vale Pereira, Msc.
Orientador

Prof. Gleydson de Azevedo Ferreira Lima, Msc.
Coordenador do Programa de Pós-Graduação em Desenvolvimento de Sistemas Corporativos

BANCA EXAMINADORA:

Prof. David Ricardo do Vale Pereira, Msc.
Orientador

Prof. Gleydson de Azevedo Ferreira Lima, Msc.

Prof. Itamir de Moraes Barroca Filho

RESUMO

Com o passar dos anos, garantir a qualidade da segurança dos sistemas de informação que são ameaçados pelas mais variadas formas de ameaças, o que é um risco às empresas atingidas, tornou-se um desafio especialmente em tempos em que a informação é o bem mais precioso. A utilização de um *firewall* e de serviços bem implementados para se manter a rede segura chega a não ser o suficiente, é preciso a conscientização e participação dos demais funcionários e direção. É necessário o emprego de um modelo de gestão que estabeleça e divida as responsabilidades de cada um nessa rede corporativa. Dai a importância de uma gestão em política de segurança. Sendo assim, este trabalho tem por objetivo estudar os conceitos de políticas de segurança baseadas em TI (tecnologia da informação) e gestão da segurança em aplicativos web.

Palavras chave: Segurança de sistemas, vulnerabilidades, Políticas de segurança, gestão da segurança em aplicativos web.

ABSTRACT

Over the years, ensuring the quality of security of information systems that are threatened by a variety of threats, which is a risk to the affected companies, has become a challenge especially in times when the information is the most precious. The use of a firewall and services, and implemented to keep the network secure enough not to be enough, we need awareness and participation of other employees and management. It is necessary to use a management model to establish and divide the responsibilities of each such corporate network. Hence the importance of a security policy management. Than, this work aims to study the concepts of security policies based on IT (information technology) and management of web application security.

Keywords: Security systems, vulnerabilities, security policies, security management for web applications.

LISTA DE ABREVIATURAS

AMA	Access Mobile Agent
ARPENET	Advanced Research and Projects Agency
BGP	Border Gateway Protocol
CMA	Computational Agent Mobile
CN	Correspondent Node
CPqD	Centro de Pesquisa e Desenvolvimento em Telecomunicações
CDPD	Cellular Digital Packet Data
CLNP	Connectionless Network Protocol
CoA	Care-of-Address
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DMA	Data Mobile Agent
DES	Data Encryption Standard
DNS	Domain Name System
DHCP	Dynamic Host Configuration Protocol
DHCPv6	Dynamic Host Configuration Protocol version 6
HA	Home Agent
HoA	Home Address
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
IHL	Internet Header Length
IETF	Internet Engineering Task Force
IPng	IP- The Next Generation
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IPX	Internetwork Packet Exchange
ISO	Open Systems Interconnection
IKMP	Internet Key Management Protocol
LAN	Local Area Network.
LLC	Logical Link Control
MAC	Media Access Control

MAN	Metropolitan Area Network.
MD5	Message Digest 5
MH	Mobile Home
MN	Mobile Node
MIP	Mobile Internet Protocol
MIPv6	Mobile Internet Protocol Version 6
MCSP	Mobile Computing Systems
NSAP	Network Service Access Point
NTP	Network Time Protocol
OGSA	Open Grid Services Architecture
OSPF	Open Shortest Path first
OSI	Open Systems Interconnection
QoS	Quality of Service
PDAs	Personal Digital Assistants
P2P	Pier To Pier
RA	Router Advertisement
RTT	Round Trip Time
RSVP	Resource Reservation Protocol
SIPP	Simple Internet Protocol Plus
TCP	Transmission Control Protocol
TANS	Tiny Area Networks
UDP	User Datagram Protocol
WAN	Wide Area Network.

LISTA DE FIGURAS

FIGURA 1: FILME “A SENHA” (SWORD FISH).....	23
FIGURA 2: WORM “ <i>BLASTER</i> ”.....	23
FIGURA 3: ILUSTRAÇÃO DA PARTE VISÍVEL DAS VULNERABILIDADES....	57
FIGURA 4: ESTRUTURA DE UM WEB SERVICE.....	58
FIGURA 5: ROBERT TAPPAN MORRIS.....	66
FIGURA 6: KEVIN DAVID MITNICK.....	67
FIGURA 7: KEVIN LEE POULSEN.....	68
FIGURA 8: GARY MCKINNON.....	69
FIGURA 9: ESTRUTURA BÁSICA DA MENSAGEM SOAP.....	74
FIGURA 10: MENSAGEM SOAP – ENVELOPE, CABEÇALHO E CORPO.....	75
FIGURA 11: INTERAÇÃO ENTRE ENTIDADES DA AOS.....	77
FIGURA 12: COLABORAÇÃO TÍPICA NA ARQUIT. DOS SERVIÇOS WEB....	81
FIGURA 13: ARQUITETURA E INTEGRAÇÃO DE WEB SERVICES.....	83
FIGURA 14: PILHA DE PROTOCOLOS QUE COMPÕE WEB SERVICES.....	84
FIGURA 15: EXEMPLO DE MAPEAMENTO DE APLICAÇÃO WEB.....	84
FIGURA 16: CONTEXTOS DE SEGURANÇA [IBM E MICROSOFT 2002].....	85
FIGURA 17: SEGURANÇA NAS DIFERENTES CAMADAS.....	86
FIGURA 18: DETECÇÃO DE INTRUSÃO EM UMA REDE LOCAL DE DIFUSÃO.....	91
FIGURA 19: DETECÇÃO DE INTRUSÃO BASEADO EM ESPELHAMENTO DE PORTAS NO COMPUTADOR.....	92
FIGURA 20: ARQUITETURA DE UM SISTEMA DE REGISTROS.....	93
FIGURA 21: DETECÇÃO DE INTRUSÃO HÍBRIDA.....	94

SUMÁRIO

1	INTRODUÇÃO	11
1.1	OBJETIVO GERAL	12
1.2	OBJETIVOS ESPECÍFICOS	12
1.3	METODOLOGIA	12
2	REFERENCIAL TEÓRICO	14
2.1	INFORMAÇÃO E ATIVOS	14
2.2	IMPORTÂNCIAS DA SEGURANÇA DE INFORMAÇÃO PARA O NEGÓCIO	14
2.3	SEGURANÇA DE COMPUTADORES	15
2.3.1	Segurança física e de entorno	16
2.3.2	Segurança da informação	17
2.4	POLÍTICA DE SEGURANÇA DA INFORMAÇÃO	18
2.5	AVALIAÇÃO DOS RISCOS DE SEGURANÇA	22
2.6	TRATAMENTOS DOS RISCOS DE SEGURANÇA	26
3	VULNERABILIDADES E AMEAÇAS	28
3.1	NEGAÇÕES DE SERVIÇO (DENIAL OF SERVICE)	29
3.2	SPAM	29
3.3	MALWARE	30
3.3.1	Vírus	31
3.3.2	Worms	32
3.3.3	Bots e botnets	33
3.3.4	Trojan horse ou cavalos de tróia	34
3.3.5	Adware e spyware	35
3.3.6	Backdoors	36
3.3.7	Keyloggers	37
3.3.8	Rootkits	38
3.4	SNIFFERS	39
3.5	PORT SCAN	40
3.6	ENGENHARIA SOCIAL	41
4	RECURSOS DE DEFESA DA REDE	43
4.1	FIREWALLS	43

4.2	SISTEMAS DETECTORES DE INTRUSÃO.....	43
4.2.1	Honeypot	44
4.3	CRIPTOGRAFIA.....	44
4.3.1	Criptografia de chaves pública	45
4.3.1.1	Chaves simétricas.....	46
4.3.1.2	Chave assimétrica.....	47
4.4	ASSINATURA DIGITAL.....	48
4.5	CERTIFICADOS DIGITAIS.....	48
4.5.1	SSL	49
4.6	ANTIVÍRUS.....	50
4.7	REALIZAÇÃO DE CÓPIAS DE SEGURANÇA.....	51
4.7.1	Cuidados com as cópias de segurança	52
4.8	SERVIDORES DE ALTA DISPONIBILIDADE.....	53
4.9	EDUCAÇÃO DOS USUÁRIOS.....	53
5	SEGURANÇA APLICADA A APLICAÇÕES WEB	55
5.1	INTRODUÇÃO, CONCEITOS E DEFINIÇÕES.....	55
5.2	IMPORTÂNCIA DA SEGURANÇA EM AMBIENTES WEB.....	56
5.2.1	Ataques	59
5.2.2	Hackers ou crackers	65
5.3	RECURSOS E TECNOLOGIAS.....	69
5.3.1	Introdução	69
5.3.2	A linguagem XML	71
5.3.2.1	A sintaxe XML.....	71
5.3.2.2	XML e validação de documentos.....	72
5.3.3	SOAP	73
5.3.4	WSDL	75
5.3.5	UDDI	76
5.3.6	SOA	76
5.3.7	OOS	78
5.3.8	RPC	80
5.4	WEB SERVICES.....	80
5.5	MECANISMOS DE SEGURANÇA.....	85

5.5.1	Introdução	85
5.5.2	Firewalls	86
5.5.2.1	Filtragem de pacotes ou <i>packet filtering</i>	87
5.5.2.2	<i>Firewalls</i> de aplicação ou <i>proxies</i>	88
5.5.2.3	<i>Firewalls</i> baseados no estado ou <i>stateful inspection firewalls</i>	88
5.5.3	Sistemas detecores de intrusão	89
5.5.3.1	<i>Network-based intrusion detection system</i> ou sistema de detecção baseado na rede.....	90
5.5.3.2	<i>Host-based intrusion detection system</i> ou sistema de detecção baseada na estação ou servidor.....	92
5.5.3.3	<i>Hybrid intrusion detection system</i> ou sistema híbrido de detecção de intrusão.....	94
5.5.4	Honeypots ou potes de mel	94
5.6	PRÁTICAS DE SEGURANÇA.....	96
6	CONSIDERAÇÕES FINAIS	101
	REFERÊNCIAS	104

1 INTRODUÇÃO

A segurança dos sistemas, principalmente com relação à garantia do sigilo dos dados, nunca foi tão ameaçada quanto nas últimas décadas. Com o advento da *Internet*, a necessidade de segurança é um fato que vem transcendendo os limites da produtividade e da funcionalidade. A absorção de informações via on-line pelos negócios das empresas e a crescente automatização dos processos produtivos através do uso de computadores, faz com que medidas para preservar esse sigilo e integridade sejam cada vez mais necessárias. Enquanto a velocidade e a eficiência em todos os processos de negócios significam uma vantagem competitiva, a falta de segurança nos meios que habilitam a velocidade e a eficiência pode resultar em grandes prejuízos e falta de novas oportunidades de negócios.

Segundo a Academia Latino Americana de Segurança da Informação (ALASI, 2005), em estudo realizado pela Universidade do Texas, apenas 6% das empresas que sofrem um desastre informático sobrevivem. Os demais 94% desaparecem, mais cedo ou mais tarde. Pesquisas do Gartner Group (apud ALASI 2005) confirmam essa tendência ao indicar que duas, de cada cinco empresas que enfrentam ataques ou danos em seus sistemas, deixam de existir.

Um dos principais problemas na segurança da informação está na cultura de segurança que é desenvolvida na empresa. Dentre os desastres informáticos mencionados anteriormente, muitos são compostos de vírus e vazamento de informações, seja por engenharia social, que será tratada nos capítulos posteriores, ou através de *softwares* como *trojan-horses* e *spywares*, que são “contraídos” através da navegação na *internet* ou através de *e-mails*, ou até mesmo por comutação de arquivos por *hard disk* externos ou *pen-drives*.

O presente trabalho tem por objetivo realizar um estudo sobre as práticas de gestão de segurança e procedimentos utilizando recursos de TI (Tecnologia da Informação) para segurança de aplicações *web*. Servirá de base a esse estudo a avaliação de políticas atualmente usadas nos mais diversos ramos de atividade, bem como, o aprofundamento teórico sobre essa área de conhecimento.

Esse trabalho está dividido em três partes, sendo que a primeira corresponde ao estado da arte, composto pela fundamentação e conceitos vindos da pesquisa bibliográfica. A segunda é a política de segurança proposta, que engloba os problemas mais comuns de segurança de uma empresa, e pode ser implementada

por um custo relativamente baixo. A terceira parte é um estudo sobre essas políticas de segurança aplicadas a conceitos e aplicações que são operacionalizadas via *web*.

1.1 OBJETIVO GERAL

Desenvolver uma pesquisa sobre a segurança de informação aplicada à aplicativos web demonstrando as suas características.

1.2 OBJETIVOS ESPECÍFICOS

- Fazer uma introdução aos conceitos de ativos, segurança, política de segurança da informação e avaliação de riscos de segurança, essenciais para uma melhor compreensão do trabalho.
- Tratar a engenharia social e vírus apresentando os tipos de vulnerabilidades mais comuns em quaisquer sistemas.
- Dissertar sobre a aplicação de uma política de gestão de segurança para a informação em qualquer sistema;
- Tratar das ferramentas mais usadas para a proteção de sistemas como *firewalls*, criptografia, antivírus, entre outros.
- Aborda o estudo proposto sobre a atualidade da gestão de política de segurança aplicada a aplicações web.

1.3 METODOLOGIA

Este estudo não tem por objetivo impor regras, mas demonstrar padrões que devem ser seguidos para se evitar riscos na comunicação de informações e na sua segurança em si pelo mundo da *internet* explorando uma fundamentação teórica que

se construí a partir de um aprofundamento teórico em bases científicas, como, artigos, livros e periódicos.

2 REFERENCIAL TEÓRICO

2.1 INFORMAÇÃO E ATIVOS

De acordo com NTC ISO/IEC 27002 (2007), “informação é um ativo que, como outros ativos, é essencial para as atividades de uma empresa e, por consequência, necessita de uma proteção. Isto é especialmente importante no interior de negócios cada vez mais interconectados. Como resultado desta interconexão, a informação se expõe as mais diversas ameaças e vulnerabilidades”.

A informação pode existir de diversas formas: escrita num papel, armazenada eletronicamente, transmitida por correio ou por meio eletrônicos, presente em películas, ou expressamente conservada. Qualquer que seja a forma ou o meio pelo qual esteja alocado, sempre deveria ter uma proteção adequada. A segurança dessa informação tem por finalidade assegurar a continuidade de um negócio, minimizar riscos e maximizar o retorno das repostas dessa informação.

Por sua vez, ativo é todo elemento que compõe os processos, incluindo o próprio processo, que manipulam e processam a informação, a contar a própria informação, o meio em que ela é armazenada, os equipamentos em que ela é manuseada, transportada e descartada.

O termo “ativo” vem da área financeira por ser um “bem” (patrimônio) de valor para a organização e, por esse motivo, necessita de proteção adequada.

Existem muitas formas de dividir e agrupar os ativos para facilitar seu tratamento. Entre elas pode-se citar: equipamentos, aplicações, usuários, ambientes, informações e processos. Dessa forma, torna-se possível identificar melhor os grupos, tratando-os com especificidade e aumentando qualitativamente as atividades de segurança.

2.2 IMPORTÂNCIAS DA SEGURANÇA DE INFORMAÇÃO PARA O NEGÓCIO

De acordo com NTC ISO/IEC 27002 (2007), por definição, a manutenção e o

armazenamento podem ser essenciais para manter a continuidade de um negócio, por consequência, sua rentabilidade e sua conclusão.

É importante tanto para o setor público como para o setor privado e para proteção de suas infra-estruturas críticas. Em ambos os setores, a Segurança de Informação atua como facilitador para alocar e evitar ou reduzir os riscos pertinentes.

As organizações e sistemas de redes de informações enfrentam ameaças a segurança, procedentes de uma variedade de fontes, incluindo fraudes originadas por computadores, espionagem, vandalismo, incêndios ou inundações. Esses danos ainda podem ser causados por códigos maliciosos ou por *hackers* (piratas do mundo virtual) que com o passar do tempo suas ações se tornam mais ambiciosas e munidas de equipamentos cada vez mais sofisticados.

Entretanto, muitos sistemas de informação não são desenhados para ser seguros. A segurança que pode ser alocada através de meios técnicos é limitada e deveria estar suportada por uma boa gestão e por procedimentos apropriados.

Segundo Thomas A. Wadlow apud ALASI (2005), “*A segurança deverá ser proporcional ao valor do que se está protegendo*”. Por consequência, um sistema de segurança da informação deve apresentar uma relação custo-benefício que na tentativa um ataque seja desestimulante ao atacante, ao mesmo tempo em que não gaste mais do que o valor da informação que está sendo protegida.

2.3 SEGURANÇA DE COMPUTADORES

Segundo os princípios da segurança da informação, um sistema computacional é considerado seguro se este atende a quatro requisitos básicos relacionados aos recursos que o compõem: confidencialidade, integridade, não-repúdio e disponibilidade.

Confidencialidade: a informação só está disponível para aqueles usuários ou processos devidamente autorizados a acessá-las. Exemplo: alguém obtém acesso não autorizado ao seu computador e lê todas as informações contidas na sua declaração de Imposto de Renda ou numa transação bancária.

Integridade: a informação não é destruída ou corrompida e o sistema tem um desempenho correto sem alterações não autorizadas. Um exemplo seria alguém obter acesso não autorizado ao seu computador e alterar informações da sua

declaração de Imposto de Renda, momentos antes de você enviá-la à Receita Federal, ou ao banco de dados de uma empresa e alterar informações referentes a contas bancárias.

Não-Repúdio: Podemos definir “não repúdio” como a garantia de que o emissor de algum dado ou informação ou o autor de alguma ação sobre a informação não possa posteriormente negar que tenha enviado o dado/informação ou que tenha alterado alguma informação. Em poucas palavras trata-se de garantir que quando for necessário provar que “alguém fez algo” tenhamos provas.

Disponibilidade: os serviços/recursos do sistema estão disponíveis sempre que forem necessários para a aplicação utilizada por um usuário que a precisa e que este esteja devidamente autorizado. Exemplo, o seu provedor sofre uma grande sobrecarga de dados ou um ataque de negação de serviço e por este motivo você fica impossibilitado de enviar sua declaração de Imposto de Renda à Receita Federal. Outro exemplo, sendo este de um fato recente, que podemos citar o mês de junho de 2011 em que *hackers* invadiram *sites* da Presidência Social e do Senado que apresentaram instabilidades na quinta-feira, 22.

Ainda devido a esse ataque, os portais mencionados não carregavam e apresentavam mensagens de erro. A página do Ministério dos Esportes também foi retirada do ar, um dia após o grupo LulzSecBrazil deixar vários outros *sites* indisponíveis. Além disso, o grupo divulgou dados pessoais da presidente Dilma Rousseff e do prefeito de São Paulo, Gilberto Kassab, como número de CPF, telefones e e-mails pessoais. Fatos como esses anteriormente citados evidenciam cada vez mais a importância de se empregar uma segurança que integre cada vez mais os requisitos básicos descritos.

A segurança de sistemas de informação ainda pode ser dividida em segurança física e a da informação.

2.3.1 Segurança física e de entorno

Tem por objetivo evitar o acesso não autorizado, o dano ou a interferência no tráfego ou na própria informação. Os serviços de processamento de informação, sensíveis ou críticas, devem estar integrados em áreas seguras, protegidas por perímetros de segurança definidos, com barreiras e controles de entrada adequados.

O Controle de acesso físico se utiliza de perímetros para proteção às áreas onde a informação está contida e serviços de processamento de informação (NTC ISO/IEC 27002).

“A segurança física tem como principais objetivos: garantir a continuidade das rotinas; assegurar a integridade dos ativos; manter a integridade e confidencialidade das informações” (Fontes, 1991, pg 21).

Um ponto normalmente negligenciado por quem pensa em segurança é o acesso físico a terminais e outros dispositivos da rede. Manter cópias de *backup* seguras contra roubo, além dos cuidados normais para manter a integridade do *backup*, assim como, *notebooks* e outros terminais portáteis também representa riscos em caso de roubo ou acesso não autorizado. Deve-se ter cuidado em relação aos dados armazenados nos discos rígidos e memórias desses terminais, outros lugares a serem protegidos são os *switches* de distribuição em rede, servidores, roteadores e servidores. “*Um dos pontos fundamentais da segurança física é o controle de acesso. Ele está baseado no fato de que nem todas as pessoas devem ou precisam ter acesso a todas as áreas da organização*” (Loss Control apud Lemos, 2001).

O acesso remoto, seja por um terminal estabelecido pelo usuário, ou por um ponto de rede conectado a equipamentos portáteis, permite o uso de falsos protocolos de *internet*, rastreamento de pacotes, entre outras atividades. Quando oferecido esse tipo de serviço ao público, torna-se interessante executá-lo separadamente, assim como se devem vigiar locais de acesso vazios e desconectá-los fisicamente do quadro de distribuição, monitorando toda tentativa de conexão não autorizada.

2.3.2 Segurança da informação

NTC ISO/IEC 27002 (2007), a segurança da informação é a proteção da informação contra uma grande variedade de ameaças com a finalidade de assegurar a continuidade do processo, minimizar os riscos e maximizar o retorno e a ocorrência dos processos.

Apóia-se implementando um conjunto adequado de controles, incluindo políticas, processos, procedimentos, estruturas organizacionais e funções de

softwares e *hardwares*. Os controles necessitam ser estabelecidos, implementados, monitorados, revisados e armazenados, onde seja necessário, para assegurar os cumprimentos dos objetivos específicos da segurança e dos negócios de uma empresa.

Apesar das tecnologias de armazenamento, manipulação e transporte de dados terem proporcionado uma maior agilidade às continuidades dos processos, surgiram, apesar disso, novos riscos. As ações de determinados profissionais, *hackers*¹, da engenharia social, vírus, *worms*, negação de serviço ou DoS (*Deny of Service*) e espionagem eletrônica são exemplos dos riscos a que as empresas estão sendo expostas diariamente. Diante deste cenário, a segurança da informação torna-se imprescindível para as organizações, sejam elas do setor público ou privado.

Uma das primeiras medidas de segurança que as empresas devem adotar é o armazenamento de suas informações em meios externos aos de uso normal. O armazenamento das informações pode ser feito das seguintes formas: impresso em papel, em meios digitais (discos, fitas, CDs, DVDs, disquetes), em imagens fotografias e filmes. Porém esses meios também exigem proteção e manutenção para garantia da segurança.

Dessa forma, segurança da informação pode ser entendida como a proteção da informação em si, dos sistemas, da infra-estrutura e dos serviços a que ela suporta, contra acidentes, roubos e incidentes para assim minimizar os possíveis impactos à segurança. Um dos recursos mais estudados e utilizados atualmente é o das políticas de segurança que será abordado a seguir.

2.4 POLÍTICA DE SEGURANÇA DA INFORMAÇÃO

“Uma política de segurança é um conjunto de diretivas, normas, procedimentos e instruções que comanda as atuações de trabalho e define os critérios de segurança para que sejam adotados em nível local ou institucional com o objetivo de estabelecer, padronizar e normalizar a segurança tanto no escopo humano como no tecnológico. A partir desses princípios, é possível fazer da segurança das informações um esforço comum, desde que todos possam contar com um arsenal informativo documentado e normalizado dedicado à padronização do método de

1 Neste trabalho, consideraremos *hacker* aquela pessoa que invade ou tenta invadir o sistema de informação de uma empresa. Pois para a segurança das informações, não importam os motivos da invasão e sim a vulnerabilidade explorada.

operação de cada um dos indivíduos envolvidos na gestão da segurança da informação”. (Academia Latino Americana de Segurança da Informação, 2006, modulo3, p.3)

Segundo a ALASI (2006), uma política de segurança deve ser elaborada considerando-se o ambiente em que se está trabalhando como a tecnologia da segurança das informações, para que os critérios estabelecidos estejam de acordo com as práticas internas mais recomendadas da organização e com as práticas de segurança atualmente adotadas a fim de buscar uma conformidade maior com critérios atualizados e reconhecidos globalmente.

“Convém que a direção estabeleça uma política clara e demonstre apoio e comprometimento com a segurança da informação através da emissão e manutenção de uma política de segurança da informação para toda a organização”. (NBR ISO/IEC 17799, 2001, p. 4)

Para a implantação da política é necessário que o alto comando da empresa a tenha aprovado, comunicado e publicado, adequadamente aos funcionários. Além disso, se faz necessário que o alto comando esteja envolvido com o todo o processo em todos os momentos e estabeleça, aprovando as diretrizes para a gestão da segurança da informação, onde devem estar contidas, no mínimo, as seguintes orientações (ALASI, 2006):

- Definição de segurança da informação, resumo das metas e *escopo*² e a importância da segurança como um mecanismo que habilita o compartilhamento da informação;
- Declaração do comprometimento da alta direção, apoiando as metas e princípios da segurança da informação;
- Breve explanação das políticas, princípios, padrões e requisitos de conformidade de importância específica para a organização.
- Conformidade com a legislação e cláusulas contratuais;
- Requisitos na educação de segurança;
- Prevenção e detecção de vírus e *softwares* maliciosos;
- Gestão da continuidade no negócio;
- Conseqüências das violações na política de segurança da informação.

2 Escopo: escopo é todo e qualquer conjunto de ativos, independente do critério utilizado.

- Definição das responsabilidades gerais e específicas na gestão da segurança da informação, incluindo o registro dos incidentes de segurança;
- Referências à documentação que possam apoiar a política, por exemplo, políticas e procedimentos de segurança mais detalhados de sistemas de informação específicos ou regras de segurança que convém que os usuários sigam. (ISO/IEC 17799, 2001, p. 4).

O método de planejamento de gestão e monitoramento de segurança de Tecnologia da Informação, pode sofrer variações a depender da necessidade da própria empresa. Devido aos diferentes estilos, tamanhos e estruturas das organizações, o método deve ser adequado ao ambiente no qual ele será posto em prática. Para uma melhor eficiência dessa adequação, faz-se necessário seguir alguns direcionamentos:

- Identificar os requisitos de segurança da informação. Basicamente, existem três fontes principais de requisitos de segurança da informação:
 - obtida através da análise/avaliação de riscos para a organização;
 - obtida a partir da legislação vigente a que a organização, seus parceiros comerciais e provedores de serviço devem atender;
 - obtida a partir dos princípios, objetivos e requisitos do negócio.
- Analisar o ambiente o qual a segurança está relacionada. Devem-se realizar levantamentos periodicamente dos riscos da segurança da informação, identificando possíveis riscos e ameaças. Os resultados obtidos deverão direcionar as ações gerenciais que nortearão todo o processo de segurança da informação.
- Seleção de controles. A partir da identificação dos riscos e com as medidas de prevenção previamente providenciadas, se faz necessário implementar controles que assegurarão a redução dos riscos a níveis aceitáveis. A seleção desses controles pode ser realizada a partir dessas medidas ou de outras que atendam às necessidades da empresa. Esses controles incluem:
 - Proteção de dados e privacidade de informações pessoais;
 - Proteção dos registros organizacionais;

- Direitos de propriedade intelectual;
 - Documento de política de segurança da informação;
 - Atribuição de responsabilidades;
 - Treinamento e educação em segurança da informação;
 - Processamento correto nas aplicações, a fim de prevenir erros, perdas, modificação não autorizada ou mau uso de informações em aplicações;
 - Gestão de vulnerabilidades técnicas;
 - Gestão de continuidade de negócios;
 - Gestão de incidentes de segurança e melhorias.
- Implementar um ambiente de segurança consistente na criação, educação e disseminação da política interna de segurança da informação para todos os envolvidos. Além disso, criar uma estrutura para manutenção, monitoramento e melhoria da segurança da informação e do comprometimento de todos os níveis administrativos, e ainda prover os recursos financeiros necessários às atividades de gestão da segurança da informação. A administração do ambiente de segurança ainda deve incluir:
 - Estabelecimento de um processo de gestão de incidentes de segurança;
 - Implementação de um sistema de medição, que colha dados para a avaliação de desempenho da gestão de segurança;
 - Obtenção de sugestões de melhorias;
 - Implementação de melhorias levantadas no processo.
 - Treinamento e educação em segurança da informação;

“De nada serve gerar uma política completa e correta em todos os aspectos se os funcionários da empresa não a conhecem ou aplicam. É importante deixar muito claras as sanções que podem sofrer os usuários que não cumprirem as políticas de segurança da empresa, tanto funcionários quanto clientes.” (HARPIA Consultoria, 2004, Modulo 3, Cap. 1, p 6).

Antes de se iniciar a criação de uma política de segurança, faz-se necessário avaliar os riscos que a empresa está exposta segundo a abordagem a seguir.

2.5 AVALIAÇÃO DOS RISCOS DE SEGURANÇA

“Os requisitos de segurança são identificados através de uma avaliação sistemática dos riscos de segurança. Os gastos com os controles necessitam ser balanceados de acordo com os danos causados aos negócios gerados pelas potenciais falhas de segurança. As técnicas de avaliação de riscos podem ser aplicadas em toda a organização ou apenas em parte dela, assim como em um sistema de informações individuais, componentes de um sistema específico ou serviços, quando for viável, prático e útil”. (ISO/IEC 17799, 2001, p. 2).

A cada dia vemos um aumento na quantidade de incidentes relacionados com a segurança dos sistemas de informação que comprometem os ativos das empresas.

O que antes era ficção, em muitos casos, virou realidade. As ameaças sempre existiram, a diferença é que agora o inimigo ficou mais rápido, mais difícil de ser detectado e muito mais atrevido. Por isso, é importante que toda organização deva estar sempre alerta e preparada para saber a maneira de como serão implementados os sistemas de segurança baseados em uma análise de riscos, a fim de evitar ou minimizar consequências indesejadas.

A partir disso, torna-se cada vez mais fundamental conhecer com detalhes o ambiente que oferece suporte aos processos de negócio das empresas em relação a sua composição e seu caráter crítico, priorizando as ações de segurança dos principais processos empresariais, mais urgentes e vinculados ao alcance dos objetivos da organização. Muitos são os cenários em que podemos avaliar os termos de segurança da informação e da análise de riscos. Com isso, podemos citar alguns exemplos, nos quais fica difícil distinguir a ficção da realidade:

- i) Depois de 60 segundos de tentativas, um *hacker* consegue entrar no sistema de segurança do Departamento de Defesa dos Estados Unidos. Você considera esse cenário ficção ou realidade? Talvez tenha pensado que é uma situação que realmente aconteceu, mas estamos apenas citando uma cena do filme “A Senha” (*Sword Fish*), portanto, neste caso, um intruso que tem acesso não-autorizado ao sistema de informações do Departamento de Defesa é só ficção (conforme a figura 1). Entretanto, os cenários que enfrentamos diariamente são cada vez mais parecidos com esse tipo de ficção. Conseqüentemente, é importante agir e estar conscientes com relação às possíveis falhas de segurança da empresa.



FIGURA 1 – Filme “A Senha” (*Sword Fish*)
Fonte: technetbrasil, 2010

- ii) Vinte e cinco dias depois de detectar uma vulnerabilidade em um aplicativo de uso comum na rede, foi criado um vírus para aproveitar essa fragilidade. Quatorze dias depois, ele é espalhado pela *Internet* e milhares de computadores de muitos usuários em todo o mundo que têm esse aplicativo vulnerável são infectados. Será realidade ou ficção? Claro que estamos diante de um caso que, embora não pareça ser possível, é uma realidade e ocorreu de verdade. Essa é a história da criação do vírus do tipo *worm* conhecido como *Blaster*, segundo notícia mostrada a seguir na figura 2. Hoje em dia, as vulnerabilidades são exploradas a uma velocidade muito maior pelas pessoas cuja intenção é danificar os ativos das empresas. Essa velocidade implica que devemos estar mais alertas do que nunca, não só agindo, como também prevenindo possíveis riscos e problemas de segurança.



FIGURA 2 – Worm “*Blaster*”
Fonte: technetbrasil, 2010

Segundo a NTC ISO/IEC 27002 (2007), A avaliação de riscos deve identificar, quantificar e priorizar os riscos diante dos critérios para a verificação das ameaças e objetivos pertinentes a empresa. Os resultados devem direcionar e determinar as ações adequadas à gestão e as prioridades tanto para a gestão das ameaças como para implementação dos controles selecionados para execução da proteção.

É recomendável que a avaliação dos riscos inclua o enfoque sistemático para estimar a magnitude e o processo de comparação das ameaças perante os critérios para determinar a importância dessas ameaças.

Por isso, é conveniente realizar periodicamente a avaliação de riscos para abordar os meios e os requisitos de segurança e a situação de risco, por exemplo: em ativos, ameaças, vulnerabilidades, impactos, valorização de riscos e quando produzem meios significativos. Estas avaliações de riscos devem executar de forma metódica que produzam resultados comparáveis e reproduzíveis. É objetivo da análise de riscos identificar e avaliar os riscos e ameaças pelo qual o sistema de TI e seus ativos estão expostos, a fim de identificar e selecionar medidas apropriadas.

Segundo Bauer (2006), as ameaças à segurança podem ser classificadas como:

- **Acesso não autorizado:** descoberta da informação de um usuário por outro que o utiliza para acessar os recursos do primeiro.
- **Acesso por imitação** (*"Spoofing Attacks"*): um usuário ou sistema comporta-se como um outro, para obtenção de informação, recursos ou prejudicar o serviço.
- **Negação de Serviço** (*"Denial of Service"*, DoS): A forma mais conhecida de ataque, que consiste na perturbação de um serviço, devido a danos físicos ou lógicos causados no sistema que o suportam. Para provocar um DoS, os atacantes disseminam vírus, geram grandes volumes de tráfego de forma artificial, ou muitos pedidos aos servidores, o que causa sobrecarga e estes últimos ficam impedidos de processar os pedidos normais.

Independente do tipo de risco a ser considerado, uma avaliação desta geralmente incluem os seguintes passos:

- Identificar ameaças que podem causar danos e afetar ativos e operações críticas. Ameaças incluem itens como intrusões, crimes,

empregados insatisfeitos, terrorismo e desastres naturais;

- Estimar a probabilidade da concretização das ameaças, baseado em informações históricas e julgamento de conhecimentos individuais;
- Identificar e qualificar o valor, susceptibilidade e criticidade da operação e do ativo que poderá ser afetado se a ameaça se concretizar, a fim de determinar quais operações e ativos são mais importantes;
- Identificar o custo das ações para eliminar ou reduzir o risco. Isso poderá incluir a implementação de novas políticas organizacionais e procedimentos, bem como controles físicos ou técnicos;
- Documentar os resultados e desenvolver planos de ação.

O nível do risco existente em uma organização pode ser categorizado como:

- **Alto:** requer imediata atenção e implementação de contramedidas;
- **Médio:** Requer atenção e implementação de contramedidas em um futuro próximo;
- **Baixo:** Requer alguma atenção e consideração para implementação de contramedidas como boas práticas de negócios.

Cada ameaça e vulnerabilidade identificada, também deve ser qualificada. Essa classificação varia conforme a organização e o departamento. Os níveis de qualificação das ameaças podem ser definidos como:

- **Não aplicável:** significa que a ameaça considerada não é relevante para a situação examinada;
- **Baixo:** não há histórico e considera-se que é improvável a concretização da ameaça;
- **Médio:** significa que há algum histórico e probabilidade que a ameaça se concretize;
- **Alto:** significa que há um histórico significativo e uma avaliação de que a ameaça está por acontecer.

Ao alcance da avaliação de riscos estão diretrizes para a tecnologia de segurança da informação: técnicas para gestão que podem ser exemplificadas em metodologias de tratamento desses riscos que veremos a seguir.

2.6 TRATAMENTOS DOS RISCOS DE SEGURANÇA

Antes de considerar o tratamento de um risco, a empresa deve decidir os critérios para determinar a possibilidade de ocorrência de falhas. Essas falhas podem ocorrer se a avaliação considerar riscos de baixa expectativa, o que é um custo aos métodos não efetivos em termos financeiros da organização.

Para cada um dos riscos identificados depois da avaliação é necessário tomar decisões para o tratamento. As opções possíveis são:

- Aplicação dos controles apropriados para redução de erros;
- Direcionamento objetivo e conhecimento das falhas, sempre e quando elas satisfaçam a política de segurança da corporação e seus critérios de aceitação;
- Evitar os riscos de acessos não permitidos que possam estar presentes;
- Transferência dos riscos acionados a outras partes, assegurando, por exemplo, os servidores de dados uma empresa.

Para aqueles riscos onde o tratamento exige uma aplicação de controles, devemos selecionar e implementá-los de modo que satisfaçam os critérios de avaliação de riscos. Os controles devem garantir a redução de falhas a níveis aceitáveis atendendo aos seguintes elementos:

- Requisitos e restrições da legislação e das regulamentações nacionais e internacionais;
- Objetivos da corporação;
- Requisitos e restrições operacionais;
- Custo de implementação e a operação com relação aos riscos reduzidos, e que permaneça proporcional aos resultados e restrições da empresa;
- Necessidade de equilibrar a inversão na implementação e operação dos controles diante da probabilidade do dano que resultará devido às falhas de segurança.

É necessário reconhecer que é possível que alguns controles podem não se aplicar a todos os sistemas e escopos de informações e podendo não ser viável para todas as organizações. Um exemplo disso são as formas de segregar as funções para se evitar fraudes e erros. Os controles selecionados, para registros de eventos, podem entrar em conflito e expor, por exemplo, a proteção de dados privados de clientes guardados num banco de dados.

Uma forma de se ter uma melhor eficiência no tratamento dos riscos é conhecendo os tipos de vulnerabilidades e ameaças mais difundidas a que a segurança da informação pode ser exposta ou atingida durante uma falha, uma negação de acesso de pessoas autorizadas ou uma invasão de um *hacker*.

3 VULNERABILIDADES E AMEAÇAS

Segundo Bauer apud CERT.br (2006), no contexto da tecnologia da informação, vulnerabilidade é definida como uma falha no projeto, implementação ou configuração de um *software* ou sistema operacional que, quando explorada por um *hacker*, resulta na violação da segurança de um computador ou rede de computadores.

De acordo com o livro Universidade H4ck3r (2003), existem três tipos de falhas mais comuns que podem ser exploradas pelos *hackers*:

- Falhas que afetam a disponibilidade;
- Falhas que dão acesso limitado ao sistema, exemplificadas tipicamente pelas falhas de configuração ou num erro de atualização do sistema;
- Falhas que permitem a execução de códigos arbitrários.

Os desenvolvedores de *softwares*, em muitos casos, não disponibilizam novas versões de seus produtos quando alguma vulnerabilidade é descoberta, mas correções, comumente conhecidas de *patches*.

Algumas vulnerabilidades, as mais comuns, são causadas devido ao mau uso, descuido, ou inadvertidas, por parte dos funcionários de recursos como: o correio eletrônico, a *internet*, ou instalação de *softwares* não confiáveis. São através de meios como esses que se propagam os vírus, *spywares*, *trojan-horses* entre outros.

Segundo Nakamura e Geus (2007), É preciso ter em mente que novas vulnerabilidades surgem diariamente. Como novas tecnologias e novos sistemas são sempre criados, é razoável considerar que novas vulnerabilidades sempre existirão e, portanto, novos ataques também serão sempre criados. A própria história mostra uma evolução constante das técnicas usadas para ataques, que estão cada vez mais sofisticadas. A mistura de diferentes técnicas, o uso de tecnologia para cobrir vestígios a cooperação entre atacantes e a criatividade são fatores que tornam a defesa mais difícil do que o habitual.

Para o *hacker*, basta que ele consiga explorar apenas um ponto de falha da organização para realizar um ataque. Um exemplo de ataque é o de negação de serviço.

3.1 NEGAÇÕES DE SERVIÇO (DENIAL OF SERVICE)

De acordo com CERT.br (2006), nos ataques de negação de serviço (DoS -- *Denial of Service*), o *hacker* utiliza um computador para tirar de operação um serviço ou computador conectado à *Internet*.

Exemplos deste tipo de ataque são:

- gerar uma grande sobrecarga no processamento de dados de um computador, de modo que o usuário não consiga utilizá-lo;
- gerar um grande tráfego de dados para uma rede, ocupando toda a banda disponível, de modo que qualquer computador dessa rede fique indisponível;
- tirar serviços importantes de um provedor do ar, impossibilitando o acesso dos usuários a suas caixas de correio no servidor de correio eletrônico ou ao servidor *de internet*.

Constitui-se de um ataque de negação de serviço distribuído (DDoS - *Distributed Denial of Service*) um conjunto de computadores utilizado para tirar de operação um ou mais serviços ou computadores conectados à *Internet*. É um tipo de ataque que procura ocupar a banda disponível para o acesso a um computador ou a uma rede, causando lentidão ou até mesmo indisponibilidade de comunicação com esse computador ou rede.

3.2 SPAM

Spam conforme definição da CERT.br (2006), é o termo usado para se referir às mensagens eletrônicas não solicitadas, que geralmente são enviados para um grande número de pessoas. Quando o conteúdo é exclusivamente comercial, esse tipo de mensagem também é referenciada como UCE (do inglês *Unsolicited Commercial E-mail*).

Alguns dos problemas sentidos pelos provedores e empresas são:

- **Impacto na banda:** Para as empresas e provedores o volume de tráfego gerado por causa de *spams* os obriga a aumentar a capacidade de seus *links* de conexão com a *Internet*. Como o custo dos *links* é alto, isso diminui os lucros do provedor e muitas vezes podem refletir no aumento dos custos para o usuário;
- **Má utilização dos servidores:** Os servidores de *e-mail* dedicam boa parte do seu tempo de processamento para tratar das mensagens não solicitadas. Além disso, o espaço em disco ocupado por mensagens não solicitadas enviadas para um grande número de usuários é considerável;
- **Inclusão em listas de bloqueio:** O provedor que tenha usuários envolvidos em casos de *spam* pode ter sua rede incluída em listas de bloqueio. Essa inclusão pode prejudicar o recebimento de *e-mails* por parte de seus usuários e ocasionar a perda de clientes;
- **Investimento em pessoal e equipamentos:** Para lidar com todos os problemas gerados pelo *spam*, os provedores necessitam contratar mais técnicos especializados, comprar equipamentos e acrescentar sistemas de filtragem de *spam*.

3.3 MALWARE

Segundo CERT.br (2006), *Malware* (**Malicious Software**) é um termo genérico que abrange todos os tipos de programa especificamente desenvolvidos para executar ações maliciosas em um computador. Na literatura de segurança o termo *malware* também é conhecido por "*software* malicioso".

São exemplos de *malware*:

- vírus;
- *worms* e *bots*;
- *backdoors*;
- cavalos de tróia;
- *keyloggers* e outros programas *spyware*;
- *rootkits*.

3.3.1 Vírus

De acordo com CERT.br (2006), um vírus é um código de computador que se anexa a um programa ou arquivo para poder se espalhar entre os computadores, infectando-os à medida que se desloca. Os vírus podem danificar os *softwares*, *hardwares* e arquivos.

O vírus depende da execução do programa ou arquivo hospedeiro para que possa se tornar ativo, dando continuidade ao processo de infecção.

Normalmente o vírus tem controle total sobre o computador, podendo fazer de tudo, desde mostrar uma mensagem, até alterar ou destruir programas e arquivos do disco.

Para que um computador seja infectado por um vírus, é preciso que um programa que contenha o vírus seja executado. Isso pode ocorrer de diversas maneiras, tais como:

- abrir arquivos anexados aos *e-mails*;
- abrir arquivos do Word, Excel, etc;
- abrir arquivos armazenados em outros computadores, através do compartilhamento de recursos;
- instalar programas de procedência duvidosa ou desconhecida, obtidos pela Internet, de disquetes, *pen drives*, CDs, DVDs, etc;
- ter alguma mídia removível (infectada) conectada ou inserida no computador, quando ele é ligado.

Existem vírus que procuram permanecer ocultos, infectando arquivos do disco e executando uma série de atividades sem o conhecimento do usuário. Ainda existem outros tipos que permanecem inativos durante certos períodos, entrando em atividade em datas específicas.

Um vírus propagado por *e-mail* (*e-mail borne vírus*), é recebido como um arquivo anexado a uma mensagem de correio eletrônico. O conteúdo dessa mensagem procura induzir o usuário a clicar sobre o arquivo anexado, fazendo com que o vírus seja executado. Quando esse tipo de vírus entra em ação, ele infecta arquivos e programas e envia cópias de si mesmo para os contatos encontrados nas listas de endereços de *email* armazenadas no computador do usuário.

Esse tipo específico de vírus não é capaz de se propagar automaticamente. O usuário precisa executar o arquivo anexado que contém o vírus, ou o programa leitor de *emails* precisa estar configurado para auto-executar arquivos anexados.

Denomina-se com um vírus de macro um conjunto de comandos armazenados em alguns aplicativos e utilizados para automatizar algumas tarefas repetitivas.

Para que o vírus possa ser executado, o arquivo que o contém precisa ser aberto e, a partir daí, o vírus pode executar uma série de comandos automaticamente e infectar outros arquivos no computador.

De acordo com a CERT.br (2006); algumas das medidas de prevenção contra a infecção por vírus são:

- instalar e manter atualizados um bom programa antivírus e suas assinaturas;
- desabilitar no seu programa leitor de *e-mails* a auto-execução de arquivos anexados às mensagens;
- não executar ou abrir arquivos recebidos por *e-mail* ou por outras fontes, mesmo que venham de pessoas conhecidas. Caso seja necessário abrir o arquivo, certifique-se que ele foi verificado pelo programa antivírus;
- procurar utilizar na elaboração de documentos formatos menos suscetíveis à propagação de vírus, tais como RTF, PDF ou PostScript;
- procurar não utilizar, no caso de arquivos comprimidos, o formato executável. Utilize o próprio formato compactado, como por exemplo Zip ou Rar.

3.3.2 Worms

Segundo a CERT.br (2006), *Worm* é um programa capaz de se propagar automaticamente através de redes, enviando cópias de si mesmo de computador para computador. Diferente do vírus, o *worm* não introduz cópias de si mesmo em outros programas ou arquivos e não necessita ser explicitamente executado para se propagar. Sua propagação se dá através da exploração de vulnerabilidades

existentes ou falhas na configuração de *softwares* instalados em computadores.

Geralmente, o *worm* não tem como consequência os mesmos danos gerados por um vírus, como, por exemplo, a infecção de programas e arquivos ou a destruição de informações. Isso não quer dizer que não represente uma ameaça à segurança de um computador, ou que não cause qualquer tipo de dano.

Worms são notadamente responsáveis por consumir muitos recursos. Degradam sensivelmente o desempenho de redes e podem lotar o disco rígido de computadores, devido à grande quantidade de cópias de si mesmo que costumam propagar. Além disso, podem gerar grandes transtornos para aqueles que estão recebendo tais cópias.

Detectar a presença de um *worm* em um computador não é uma tarefa fácil. Muitas vezes ele realiza uma série de atividades, incluindo sua propagação, sem que o usuário tenha conhecimento. Embora alguns programas antivírus permitam detectar a presença de *worms* e até mesmo evitar que eles se propaguem isto nem sempre é possível. Portanto, o melhor é evitar que seu computador seja utilizado para propagá-los.

3.3.3 Bots e botnets

De modo similar ao *worm*, o *bot* é um programa capaz de se propagar automaticamente, explorando vulnerabilidades existentes ou falhas na configuração de *softwares* instalados em um computador. Adicionalmente ao *worm*, dispõe de mecanismos de comunicação com o invasor, permitindo que o *bot* seja controlado remotamente.

Segundo a CERT.br (2006), normalmente o *bot* se conecta a um servidor de IRC (*Internet Relay Chat*) e entra em um canal determinado. Então, ele aguarda por instruções do invasor, monitorando as mensagens que estão sendo enviadas para esse canal. O invasor, ao se conectar ao mesmo servidor de IRC e entrar no mesmo canal, envia mensagens compostas por seqüências especiais de caracteres, que são interpretadas pelo *bot*. Essas seqüências de caracteres correspondem a instruções que devem ser executadas pelo *bot*.

Um invasor, ao se comunicar com um *bot*, pode enviar instruções para que ele realize diversas atividades, tais como:

- desferir ataques na *Internet*;
- executar um ataque de negação de serviço;
- furtar dados do computador onde está sendo executado, como por exemplo números de cartões de crédito;
- enviar *emails* de *phishing*;
- enviar *spam*.

Aprofundando o estudo sobre *bots*, *Botnets* são redes formadas por computadores infectados com *bots*. Essas redes podem ser compostas por centenas ou milhares de computadores. Um invasor que tenha controle sobre uma *botnet* pode utilizá-la para aumentar a potência de seus ataques, por exemplo, para enviar centenas de milhares de *emails* de *phishing* ou *spam*, desferir ataques de negação de serviço, etc. O *bot* é projetado para realizar as instruções passadas pelo invasor sem que o usuário tenha conhecimento.

A melhor forma de se proteger dos *bots* é manter o sistema operacional e os *softwares* instalados em seu computador sempre atualizados e com todas as correções de segurança (*patches*) disponíveis aplicadas, para evitar que possuam vulnerabilidades.

3.3.4 Trojan horse ou cavalos de tróia

De acordo com CERT.br (2006), um cavalo de tróia ou *trojan horse* é um programa, recebido como: um cartão virtual, álbum de fotos, protetor de tela, jogo, etc. Além de executar funções para as quais foi aparentemente projetado, também executa outras funções, normalmente maliciosas e sem o conhecimento do usuário.

Algumas das funções maliciosas que podem ser executadas por um cavalo de tróia, são:

- Instalação de *keyloggers* ou *screenloggers* (que serão abordados em seguida);
- Furto de senhas e outras informações sensíveis, como números de cartões de crédito;

- Inclusão de *backdoors* (também será abordado em seguida), para permitir que um *hacker* tenha total controle sobre o computador;
- Alteração ou destruição de arquivos.

Por definição, o cavalo de tróia distingue-se de um vírus ou de um *worm* por não infectar outros arquivos, nem propagar cópias de si mesmo automaticamente. Normalmente consiste em um único arquivo que necessita ser explicitamente executado.

Podem existir casos onde um cavalo de tróia contenha um vírus ou *worm*. Mas mesmo nestes casos é possível distinguir as ações realizadas como consequência da execução do cavalo de tróia propriamente dito, daquelas relacionadas ao comportamento de um vírus ou *worm*.

Enquanto estão sendo executados, podem executar diversas ações simultaneamente como: enviar dados confidenciais para outro computador, instalar *backdoors*, alterar informações, apagar arquivos, formatar o disco rígido ou apenas exibir uma mensagem de erro.

Uma medida preventiva contra *trojan horse* é a utilizar de um *firewall* pessoal. Alguns *firewalls* permitem bloquear o recebimento desses programas.

3.3.5 Adware e spyware

Segundo CERT.br (2006), *Adware* (*Advertising software*) é um tipo de *software* especificamente projetado para apresentar propagandas, seja através de um *browser*, seja através de algum outro programa instalado em um computador.

Em muitos casos, os *adwares* têm sido incorporados a *softwares* e serviços, constituindo uma forma legítima de patrocínio ou retorno financeiro para aqueles que desenvolvem *software* livre ou prestam serviços gratuitos. Um exemplo do uso legítimo de *adwares* pode ser observado no programa de troca instantânea de mensagens MSN Messenger.

Spyware é o termo utilizado para se referir a uma grande categoria de *software* que tem o objetivo de monitorar atividades de um sistema e enviar as informações coletadas para terceiros. Existem *adwares* que também são considerados um tipo de *spyware*, pois são projetados para monitorar os hábitos do

usuário durante a navegação na *Internet*, direcionando as propagandas que serão apresentadas.

Seguem algumas funcionalidades implementadas em *spywares*, que podem ter relação com o uso legítimo ou malicioso:

- monitoramento de URLs acessadas enquanto o usuário navega na *Internet*;
- alteração da página inicial apresentada no *browser* do usuário;
- varredura dos arquivos armazenados no disco rígido do computador;
- monitoramento e captura de informações inseridas em outros programas, como IRC ou processadores de texto;
- instalação de outros programas *spyware*;
- monitoramento de teclas digitadas pelo usuário ou regiões da tela próximas ao clique do mouse;
- captura de senhas bancárias e números de cartões de crédito;
- captura de outras senhas usadas em sites de comércio eletrônico.

Existem ferramentas específicas, conhecidas como “anti-*spyware*”, capazes de detectar e remover uma grande quantidade de programas *spyware*. Algumas destas ferramentas são gratuitas para uso pessoal e podem ser obtidas pela *Internet*.

3.3.6 Backdoors

“Aos programas que permitem o retorno de um invasor a um computador comprometido, utilizando serviços criados ou modificados para este fim, dá-se o nome de *backdoor*.” CERT.br (2006).

Conforme CERT.br (2006), normalmente um *hacker* procura garantir uma forma de retornar a um computador comprometido, sem precisar recorrer aos métodos utilizados na realização da invasão. Na maioria dos casos, também é intenção do *hacker* poder retornar ao computador comprometido sem ser notado.

A forma usual de inclusão de um *backdoor* consiste na disponibilização de um novo serviço ou substituição de um determinado serviço por uma versão alterada,

normalmente possuindo recursos que permitam acesso remoto (através da *Internet*). Pode ser incluído por um invasor ou através de um cavalo de tróia.

Outra forma de utilização remota é a instalação de pacotes de *software*, tais como o *BackOrifice* e *NetBus*, da plataforma *Windows*. Se mal configurados ou utilizados sem o consentimento do usuário, podem ser classificados como *backdoors*.

Há casos onde a existência de um *backdoor* não está associada a uma invasão são:

- instalação através de um cavalo de tróia;
- inclusão como consequência da instalação e má configuração de um programa de administração remota.

Alguns fabricantes incluem *backdoors* em seus produtos alegando necessidades administrativas. É importante ressaltar que estes casos constituem uma séria ameaça à segurança de um computador que contenha um destes produtos instalados, mesmo que *backdoors* sejam incluídos por fabricantes conhecidos.

Backdoors podem ser incluídos em computadores executando nos mais diversos sistemas operacionais, tais como *Windows*, *Unix*, *Mac OS*, *Linux*, entre outros.

3.3.7 Keyloggers

De acordo com CERT.br (2006), um *Keylogger* é um programa capaz de capturar e armazenar as teclas digitadas pelo usuário no teclado de um computador. Dentre as informações capturadas podem estar senhas bancárias e números de cartões de crédito. Dentre as informações capturadas podem estar o texto de um *e-mail*, dados digitados na declaração de Imposto de Renda e outras informações sensíveis, como senhas bancárias e números de cartões de crédito.

Em muitos casos, a ativação do *keylogger* é condicionada a uma ação prévia do usuário, como por exemplo, após o acesso a um *site* específico de comércio

eletrônico ou *Internet Banking*³. Muitas instituições financeiras (bancos *on-line*) desenvolveram os teclados virtuais para evitar que os *keyloggers* pudessem capturar informações de conta e senhas de usuários. Então, foram desenvolvidas novas modalidades de *keyloggers* os *screenloggers*, capazes de:

- Armazenar a posição do cursor e a tela apresentada no monitor, nos momentos em que o *mouse* é clicado;
- Armazenar a região que circunda a posição onde o *mouse* é clicado.

De posse destas informações um *hacker* pode, por exemplo, descobrir a senha de acesso ao banco, utilizada por um usuário.

Um *keylogger* normalmente vem como parte de um programa *spyware* ou *trojan horse*. Dessa forma, é necessário que este programa seja executado para que o *keylogger* se instale no computador. Geralmente, tais programas vêm anexados a *emails* ou estão disponíveis em *sites* na *Internet*.

3.3.8 Rootkits

Um *hacker*, ao realizar um ataque, pode se utilizar de mecanismos para esconder e assegurar a sua presença no computador invadido. O conjunto de ferramentas que fornece meios para assegurar essa presença, dar-se o nome de *rootkit*.

O nome *rootkit* não indica que as ferramentas que o compõem são usadas para obter acesso privilegiado (*root* ou Administrador) em um computador, mas sim para mantê-lo. Isto significa que o *hacker*, após inicializar o *rootkit*, terá acesso privilegiado ao computador previamente comprometido, sem precisar recorrer novamente aos métodos utilizados na realização do ataque, e suas atividades ficaram invisíveis aos usuários do computador.

Um *rootkit* pode fornecer as mais diversas funcionalidades. Dentre elas, podemos citar as seguintes:

³ *Internet Banking* : banco que disponibiliza acesso a movimentação ao cliente (banco on-line)

- programas para camuflar ações e informações deixadas pelo invasor (normalmente presentes em todos os *rootkits*), tais como: arquivos, diretórios, processos, conexões de rede;
- *backdoors*, para assegurar o acesso futuro do invasor ao computador comprometido (presentes na maioria dos *rootkits*);
- programas para remoção de evidências em arquivos de *logs*;
- *sniffers*, para capturar informações na rede onde o computador está localizado, como por exemplo senhas que estejam trafegando em claro, ou seja, sem qualquer método de criptografia;
- *scanners*, para mapear potenciais vulnerabilidades em outros computadores;
- diversos tipos de *malware*, como: cavalos de tróia, *keyloggers*, ferramentas de ataque de negação de serviço.

Existem programas que são capazes de detectar a presença de *rootkits*, porem não quer dizer que serão capazes de detectar todos os existentes (principalmente aos mais recentes). Alguns desses programas são gratuitos e podem ser obtidos pela *Internet* (antes de obter esse tipo de programa, certifique que o fabricante é confiável).

Como os *rootkits* são projetados para ficar ocultos, sua identificação é uma tarefa difícil. Dessa forma procure evitar que um *rootkit* seja instalado em seu computador.

3.4 SNIFFERS

Sniffers são programas de computador que monitoram passivamente o tráfego de rede. Eles podem ser utilizados, legitimamente, pelo administrador do sistema para verificar problemas de rede, ou podem ser usados ilegalmente por um intruso para roubar nomes de usuários e senhas. Esse tipo de programa explora o fato dos pacotes das aplicações *TCP/IP*⁴ não serem criptografados.

⁴ TCP (*Transmission Control Protocol*): Protocolo de Controle de Transmissão - IP: *Internet Protocol*. Endereço de computador em uma rede, utilizado para a comunicação entre computadores na rede.

Entretanto, para utilizar o *sniffer*, é necessário que ele esteja instalado em um ponto da rede onde passe tráfego de pacotes de interesse para o invasor ou ao administrador da rede.

3.5 PORT SCAN

Segundo Bauer apud Coimbra (2006), O *portscan* é o ponto inicial de muitos ataques, a partir do qual estratégias são elaboradas, com base na identificação de quais *daemons*⁵ rodam na máquina e suas respectivas versões e portas. Portanto, com uma identificação positiva da origem, pode-se muito provavelmente identificar tentativas de ataque e seus autores e bloqueá-las.

O papel de um *portscan* é percorrer todas as portas de um endereço IP especificado, com um intervalo geralmente determinado pelo utilizador. Então é feita uma checagem a procura de conexões nessas portas, a fim de identificar qual *daemon* e versão responde por determinada porta.

O objetivo de um *hacker* ao disparar um *portscan* contra um IP específico, é o de tentar encontrar *daemons* que contenham vulnerabilidades exploráveis através de um código *brute force*⁶ ou mesmo de um *exploit*⁷.

Programas conhecidos por *detect-scan*, têm o papel de evitar que pessoas estranhas tomem conhecimento de quais serviços e versões sua máquina ou servidor disponibiliza. Para isso, toma como princípio básico evitar os *portscans*, colocando "escutas" em portas *TCP* e/ou *UDP* especificadas. Essas "escutas", logicamente, são feitas através do *detect-scan* que roda em *background*, monitorando as portas específicas. Quando detecta algo semelhante a um *portscan*, pode executar diversas funções, tais como *logar*, bloquear ou executar comandos externos pré-estabelecidos em sua configuração.

5 *Daemon*: é um monitor de execução e disco, programa permanentemente ativo em um sistema ligado na rede que aguarda instruções externas para disparar ações específicas.

6 *Brute force* (força bruta): testar combinações de chaves até que a correta seja descoberta

7 *Exploit* é um programa de computador que permite tirar partido de vulnerabilidades de outros programas como o próprio sistema operacional ou serviços de interação de protocolos (ex: Servidores Web).

3.6 ENGENHARIA SOCIAL

“Engenharia Social é a ciência que estuda como o conhecimento do comportamento humano pode ser utilizado para induzir uma pessoa a atuar segundo seu desejo. Não se trata de hipnose ou controle da mente, as técnicas de Engenharia Social são amplamente utilizadas por detetives (para obter informação), e magistrados (para comprovar se um declarante fala a verdade). Também é utilizada para lograr todo tipo de fraudes, inclusive invasão de sistemas eletrônicos” (KONSULTEX Informática, 1993).

Conforme um dos maiores especialistas em engenharia social, Kevin Mitnick: *“É um termo diferente para definir o uso de persuasão para influenciar as pessoas a concordar com um pedido”*(Mitnick, 2004).

Segundo Kevin Mitnick (2003), apenas 50% dos problemas com segurança em sistemas de computação ocorrem devido à tecnologia. A outra parte é decorrente de falhas humanas. É necessário que seja feito um trabalho de conscientização e educação nas pessoas para garantir que o investimento em segurança técnica não seja desperdiçado. As pessoas precisam saber o valor da informação e a segurança precisa fazer parte da função dos funcionários.

O termo engenharia social está sendo utilizado para definir técnicas utilizadas para tirar proveito de falhas que as pessoas cometem ou que sejam levadas a cometer com relação às informações da área de tecnologia da informação. Segundo o especialista Rich Mogull, diretor de pesquisas do Gartner Research, “essa é a maior ameaça à continuidade dos negócios para a próxima década” (*InformationWeek* Brasil, 2003).

Segundo Bauer (2006), podemos citar as seguintes ferramentas utilizadas pelo praticante da engenharia social:

- **Telefone:** se passar por alguém que não é. Seria um dos típicos ataques de engenharia social, como na personificação – *help-desk*;
- **Internet:** coleta de informações – como, por exemplo, *sites* que fornecem ID e *passwords default*, *sites* clonados, etc.;
- **E-mail:** *Fakemail* (e-mails com origem falsa);
- **Pessoalmente:** Usando o poder de persuasão (tipo de ataque mais raro);
- **Chats:** Assim como no telefone, fazer-se passar por alguém que na verdade não é, fica muito mais fácil pelos canais de bate-papo. Pois

além de tudo, mandar fotos fica bem mais atrativo e seduz mais facilmente a conseguir informações que deseja;

- **Cartas/correspondência:** Não é o meio mais moderno sem dúvida, mas um recurso poderoso que faz como suas maiores vítimas, pessoas mais velhas. Não somente idosos, mas principalmente aqueles que têm resistência à tecnologia;
- **Spyware:** *software* especializado, usado para monitorar de modo oculto as atividades do computador de um alvo;
- **Mergulho no lixo:** O lixo muitas vezes contém informações essenciais ao engenheiro social;
- **“Surfar sobre os ombros”:** É o ato de observar uma pessoa digitando no teclado do computador para descobrir e roubar sua senha ou outras informações de usuário.

4 RECURSOS DE DEFESA DA REDE

4.1 FIREWALLS

Segundo Bauer apud Compy (2006), *Firewalls* são dispositivos constituídos pela combinação de *software* e *hardware*, utilizados para dividir e controlar o acesso entre redes de computadores. Alguns programas de *firewall* permitem analisar continuamente o conteúdo das conexões, filtrando vírus de *email*, cavalos de tróia e outros tipos de *malware*, antes mesmo que os antivírus entrem em ação. Também existem pacotes de *firewall* que funcionam em conjunto com os antivírus, provendo um maior nível de segurança para os computadores em que são utilizados.

Os *firewalls* criam arquivos no computador, denominados arquivos de registro de eventos (*logs*). Nesses arquivos, são armazenadas as tentativas de acesso não autorizado à rede, para serviços que podem ou não estar habilitados.

Tão importante quanto proteger o sigilo das informações é saber se ele está vulnerável.

4.2 SISTEMAS DETECTORES DE INTRUSÃO

“Os sistemas detectores de intrusão são sistemas automáticos baseados em mecanismos de monitoração em tempo real que observam o comportamento do tráfego de rede e ambientes operacionais, que em seu conjunto procuram identificar padrões considerados hostis.”(ANTONIO ALEXANDRE DE CASTRO SOARES, Sistemas Detectores de Intrusão em Redes Ad Hoc, p.2)

Segundo Bauer (2006), ferramentas IDS, ou *Intrusion Detection System*, trabalham analisando os pacotes que trafegam na rede e comparando-os com assinaturas de ataques conhecidas.

Uma ferramenta IDS serve para fornecer informações sobre a rede, tais como:

- Tentativas de ataques diárias;
- Tipo de ataque usado;

- Origem dos ataques;

A partir dele, pode-se tomar conhecimento do que realmente se passa na rede e, em casos extremos, tomar as medidas cabíveis para tentar solucionar qualquer problema.

4.2.1 Honeypot

“Um *honeypot* é um recurso de rede cuja função é de ser atacado e comprometido (invadido). Significa dizer que um *Honeypot* poderá ser testado, atacado e invadido. Os *honeypots* não fazem nenhum tipo de prevenção, os mesmos fornecem informações adicionais de valor inestimável” (Lance Spitzner – 2003)

É um sistema que possui falhas de segurança reais ou virtuais, colocadas de maneira proposital, a fim de que seja invadido.

Os *Honeypots* podem ser classificados da seguinte forma:

- Baixa Interatividade: Serviços Falsos. *Listener* TCP/UDP. Gerar respostas Falsas.
- Média Interatividade: Ambiente falso. Cria uma ilusão de domínio da máquina. Invasão real do sistema.
- Alta Interatividade: Sistema Operacional com serviços comprometidos, não perceptíveis ao atacante. Contem riscos: Utilização como trampolim ou como repositório de informações roubadas

4.3 CRIPTOGRAFIA

Criptografia é a técnica de escrever uma informação e transformá-la em uma mensagem cifrada ou em um código.

A criptografia pode ser usada para proporcionar as seguintes características:

- **Confidencialidade:** Garantir que os dados permaneçam privados. Os algoritmos de criptografia são usados para converter texto sem formatação em texto codificado e o algoritmo de decodificação é usado

para converter o texto codificado em texto ao formato original.

- **Integridade de dados:** Garantir que os dados sejam protegidos contra modificação accidental ou mal-intencionada. A integridade, geralmente, é fornecida por códigos de autenticação de mensagem ou *hashes*. Os *hashes* são usados para verificar a integridade dos dados enviados por canais não seguros.
- **Autenticação:** Garantir que os dados se originem de uma parte específica. Os certificados digitais são usados para fornecer autenticação. As assinaturas digitais geralmente são aplicadas a valores de *hash*, uma vez que eles são significativamente menores que os dados de origem que representam.

De acordo com CERT.br (2006), uma mensagem codificada por um método de criptografia deve ser privada, ou seja, somente aquele que enviou e aquele que recebeu devem ter acesso ao conteúdo da mensagem. Além disso, uma mensagem deve poder ser assinada, ou seja, a pessoa que a recebeu deve poder verificar se o remetente é mesmo a pessoa que diz ser, tendo a capacidade de identificar se a mensagem foi modificada.

Os métodos de criptografia atuais são seguros e eficientes e baseiam-se no uso de uma ou mais chaves. A chave é uma seqüência de caracteres que podem conter letras, dígitos e símbolos (como uma senha), e que é convertida em um número, utilizada pelos métodos de criptografia para codificar e decodificar mensagens.

Os métodos criptográficos podem ser subdivididos em duas grandes categorias, de acordo com o tipo de chave utilizada: a criptografia de chave única e a criptografia de chave pública e privada (Ferreira e Araújo, 2005).

4.3.1 Criptografia de chaves pública

Segundo Bauer apud Modulo (2006), A criptografia de chaves pública e privada utiliza duas chaves distintas: uma para codificar e outra para decodificar mensagens. Nesse método, cada pessoa ou entidade mantém duas chaves: uma pública, que pode ser divulgada livremente, e outra privada, que deve ser mantida

em segredo pelo seu dono. As mensagens codificadas com a chave pública só podem ser decodificadas com a chave privada correspondente.

De acordo com CERT.br (2006), Apesar desse método ter o desempenho bem inferior em relação ao tempo de processamento, quando comparado ao método de criptografia de chave única, apresenta como principal vantagem a livre distribuição de chaves públicas, não necessitando de um meio seguro para que chaves sejam combinadas antecipadamente. Além disso, pode ser utilizado na geração de assinaturas digitais.

Os métodos de criptografia atualmente utilizados, e que apresentam bons níveis de segurança, são publicamente conhecidos e são seguros pela robustez de seus algoritmos e pelo tamanho das chaves que utilizam.

Para que um atacante descubra uma chave ele precisa utilizar algum método de “força bruta”, ou seja, testar combinações de chaves até que a correta seja descoberta. Portanto, quanto maior for a chave, maior será o número de combinações a testar, inviabilizando assim a descoberta de uma chave em tempo hábil. Além disso, chaves podem ser trocadas regularmente, tornando os métodos de criptografia ainda mais seguros.

Os métodos de criptografia ainda podem ser classificados pelo tipo de sua chave: Simétrica ou Assimétrica, e os algoritmos por elas utilizados.

4.3.1.1 Chaves simétricas

De acordo com Alecrim (2005), esse é um tipo de chave mais simples, onde o emissor e o receptor fazem uso da mesma chave, isto é, uma única chave é usada na codificação e na decodificação da informação. Existem vários algoritmos que usam chaves simétricas, como o DES, o IDEA, e o RC:

- DES (Data Encryption Standard): criado pela IBM em 1977, faz uso de chaves de 56 bits. Isso corresponde a 72 quatrilhões de combinações. É um valor absurdamente alto, mas não para um computador potente. Em 1997, esse algoritmo foi quebrado por técnicas de “força bruta” (tentativa e erro) em um desafio promovido na internet;
- IDEA (International Data Encryption Algorithm): criado em 1991 por James Massey e Xuejia Lai, o IDEA é um algoritmo que faz uso de

chaves de 128 bits e que tem uma estrutura semelhante ao DES. Sua implementação em software é mais fácil do que a implementação deste último;

- **RC (Ron's Code ou Rivest Cipher):** criado por Ron Rivest na empresa RSA Data Security, esse algoritmo é muito utilizado em e-mails e faz uso de chaves que vão de 8 a 1024 bits. Possui várias versões: RC2, RC4, RC5 e RC6. Essencialmente, cada versão difere da outra por trabalhar com chaves maiores.

Há ainda outros algoritmos conhecidos, como o AES (*Advanced Encryption Standard*) - que é baseado no DES - , o 3DES, o *Twofish* e sua variante *Blowfish*, entre outros.

O uso de chaves simétricas tem algumas desvantagens, fazendo com que sua utilização não seja adequada em situações onde a informação é muito valiosa. Para começar, é necessário usar uma grande quantidade de chaves caso muitas pessoas ou entidades estejam envolvidas. Ainda, há o fato de que tanto o emissor quanto o receptor precisam conhecer a mesma chave. A transmissão dessa chave de um para o outro pode não ser tão segura e cair em "mãos erradas".

4.3.1.2 Chave assimétrica

Também conhecida como "chave pública", a chave assimétrica trabalha com duas chaves: uma denominada privada e outra denominada pública. Neste método, um emissor deve criar uma chave de codificação e enviá-la ao receptor. Essa é a chave pública. Outra chave deve ser criada para a decodificação. Esta, a chave privada, é secreta.

Entre os algoritmos que usam chaves assimétricas, têm-se o RSA (o mais conhecido) e o ElGamal:

- **RSA (Rivest, Shamir and Adleman):** criado em 1977 por Ron Rivest, Adi Shamir e Len Adleman nos laboratórios do MIT (**M**assachusetts **I**nstitute of **T**echnology), é um dos algoritmos de chave assimétrica mais usados. Nele, números primos (número primo é aquele que só pode ser dividido por 1 e por ele mesmo) são utilizados da seguinte

forma: dois números primos são multiplicados para se obter um terceiro valor. Porém, descobrir os dois primeiros números a partir do terceiro (ou seja, fazer uma fatoração) é muito trabalhoso. Se dois números primos grandes (realmente grandes) forem usados na multiplicação, será necessário usar muito processamento para descobri-los, tornando essa tarefa praticamente inviável. Basicamente, a chave privada no RSA são os números multiplicados e a chave pública é o valor obtido;

- ElGamal: criado por Taher ElGamal, esse algoritmo faz uso de um problema matemático conhecido por "logaritmo discreto" para se tornar seguro. Sua utilização é freqüente em assinaturas digitais.

Existem ainda outros algoritmos, como o DSA (Digital Signature Algorithm), o Schnorr (praticamente usado apenas em assinaturas digitais) e Diffie-Hellman.

4.4 ASSINATURA DIGITAL

Segundo CERT.br (2006), consiste na criação de um código, através da utilização de uma chave privada, de modo que a pessoa ou entidade que receber uma mensagem contendo esse código possa verificar se o remetente é mesmo quem diz ser e identificar qualquer mensagem que possa ter sido modificada.

É importante ressaltar que a segurança do método baseia-se no fato de que a chave privada é conhecida apenas pelo seu dono. Também é importante ressaltar que o fato de assinar uma mensagem não significa gerar uma mensagem sigilosa.

4.5 CERTIFICADOS DIGITAIS

Segundo CERT.br (2006), certificado digital é um arquivo eletrônico que contém dados de uma pessoa ou instituição utilizados para comprovar sua identidade. Este arquivo pode estar armazenado em um computador ou em outra mídia, como um *token* ou *smart card*.

Exemplos semelhantes a um certificado digital são o CNPJ, RG, CPF e

carteira de habilitação de uma pessoa. Cada um deles contém um conjunto de informações que identificam a instituição ou pessoa e a autoridade (para estes exemplos, órgãos públicos) que garante sua validade.

Algumas das principais informações encontradas em um certificado digital são:

- dados que identificam o dono (nome, número de identificação, estado, etc);
- nome da Autoridade Certificadora (AC) que emitiu o certificado;
- o número de série e o período de validade do certificado;
- a assinatura digital da AC.

A Autoridade Certificadora (AC) pode ser emitida para diversos tipos de entidades, tais como: pessoa, computador ou departamento de uma instituição.

Além da questão referente à segurança contra *hackers*, também os vírus merecem destaque por poderem comprometer a segurança da rede.

4.5.1 SSL

Secure Sockets Layer é uma tecnologia de segurança que é comumente utilizada para codificar os dados trafegados entre o computador do usuário e um *Website*. O protocolo SSL, através de um processo de criptografia dos dados, previne que os dados trafegados possam ser capturados, ou mesmo alterados no seu curso entre o navegador (*browser*) do usuário e o site com o qual ele está se relacionando, garantindo desta forma, informações sigilosas como os dados de cartão de crédito.

Quando um visitante de um *website* se conecta a um servidor que está utilizando o protocolo SSL, eles irão notar na barra de endereços, que o protocolo passa a ser `https://` (no lugar do `http://` padrão). Aliado a isto, a maioria dos *browsers* (como o *internet explorer* por exemplo) mostram no browser um tradicional cadeado (como ilustrado na imagem abaixo). Quando este cadeado está sendo mostrado, o usuário passa a ter a tranquilidade de saber que as informações fornecidas aquele *Website* não poderão ser interceptadas no seu trajeto.

Para obter um certificado digital SSL é necessário fornecer informações sobre o detentor do *Website*, tais como endereço, documentação e pessoa de contato. Com estes dados, é gerado um par de chaves de criptografia, que irão garantir o processo de codificação dos dados. Estas chaves são duas: Uma chave privada, que deverá ficar somente no servidor e uma chave pública que será utilizada, em conjunto com as informações cadastrais, para gerar um CSR (*Certificate Signing Request*).

Este CSR é então submetido a uma autoridade certificadora (CA) que irá validar os dados cadastrais, através da comprovação da autenticidade dos documentos e da propriedade do *Website*, garantindo que aquele certificado foi realmente emitido para o proprietário do *Website*.

A autoridade certificadora, gera então o certificado definitivo, que deveria ser instalado pelo seu provedor responsável pela hospedagem de seu *Website*. Todo este processo é realizado pela *Laniway*, de forma rápida e prática para os clientes hospedados em nossos servidores, ou então o certificado é remetido para o cliente, para que ele remeta o mesmo para o administrador de seu provedor visando a instalação do mesmo.

4.6 ANTIVÍRUS

Os antivírus são programas que procuram detectar e remover vírus de computador. Novas funcionalidades têm sido adicionadas, de modo que alguns antivírus procuram detectar e remover cavalos de tróia e outros tipos de códigos maliciosos, barrar programas hostis e verificar *e-mails*.

Um bom antivírus deve:

- identificar e eliminar todos os vírus conhecidos além de outros tipos de *malware*,
- analisar em tempo real os arquivos que estão sendo obtidos pela *Internet*,
- Realizar verificação agendada de discos rígidos, ou *Hard Drives (Hds)*, e continua de disquetes e unidades removíveis, como CDs, DVDs e *pen drives*, de forma transparente ao usuário,

- verificação de *e-mails* e anexos,
- verificação *heurística*⁸ de arquivos,
- atualizar as assinaturas de vírus e *malwares* conhecidos, pela rede, de preferência diariamente.

Alguns antivírus, além das funcionalidades acima, permitem verificar *e-mails* enviados, podendo detectar e barrar a propagação por *e-mail* de vírus, *worms*, e outros tipos de *malware*, como dito anteriormente.

Apesar do que já foi dito, um antivírus não é capaz de impedir que um atacante tente explorar alguma vulnerabilidade existente em um computador. Também não é capaz de evitar o acesso não autorizado a um *backdoor* instalado em um computador.

4.7 REALIZAÇÃO DE CÓPIAS DE SEGURANÇA

Cópias de segurança (*backups*) dos dados armazenados em um computador são importantes, não só para se recuperar de eventuais falhas, mas das conseqüências de uma possível infecção por vírus, ou de uma invasão.

Podem ser dos mais diversos tipos como: armazenamento de arquivos em CDs, DVDs ou DATs; ou mais complexas como: espelhamento de um disco rígido ou de servidores.

Também existem equipamentos e *softwares*, mais sofisticados e específicos, que, dentre outras atividades, automatizam todo o processo de realização de cópias de segurança, praticamente sem intervenção do usuário. A utilização de tais equipamentos e *softwares* envolve custos mais elevados e depende de necessidades particulares de cada usuário.

A frequência com que é realizada uma cópia de segurança e a quantidade de dados armazenados neste processo depende da periodicidade com que o usuário cria ou modifica arquivos. Cada usuário deve criar sua própria política para a realização de cópias de segurança.

⁸ Heurísticas é uma pesquisa realizada por meio da quantificação de proximidade a um determinado objetivo. Em outras palavras, apresenta comportamento semelhante a um vírus.

4.7.1 Cuidados com as cópias de segurança

Os cuidados com cópias de segurança dependem das necessidades do sistema. Alguns dos cuidados serão discutidos abaixo:

- **Escolha dos dados.** Cópias de segurança devem conter apenas arquivos confiáveis. Arquivos do sistema operacional que façam parte da instalação dos *softwares* de um computador não devem fazer parte das cópias de segurança. Eles podem ter sido modificados ou substituídos por versões maliciosas, que quando restauradas podem trazer uma série de problemas de segurança para um computador. O sistema operacional e os *softwares* de um computador podem ser reinstalados de mídias idôneas, fornecidas por fabricantes confiáveis;
- **Mídia utilizada.** A escolha da mídia para a realização da cópia de segurança é extremamente importante e depende da importância e da vida útil que a cópia deve ter;
- **Local de armazenamento.** Cópias de segurança devem ser guardadas em um lugar condicionado (longe de muito frio ou muito calor) e restrito, de modo que apenas pessoas autorizadas tenham acesso a esse local (segurança física);
- **Cópia em outro local.** Cópias de segurança podem ser guardadas em locais diferentes. Um exemplo seria manter uma cópia em casa e outra no escritório. Também existem empresas especializadas em manter áreas de armazenamento com cópias de segurança de seus clientes. Nesses casos é muito importante considerar a segurança física de suas cópias, como discutido no item anterior;
- **Criptografia dos dados.** Os dados armazenados em uma cópia de segurança podem conter informações sigilosas. Nesse caso, os dados que contenham informações sigilosas devem ser armazenados em algum formato criptografado;
- **Descarte de cópias antigas.** Uma coisa já tratada anteriormente no tópico engenharia social é o fato de *hackers* mexerem no lixo das empresas atrás de senhas ou pistas sobre a rede. Uma fita de *backup* seria um achado.

4.8 SERVIDORES DE ALTA DISPONIBILIDADE

Servidores de alta disponibilidade visam eliminar as paradas não planejadas. Consiste em dois servidores interligados. No travamento de um, o outro entra em ação, trabalhando com redundância total de *hardware* com espelhamento de conteúdo. Com isso, se obtém dois *nodos* com discos exatamente iguais, até o instante da falha. As aplicações que trabalham com dados dinâmicos ou atualizados com muita frequência se beneficiam desse *driver*.

O processo no qual uma máquina assume os serviços de outra, quando se apresenta falha, é chamado de *failover*. Ao ser percebida a falha, além do *failover*, é obviamente necessário que se faça manutenção no servidor falho. Ao ser recuperado de uma falha, esse servidor será recolocado em serviço, e então se tem a opção de realizar o processo inverso do *failover*, que se chama *failback*.

4.9 EDUCAÇÃO DOS USUÁRIOS

Ferreira e Araújo (2006) salientam que educar os usuários é uma tarefa extremamente importante e que deve fazer parte do cotidiano dos administradores de redes. Sabe-se que grande parte dos problemas de segurança são originados na rede interna da organização e, muitas vezes, é causada pelo desconhecimento de conceitos e procedimentos básicos de segurança por parte dos usuários.

Um exemplo clássico desse problema é a má configuração do programa de leitura de *e-mails* de um usuário, que faz com que qualquer arquivo anexado a uma mensagem seja automaticamente aberto ou executado, permitindo a instalação de *backdoors*, cavalos de tróia, disseminação de vírus, etc.

O primeiro fator que contribui diretamente para o processo de educação é o estabelecimento de políticas de segurança e de uso aceitáveis, claras, sem ambigüidades, conhecidas e completamente entendidas pelos usuários da rede.

Outro fator importante é o estabelecimento de um canal de comunicação, por exemplo, através de listas de *e-mails*, em que informações sobre questões relevantes de segurança são freqüentemente passadas para os usuários da rede. A descoberta de uma vulnerabilidade de segurança que afeta o servidor *Web* da

organização pode não ser relevante para os usuários, mas a notificação da descoberta de um novo vírus, sua forma de infecção e métodos de prevenção são informações que devem ser conhecidas e aplicadas por todos os usuários.

5 SEGURANÇA APLICADA A APLICAÇÕES WEB

5.1 INTRODUÇÃO, CONCEITOS E DEFINIÇÕES

A partir do estouro da bolha da internet em 2001, uma mudança importante começou a ser percebida na arquitetura de aplicações web que passaram a ser desenvolvidas. Páginas web antes estáticas, foram substituídas por sites onde o usuário pode interferir na produção de conteúdo através da publicação de vídeos, inserção de informações pessoais, transferência de informações corporativas, dados bancários e imagens, ou participação em projetos colaborativos, correio eletrônico e redes sociais.

O ponto chave que guiou esta transformação foi a percepção por parte dos desenvolvedores da existência de uma característica em comum entre as empresas “ponto com” que sobreviveram à crise da internet: o fornecimento de aplicações com alto grau de interatividade e usabilidade.

Nesses ambientes, o Ajax, sigla para *Asynchronous JavaScript e XML*, surgiu como a técnica de programação que viabilizou a existência dos sites interativos.

“A arquitetura de *software* é resultado de influências sociais, técnicas e de negócio. Estas influências se dão através dos requisitos funcionais (aquilo que o sistema deve fazer) e também pelos requisitos de qualidade (muitas vezes chamados de requisitos não funcionais) solicitados pelos *stakeholders* no processo de desenvolvimento de um *software*. Como diferentes *stakeholders* possuem diferentes preocupações e objetivos é de se esperar que conflitos (*tradeoffs*) surjam deste processo.” (Campos, Signorini Neto e Canto, 2008)

Para acesso as aplicações web, os usuários necessitam passar por um processo de autenticação criterioso. Os dados utilizados nesse processo, por sua vez, devem ser protegidos de forma que não possam ser conhecidos por usuários ilegítimos. Por outro lado, os desenvolvedores dessa categoria de aplicações estão cada vez mais preocupados com a segurança dos sistemas e atuando fortemente para reforçá-la, principalmente dentro do ambiente da aplicação. Como consequência dessa atitude, nos últimos tempos, o foco de ataques voltou-se para o elo mais fraco, ou seja, o usuário final.

Senhas são facilmente roubadas de usuários inadvertidos, bastando para

isso, verificar a grande quantidade de fraudes ocorridas no sistema bancário nos últimos anos através da Internet com utilização de dados roubados.

Todo o processo de autenticação pode ser visualizado como sendo baseado em confiança, ou seja, uma vez autenticado, o usuário é considerado confiável para a aplicação. Em um cenário onde os dados de autenticação não são totalmente confiáveis, não se pode confiar cegamente no usuário, portanto, deve-se encontrar uma técnica complementar para comprovar a veracidade das informações de autenticação e a própria validade dessa autenticação continuamente.

5.2 IMPORTÂNCIA DA SEGURANÇA EM AMBIENTES WEB

Com o advento das transações e trocas de informações realizadas pela internet, a segurança tornou-se primordial em ambientes *web*. Em 9 de janeiro de 2000, a Diretoria da empresa *CD Universe*, varejista de CD's pela *Internet*, muito conhecida nos EUA, recebeu um *e-mail* de um certo Maxus, dizendo que havia roubado o cadastro de Cartões de Crédito da empresa, com mais de 300 mil nomes.

O cadastro incluía o nome do cliente, número do cartão, data de validade, endereço e outros dados confidenciais. Como prova desse roubo, divulgava em um site na Internet, 25 mil cartões utilizáveis.

Exigia certa quantia em dinheiro para não divulgar os outros quase 300 mil cartões. A empresa chamou o FBI, para constatar que o *hacker* se encontrava na Rússia, ao abrigo das leis americanas contra *hacker* e fraude e que nada poderia ser feito.

Foi um ataque nas Portas Públicas (Portas 80 e 443) e aparentemente o mecanismo usado foi o *SQL Injection*, que possibilitou ao *hacker* conseguir as informações diretamente do banco de dados. O crescimento do *cybercrime* nestes últimos anos deveu-se à crescente profissionalização dos criminosos. De simples e românticos *hackers*, para quadrilhas que procuram sistematicamente vulnerabilidades em sites de *ecommerce* e *Internet banking*.

Segundo a *SpiDynamics*, fabricante do software *WebInspect*, que simula um *hacker* ético e descobre as vulnerabilidades das aplicações *web*, existem mais de 4.500 formas de ataques que podem atingir os sites de *ecommerce* e de instituições financeiras.



FIGURA 3 – Ilustração da parte visível das vulnerabilidades.
Fonte: risco, 2010.

Porque segurança de aplicações deve ser prioridade?

- Aplicações web são o foco número 1 dos hackers:
 - 75% dos ataques acontecem na camada da aplicação
- A maior parte das páginas web estão vulneráveis:
 - 90% dos sites são vulneráveis à ataques na aplicação
 - 78% das vulnerabilidades facilmente exploráveis afetam aplicações Web
 - 80% das organizações irão experimentar um incidente em segurança de aplicações até 2010
- Aplicações web são alvos de alto valor para hackers:
 - Dados de clientes,
 - cartão de crédito,
 - roubo de identidade,
 - fraude, etc.
- No Brasil 13% das empresas que sofreram ataques tiveram prejuízos que ultrapassam R\$ 1 Milhão (Módulo)
- Estudo realizado com 200 empresas brasileiras afirma que 67,5% sofreram algum tipo de ataque nos últimos 12 meses (ISS) – Instituições financeiras nacionais têm sofrido até 100 ataques

frustrados por dia.

- Principais vulnerabilidades:
 - Imaturidade em segurança
 - Desenvolvimento *in-house*
 - Percepção de simplicidade
 - Restrições de recursos e tempo
 - Mau uso de tecnologias

A segurança das aplicações, principalmente aquelas conectadas a uma rede aberta e perigosa como é a *Internet* é bastante complexa. Essa complexidade advém do fato que as aplicações *web*, *ecommerce*, *Internet bank*, na realidade são agrupamentos bastante heterogêneos de plataformas, bancos de dados, servidores de aplicação, etc..

Uma aplicação típica, geralmente, está distribuída em vários servidores, rodando diversos aplicativos e para funcionar na velocidade adequada, a aplicação precisa que as *interfaces* entre os diversos sistemas sejam construídas com a premissa que os dados passados através da mesma são confiáveis e não hostis. Não há tempo hábil para duplas verificações.

O “calcanhar de Aquiles” destas aplicações é a necessidade de haver “confiança” entre os diversos subsistemas e é disso que os *hackers* e outros cibercriminosos se aproveitam.

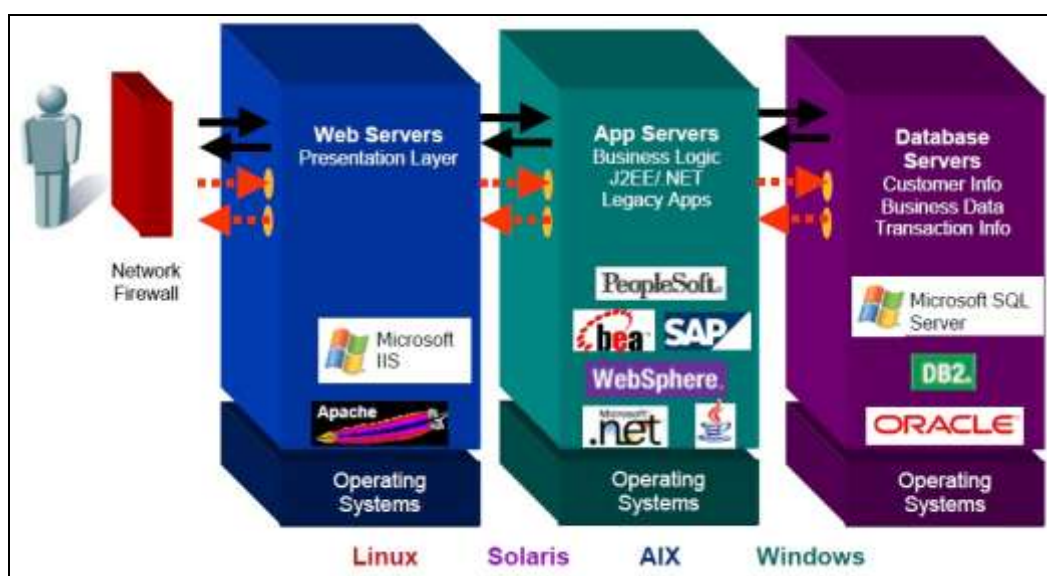


FIGURA 4 – Estrutura de um WEB Service.
Fonte: risco, 2010.

Nestes sistemas complexos, a segurança dos produtos disponíveis no mercado é assegurada pelos fabricantes, que fornecem periodicamente *patches* que os atualizam.

Para o sistema aplicativo, freqüentemente desenvolvido *in house* ou por terceiros, especificamente para a empresa, não existem *patches* de segurança.

Segundo a revista americana *Software Magazine*, uma aplicação *web* típica tem de 150 mil a 250 mil linhas de código. São grandes e complexas. Lidam com vários subsistemas, bancos de dados e sistemas operacionais.

Na sua especificação foi pedido rapidez de processamento (os visitantes não gostam de esperar), beleza e usabilidade (a idéia é transformar visitantes em clientes), prazo de entrega e custo baixo.

Normalmente segurança não faz parte dos requisitos do sistema, a não ser vagamente, "...assegurar que os visitantes e clientes se identifiquem com *login* e *password*...", "...as *passwords* serão armazenadas criptografadas no sistema..." ou se faz, não se prevê nenhum mecanismo periódico de atualização dos sistemas em função de novas ameaças.

Normalmente segurança é subentendida como um problema do pessoal da segurança, da infra-estrutura e não do pessoal de desenvolvimento ou produto, a não ser em termos de generalidades.

Segundo estudos do Ministério da Defesa Americano (DoD ou Pentágono), "cada 1.000 linhas de código embutem 15 defeitos de segurança !"

Numa aplicação típica de 200 mil linhas, existirão, portanto, 3.000 defeitos de segurança, que precisarão ser identificados e corrigidos.

O processo de correção fica parecido com "enxugar gelo", mal os erros são detectados e corrigidos, uma nova versão da aplicação é liberada por pressão dos usuários, normalmente as áreas de *marketing* e comercial, que conseguem toda a atenção administração da empresa.

5.2.1 Ataques

Os ataques que conhecemos são baseados em vulnerabilidades típicas de aplicações *web* complexas. Mesmo os sistemas operacionais (Windows, AIX, Solaris) que são mantidos por grandes empresas, empregando milhares de

profissionais, têm vulnerabilidades que são periodicamente descobertas por *hackers* e só se transformam em patches depois que os *hackers* já atacaram algumas vezes, que o problema foi comunicado ao fabricante e devidamente corrigido.

A *Internet* agregou outros componentes de risco, sendo muito importante o “efeito comunidade” em que os *hackers* e outros criminosos se julgam fazendo parte de uma “comunidade” e obrigados a compartilhar rapidamente suas descobertas. Isto significa que qualquer vulnerabilidade descoberta nas suas aplicações será rapidamente divulgada, com as ferramentas necessárias para atacá-la, e outros *hackers* e cibercriminosos aproveitarão as vulnerabilidades da sua aplicação.

Os ataques podem causar uma série de problemas, entre os quais se pode citar:

- Perdas Financeiras;
- Transações Fraudulentas;
- Acesso não autorizado a dados, inclusive confidenciais;
- Roubo ou modificação de Dados;
- Roubo de Informações de Clientes;
- Interrupção do Serviço;
- Perda da confiança e lealdade dos clientes;
- Dano à imagem da marca.

Os tipos mais comuns de ataques são:

1. Cross-Site Scripting: é uma vulnerabilidade muito comum encontrada em aplicativos web. CSS permite ao atacante inserir códigos maliciosos nessas páginas, para que sejam executados no momento em que tais páginas forem acessadas.

O ataque permite que conteúdos (*scripts*) em uma zona sem privilégio seja executado com permissão de uma zona privilegiada - i.e. escalação de privilégios no cliente (*web browser*) executando o *script*.

A vulnerabilidade poderia ser:

- Um *bug* do *browser* que sob determinadas condições permite conteúdos (*scripts*) de determinado nível ser executado com permissões de níveis mais altos.
- Um erro na configuração do *browser*; sites não-seguros listados em zonas privilegiadas.
- Vulnerabilidade de *cross-site scripting* em uma zona privilegiada.

Um cenário comum de ataque envolve dois passos:

O primeiro passo é utilizar uma vulnerabilidade *Cross-Site Scripting* para executar *scripts* em zonas privilegiadas. Para completar o ataque, realizar ações maliciosas no computador utilizando controles *ActiveX* não-seguros.

Esse tipo de vulnerabilidade tem sido explorada para instalar silenciosamente vários *malware* (tais como *spyware*, *softwares* de controle remoto, *worms* e coisas semelhantes) em computadores enquanto navegam em páginas *web* maliciosas.

O ataque mais comum utilizado contra vulnerabilidades CSS é a execução de códigos *javascript* para permitir o seqüestro de sessão (roubo de *cookie*). Utilizando *javascript* também seria possível fazer coisas com contas de usuários tal como alterar detalhes de seu perfil.

2. SQL Injection: Trata-se da manipulação de uma instrução SQL através das variáveis quem compõem os parâmetros recebidos por um *script server-side*, tal como PHP, ASP, *ColdFusion* e outros.

O principal motivo pelo qual deve-se impossibilitar a utilização da *SQL Injection* está no fato de que, através de uma simples instrução SQL, como por exemplo, uma projeção de dados, outras operações podem ser executadas, podendo impactar sobre o esquema das tabelas, os dados armazenados, e até mesmo sobre elementos do sistema operacional, tendo em vista que alguns bancos de dados permitem a execução de comandos do *shell* do próprio sistema operacional.

Deve-se tomar cuidado com a configuração do usuário que estabelece a conexão com o banco de dados. O ideal é que as permissões de acesso deste usuário estejam restritamente limitadas às funções que irá realizar, ou seja, para a exibição de um relatório, a conexão com o banco de dados deve ser realizada por um usuário com permissões de leitura e acesso somente às tabelas necessárias para sua operação.

3. Command Injection: Esta classe de ataques, muito comum em ambiente web, consiste na inserção de comandos nos dados passados como entrada para a aplicação de forma que estes comandos sejam executados pela ou através da aplicação. As subclasses mais comuns são *SQL injection* e *Shell Command Injection*.

Injeção de SQL, ou *SQL Injection*, consiste em inserir comandos SQL nos dados enviados para a aplicação de modo a acessar o banco de dados subjacente.

Um ataque de uma sentença SQL retornará a lista de todos os usuários da aplicação, que poderá ser usada posteriormente como subsídio para outros ataques. O adversário poderia também inserir comandos visando alterar ou apagar dados do banco, causando estragos ainda maiores. A injeção de comandos *shell* acontece de forma similar se a aplicação acessa o sistema operacional através de um interpretador de comandos.

A maneira mais simples de proteger uma aplicação contra injeção de comandos é evitar o uso de interpretadores externos, como um interpretador SQL ou um *shell*. Para evitar o uso de comandos de *shell*, devemos usar as bibliotecas do sistema que realizem as funções específicas necessárias.

Nos casos em que não há alternativa, tais como as chamadas a interpretadores SQL de bancos de dados, devemos validar criteriosamente os dados recebidos e certificar que estes dados não incluam nenhum comando ou conteúdo potencialmente nocivo. Devemos também estruturar as requisições de maneira que os parâmetros recebidos sejam efetivamente tratados como dados e não como comandos.

No caso de Java, pode-se empregar também *stored procedures* e *prepared statements* (representados em Java pelas classes *Callable Statement*, quando são usadas *stored procedures*, e *PreparedStatement*, quando se usa SQL) para aumentar o nível de proteção à informação.

Nos casos em que é usado o *framework Hibernate*, as manipulações de dados mais simples ou corriqueiras são tratadas diretamente pelo *framework*. No entanto, o *Hibernate* disponibiliza a linguagem HQL (ou *Hibernate Query Language*) para que os programadores possam definir consultas ou manipulações complexas dos dados da aplicação. Embora o HQL apresente algumas restrições no conteúdo dos comandos que diminuam o risco de ataques de injeção de comandos, os riscos não são totalmente eliminados. Assim, são necessários os mesmos cuidados que no caso do uso direto de SQL para acesso à base de dados.

4. Cookie/Session Poisoning: Na Web, é a modificação de um *cookie* (informações pessoais no computador de um usuário) por um invasor para obter informações sobre o usuário não autorizado para fins tais como roubo de identidade. O atacante pode usar as informações para abrir novas contas ou para obter acesso a contas existentes do usuário.

Cookies armazenados no disco rígido do seu computador manter *bits* de informação que permitem visitar o nosso site para autenticar a sua identidade,

acelerar suas transações, monitorar o seu comportamento, e personalizar suas apresentações para você. No entanto, os cookies também podem ser acessados por pessoas não autorizadas a fazê-lo. A menos que medidas de segurança estão no lugar, um atacante pode examinar um cookie para determinar a sua finalidade e editá-lo para que ele ajuda-los a obter informações do usuário a partir do site que enviou o cookie.

Para protegerem-se contra o *Session Poisoning*, os sites que usá-los deve proteger *cookies* (através de criptografia, por exemplo) antes de serem enviados ao computador de um usuário. Plataforma *Ingrian Networks* 'Segurança Ativa do aplicativo é um meio de *cookies* de segurança. Quando os *cookies* passar pela plataforma, informações sigilosas são criptografadas. A assinatura digital é criada, que é usado para validar o conteúdo de todas as futuras comunicações entre o remetente e o destinatário. Se o conteúdo for violado, a assinatura já não corresponde ao conteúdo e será recusado o acesso pelo servidor.

5. *Parameter/Form Tampering*: Esta categoria envolve a manipulação de parâmetros de URL, de forma a obter informação que normalmente não se encontra disponível para o usuário. Acesso a bases de dados dos servidores de *back-end* através de chamadas SQL (*Structured Query Language*) normalmente incluídas na URL. Usuários maliciosos podem manipular o código SQL para eventualmente obter a lista de todos os usuários, senhas, números de cartões de crédito, ou qualquer outra informação guardada na base de dados. Também é conhecido como *URL Manipulation*, pela facilidade de manipular informações na URL.

6. *Buffer Overflow*: é uma técnica de ataque clássica, onde um usuário malicioso envia mais informação do que aquela que a aplicação pode e espera receber e, conseqüentemente, causar um *buffer overflow*. Este ataque pode estar relacionado com a aplicação ou com algum dos componentes que ela utiliza, como o servidor *Web*, o servidor da aplicação, etc. É utilizado para parar um ou mais serviços, o servidor ou ainda ganhar controle absoluto sobre este através da execução de código malicioso. Pode-se ainda, derrubar um servidor mal configurado, fazendo com que ele armazene *LOGs* até esgotar o espaço em disco. Imagine um arquivo de *LOG* de 80 GB que ocupa todo o espaço do seu disco rígido. Atualmente a vulnerabilidade de *buffer overflow* é encontrada com menos frequência, pois foram criados vários mecanismos de proteção contra ela.

7. *Directory Traversal/Forceful Browsing*: Quando as aplicações não

limitam a navegação que o cliente pode efetuar, um *hacker* pode explorar esta fraqueza e “saltar” diretamente para páginas que, em condições normais, só poderiam ser acessadas através de mecanismos de autenticação. As aplicações *Web* que não estão corretamente configuradas permitem que usuários maliciosos acessem diretamente a URLs que podem conter informação sigilosa.

8. Connection hijacking: Conhecido “Roubo de Conexões”, quando um invasor intercepta um pacote no contexto de uma conexão e passa a participar da comunicação.

9. Session Hijacking: O Termo “*Session Hijacking*” refere-se à exploração de uma sessão válida de um computador – as vezes também chamada de *Session Key* ou ID – para conseguir acesso não-autorizado a informações ou serviços em um sistema de computador.

Em particular, é usado para referir-se ao roubo do *Magic Cookie*, utilizado para autenticar um usuário em um servidor remoto. Isto é particularmente relevante para os desenvolvedores *Web*, já que os HTTP *Cookies* usados para manter uma sessão em diversos sites *Web* podem facilmente ser roubados por um atacante utilizando um computador que esteja no meio da comunicação ou então acessando os *cookies* salvos no computador da vítima.

Muitos *web* sites permitem que usuários criem e gerenciem suas próprias *accounts*, logando utilizando um *username* e *password* (que pode ou não ser criptografada durante o transito) ou outro método de autenticação. Para que o usuário não tenha que re-digitar seu usuário/senha em todas as página para manter sua sessão, muitos *web* sites utilizam-se de *cookies*: um *token* de informações solicitadas pelo *server* e retornada pelo *user browser* para confirmar sua identidade.

Se um atacante estiver habilitado a roubar este *cookie*, ele pode fazer requisições como se fosse o usuário legítimo, ganhando acesso à informações privilegiadas ou até modificando dados. Se o *cookie* for um *Cookie Persistente*, o roubo de identidade pode se estender por um tempo muito longo.

É claro que *Session Hijacking* não é limitada para a *Web*, qualquer protocolo cujo o estado é mantido através de uma chave que é checada através de dois computadores é vulnerável, especialmente se não for criptografada.

10. Ataques de dicionários: Trata-se de um ataque baseado em senhas que consiste na cifragem das palavras de um dicionário e posterior comparação com os arquivos de senhas de usuários. De posse do arquivo de senhas */etc/passwd*, o *cracker* (veremos no próximo tópico detalhes sobre este tipo de profissional) utiliza

um arquivo que contém diversas palavras que serão tentadas como senha. Este trabalho é feito automaticamente por ferramentas dedicadas a este tipo de tarefa e pode levar dias dependendo da lista de senhas do *cracker* e quantidades de usuários existentes no arquivo de senha.

Note que o uso de criptografia md5 e senhas ocultas dificultam bastante ao arquivo de senhas eo sucesso de um ataque bem sucedido.

Você pode configurar seu sistema para garantir que as senhas não sejam adivinhadas facilmente. Isto pode ser feito usando um módulo PAM chamado `pam_cracklib`.

Este módulo irá comparar a senha com palavras de um dicionário. O usuário não será autorizado a definir uma nova senha a menos que certas condições sejam satisfeitas, ou seja, senhas facilmente adivinháveis não serão permitidas.

5.2.2 *Hackers ou crackers*

Qual a primeira lembrança ao se ouvir a palavra “*hacker*”? A de um criminoso que invade *sites* e sistemas para roubar informações? Provavelmente. O termo *hacker* é associado a “crimes”, devido ao desconhecimento do termo “*cracker*”.

Primeiramente, “*hacker*” ou “*cracker*” são indivíduos que elaboram e modificam *software* e *hardware* de computadores, seja desenvolvendo novas funcionalidades ou adaptando antigas.

Hacker é um indivíduo que se utiliza seu conhecimento para testar os recursos de segurança instalados em uma empresa. Por sua vez, os “*crackers*” são indivíduos que utilizam seu conhecimento para invadir e sabotar sistemas, quase sempre objetivando a captação de dados que possam ser usadas como moeda de troca. As intenções dos *crackers* podem ser de um simples roubo eletrônico, estelionato ou até o roubo de informações confidenciais. Geralmente essas informações são vendidas ou utilizadas para aplicar golpes na *Internet*.

Veremos um breve histórico de alguns *Crackers* famosos:

a) Robert Tappan Morris

- criou em 1988 o primeiro código malicioso disseminado pela internet o “Morris worm”;
- Estima-se que a praga infectou 10% dos 60 mil computadores que

formavam a rede mundial da época.

- O vírus não tinha nenhuma carga maliciosa, só que um problema em sua programação sobrecarregava o sistema infectado impedindo sua operação.
- O “criador do primeiro vírus” atualmente é professor do Instituto Tecnológico de Massachusetts (MIT) – mesma instituição que ele iniciou a propagação do seu worm.
- Atualmente, o código fonte do worm está armazenado em um disquete exibido em destaque no Museu de Ciência de Boston.



FIGURA 5 – Robert Tappan Morris.
Fonte: krishnakummar, 2008

b) Kevin David Mitnick

- É considerado o pai dos hackers, também era engenheiro social (habilidade de se manipular pessoas para obter informações para acessar um sistema, roubar dados de bancos ou qualquer outra coisa).
- Sua carreira começou durante os anos 70, quando aos 16 anos, conseguiu invadir o sistema da sua escola e alterou algumas notas.
- Sua notoriedade foi alcançada em virtude de uma nova profissão: hacker em tempo integral.
- Invadiu vários computadores, obtendo ilegalmente códigos de acesso à

telefonica de longa distância, criando números telefônicos isentos de contas,

- Apropriou-se de mais de 20 mil números de cartões de créditos, enganou o FBI por 2 anos portando apenas um notebook e um telefone celular, sendo um dos cibercriminosos mais procurados da história.
- Foi preso em 1995 e libertado em 2000 após pagar fiança.
- Ficou três anos em liberdade condicional, com a condição de se manter longe de computadores e sem poder conectar-se à internet durante 8 anos.
- Atualmente Mitnick é dono de uma empresa de segurança a Mitnick Security Consulting (<http://mitnicksecurity.com/index.php>), faz palestra em diversos países, trabalha como consultor em segurança de sistemas. É autor dos livros “A arte de invadir” e “A arte de enganar”.
- Seu ataque foi muito sofisticado e sem “solução” até os dias atuais



FIGURA 6 – Kevin David Mitnick.
Fonte: blogdodito, 2010.

c) Kevin Lee Poulsen - “Watchman”

- Seu “trabalho” mais reconhecido foi controlar as linhas telefônicas de uma rádio da cidade de Los Angeles e, assim, garantir que seria a 102ª pessoa a ligar e ganhar um Porsche.
- Na época, a rede de TV NBC mostrou em um programa sobre crimes não-solucionados, sofrendo um pane nas mesas telefônicas abertas

para telefonemas a respeito de pistas. Quem estava por trás da falha? Poulsen.

- Em 1991, o hacker finalmente foi preso e condenado por fraudes em computadores, lavagem de dinheiro e obstrução de justiça.
- Libertado, reinventou-se como jornalista, criando um site, Security Focus News, respeitado na cobertura de tecnologia. Desde 2005 é editor da Wired.
- Em outubro de 2006, Poulsen montou uma operação contra pedófilos no MySpace, usando páginas para rastrear quem estava em busca de crianças, levando à prisão um deles, Andrew Lubrano.



FIGURA 7 – Kevin Lee Poulsen.
Fonte: dymzworld, 2010.

d) Gary McKinnon

- conhecido como "Solo", é considerado o maior hacker de todos os tempos.
- Foi acusado pelos Estados Unidos de perpetrar o "maior ataque de todos os tempos a um computador militar". Ele é acusado de ter acessado ilegalmente e danificado 53 computadores do exército

americano, da aeronáutica, do Pentágono e da Agência Espacial Americana (Nasa) entre fevereiro de 2001 e março de 2002.

- O motivo do ataque era comprovar a existência dos OVNI's e demonstrar as falhas de segurança do sistema americano.
- Um detalhe interessante é que para fazer isso ele utilizou internet discada de 56k.
- McKinnon é portador da Síndrome de Asperger (autista), o que pode ser um dos motivos da habilidade.

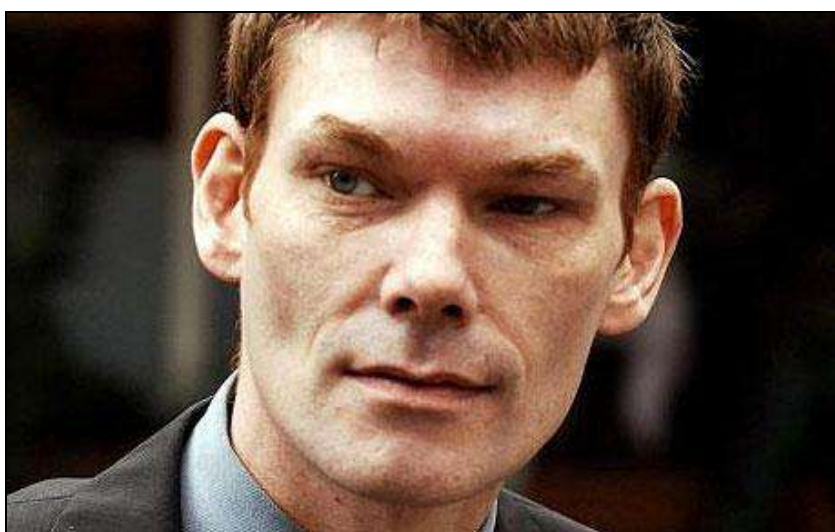


FIGURA 8 – Gary McKinnon.
Fonte: telegraph, 2010.

Veremos a seguir um detalhamento sobre recursos e tecnologias usados em ambiente *web* assim como uma introdução geral aos conceitos de *Web Services*.

5.3 RECURSOS E TECNOLOGIAS

5.3.1 Introdução

No começo dos anos 60, o modelo de computação vigente era apenas um modelo centralizado, com um grande computador central (*mainframe*) que possuía um grande poder de processamento e tinha vários computadores de pouco poder de

processamento, conectados a ele. Não havia preocupação com a padronização de protocolos e *interfaces*, cada fabricante tinha seus próprios produtos e soluções que, na maioria das vezes, não eram compatíveis com os produtos de outros fabricantes.

Com o passar dos anos, a necessidade de padronização foi ficando cada vez mais clara. A diversidade de soluções dificultava a interconexão de redes de computadores. Neste contexto, em meados dos anos 70, começam os trabalhos de padronização da ISO, que propôs uma arquitetura de conexão dividida em sete camadas conhecida como Modelo OSI.

O modelo de computação centralizada foi perdendo força com o surgimento e a proliferação da *Internet*. Hoje em dia, existe um modelo de computação distribuída vigente na *Internet*, com a arquitetura cliente-servidor desempenhando papel fundamental em um ambiente de rede descentralizado, com máquinas autônomas e independentes. Este ambiente descentralizado utiliza o protocolo TCP/IP como principal meio de comunicação entre máquinas de arquiteturas diferentes executando sistemas operacionais e aplicações diferentes.

Com a evolução da computação apoiada em aplicações distribuídas, surgiram novos padrões para o desenvolvimento de aplicações orientadas ao uso em rede, como, por exemplo, CORBA (*Common Object Request Broker Architecture*)[CORBA98]. Este padrão proposto pela OMG (*Object Management Group*) especifica uma arquitetura para o desenvolvimento de aplicações distribuídas. A *Microsoft* tem sua proposta de objetos distribuídos que é o DCOM. Porém, todas estas propostas levaram ao desenvolvimento de sistemas fortemente acoplados, onde é necessário um pré-acordo entre as organizações para determinar o tipo de dados e a *interface* dos métodos. Como resultado, tem-se agora um cenário onde os sistemas das organizações consistem em uma mistura de soluções particulares, executando sob uma variedade de plataformas. Integrar sistemas existentes com novo *software* é uma tarefa complicada, além dos rígidos formatos de dados e *interfaces* não serem facilmente adaptados aos novos requisitos. Desenvolvedores são desafiados com *software* escrito em diferentes linguagens, para diferentes sistemas operacionais, diferentes redes, e *hardwares*.

Sendo assim, fez-se necessário a criação de padrões para a especificação de um meio de compartilhar informações em um ambiente de aplicações heterogêneas como a *Internet*. A criação de uma ferramenta, a linguagem XML, tornou este compartilhamento de informações possível.

Recentemente, a indústria assiste a uma evolução de novos padrões para

melhorar a integração entre organizações. A utilização de protocolos de comunicação, TCP/IP e HTTP, fazem-se necessários para uma integração da rede pública e mundial. Por sua vez, a tecnologia Java permite a criação de aplicações independentes de plataforma. XML complementando Java criam dados independentes de plataforma específica.

Porém, as organizações não podem simplesmente abandonar seus investimentos em tecnologia e nos seus sistemas. A necessidade de integrar *software* com novas soluções tem levado à criação de novos produtos e padrões, permitindo às aplicações existentes trabalharem de uma forma fracamente acoplada. Esta integração flexível entre sistemas pode ser alcançada somente com a criação de um padrão para a troca de dados independente de plataforma. Uma forma de se alcançar esta integração é através da utilização de *Web Services*, que permite de integrar sistemas já existentes com novos sistemas.

5.3.2 A linguagem XML

XML (*EXtensible Markup Language*) ou linguagem extensível de marcação, é uma linguagem designada para descrever e estruturar informações. Como uma linguagem de marcação, XML se assemelha com a linguagem HTML, possuindo marcações para descrever os dados. Porém, estas marcações não são pré-definidas na linguagem, tornando possível a criação de marcações de acordo com necessidades específicas.

XML é uma recomendação da W3C (*World Wide Web Consortium*) e o seu desenvolvimento e especificação tem sido supervisionado pelo *XML Working Group*. XML é uma especificação pública, ou seja, não é propriedade de nenhuma companhia. O desenvolvimento de XML começou em 1996 e é um padrão W3C desde fevereiro de 1998.

5.3.2.1 A sintaxe XML

Um arquivo XML é composto por marcações aninhadas e delimitadas. A

primeira linha descreve a versão do documento. A próxima linha descreve o elemento raiz. As próximas linhas definem os elementos derivados do elemento raiz. A listagem a seguir mostra um exemplo de arquivo XML:

```
<?xml version="1.0"?>
<computador marca= "DELL">
  <preço> 3000,00 </preço>
  <hd un="GB"> 20 </hd>
  <memória> 128 </memória>
  <monitor marca= "LG"> 15
</monitor>
</computador>
```

Um elemento XML localiza-se entre de uma marcação de início e uma marcação de fim, incluindo a própria marcação. Serve para delimitar conteúdo e todo documento XML deve possuir pelo menos um elemento raiz numa relação de herança, formando assim forma uma estrutura de árvore.

Documentos XML geralmente possuem correspondência com uma tabela em uma base de dados. Pode-se utilizar, por exemplo, os nomes dos campos da tabela como nomes de elementos. Elementos XML podem conter atributos que armazenam informações adicionais sobre esses elementos.

XML *Namespaces* são atributos especiais de elementos que servem para evitar conflitos de nomes de elementos, desde que estes nomes não sejam pré-definidos (palavras reservadas).

Quando um *namespace* é definido na marcação de início de um elemento, todos os elementos filhos com o mesmo prefixo estarão associados ao mesmo *namespace*. Os endereços utilizados para identificar um *namespace* não são utilizados pelo *parser* para processar informações. A principal finalidade destes endereços é fornecer um único nome ao *namespace*. Geralmente, um *namespace* é utilizado como um ponteiro para uma página *Web* a qual referencia informações a seu respeito.

5.3.2.2 XML e validação de documentos

Um documento XML “bem formado” é um documento que está em conformidade com as regras sintáticas. Um documento XML “válido” é um documento que está em conformidade com as regras de um DTD (*Document Type Definition* ou Definição do Tipo do Documento). Um DTD define os elementos permitidos em um documento XML. O propósito de um DTD é definir a estrutura do documento com uma lista de elementos possíveis. Com o uso de uma definição de documento, cada arquivo XML pode carregar uma descrição do seu próprio formato. Deste modo, grupos independentes podem concordar em usar um DTD comum para a troca de informação. Uma aplicação pode usar um DTD para verificar se os dados que recebeu são válidos.

O ponto principal de um documento DTD é a declaração de elementos e da estrutura do documento XML. É possível declarar um elemento ou um conjunto de elementos como uma expressão regular, permitindo a definição da ocorrência de um determinado elemento.

5.3.3 SOAP

Segundo Wathier (2005), o protocolo SOAP foi criado para transportar mensagens XML de um computador para outro, via vários protocolos padrões de transporte. HTTP é o mais comum destes protocolos, obviamente porque predomina no uso da Web.

SOAP se define usando XML, que proporciona com simplicidade e coerência uma maneira de uma aplicação enviar mensagem XML para outra. SOAP é o que faz a integração entre aplicações ser possível, pois após a definição do conteúdo do XML, é o SOAP que transfere os dados de um lugar para outro pela rede. Permite enviar e receber documentos XML que suportam um protocolo comum de transferência de dados. Além disso, SOAP permite tratar mensagens XML retornadas de um serviço remoto e seu modelo possibilita de forma clara a separação entre os dados de processamento de infra-estrutura e processamento de mensagens de aplicação. (WATHIER, 2005)

Segundo Freire (2002), o SOAP é um protocolo que permite o uso de chamadas escritas em linguagens distintas, definindo chamadas de procedimentos e respostas na forma de elementos XML. O SOAP baseia-se na idéia de um mecanismo RPC (*Remote Procedure Call*) em XML. Desenvolvido com a colaboração da *Microsoft* e da *DevelopMentor*, sua implementação começou a ser difundida no fim dos anos 90.

A figura abaixo apresenta a estrutura básica de uma mensagem SOAP de acordo com (ROSENBERG, 2004).

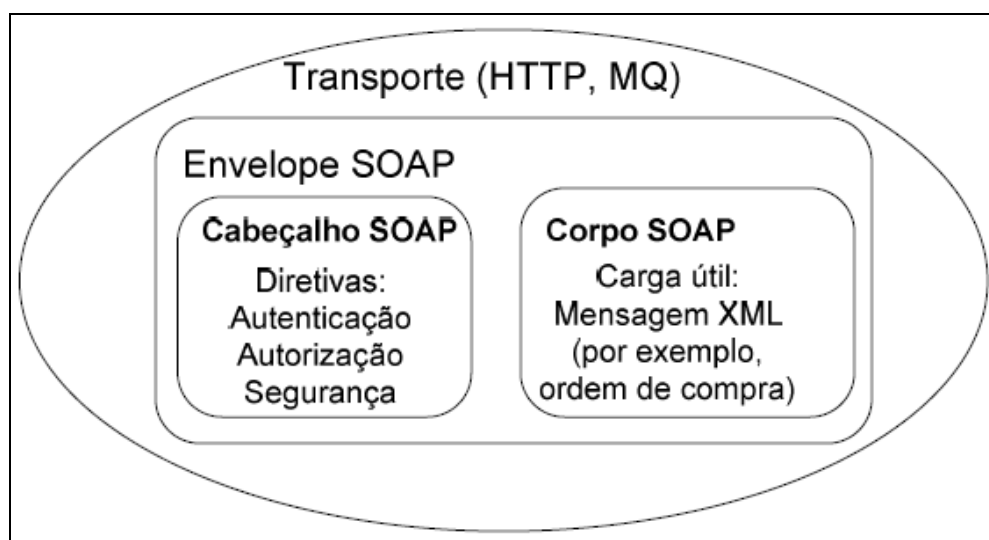


FIGURA 9 – Estrutura básica da mensagem SOAP.
Fonte: ufrs, 2005.

Segundo Wathier (2005), SOAP fornece um envelope (Envelope SOAP) para a mensagem XML, este envelope é um *container*⁹ que protege os dados XML. O objetivo é criar um *container* uniforme para que mensagens SOAP possam ser transmitidas por qualquer protocolo de transporte (HTTP, MQ). O protocolo evita que aplicação conheça o protocolo de transporte e se mantém coerente com o envelope SOAP.

O envelope SOAP é composto por duas partes: o cabeçalho (Cabeçalho SOAP) e o corpo da mensagem (Corpo SOAP).

Cabeçalho SOAP (*header*) – contém informações sobre a mensagem SOAP. Estas informações são usadas para gerenciar ou prover segurança para o pacote.

Corpo SOAP (*body*) – contém a mensagem SOAP propriamente dita (Carga útil).

⁹ Área delimitada e protegida destinada para armazenar algo.

Esta informação é enviada de uma aplicação para outra. Poderia ser um documento com uma ordem de compra ou um contrato, poderia ser uma descrição de uma classe com métodos remotos e seus parâmetros.

Um exemplo de mensagem SOAP é listado na figura abaixo (ROSENBERG, 2004):

```
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2001/12/soap-
  envelope">
  <env:Header>
    <n:alertcontrol xmlns:n="http://example.org/alertcontrol">
      <n:priority>1</n:priority>
      <n:expires>2004-06-22T14:00:00-5:00</n:expires>
    </n:alertcontrol>
  </env:Header>
  <env:Body>
    <m:alert xmlns:m="http://example.org/alert">
      <m:msg>Pick up Bobby at school at 2PM</m:msg>
    </m:alert>
  </env:Body>
</env:Envelope>
```

FIGURA 10 – Mensagem SOAP – envelope, cabeçalho e corpo.
Fonte: ufrs, 2005.

A mensagem transportada deve ser de conhecimento somente dos seus receptores. O serviço remoto deve conhecer quem está requisitando seu serviço e se está autorizado. SOAP é um mecanismo de pacote para mensagens XML e documentos. Muitos pacotes necessitam descrever importantes informações sobre o que o pacote todo contém, por exemplo: o remetente, como o receptor vai validar o remetente, quais permissões o remetente possui e assim por diante (ROSENBERG, 2004).

5.3.4 WSDL

WSDL (*Web Services Description Language*) é uma linguagem XML, extensível, que define as operações que um serviço *Web* provê e a estrutura dessas operações em relação às mensagens SOAP. Isto é, define as estruturas de entrada e saída de um serviço *Web*, associação entre os parâmetros e tipos de dados.

Segundo Camargo, Mello, Fraga e Wingham (2006), um documento WSDL é independente de linguagem e de plataforma e tem por objetivo: descrever quais são os serviços oferecidos; mostrar como os clientes e provedores irão processar as requisições; e, indicar em qual formato o serviço deve enviar as informações para um cliente.

WSDL contém informações de um serviço para que outros possam interagir com este serviço, como a localização do serviço, o que o serviço pode fazer e como poderá ser chamado (WATHIER, 2005).

5.3.5 UDDI

Na construção de serviços Web, esses serviços necessitam ser acessados em algum lugar na Web por uma aplicação cliente. Uma forma de acessar um serviço Web é fazer com que a aplicação cliente conheça a URI (*Uniform Resource Identifier*) do serviço, desta maneira caracteriza o modo estático de localizar e acessar um serviço. Entretanto, quando a aplicação cliente não detém a localização de um serviço Web, este pode ser descoberto antes de ser acessado, caracterizando o modo dinâmico de descobrir a localização de um serviço.

O UDDI provê um método padronizado para a publicação e descoberta de informações sobre serviços Web. É uma iniciativa de criar um *framework* padrão para descrição de serviços, descoberta de negócios e integração de serviços comerciais focando na pesquisa e na arquitetura orientada a serviços (SOA - *Service Oriented Architecture*).

5.3.6 SOA

Segundo Papazoglou (2003), a Arquitetura Orientada a Serviços (AOS) (*Service Oriented Architecture* – SOA) é uma caracterização de sistemas distribuídos, em que as

funcionalidades do sistema são expostas via descrição de uma *interface*, permitindo a publicação, localização e a invocação por meio de um formato padronizado.

Segundo Camargo, Mello, Fraga e Wingham (2006), a AOS é constituída de relações entre três tipos de participantes: o diretório para registro de serviços, repositório que é utilizado para publicar e localizar as *interfaces* dos serviços; o provedor de serviços, entidade responsável por publicar as *interfaces* dos serviços providos por esta no registro de serviços e também responsável por atender as requisições originadas pelos clientes; e o cliente, aplicação ou outro serviço que efetua requisições a um serviço. Cada participante da arquitetura pode ainda assumir um ou mais papéis, podendo ser, por exemplo, um provedor e um cliente de serviços.

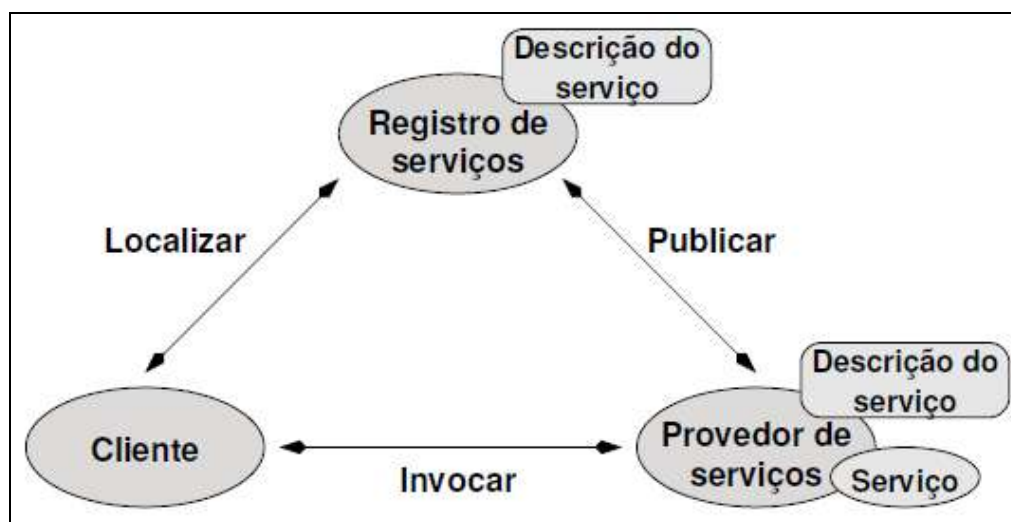


FIGURA 11 – Interação entre entidades da AOS
Fonte: SBSeg 2006.

Os participantes se relacionam através de três operações: publicar, localizar e invocar, como pode ser visto na Figura 12. Inicialmente, o provedor de serviço publica a *interface* do seu serviço junto ao diretório para registro de serviços. Desta forma, em algum momento posterior, o cliente pode efetuar uma busca por um determinado serviço (operação localizar), especificando as características desejadas, no diretório de registros.

Se o serviço existir, a *interface* e a localização do respectivo serviço são retornadas para o cliente. Por fim, o cliente efetua uma invocação ao provedor do serviço (operação invocar).

Os serviços estão baseados nas trocas de mensagens entre os provedores e os clientes, sendo assim, as mensagens seguem um formato padrão, garantindo aos serviços a neutralidade da tecnologia e permitindo que provedores e clientes utilizem diferentes implementações nas camadas inferiores. Os serviços também são definidos como fracamente acoplados, isso indica que possuem uma localização transparente e que não necessitam conhecer as estruturas internas presentes no lado do provedor e do cliente.

As *interfaces* dos serviços são auto-descritivas e baseadas em padrões abertos. Ou seja, a *interface* de um serviço define um conjunto de métodos públicos, juntamente com seus parâmetros, valores de retorno e meios para tratar possíveis exceções, porém não provê uma implementação. Com isto, pode-se assumir que a *interface* é um contrato entre o provedor do serviço e o cliente, sendo que o provedor deverá implementar todos os métodos ali descritos e o cliente só poderá invocar tais métodos.

Por estarem relacionados diretamente às funções de negócios, os serviços representam uma forma de modularidade diferente daquelas existentes nas linguagens de programação como os módulos, componentes e objetos. Componentes representam entidades e regras de negócio, já um serviço representa uma função de negócio completa, sendo composto por uma coleção de componentes. Serviços podem ser reutilizados e empregados em novas transações, na camada de negócios, dentro de uma organização ou através de organizações.

5.3.7 OOS

Segundo Kropiwiiec (2003), OOS apresenta um modelo de segurança que permite representar a autorização ou bloqueio ao acesso a elementos de um documento XML. Essa representação pode ser realizada indicando diretamente elementos de um determinado documento ou através do esquema desse documento (representado por um *DTD*¹⁰).

Uma autoridade central utiliza um conjunto de esquemas para especificar o formato da informação que trafega dentro da organização. Os documentos XML são

instâncias dos esquemas definidos e mantidos em cada setor, descrevendo as informações específicas deste setor.

A relação entre a instância e o esquema permite uma distinção entre dois níveis de autorização, ambos permitindo uma alta granularidade nas especificações. O primeiro nível (associado aos documentos XML) permite controlar autorizações de baixo nível, provendo total controle sobre as autorizações de cada documento (são chamadas de autorizações *hard*, pois têm uma prioridade maior que as autorizações do segundo nível). O segundo nível (associado ao DTD) são autorizações de alto nível e provêm controle de autorizações para todos os documentos da organização ou departamento (são chamadas de autorizações *soft*, pois têm prioridade menor).

As autorizações para o esquema e para o documento XML podem ser especificadas referenciando cada elemento/atributo único em um documento. As autorizações de um elemento podem ser declaradas recursivamente (aplicam-se também aos subelementos) ou locais (aplicam-se somente aos atributos diretos).

As autorizações especificadas para os esquemas ou documentos XML são armazenadas em documentos chamados *XML Access Sheet* (XAS). Cada autorização indica se há permissão ou proibição para um usuário executar certa ação sobre um objeto, junto com a prioridade (*soft* ou *hard*) e o tipo (recursivo ou local). O objeto identifica um elemento ou um conjunto de elementos em um documento ou conjunto de documentos.

Os documentos e esquemas são caracterizados por URIs (*Uniform Resource Identifier*). Como existe alta granularidade, é necessário referenciar elementos e atributos específicos dentro dos documentos. Esses elementos e atributos são referenciados pela linguagem *XPath* (linguagem para representar expressões de caminhos).

Os usuários são caracterizados pela tripla <identificação do usuário, endereço IP, nome simbólico máquina>, onde a identificação do usuário é o nome utilizado no *login*, o endereço IP é o endereço da máquina de onde o usuário fez a solicitação e o nome simbólico é o nome da máquina registrado no servidor DNS.

As autorizações podem ser especificadas através de grupos de usuários ou padrões de localização. Os grupos de usuário são os grupos definidos no servidor. O

padrão de localização é uma expressão identificando um conjunto físico de localizações, referenciando o nome simbólico ou o endereço IP.

O processamento utiliza a linguagem XSL e pode ser realizado tanto no servidor quanto no cliente. O cenário principal envolve um usuário solicitando um conjunto de elementos XML de um computador remoto. O processador usa como entrada o documento XML válido solicitado pelo usuário, junto com o documento XAS associado. A saída do processador é um documento XML válido incluindo somente as informações às quais o usuário tem permissão de acesso. Esse documento é então enviado ao usuário como resultado da solicitação.

5.3.8 RPC

RPC (*Remote Procedure Call*), segundo Becker, Claro e Sobral (2003), é um conjunto de regras para a construção de sistemas distribuídos. Estas regras especificam meios de codificar e decodificar informação transmitida entre duas aplicações. Efetivamente, RPC fornece um mecanismo para definição de *interfaces* que podem ser invocadas através de uma rede.

Uma operação RPC possui uma assinatura (o nome da operação), seus parâmetros de entrada, resultados e exceções que podem acontecer. A assinatura de uma operação é encapsulada em uma estrutura chamada IDL (*Interface Definition Language*).

A chamada de um procedimento que está localizado em um sistema remoto requer especificar qual sistema contatar, como codificar os parâmetros, como receber a resposta e como decodificar a resposta para utilização em um sistema específico.

Uma das primeiras especificações para construção de sistemas distribuídos foi feita em 1990 pela OSF (*Open Software Foundation*) com DCE (*Distributed Computing Environment*). Dois outros produtos de desenvolvimento de aplicações distribuídas que surgiram em seguida foram CORBA(OMG) e DCOM(*Microsoft*). Estes produtos especificam mecanismos que forçam a utilização de um protocolo específico para a comunicação entre as aplicações.

5.4 WEB SERVICES

“Trata-se de uma aplicação identificada através de uma URI, que possui *interfaces* bem definidas e descritas em XML. As interações com outras aplicações se faz através de trocas de mensagens XML utilizando protocolos padrões da Internet.” (W3C, 2004)

A segurança de um sistema computacional é vital para a sua utilização. Aplicações seguras na web são implementadas com utilização de serviços fornecidos pelo protocolo HTTP (*HyperText Transfer Protocol*), acessíveis a partir de um *browser* e com restrições de segurança bem definidas, como caracterizado pelos seguintes requisitos de segurança: Confidencialidade, Integridade, Disponibilidade e Legitimidade. Lojas virtuais e aplicações bancárias on-line são exemplos mais comuns dessas aplicações. Este é um protocolo de requisição/resposta e não foi originalmente desenvolvido para servir como base para as aplicações complexas existentes atualmente. Ele não mantém estado, o que torna cada requisição independente das demais.

Para acesso a essas aplicações, os usuários necessitam passar por um processo de autenticação criterioso. Os dados utilizados nesse processo, por sua vez, devem ser protegidos de forma que não possam ser conhecidos por usuários ilegítimos. Por outro lado, os desenvolvedores dessa categoria de aplicações estão cada vez mais preocupados com a segurança dos sistemas e atuando fortemente para reforçá-la, principalmente dentro do ambiente da aplicação.

Como consequência dessa atitude, o foco de ataques voltou-se para o elo mais fraco, ou seja, o usuário final. Conforme discutido por [Schneier 2001], senhas são facilmente roubadas de usuários inadvertidos, bastando para isso, verificar a grande quantidade de fraudes ocorridas no sistema bancário nos últimos anos através da Internet com utilização de dados roubados.

Todo o processo de autenticação pode ser visualizado como sendo baseado em confiança, ou seja, uma vez autenticado, o usuário é considerado confiável para a aplicação. Em um cenário onde os dados de autenticação não são totalmente confiáveis, não se pode confiar cegamente no usuário, portanto, deve-se encontrar uma técnica

complementar para comprovar a veracidade das informações de autenticação e a própria validade dessa autenticação continuamente.

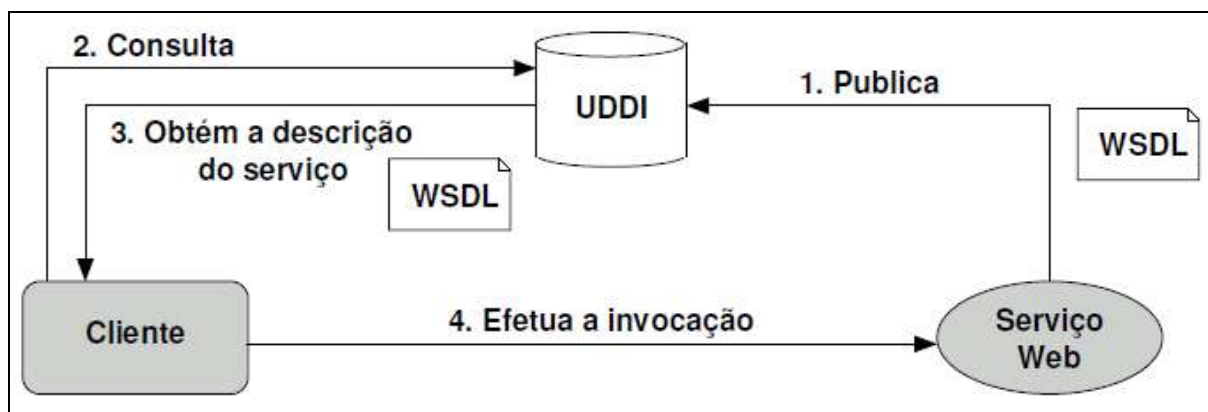


FIGURA 12 – Colaboração típica na Arquitetura dos Serviços Web
Fonte: SBSeg 2006

Segundo Camargo, Mello, Fraga e Wangham (2006), Os *Web Services* seguem uma Arquitetura Orientada a Serviços (AOS) e suas principais características são:

- Possuem um modelo fracamente acoplado e transparente que garante a interoperabilidade entre os serviços, sem que estes necessitem ter o conhecimento prévio de quais tecnologias estão presentes em cada lado da comunicação;
- São auto-contidos e auto-descritivos;
- Usam padrões abertos como o HTTP e o XML, permitindo assim que aplicações sejam integradas através de linguagens e protocolos amplamente aceitos;
- Tornam mais fácil a composição ou a combinação de diferentes provedores, visando formar serviços mais complexos e sofisticados.

O termo *web service* cria uma expectativa para o presente e ainda mais para o futuro das aplicações distribuídas. Promovida por grandes empresas como: *Microsoft, Sun, IBM e Borland*; os *Web Services* mudaram a forma de como enxergamos e utilizamos a internet.

A proposta dos *Web Services* é criar uma coleção de funções empacotadas e disponíveis em um ponto da rede, e que possam ser utilizadas livremente por outros

programas através de chamadas remotas de procedimentos. Padrões como o CORBA funcionam sob o mesmo princípio.

Os *Web Services* possuem uma arquitetura baseada na interação de três categorias: provedor do serviço (*service provider*), provedor de registro (*registry provider* ou *registry broker*) e o cliente do serviço (*service requestor*). Estas categorias envolvem as operações de procura (*find*), publicação (*publish*) e acoplamento (*bind*) do serviço. O provedor publica o serviço com a operação *publish*. O cliente do serviço utiliza a operação *find* para obter uma descrição de serviço do provedor de registro e usa esta descrição para, dinamicamente, acessar e interagir (*bind*) com o provedor do serviço. A figura 13 ilustra esta arquitetura:

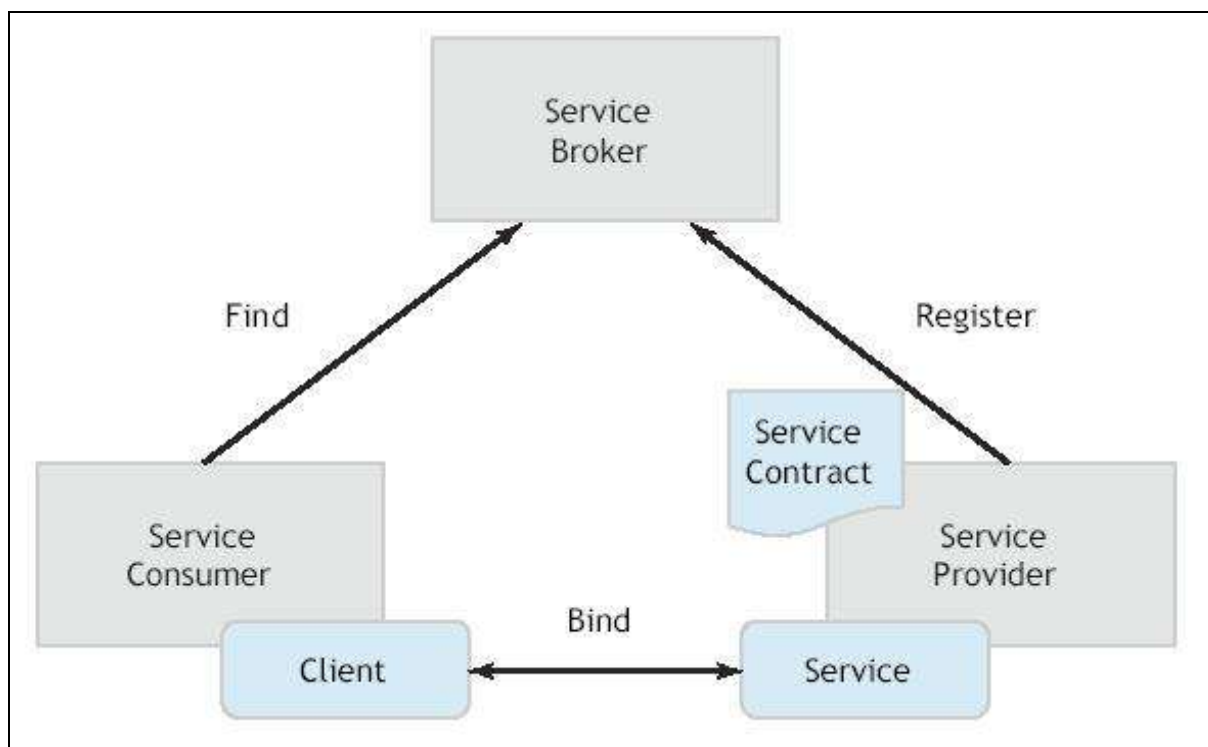


FIGURA 13 – Arquitetura e Integração de Web Services
Fonte: gta.ufrj, 2005.

De acordo com Freire (2002), o que difere os *Web Services* das demais arquiteturas são as características de implementação: seus protocolos são baseados em padrões aceitos e adotados pelo mercado – o protocolo HTTP (*HyperText Transfer Protocol*) como padrão para o transporte e o XML (*EXtensible Markup Language*) como padrão de representação de dados.

De acordo com Becker, Claro e Sobral (2003), a interação entre *Web Services* pode ser feita estaticamente ou dinamicamente em tempo de execução. Um

solicitante de um serviço descreve as características do serviço procurado e utiliza o provedor de registro para localizar um serviço apropriado. Uma vez localizado o serviço, a informação na descrição do serviço é utilizada para a interação entre cliente e servidor. A descoberta, a invocação dinâmica de serviços (*publish*, *find*, *bind*) e uma colaboração baseada em mensagens permitem o desenvolvimento de aplicações distribuídas fracamente acopladas com um enorme grau de interoperabilidade. A figura 14 retrata a interoperabilidade e padrões na qual são desenvolvidos *Web Services*:

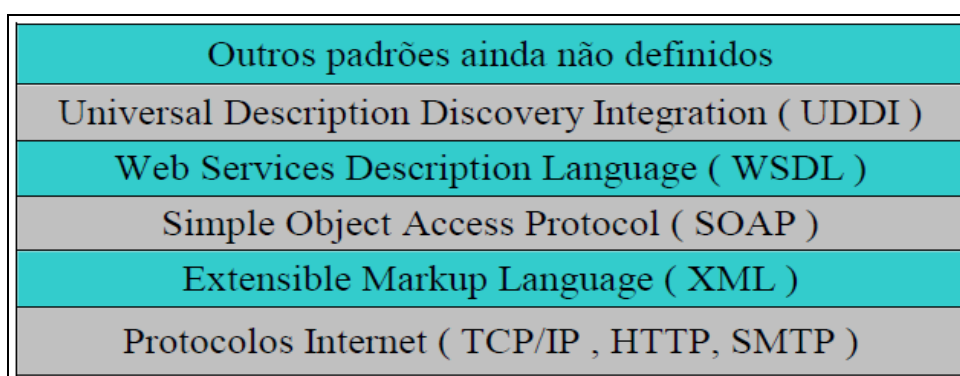


FIGURA 14 – Pilha de protocolos que compõe Web Services
Fonte: lasid, 2005.

Por sua vez, segundo Vêras e Ruggiero (2005), As aplicações seguras na *web* se diferenciam das demais aplicações *web* pela sua importância e restrições de segurança. O conjunto de regras e procedimentos de segurança aplicáveis a todas as atividades relacionadas com a aplicação segura na *web* dentro de um domínio define a política de segurança da aplicação. Ela formaliza as necessidades de segurança, e como mantê-la. Determina as expectativas para segurança fornecendo direção e suporte ao gerenciamento da mesma.

Para a utilização da confiança em aplicações *web*, é necessário entender e conhecer a aplicação, sua política de segurança, seus fluxos de interação e padrão de requisições para cada funcionalidade. Disso, advém a necessidade de se realizar um mapeamento da aplicação em uma máquina de estados finitos, utilizada para o acompanhamento criterioso dos usuários nas tentativas de operar as suas funcionalidades durante uma sessão *web* (VÉRAS e RUGGIERO; 2005).

Segundo Freire (2002), inúmeras aplicações surgiram desde a adoção dos *Web Services* e certamente outras ainda impensadas devem estar por vir.

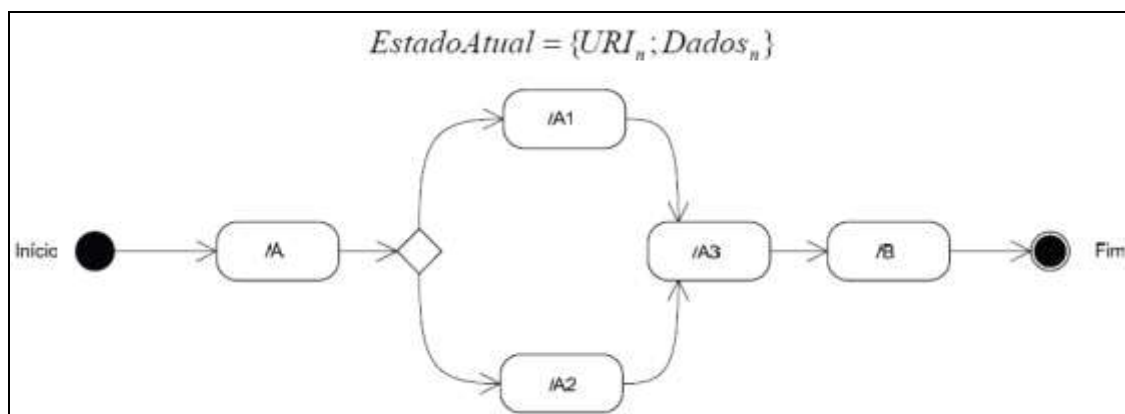


FIGURA 15 – Exemplo de mapeamento de aplicação web
Fonte: SBSeg 2005

5.5 MECANISMOS DE SEGURANÇA

5.5.1 Introdução

Segundo Camargo, Mello, Fraga e Wingham (2006), os *Web Services* permitem que as aplicações se comuniquem sem a necessidade de qualquer tipo de interação com o usuário final. O problema está em como garantir que as informações do usuário cheguem até o sistema da empresa de forma segura, visto que as informações sensíveis do usuário, que só interessam a estes sistemas, estariam sendo roteadas e disponíveis pelo sistema do portal em si.

O roteamento entre múltiplos *Web Services* é comumente utilizado para obter escalabilidade e também para agir como uma ponte entre diferentes protocolos. Tecnologias como o TLS/SSL permitem garantir a confidencialidade entre duas partes, porém não proporcionam segurança fim-a-fim, uma vez que a mensagem, para atingir o destinatário final, passa por diversos nós intermediários a nível de aplicação.

As lacunas de segurança não ocorrem no transporte dos dados, mas sim quando os mesmos estão disponíveis nos nós intermediários. Assim, as informações confidenciais presentes nas mensagens SOAP, que deveriam permanecer confidenciais durante todo o percurso através dos nós SOAP intermediários, poderiam ficar expostas. Para tratar tal desafio, princípios de segurança devem ser aplicados em um contexto de segurança, que inclui muito mais que uma simples troca de mensagens SOAP. A Figura 17 ilustra diferentes contextos de segurança,

em que o 1º contexto representa uma configuração ponto a ponto e o 2º uma configuração fim-a-fim.



FIGURA 16 – Contextos de segurança [IBM e Microsoft 2002]

Fonte: SBSeg 2006

Outro desafio é como garantir os limites de segurança, antes determinados pelos *firewalls*. Os filtros de pacotes tradicionais se ocupam basicamente com a segurança na camada de rede, analisando se o pacote vem de uma origem confiável, porém não se preocupa com o conteúdo dos pacotes. Assim, toda e qualquer requisição a um *Web Service* irá transpor o *firewall*. Os *Web Services* também estão suscetíveis a tipos de ataques já conhecidos como negação de serviço, mensagens antigas, estouros de pilha, entre outros.

Para garantir a segurança neste novo tipo de ambiente, novos mecanismos de segurança devem ser implantados também nas camadas superiores da pilha TCP/IP e devem operar em conjunto com os mecanismos presentes nas camadas inferiores (veja Figura 17).

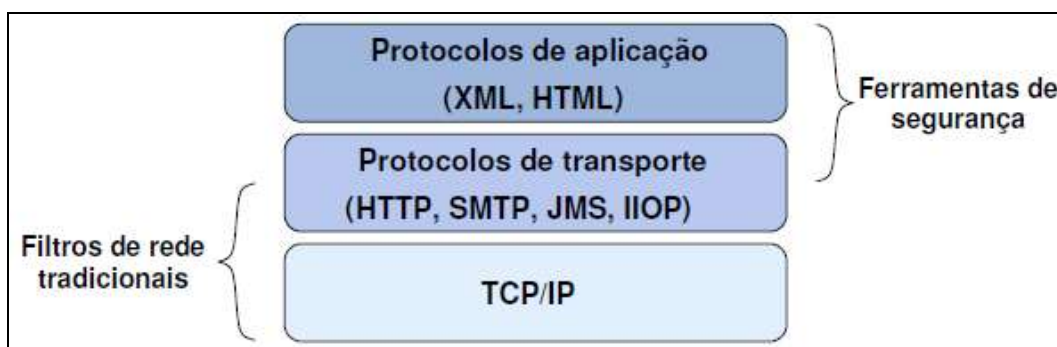


FIGURA 17 – Segurança nas diferentes camadas

Fonte: SBSeg 2006

5.5.2 Firewalls

Firewall é um quesito de segurança cada vez mais imprescindível no mundo da computação. À medida que o uso de informações e sistemas cresce, a proteção destes requer a aplicação de ferramentas e conceitos de segurança cada vez eficientes.

Pode ser definido como uma barreira de proteção, que controla o tráfego de dados entre seu computador e a *Internet* (ou entre a rede onde seu computador está instalado e a *Internet*). Seu objetivo é permitir somente a transmissão e a recepção de dados autorizados. Existem *firewalls* baseados na combinação de *hardware* e *software* e *firewalls* baseados somente em *software*. Este último é o tipo recomendado ao uso doméstico e também é o mais comum.

Explicando de maneira mais precisa, o *firewall* é um mecanismo que atua como "defesa" de um computador ou de uma rede, controlando o acesso ao sistema por meio de regras e a filtragem de dados. A vantagem do uso de *firewalls* em redes, é que somente um computador pode atuar como *firewall*, não sendo necessário instalá-lo em cada máquina conectada.

Há mais de uma forma de funcionamento, que varia de acordo com o sistema, aplicação ou do desenvolvedor do programa. No entanto, existem dois tipos básicos de *firewalls*: o que é baseado em filtragem de pacotes e o que é baseado em controle de aplicações ou estados do sistema. Ainda há o tipo baseado em aplicativos de *Firewall* e de *Proxy*. Cada um trabalha para um determinado fim, fazendo que a comparação entre eles não seja aplicável.

5.5.2.1 Filtragem de pacotes ou *packet filtering*

Um *firewall* com filtragem de pacotes é essencialmente um *router* com *software* de filtragem de pacotes. A filtragem de pacotes trabalha ao nível de rede do modelo OSI; cada pacote de dados é examinado quando a transferência de dados é feita de uma rede para a outra.

Os pacotes de dados são analisados e confrontados com um conjunto de filtros predefinidos pela configuração do *firewall* por parte do utilizador. Os pacotes de dados que estiverem de acordo com os padrões pré estabelecidos pela configuração passam pelo *firewall*, caso contrário serão pura e simplesmente recusados. É muito utilizado em redes pequenas ou de porte médio.

Através do *firewall* com filtragem de pacotes, alguns sistemas ou serviços podem ser liberados completamente, enquanto outros são bloqueados por padrão, por terem riscos elevados. O grande problema desse tipo de *firewall*, é que as regras aplicadas podem ser muito complexas e causar perda de desempenho da rede ou não serem eficazes o suficiente.

Ele se restringe a trabalhar nas camadas TCP/IP, decidindo quais pacotes de dados podem passar e quais não. Tais escolhas são regras baseadas nas informações endereço IP remoto, endereço IP do destinatário, além da porta TCP usada.

5.5.2.2 *Firewalls* de aplicação ou *proxies*

Firewalls de controle de aplicação são instalados geralmente em computadores servidores e são conhecidos como *proxy*. Este tipo não permite comunicação direto entre a rede e a *Internet*. Tudo deve passar pelo *firewall*, que atua como um intermediador. O *proxy* efetua a comunicação entre ambos os lados por meio da avaliação do número da sessão TCP dos pacotes.

Este tipo de *firewall* é mais complexo, porém muito seguro, pois todas as aplicações precisam de um *proxy*. Caso não haja, a aplicação simplesmente não funciona. Em casos assim, uma solução é criar um "*proxy* genérico", através de uma configuração que informa que determinadas aplicações usarão certas portas. Essa tarefa só é bem realizada por administradores de rede ou profissionais de comunicação qualificados.

O *firewall* de aplicação permite um acompanhamento mais preciso do tráfego entre a rede e a *Internet* (ou entre a rede e outra rede). É possível, inclusive, contar com recursos de log e ferramentas de auditoria. Tais características deixam claro que este tipo de *firewall* é voltado a redes de porte médio ou grande e que sua configuração exige certa experiência no assunto. a informação da *Internet* é recolhida pelo *firewall* e seguidamente enviada para o sistema requisitante e vice-versa.

5.5.2.3 *Firewalls* baseados no estado ou *stateful inspection firewalls*

Ao contrário da filtragem de pacotes, este método inspeciona cada ligação que atravessa todos os *interfaces* do *firewall* assegurando-se que é fidedigna. Este *firewall* analisa o tráfego ao nível do IP e TCP/UDP, construindo tabelas de estado das ligações à *Internet* de modo a que estejam salvaguardados os ataques do tipo *spoofing*, *replaying*.

Podemos relacionar algumas vantagens do uso deste tipo de *firewall*:

- Possibilitam um funcionamento ao nível da aplicação de forma dinâmica;
- Podem incluir funcionalidades como, encriptação de dados, encapsulamento e balanceamento de carga;
- A manutenção e configuração são extremamente reduzidas e de fácil aplicação por parte do utilizador.

Em contra partida, podemos citar a seguinte desvantagem:

- A relação do custo a pagar por este tipo de tecnologia, uma vez que se trata de algo mais eficiente que os anteriores citados e com pouca necessidade de intervenção por parte do utilizador, o que acarreta uma elevação dos custos, por outro lado, em relação a segurança da informação, então esta será a melhor opção (Engenharia de redes informáticas).

5.5.3 Sistemas detecores de intrusão

Um sistema de detecção de intrusão (*intrusion detection system*, IDS) é um processo ou dispositivo ativo que analisa as atividades do sistema e da rede e identifica entradas não autorizadas e/ou atividades maléficas. A maneira como um IDS detecta anomalias pode variar amplamente; o objetivo final de qualquer IDS é capturar os infratores na ação antes de realmente danificarem seus recursos.

Um IDS protege um sistema de ataques, mal-uso e danos. Também pode monitorar as atividades da rede, auditoria nas configurações da rede e do sistema

para detectar vulnerabilidades, analisar integridade de dados e muito mais. Dependendo dos métodos de detecção que você escolher aplicar, há diversos benefícios diretos e casuais em utilizar um IDS.

Entender o que é um IDS e as funcionalidades que oferece é essencial para determinar qual será o tipo apropriado para incluir em suas normas de segurança em informática. Esta seção aborda os conceitos por trás dos IDSs, as funcionalidades de cada tipo de IDS e a emergência de IDSs híbridos que aplicam diversas técnicas de detecção e ferramentas em um pacote.

Alguns IDSs são baseados no conhecimento, que alertam prioritariamente os administradores de segurança antes de uma intrusão ocorrer utilizando um banco de dados de ataques comuns. Alternativamente há alguns IDSs comportamentais que rastreiam todos os usos de recursos para encontrar anomalias, que normalmente são sinais positivos de atividades maldosas. Alguns IDSs são serviços isolados (*standalone*) que trabalham por trás do cenário e monitoram passivamente as atividades, registrando quaisquer pacotes suspeitos vindos de fora. Outros IDSs combinam ferramentas padrão de sistema, configurações alteradas e registro de palavras, juntamente à intuição do administrador e sua experiência em criar um *kit* de detecção de intrusão poderoso. Avaliar as diferentes técnicas de detecção pode ajudar a encontrar uma que seja correta para sua organização.

Os tipos mais comuns de IDSs mencionados no campo da segurança são conhecidos como IDSs baseados na máquina e baseados na rede. Um IDS baseado na máquina é o mais detalhado dos dois, envolvendo a implementação de um sistema de detecção em cada máquina separadamente. Independente do ambiente de rede no qual a máquina reside, ela está protegida. Um IDS baseado na rede afunila pacotes através de um único dispositivo antes de enviá-los a máquinas específicas. IDSs baseados na rede são frequentemente encarados como menos detalhados, já que muitas máquinas num ambiente móvel impossibilitam uma filtragem e proteção confiável dos pacotes na rede.

5.5.3.1 *Network-based intrusion detection system* ou sistema de detecção baseado na rede

SDIs baseados em Rede (SDIRs) monitoram todo o tráfego que passa através da rede. Os pacotes são capturados e é feita uma análise em cada um deles para verificar se este está dentro de padrões pré-determinados ou não, indicando respectivamente tráfego normal ou uma tentativa de ataque.

Em um meio compartilhado de difusão, os pacotes passam livremente e todas as *interfaces* a ele conectadas recebem estes pacotes. Dependendo do endereço de destino do nível de enlace do pacote, a *interface* o passa para o barramento interno da máquina para processamento posterior. Mas, a princípio, todas as *interfaces* recebem todo o tráfego. Sendo assim, é necessário posicionar o SDIR na rede que se quer proteger para que ele possa receber os pacotes necessários para sua análise, como mostrado na Fig. 18.

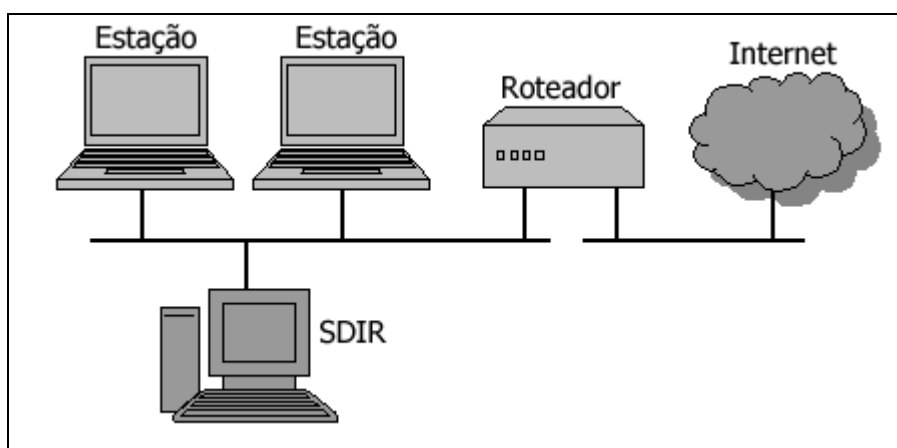


FIGURA 18 – Detecção de intrusão em uma rede local de difusão
Fonte: gta.ufrj, 2003.

É importante frisar que estações de trabalho conectadas à rede também podem receber pacotes destinados a outras estações. Para que uma *interface* de rede consiga receber pacotes não destinados a ela, é necessária que ela trabalhe em um modo especial, chamado modo promíscuo. Existem basicamente dois modos de operação em uma *interface* de rede:

- Normal, onde cada *interface* de rede verifica o endereço de destino do nível de enlace, também chamado endereço MAC (*Medium Access Control*), ou ainda endereço físico. Sendo o endereço do pacote idêntico ao endereço da *interface*, o pacote é repassado para processamento interno.
- Promíscuo, onde a verificação feita no modo anterior é completamente ignorada e todos os pacotes são repassados para processamento.

Esse processo também é chamado de "farejar" (*sniff*) a rede. Isso pode trazer problemas tanto para a estação de trabalho, que pode não suportar o processamento de todos os pacotes de uma rede local (seria necessário verificar o endereço de destino da camada de rede por *software*), quanto para a segurança da rede, que pode ficar comprometida devido a exposição de dados sigilosos, como autenticações.

Os SDIRs precisam trabalhar em modo promíscuo para que mesmo os pacotes não destinados a ele possam ser analisados. Tendo todas as informações que trafegam pela rede é possível ao sistema, por exemplo, detectar varreduras de portas (e.g., tentativas sucessivas de abertura de conexões TCP a diferentes portas), monitorar conexões ou datagramas maliciosos (verificação do conteúdo no pacote com destino ao servidor *web*) ou ainda verificar ataques de negação de serviço (através da falta de resposta de um servidor a uma requisição de um serviço).

Em redes comutadas, cada máquina recebe o tráfego que corresponde somente a ela. Ou seja, uma máquina A não receberia tráfego destinado a máquina B e vice-versa. Isso se deve ao conhecimento da localidade física de onde está conectada cada estação pelo comutador, ou seja, em qual porta está qual máquina, e ao fato da análise do endereço de destino do cabeçalho da camada de enlace. Para detectar intrusões em redes desse tipo, seria necessário algum tipo de artifício no comutador com o objetivo de fazer com que todo o tráfego que passa por ele seja enviado, além da porta do destinatário, para a porta onde está conectado o SDIR. Esse artifício é chamado de espelhamento de portas e tem objetivo de refletir o tráfego de outras portas para uma porta específica. O modo como ele é feito é mostrado na figura 19 onde a estação A passa um quadro para a estação B e este é refletido na porta do SDIR.

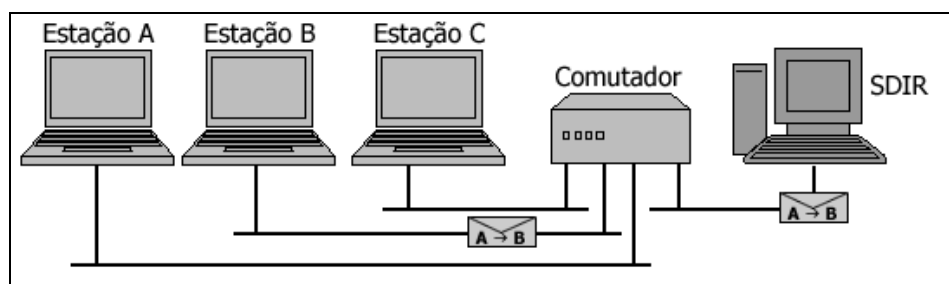


FIGURA 19 – Detecção de intrusão baseado em espelhamento de portas no computador
Fonte: gta.ufrj, 2003.

5.5.3.2 *Host-based intrusion detection system* ou sistema de detecção baseada na estação ou servidor

Segundo Laufer (2003), SDIs baseados na Estação ou Servidor (SDIEs) foram os primeiros sistemas de detecção de intrusão implementados. O objetivo deles é monitorar toda a atividade existente em uma estação específica. O funcionamento desses sistemas se dá através da coleta e análise de dados originados em uma máquina que hospeda um serviço. Depois de coletados, esses dados podem ser analisados localmente ou até enviados para uma máquina remota responsável pelo exame.

Um exemplo típico de um sistema de coleta de dados seria um sistema de registros (*logs*). Esses sistemas são responsáveis pelo armazenamento de ocorrências em aplicações que rodam na estação. Ou seja, as aplicações estão configuradas para enviar seus registros para este sistema e ele é o encarregado de armazená-los em uma série de arquivos. Um eventual sistema de análise usaria essa base de registros do sistema comparando-a com padrões pré-estabelecidos com o objetivo de detectar intrusões. A arquitetura de um sistema desse tipo pode ser visualizada na figura 20.

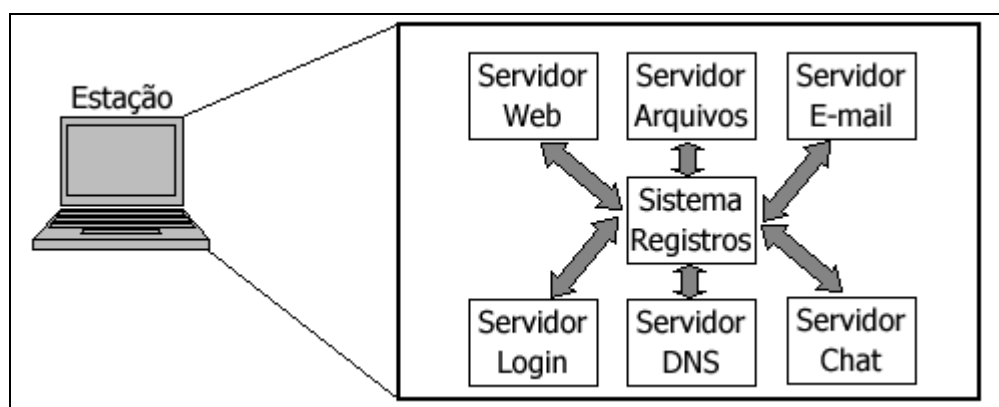


FIGURA 20 – Arquitetura de um sistema de registros
Fonte: gta.ufrj, 2003.

Os SDIEs podem atuar em diversas áreas dentro de uma mesma estação com o objetivo comum de detectar intrusões. Eles podem monitorar todo um sistema de arquivos através de uma comparação desse sistema com dados recolhidos

anteriormente referentes a esse sistema detectando modificações. Isso poderia ser feito se as informações de arquivos críticos do sistema, como data da última modificação fossem recolhidos com antecedência a um teste de verificação da integridade do sistema de arquivos. Existem outros meios que podem ajudar na detecção de intrusos em estações, como o monitoramento da atividade de rede, monitoramento de atividades de *login* e do super-usuário (*root*), entre outras.

5.5.3.3 *Hybrid intrusion detection system* ou sistema híbrido de detecção de intrusão

De acordo com Laufer (2003), os dois tipos de detecção de intrusão apresentados se diferenciam, mas se complementam. À medida que os SDIEs atuam somente em estações críticas, o SDIR atua analisando todo o tráfego de rede, inclusive aquele para estações que não contem um sistema de detecção rodando. Uma configuração bastante comum seria a de um SDIR para a rede local e SDIEs rodando nos servidores principais, como é mostrado na figura 21.

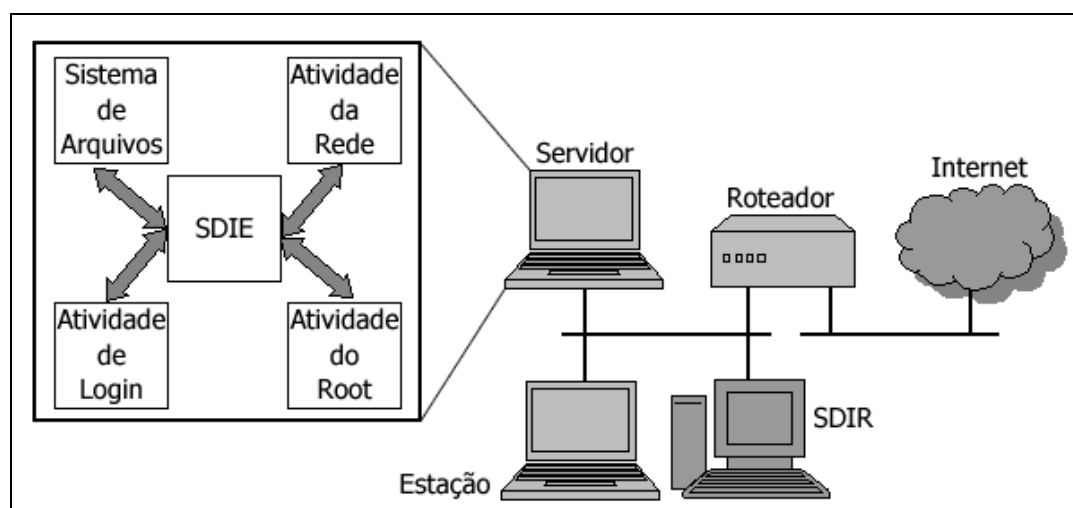


FIGURA 21 – Detecção de intrusão híbrida
Fonte: gta.ufrj, 2003.

5.5.4 *Honeypots* ou potes de mel

No mundo real, *Honeypots* atraem muitos tipos de animais que têm o objetivo de se alimentar, pelo menos esse é o comportamento esperado. Porém, sendo o mel

muito viscoso, muitos animais acabam ficando relativamente presos ali, ficando evidente que este animal tentou se alimentar do mel. O conceito é análogo quando levado para o mundo da segurança de redes: uma estação é colocada a disposição para quem quiser invadi-la, sendo esta estação apelidada de *honeypot*. Esta estação, com alguns serviços vulneráveis, tem por objetivo atrair usuários dispostos a invadi-la, ou seja, é uma máquina que serve exclusivamente para ser testada, atacada e comprometida. E toda a informação usada para a invasão será armazenada, de preferência em outra máquina, deixando visíveis os passos seguidos pelo invasor, até onde ele conseguiu chegar e o modo como isso foi feito.

Os potes de mel, por si só, não são uma melhoria para a segurança de rede. O que eles fornecem são informações valiosas de como são feitas algumas intrusões, como determinados serviços vulneráveis são explorados e ainda como é o comportamento dos invasores após terem comprometido a máquina. Somente isso não agrega nenhuma segurança a rede, pelo contrário, pode até chamar a atenção para a rede onde o pote de mel está localizado. A grande vantagem de se utilizar potes de mel é que, tendo informações de como determinados tipos de invasões são feitas, regras de como evitá-las, detectá-las e reagir a elas podem ser criadas contribuindo para uma melhoria da segurança da rede.

Os *Honeypots* apresentam vantagens e desvantagens. As vantagens são:

- Um *Honeypot* coleta muito poucos dados e o que ele coleta geralmente é de alto valor. Sendo, pela definição de pote de mel, uma estação destinada a ser comprometida, assume-se que praticamente todo o tráfego que entra e sai dela é pelo menos suspeito. Isso significa que não existe nenhum tráfego de produção atravessando aquela máquina, ou seja, qualquer tráfego que esteja entrando provavelmente é um teste, uma varredura ou até mesmo um ataque; qualquer tráfego que esteja saindo da máquina significa que o pote de mel foi comprometido. Esse modelo é bem simples, o que garante uma grande facilidade na captura, armazenamento e filtragem dos dados importantes, dado que um dos grandes problemas em segurança é achar algum dado útil no meio de *gigabytes*, ou até *terabytes*, de informação.
- Funcionam como fonte de recursos. Assim como já foi exposto na seção anterior, a principal função de um pote de mel é fornecer recursos para aprender o modo como as invasões são feitas. Tendo essas informações, regras para um sistema de detecção de intrusão

poderiam ser elaboradas e a rede estaria mais segura.

As desvantagens são:

- Caso nenhum ataque venha a ocorrer nos *Honeypots*, eles não tem sentido algum. Os objetivos que um pote de mel propõe somente serão atingidos se a máquina realmente for submetida a algum tipo de teste, varredura ou ataque. Caso isso nunca aconteça, não será possível a captura de dados importantes relacionados a invasões e a máquina não terá nenhuma função. Para isso, é importante que seja liberado o acesso àquela estação, tendo sempre em mente que a rede de produção deve ser isolada de onde está situado o pote de mel, preferencialmente através de um *firewall*.
- Os *Honeypots* apresentam riscos na rede do usuário. Isso se deve ao fato de um pote de mel chamar a atenção para a rede onde ele está localizado. Sendo assim, além do *Honeypot*, outras máquinas podem ser testadas contra falhas de segurança e talvez possam também ser invadidas. Antes de implementar qualquer tipo de *Honeypot*, é necessário um planejamento de toda a segurança da rede para evitar surpresas indesejáveis.

5.6 PRÁTICAS DE SEGURANÇA

A adoção de boas práticas de segurança passa criação e utilização de regras as quais os envolvidos (tanto o produtor quanto o consumidor de tal informação) devem seguir para garantir a confiabilidade, a integridade, o não-repúdio e a disponibilidade dos dados.

A política de segurança não define procedimentos específicos de manipulação e proteção da informação, mas atribuem direitos e responsabilidades às pessoas (usuários, administradores de redes e sistemas, funcionários, gerentes, etc.) que lidam com essa informação. Desta forma, elas sabem quais as expectativas que podem ter e quais são as suas atribuições em relação à segurança dos recursos computacionais com os quais trabalham. Além disso, a política de

segurança também estipula as penalidades às quais estão sujeitos aqueles que a descumprem.

Antes de descrever que práticas devem ser adotadas, é necessário estipular e definir que informações serão influenciadas ou protegidas.

Isso deve ser feito analisando os riscos identificando:

- Recursos e informações que serão protegidas de acordo com critérios e prioridades de controle;
- Ameaças as quais estes recursos estão suscetíveis;
- Vulnerabilidades que podem viabilizar a concretização dessas ameaças, analisando-as individualmente.

Determinados aspectos devem ser seguidos para se garantir boas práticas de segurança:

- Aspectos preliminares
 - Escopo e níveis de abrangência e atuação das práticas;
 - Normas e regulamentos a que estão subordinadas;
 - Determinar os papéis de cada envolvido no cenário;
 - Meios que serão usados na distribuição da informação;
 - Como e com que frequência serão revisadas as práticas em vigência.
- Políticas de acesso e criação de senhas
 - Criação de regras e requisitos na utilização de contas de acesso às informações, assim como privilégios de acesso compatíveis ao cargo ou função;
 - Educação dos funcionários, parceiros, fornecedores e clientes e dissemine uma "cultura de segurança da informação" (o sucesso da proteção do meio é resultado de campanhas de conscientização). A 1ª ameaça à execução de qualquer processo é o ser humano - sobretudo se ele não entender o porquê do que está fazendo;
 - Determinação de regras de criação de senhas válidas e tempo validade das mesmas;
 - Normas de proteção e reuso de senhas;
 - Critérios para utilização de senhas *default*;

- Regras para instalação, replicação e utilização de *softwares* e informações;
- Proteção das informações e recursos por senhas;
- Remoção das contas de rede inativas (ex: contas de ex-funcionários).
- Normatização no uso de correios eletrônicos e páginas da *internet*;
- Direito a privacidade e condições nas quais esse direito pode ser violado pelo provedor dos recursos e informações;
- Uso de antivírus, *Firewalls* e Sistemas de detecção de intrusão.
- Direitos e responsabilidades de quem detém a informação
 - Criar o hábito de realizar cópias de segurança periodicamente e testes em que as mesmas possam ser recuperadas com integridade;
 - Diretrizes para configuração e instalação de sistemas e equipamentos de rede;
 - Adoção e criação de normas de planos de contingência para controle e gerenciamento de emergências;
 - Remover de seus computadores todos os programas e recursos que não façam parte das necessidades de rotina trabalho ou que não contribuam para o aumento de produtividade de seus funcionários (ou não permita a instalação). Lembrando, quanto mais programas instalados um computador possuir, mais lento e mais sujeito à incidentes de segurança ele está;
 - Autoridade para conceder e revogar autorizações de acesso, conexão, alocação, registro e enderçamento de sistemas, equipamentos e usuários;
 - Monitoramento das atividades dos sistemas, equipamentos e usuários;
 - Normas de segurança física e lógica.
- Ações a serem tomadas em caso de descumprimento de tais práticas
 - Diretrizes para tratamento e resposta de incidentes de segurança;
 - Estabelecer penalidades a serem usadas em atitudes de descumprimento das práticas de segurança.

É necessário ressaltar que os tópicos acima não são exaustivos, nem

tampouco se aplicam a todos os casos. Cada organização possui seu ambiente próprio e distinto e os seus próprios requisitos de segurança, e por isso, deve desenvolver uma política de segurança que se adeque a essas peculiaridades. É recomendável que organizações que possuam uma rede *wireless* incorporem uma política específica para uso deste tipo de rede à sua política de segurança.

Alguns fatores são importantes para o sucesso de uma política de práticas de segurança:

- Apoio por parte da administração superior;
- As práticas devem ser amplas, cobrindo todos os aspectos que envolvem a segurança dos recursos computacionais e da informação sob responsabilidade da organização;
- A política das práticas deve ser periodicamente atualizada de forma a refletir as mudanças na organização;
- Deve haver um indivíduo ou grupo responsável por verificar se as práticas estão sendo respeitadas;
- Todos os usuários da organização devem tomar conhecimento e manifestar a sua concordância em submeter-se antes de obter acesso aos recursos computacionais;
- A política das práticas deve estar disponível em um local de fácil acesso aos usuários, tal como a *intranet* da organização.

Dentre os itens acima, o apoio por parte da administração superior é essencial. Se a política de segurança não for encampada pela administração, ela rapidamente será deixada de lado pelos demais setores da organização. Além disso, é importante que os seus membros dêem o exemplo no que diz respeito à observância da política de segurança.

Os seguintes fatores influem negativamente na aceitação de uma política de segurança e podem levá-la ao fracasso:

- As práticas não devem ser demasiadamente detalhadas ou restritivas;
- O excesso de detalhes pode causar confusão ou dificuldades na sua implementação;
- Não devem ser abertas exceções para indivíduos ou grupos;
- A política das práticas não deve estar atrelada a *softwares* e/ou *hardwares* específicos.

Garantir a segurança não implica em criar um ambiente paranóico dentro da organização. A correta implementação das medidas e procedimentos, aliadas ao gerenciamento contínuo de processos cotidianos, permitem minimizar consideravelmente a ocorrência de incidentes de segurança em sua empresa e, mesmo nos caso em que eles ocorram evitar a perda de informações preciosas, recuperando os dados e restaurar rapidamente os ambientes de trabalho.

"Boas práticas" são aquelas que provaram ser eficazes através de sua utilização e, por isso, são consideradas essenciais em praticamente qualquer contexto em que se apliquem. Muitas são tão básicas que deveriam ser implementadas por qualquer empresa de forma direta ou indireta.

6 CONSIDERAÇÕES FINAIS

Como foi visto durante a produção deste trabalho, a segurança da informação ganha uma importância maior em decorrência do advento da internet.

Para garantir a continuidade do processo, a segurança de informação aplicada a aplicações *Web* visa atender os requisitos de confiabilidade, integridade e disponibilidade.

Em decorrência das vulnerabilidades, pesquisas e estudos mostram que apenas uma pequena parcela das empresas, cerca de 6%, conseguem ter um *know-how* necessário para se montar em sobrevida diante dos milhares e mais diversas ameaças a que podem ser expostas numa falha de algum dos métodos, ou processos da política de segurança.

As ameaças variam desde um simples “vírus”, até um *hacker*, suficientemente inteligente para perceber e detectar os erros do sistema; ou até de dentro da própria instituição quando um empregado mal intencionado manipula e altera ou danifica um arquivo ou informação do sistema.

“A segurança deverá ser proporcional ao valor do ativo que está sendo protegido” (ALASI, 2005).

Habitualmente, vemos através dos meios de comunicação notícias de bancos de dados de empresas, agências bancárias, órgãos públicos, governamentais e militares de todas as partes do mundo sendo invadidos e violados por profissionais cada vez mais ousados, e que cada vez menos se contentam em apenas quebrar a segurança de computadores ou servidores de arquivos em rede.

Cópias de segurança apenas garantem que uma informação até aquele momento da gravação não serão perdidas ou comprometidas. O mesmo não poder ser correto a partir do instante de segundo imediatamente após.

Para obter-se sucesso na proteção de informações a adoção de uma política clara e que contenha o compromisso dos envolvidos, faz-se necessária e indispensável para a manutenção e continuidade dos processos de organização.

Como foi exposto, equipamentos como *firewalls* e métodos como sistemas de detecção de intrusão são eficientes para evitar formas simples de invasão aos sistemas. Porém pouca será a serventia dessa tecnologias se o profissional que os mantém não tiver a competência e conhecimento exigidos e aplicáveis no cotidiano das funções.

A negação de serviço consiste em um tipo de ameaça à segurança onde ocorrem perturbações no acesso a um serviço que originam sobrecarga e até impedimento no processamento dos pedidos de acesso.

As falhas que podem ser exploradas por *hackers* variam desde aquelas que afetam a disponibilidade até as que permitem a execução de códigos maliciosos.

Alguns tipos de ameaças como *spam* e vários tipos de *malware* necessitam da ação humana para se concretizar e propagar a outros computadores e redes.

Diferente deste raciocínio há também ameaças que não dependem de falhas humanas e que camuflam a presença do invasor, garantindo a ele um retorno seguro sem a percepção dos métodos de proteção. A esse tipo de ameaça dar-se o nome de *backdoors*.

Dos problemas com segurança em sistemas de computação, metade ocorrem devido à tecnologia. A outra decorre de falhas humanas que são geradas pelo descuido ou mau uso dos recursos computacionais ou ainda do desconhecimento ou desrespeito das regras de segurança.

Vimos anteriormente que diversos recursos de defesa podem ser empregados às redes. Enquanto *firewalls*, SDIs e criptografia tentam garantir a integridade e não violação dos ativos; a assinatura e o certificado digital tentam assegurar a autenticidade da informação. E por fim, os *backups* garantem a disponibilidade e continuidade.

Por sua vez, os servidores de alta disponibilidade visam eliminar paradas não programadas, assegurando assim também a disponibilidade e continuidade do serviço e da informação.

Apesar de existirem várias formas de proteção á informação, citações como a de Eugene Spafford onde ele diz: Que mesmo um sistema estando desligado, lacrado, enterrado, e vigiado; ainda assim sua total segurança não está garantida e inspira cuidados permanentemente.

A segurança da informação ganha importância, pois ela tenta assegurar que informações vitais não cheguem al alcance de pessoas mal intencionadas, representadas nas figuras dos *hackers* e *crackers*.

Esses profissionais atacam sistemas das mais variadas formas explorando vulnerabilidades que, na maioria, só uma parcela pequena é visível aos desenvolvedores inicialmente.

Algumas dessas pessoas ganharam destaque mundial devido às suas capacidades de explorar as deficiências dos sistemas e tirar vantagens delas.

Diversos recursos, linguagens e tecnologias foram incorporadas aos métodos anteriormente citados para reforçar a segurança dos dados dos sistemas.

Os *Web Services*, pelas informações por eles transportadas e inseridas requerem cuidado maior nos quesitos de segurança. Cada grande corporação buscou criar sua própria tecnologia ou ferramenta que garantisse a confiabilidade dessa segurança.

Porém, como vimos o sucesso da vigilância dos sistemas está diretamente ligado ao comportamento e as atitudes de quem administra e mantém os sistemas. Criar uma cultura de segurança da informação passa a ser o objetivo inicial de cada empresa. Cada profissional passa a ser decisivo nessa cultura, pois os que seguem representam menos vulnerabilidades que podem ser exploradas por quem tentar invadir os sistemas da empresa.

Adotar boas práticas de segurança custa menos do que o prejuízo de expor suas informações e tornar-se a próxima vítima das ameaças de segurança da informação.

REFERÊNCIAS

ACADEMIA latino americana de segurança da Informação, <<http://www.tecnet.com.br/academia/>> acesso em: 06 ago. 2011.

ALECRIM, Emerson; 2004. **Firewall: conceitos e tipos** - Disponível em: <<http://www.infowester.com/firewall.php>> acesso em: 30 ago. 2011.

ALECRIM, Emerson; 2005. **CRİPTOGRAFIA** - Disponível em: <<http://www.infowester.com/criptografia.php>> acesso em: 30 ago. 2011.

ASSAD, Rodrigo. **Segurança em aplicações WEB: A visão do invasor** – disponível em: <www.cesar.edu.br/docs/PalestraAssad.pdf> acesso em: 20 ago. 2011.

BAUER, César Adriano; 2006. **Política de segurança da informação para redes corporativas** - <<http://svn.assembla.com/svn/odinIDS/Egio/artigos/PraticasSeguranca/politicasSeguranca.doc>>. Acesso em: 02 ago. 2011.

BECKER, Aleksader Knabben; CLARO, Daniela Barreiro; SOBRAL, João Bosco. **Web Services e XML: Um Novo Paradigma da Computação Distribuída** – disponível em : <<http://www.lasid.ufba.br/pessoal/danielaclaro/download/ArtigoWebServices.pdf>>. Acesso em: 23 ago. 2011.

CAMPOS, Fernando; SIGNORINI Neto, Nery; CANTO, Wagner F.. **Web 2.0 Sob a Perspectiva da Segurança** – disponível em <<http://www.fernandocampos.pro.br/artigos/Web%202.0%20Sob%20a%20Perspectiva%20da%20Seguran%C3%A7a.pdf>> acesso em: 20 ago. 2011

CAMARGO, Francisco; 2004. **Segurança das Aplicações Web: Roubo, Chantagem, Fraude.** – disponível em <<http://thiagogenez-tcc.googlecode.com/svn/trunk/tcc/>> acesso em: 20 ago. 2011.

CAMARGO, Edson; MELLO, Emerson Ribeiro de; FRAGA, Joni da Silva; WANGHAN, Michelle S. – 2006. **Segurança em Serviços Web.** Disponível em: <gcseg.das.ufsc.br/wssec/pubs/mello06-sbseg-mc-seg-ws.pdf> acesso em: 25 ago. 2011.

CERT.br - Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil - disponível em <<http://www.cert.br/>> acesso em: 06 ago. 2011.

FERREIRA, Fernando Nicolau Freitas; ARAÚJO, Marcio Tadeu de. **Políticas de segurança da informação** Rio de Janeiro: Ciência Moderna, 2006. 178p.

FREIRE, Herval. **Web Services: A Nova Arquitetura da Internet**, 2002. Disponível em: <www.cnnt.com.br/files/webservices.pdf> acesso em: 24 ago. de 2011.

HARPIA Consultoria Empresarial, disponível em <http://harpiaconsultoria.com.br/material/seguranca/Modulo3_Capitulo1_FINAL.pdf> acesso em: 10 ago. 2011

Hackers e Crackers – disponível em: <<http://winterwdicas.blogspot.com/2011/04/hackers-e-crackers.html>> acesso em: 21 ago. 2011.

Hackers ou Crackers: Quem são? – disponível em: <<http://www.parnamirim.rn.gov.br/tecnologia/hackers-ou-crackers-quem-sao/>> acesso em: 21 ago. 2011.

ISECURE – **O que é um honeypot?** – disponível em <<http://drkmario.blogspot.com/2005/08/o-que-um-honeypot.html>> acesso em: 18 ago. 2011

JUSBASIL - **Sites da Presidência e do Senado são hackeados** – disponível em <<http://www.jusbrasil.com.br/politica/7187789/sites-da-presidencia-e-do-senado-sao-hackeados>> acesso em: 06 ago. 2011.

KROPIWIEC, Cássio Ditzel. **Proposta de um mecanismo de segurança para web services baseado em IPsec**, 2003. Disponível em: <<http://www.ppgia.pucpr.edu.br/lib/exe/fetch.php?media=dissertacoes:2005:cassio-ditzel-003.pdf>> acesso em: 24 ago. 2011.

LAUFER, Paulo P.. **Introdução a Sistemas de Detecção de Intrusão**, 2003. Disponível em: <http://www.gta.ufrj.br/grad/03_1/sdi/index.htm>. Acesso em: 01 set. 2011.

MARINHO, Marcelo Mendes; LAHR, Thiago Canozzo. - **(In)Segurança em Aplicações Web**, 2008 – disponível em: <http://www.detaileventos.com.br/ibm_dc2008/download/%28In%29Seguranca%20em%20Aplicacoes%20Web.pdf> acesso em: 20 ago. 2011

MITNICK, Kevin. **Hacker Regenerado**. Revista Época, Globo, 2003. Entrevista disponível em: <<http://revistaepoca.globo.com/Epoca/0,6993,EPT600936-1666,00.html>>. Acesso em: 17 ago. 2011.

NAKAMURA, Emilio Tissato; GEUS, Paulo Lício de. **Segurança de Redes em Ambientes Cooperativos**, Novatec Editora, [São Paulo], 2007. 488p

NBR/ISO/IEC 17799. **Tecnologia da Informação** – Código de prática para a gestão da segurança da informação. Rio de Janeiro: Associação Brasileira de Normas Técnicas, 2001. 56p.

NTC-ISO/IEC 27002. **Tecnología de La Información** – Técnicas de seguridad, Código de práctica para la gestión de la seguridad de la información. Bogotá: Instituto Colombiano de Normas Técnicas e Certificação, 2007. 133p.

Os dez mais famosos hackers e seus feitos – disponível em: <http://www.naoaianessa.net/2010/09/os-dez-mais-famosos-hackers-e-seus_30.html> acesso em: 21 ago. 2011.

RED HAT; 2005. **Red Hat Enterprise Linux 4: Guia de Segurança** – disponível em: <http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-pt_br-4/ch-detection.html> acesso em: 31 ago. 2011.

SÊMOLA, M. **Gestão da Segurança da Informação** – Uma visão executiva. 3. Ed. Rio de Janeiro: Elsevier, 2003. 160p.

SOARES, Antonio Alexandre de Castro; DUARTE, Otto Carlos Muniz Bandeira. **Sistemas Detectores de Intrusão**. Disponível em: <www.gta.ufrj.br/seminarios/CPE825/tutoriais/asoares/Sistemas-Detectores-de-Intrusao.pdf> acesso em: 19 ago. 2011

VÉRAS, Alisson L. M.; RUGGIERO, Wilson V.. **Autenticação Contínua de Usuários em Aplicações Seguras na Web** – 2005. Disponível em: <<http://labcom.inf.ufrgs.br/ceseg/anais/2005/artigos/12462.pdf>> acesso em: 25 ago. 2011.

WATHIER, Adair José. **Segurança em Web Services com WS-Security**, 2005. Disponível em <http://saloon.inf.ufrgs.br/twiki-data/Docs/OnlineDoc20051217200943/SegurancaWebService_WSSecurity.pdf> acesso em: 24 ago. 2011.