Nilson Santos Costa

**Proteção de Sistemas Elétricos
considerando Aspectos de Segurança da
Rede de Comunicação**

Exame de Tese apresentado à Escola de Engenharia de São Carlos, da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Doutor em Engenharia Elétrica.

Área de concentração: Sistemas Elétricos de Potência
Orientador: Prof. Titular Denis Vinicius Coury

São Carlos

Abril / 2007

*Dedico esse trabalho a
todos que moram no meu
coração.*

AGRADECIMENTOS

Agradeço a Deus pelo trabalho realizado.

Ao Prof^o Tit. Denis Vinicius Coury que me orientou nesse trabalho.

Gostaria de agradecer, em especial, ao Prof. Dr. Sofiane Labidi pela , amizade e incentivo durante a realização desse trabalho.

Ao Professor Dr. Othon Bastos Filho pela amizade e incentivo.

A minha amada família, minha alegre e amada mamãe, as minhas irmãs que sempre, sempre estiveram comigo.

Ao meu pai que sei que sempre se orgulhou de mim e que hoje me acompanha os meus passos na minha mente e no meu coração.

Aos amigos que moram no meu coração, Raimundo João S. Filho, Ulysses T. B. Val, Luiz Giovani, Valtemir Emerencio, Renan Giovanini, Odilon Filho, Silvio Aparecido, Eugenia, Wesley Usida, Bruno Feres, Erika Hohn, Letícia Caldeira, Betiol, Danilo os dois, Emiliano, Ênio Segatto, Carlos F., Fernanda R. da Silva, Patrícia Teixeira, Kátia, Mario O., Mario Meireles.

A uma amiga especial Aline Bianco, e a todos aqueles que encontrei pelos caminhos da vida e que me tornaram uma pessoa melhor.

A mulher mais maravilhosa e dona do mais lindo sorriso, Marilene Spohr.

Aos amigos do LSEE – Laboratório de Sistemas de Energia Elétrica.

Aos professores, funcionários, graduandos e pós-graduandos do Departamento de Engenharia Elétrica.

A todas as pessoas que me ajudaram neste trabalho e na minha vida, meu muito obrigado.

À FAPEMA – Fundação de Amparo a Pesquisa e ao Desenvolvimento Científico e Tecnológico do Maranhão pela bolsa concedida.

E mais uma vez agradeço a Deus por permitir que todas essas pessoas amáveis e maravilhosas citadas acima estivessem ao meu lado em minha passagem por este mundo.

*“A vida é construída nos sonhos e
concretizada no amor!”
Francisco Xavier.*

RESUMO

COSTA, N.S., *Proteção de Sistemas Elétricos considerando Aspectos de Segurança da Rede de Comunicação*. São Carlos, 2007, 189p. Tese de Doutorado – Escola de Engenharia de São Carlos, Universidade de São Paulo.

O mundo moderno está cada dia mais conectado por todos os meios tecnológicos que existem hoje. Isto permite que mais e mais pessoas possam se comunicar, tornando a estrada da comunicação virtual obrigatória para a sobrevivência das pequenas, médias e grandes empresas públicas e privadas.

O grande avanço tecnológico do século 20 foi à utilização em grande escala do PC (*personal computer*) comumente chamados de microcomputadores. Este avanço também chegou aos Sistemas Elétricos de Potência, tornando as subestações digitalizadas. Estas subestações sendo digitais correm riscos de invasão cibernética interna ou mesmo externa. Embora a possibilidade de invasão cibernética externa seja pequena, ela existe. Diante dessa situação este trabalho propõe a aplicação de um Sistema de Segurança, aplicado em um Sistema Elétrico de Potência. O trabalho concentra-se especificamente no estudo dos Sistemas de Detecção de Intruso (SDI), nos seus dois modos básicos: o SDI por abuso e SDI por anomalia utilizando Redes Neurais Artificiais. Estes conceitos serão testados em um Sistema Elétrico de Potência simulado, com uma rede de comunicação baseada em microcomputadores e/ou equipamentos microprocessados, com relés digitais reais. Os Softwares, denominados SNORT e “Carcará”, foram utilizados e extensivamente testados com resultados altamente encorajadores para a função descrita.

Palavras chave: Sistema Elétrico de Potência, Sistema de Detecção de Intruso, Redes Neurais, Autômatos, Microcomputadores, SNORT, Carcará.

ABSTRACT

COSTA, N.S., Electric Power System Protection considering Safety Aspects of the Communication Network. São Carlos, 2007, 189p. Doctoral Thesis – São Carlos Engineering School, University of São Paulo.

Modern world is more connected each day by all technological means available. This allows more people to communicate, turning the virtual communication road obligatory to the survival of small, medium and large companies, whether public or private.

The great technological advance of the 20th century was the large use of the PCs (personal computer), usually called microcomputers. This advance also reached the Power Electric Systems with the digitalization of the substations. These digitalized substations, run the risk of cybernetic invasion, internal or even external. Although the possibility of external cybernetic invasion is small, it exists. In that context, the present thesis proposes the application of a Security System for an Electric Power System. The focus will be the study of Intruder Detection Systems (IDS), on its two basic forms: the IDS by abuse and the IDS by anomaly, using Artificial Neural Networks. These concepts will be tested in a simulated Electric Power System, with a communication network based on microcomputers, with actual digital relays with the digitalization of the substations.

Keywords: Electric Power System, Intruder Detection System, Neural Networks, Microcomputers, SNORT, Carcará.

LISTA DE FIGURAS

Figura 1	- Estrutura do EPOCHS em forma de dependência do UML_____	12
Figura 2	- Visão geral do NS2 _____	13
Figura 3	- Visão geral do NS em UML_____	15
Figura 4	- Visão da OTCL em UML_____	17
Figura 5	- Visão dos Pacotes do NS_____	18
Figura 6	- Visão do PSCAD em UML_____	19
Figura 7	- Diagrama de Estado para criar um projeto em UML_____	20
Figura 8	- Simulador AWSIM_____	21
Figura 9	- Simulador ModSAF _____	22
Figura 10	- Classificação dos agentes_____	24
Figura 11	- Mecanismos de funcionamento do EPOCHS_____	25
Figura 12	- Estruturação Funcional do EPOCHS_____	26
Figura 13	- Incidentes em redes de Computadores_____	31
Figura 14	- Tipos de incidentes em uma rede de Computadores_____	32
Figura 15	- Tipos de protocolos e portas mais utilizados para invasão_____	33
Figura 16	- Dia da semana em que ocorrem mais incidentes_____	34
Figura 17	- Países e a quantidade de ataques _____	35
Figura 18	- Declaração de um Certificado Falso_____	42
Figura 19	- Declaração de um Certificado Falso concluído_____	42
Figura 20	- Amostra de identificação de IP usando o Messenger e Netstat__	47
Figura 21	- Arquitetura do SNORT_____	74
Figura 22	- Funcionalidade do <i>Sniffer</i> do SNORT_____	75
Figura 23	- Pré-processador do SNORT_____	76
Figura 24	- Motor de Detecção do SNORT_____	78
Figura 25	- Componentes de Alerta do SNORT_____	79
Figura 26	- SDI com Roteador_____	81
Figura 27	- <i>Firewall</i> com SNORT_____	82
Figura 28	- <i>Firewall</i> DMZ e SNORT_____	83
Figura 29	- Ambiente de rede para a geração do tráfego dos cenários de teste_____	102

Figura 30	- Sequência de atividades a serem realizadas para coletar o tráfego de ataque_____	103
Figura 31	- Ambiente de rede para o cenário de avaliação_____	105
Figura 32	- Caso de Uso de Atividades do Operador_____	120
Figura 33	- Diagrama de Colaboração “Avaliação de Valores e Checagem de Limites” _____	125
Figura 34	- Diagramas de Colaboração “Leitura de Parâmetros e Configurações “Relés de Proteção” _____	126
Figura 35	- Diagrama de Colaboração “Operação de Equipamentos (Alta Tensão e Controle” _____	127
Figura 36	- Diagrama de Colaboração “Realizar a Indicação de <i>Status</i> ” _____	127
Figura 37	- Diagrama de Colaboração “Realizar a Manipulação de Eventos Sequenciais _____	128
Figura 38	- Os quatro turnos que os operadores trabalham (cada turno de 6 horas)_____	129
Figura 39	- Os quatro tipos de perfis de usuários (operadores)_____	130
Figura 40	- As cinco atividades Principais dos usuários (operadores)_____	131
Figura 41	- As subatividades dos usuários (operadores)_____	131
Figura 42	- A ação do usuário e a saída do usuário (operador)_____	132
Figura 43	- A ação do usuário (operador)_____	132
Figura 44	- Estruturação da ligação do Programa_____	133
Figura 45	- Estruturação da ligação do Programa _____	133
Figura 46	- Caso de uso do “Sistema de Atividades do Operador” _____	134
Figura 47	- Diagrama de Colaboração “Operação de Atividades” _____	140
Figura 48	- Diagrama de Colaboração “Configuração de Atividades” _____	140
Figura 49	- Diagrama de Colaboração “Sinalizar Alertas” _____	141
Figura 50	- Diagrama de Colaboração “Imprimir Relatórios” _____	142
Figura 51	- Diagrama de Colaboração “Realizar Auditoria” _____	143
Figura 52	- Invasor realizando <i>Scanner</i> _____	146
Figura 53	- Firewall com DMZ _____	147
Figura 54	- Canal dedicado do SEP e expandindo se necessário_____	148
Figura 55	- Sintaxe da regra <i>bad-traffic</i> _____	150
Figura 56	- Software de teste GFI Languard _____	151

Figura 57	- Estrutura da Rede de Computadores para os testes do SNORT__	152
Figura 58	- Iniciando o software SNORT no computador “Cobaia I”_____	153
Figura 59	- Microcomputador sendo encontrado pelo Microcomputador “X”	154
Figura 60	- <i>Software</i> de teste GFI Languard iniciando_____	154
Figura 61	- <i>Software</i> de teste GFI Languard resultados_____	155
Figura 62	- <i>Software</i> SNORT detectando o <i>scanner</i> do computador “X1“__	155
Figura 63	- Análise de escalabilidade do SNORT em relação aos ataques de inserção _____	157
Figura 64	- Análise de escalabilidade do SNORT em relação aos ataques de evasão _____	157
Figura 65	- Análise de escalabilidade do SNORT em relação aos ataques de varredura de portas _____	158
Figura 66	- Análise de escalabilidade do SNORT em relação aos ataques de negação de serviços _____	159
Figura 67	- Tentativas de realização de atividades de um perfil autorizado em um SEP _____	161
Figura 68	- Rede neural de várias Camadas do <i>Software</i> Carcará _____	162
Figura 69	- Rede neural de várias Camadas do <i>Software</i> Carcará_____	163
Figura 70	- Teste da Classe A1 (perfil do Operador)_____	164
Figura 71	- Teste da Classe B1 (perfil do Operador)_____	164
Figura 72	- Teste da Classe C1 (perfil do Operador)_____	165
Figura 73	- Teste da Classe D1 (perfil do Operador)_____	166
Figura 74	- Rede neural de várias Camadas do <i>Software</i> Carcará _____	166
Figura 75	- Matriz de Confusão _____	167
Figura 76	- Representação Gráfica do ator e do caso de uso_____	189
Figura 77	- Casos de uso de um Sistema escolar para professores via <i>web</i> __	190
Figura 78	- Estereótipos: Interface, Controle e Armazenamento_____	191
Figura 79	- Visualização de um diagrama de sequência_____	193
Figura 80	- Exemplo de um diagrama de colaboração_____	194
Figura 81	- Exemplo de um Diagrama de Transição de Estado_____	195

LISTA DE TABELAS

Tabela 1	Sistemas Operacionais e seus campos TTL_____	46
Tabela 2	Descrição de produtos adicionais para o SNORT_____	80
Tabela 3	Descrição técnica de ataques de evasão HTTP _____	96
Tabela 4	Descrição técnica de ataques de inserção de http_____	96
Tabela 5	Descrição técnica de ataques de varredura de Portas – Conexão _	96
Tabela 6	Descrição técnica de ataques de varredura de Portas IP _____	97
Tabela 7	Descrição técnica de ataques de varredura de Portas TCP _____	97
Tabela 8	Descrição técnica de ataques de varredura de Portas ICMP _____	97
Tabela 9	Descrição técnica de ataques de varredura de Portas UDP _____	97
Tabela 10	Descrição técnica de ataques de Negação de Serviço – Conexão _	98
Tabela 11	Descrição técnica de ataques de Negação de Serviço IP _____	98
Tabela 12	Descrição técnica de ataques de Negação de Serviço TCP _____	98
Tabela 13	Descrição técnica de ataques de Negação de Serviço ICMP _____	98
Tabela 14	Descrição técnica de ataques de Negação de Serviço UDP _____	98
Tabela 15	Seleção das ferramentas do cenário de Avaliação _____	100
Tabela 16	Operador (usuário) suas atividades e código da atividade _____	130
Tabela 17	Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de Inserção _____	156
Tabela 18	Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de Evasão _____	157
Tabela 19	Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de Varredura de Portas _____	158
Tabela 20	Resultados obtidos na análise do SNORT em relação aos ataques de negação de serviços _____	158
Tabela 21	Quantidade de Padrões dos Operadores, avaliados _____	162

LISTA DE ABREVIATURAS E SIGLAS

AgentHQ - *Agent Headquarter*

EPOCHS - *Electric Power and Communication Synchronizing Simulator*

GPS - *Global Positioning System*

IP - *Internet Protocol*

LAN - *Local Area Network*

RTI - *Runtime Infrastructure*

SCADA - *Supervisory Control and Data Acquisition System*

SEP - *Sistema Elétrico de Potência*

TCP - *Transmission Control Protocol*

UDP - *User Datagram Protocol*

VPN - *Virtual Private Networks*

WAN - *Wide Area Network*

MANS - *Metropolitan Área Network*

CERT - *Computer Emergency Response Team*

ICMP - *Internet Control Message Protocol*

SUMÁRIO

	RESUMO	p. ix
	LISTA DE FIGURAS	xiii
	LISTA DE TABELAS	xviii
	LISTA DE ABREVIATURAS E SIGLAS	xix
1	Introdução	01
1.1	Proteção dos Sistemas Elétricos de Potência (SEP)	01
1.2	A comunicação em um SEP por meio de uma Rede de <i>Intranet</i>	04
1.3	Segurança em uma Rede de Comunicação <i>Intranet</i>	06
1.4	Organização da Tese	08
2	Engenharia Reversa Funcional do EPOCHS utilizando a Linguagem de Modelagem Unificada (UML)	11
2.1	Introdução	11
2.2	Estrutura do <i>Network Simulator</i> (NS-2)	12
2.3	Estrutura do PSCAD	18
2.4	EPOCHS	20
	2.4.1 Trabalhos Relacionados	20
	2.4.2 Componentes do EPOCHS	22
2.5	Considerações Finais	26
3	Sistemas de Segurança em uma Rede de Computadores	29
3.1	Introdução	29
3.2	Tipos de ataques em uma Rede de Computadores	35
	3.2.1 Ataques não Técnicos	36
	3.2.2 Ataques Locais	40
	3.2.3 Ataques Remotos	44
3.3	Tipos de Defesa em uma Rede de Computadores	50
	3.3.1 Atualizações e Correções	51
	3.3.2 Vírus	51
	3.3.3 Códigos Seguros	52

3.3.4	Segurança dos <i>Hosts</i>	53
3.4	Segurança em Rede	54
3.5	Sistemas de Detecção de Intrusos	55
3.5.1	Definição	55
3.5.2	Vantagens dos Sistemas de Detecção de Intrusão	56
3.6	Métodos de Análise de Detecção de Intrusão	58
3.6.1	Detecção por Anomalia	58
3.6.2	Detecção por abuso	62
3.6.3	Detectores de Intrusos Baseado em Rede	66
3.6.4	Detectores de Intrusos Baseado em <i>Host</i>	67
3.6.5	Detectores de Intrusos Baseado em Aplicação	68
3.7	Considerações Finais	69
4	Definição, Estrutura e Metodologia de Avaliação do <i>Software</i> de Detecção de Intrusos SNORT	71
4.1	Definição do SNORT	72
4.2	Componentes do SNORT	73
4.3	O <i>Sniffer</i> do SNORT	74
4.4	O Pré-Processador do SNORT	75
4.5	O Motor de Detecção	77
4.6	Alertas	78
4.7	Aplicações do SNORT	81
4.8	Concorrentes do SNORT	83
4.9	Etapas da Metodologia	85
4.9.1	Seleção dos ataques	85
4.9.2	Descrição técnica do ataque	95
4.9.3	Seleção de Ferramentas	100
4.9.4	Geração do tráfego do Cenário de Avaliação	101
4.9.4.1	Coleta do tráfego de ataque	101
4.9.4.2	Geração do tráfego de fundo	104
4.10	Montagem do ambiente de avaliação	104
4.11	Considerações Finais	106
5	Estrutura de um Perfil de Usuário do SEP e o <i>Software</i> “Carcará”	109
5.1	Introdução	109

5.2	Alguns Trabalhos de Modelagem do usuário	110
5.3	Exemplos de Shells de Modelagem do usuário	111
5.4	Características Importantes para um Sistema de Modelagem do Usuário	112
5.5	Tendências nos Sistemas de Modelagem do Usuário	118
5.6	Definição de Atividades do Operador de uma Subestação de um SEP	120
5.7	A Base de dados do operador do Software “Carcará”	129
5.8	A estrutura do <i>Software</i> “Carcará”	132
5.9	Definição de casos de uso do Sistema de Detecção de Intruso “Carcará”	134
5.10	Diagramas de Colaboração dos casos de Uso do Sistema de Detecção de Intruso “Carcará”	139
5.11	Considerações Finais	143
6	Resultados da aplicação do <i>software</i> de Detecção de Intrusos SNORT e do <i>Software</i> “Carcará”	145
6.1	Resultados com o Software SNORT	145
6.1.1	Introdução	145
6.1.2	Planejamento Ideal para Instalação de um SDI para um SEP	147
6.1.3	Testes com o SNORT	149
6.1.3.1	Regras Básicas Utilizadas pelo SNORT	149
6.1.3.2	Ataque de “X1”, “X2”, “X3” e ”X4” e resposta do “Cobaia I”	152
6.2	Resultados com o <i>Software</i> “Carcará”	159
6.3	Considerações Finais	167
7	Conclusões Finais e Continuidade do Trabalho	169
8	Referências Bibliográficas	175
	Apêndice A - Conceitos básicos sobre UML	185
	Apêndice B – Segurança (Testes de Invasão)	197

1 INTRODUÇÃO

1.1 Proteção dos Sistemas Elétricos de Potência (SEP)

Os Sistemas Elétricos de Potência (SEP) são planejados para disponibilizar energia com qualidade, confiabilidade e continuidade. No entanto, os SEP estão constantemente expostos às várias contingências, tais como; descargas atmosféricas, catástrofes naturais, falhas na operação, falhas em seus dispositivos (geradores, transformadores, cabos, disjuntores, chaves de manobra, barramentos, relés, motores, etc.). Tais contingências podem prejudicar a todos os sistemas que estejam interligados, sendo necessário o isolamento da parte afetada, com o fim de minimizar seus efeitos danosos e manter a maior parte possível dos sistemas em funcionamento. Assim, é necessário um sistema de proteção seletivo e eficaz para assegurar a confiabilidade e a continuidade no suprimento de energia.

Nos SEP o mecanismo básico de proteção, denominado de relé, é utilizado para isolar as áreas envolvidas com defeito (falta). A construção básica dos relés no início era de origem mecânica, por isso foram chamados de relés eletromecânicos. Por serem eletromecânicos falhavam muitas vezes quando deveriam isolar uma área de falta. Este tipo de relé é construído com predominância dos movimentos mecânicos provenientes dos acoplamentos elétricos e magnéticos. O seu princípio de operação pode ser resumido como sendo uma relação entre sua entrada, que geralmente são sinais de tensão e corrente, e sua saída, que consiste em estados de *on-off* dos contatos do relé. Estes relés utilizam forças de atração que são produzidas por acoplamentos eletromagnéticos entre a corrente e o fluxo de sua bobina em seu interior. Alguns relés têm seu princípio de operação baseado nas forças criadas pela expansão do metal de seus contatos causada pelo aumento de temperatura durante a passagem da corrente.

Nos relés eletromecânicos, as forças atuantes são criadas pela combinação dos sinais de entrada e a energia armazenada nas bobinas dos enrolamentos internos do relé. Com a expansão dos sistemas de potência, surgiu a necessidade de sistemas de proteção mais confiáveis e com altos índices de desempenho. Isso foi alcançado com o desenvolvimento de relés utilizando-se dispositivos semicondutores, geralmente referidos como relés de estado sólido ou estático. O termo estático foi originado por oposição aos relés eletromecânicos, já que o relé estático é caracterizado essencialmente pela ausência de movimentos mecânicos, pois são construídos com dispositivos eletrônicos. Os primeiros relés estáticos colocados em operação no sistema elétrico causaram muitos problemas, produzindo operações indevidas. Esses problemas ocorreram principalmente porque eles, sendo eletrônicos, ficaram com sensibilidade muito apurada, e quaisquer transitórios ou pequenos harmônicos comuns ao sistema elétrico de potência eram suficientes para sua operação.

Todas as características e funções avaliadas nos relés convencionais estão presentes nos relés de estado sólido. Esses dois modelos de relés são constituídos por componentes que necessitam de baixa potência para operarem, porém possuem capacidade limitada na tolerância de temperaturas extremas e umidade.

Com o desenvolvimento da tecnologia digital, deu-se início ao desenvolvimento dos relés computadorizados ou digitais. Tal tipo de dispositivo é um relé gerenciado por um microprocessador específico, controlado por um *software*, onde os dados de entrada são digitais. Os princípios de funcionamento dos relés convencionais são uma referência para o seu desenvolvimento, desde que a entrada do relé consista em sinais de tensão e corrente provenientes do sistema elétrico. Portanto é necessário obter uma representação digital para esses sinais e, usando-se um algoritmo apropriado, a abertura dos disjuntores é conseguida.

Contudo, o que começou a existir foi um sistema elétrico de potência cheio de padrões, onde cada fabricante se comunicava somente com os seus próprios produtos, não tendo comunicação com o conjunto necessário. Diante desta difusão de padrões foi criada a Norma IEC 61850 [1]. O objetivo dessa norma foi estabelecer e definir ordem, uma padronização de comunicação, aos produtos do sistema de potência, visando a uma comunicação entre os elementos, independentes da origem de fabricação.

Os atuais sistemas elétricos de potência passam por uma série de mudanças com a norma IEC 61850. Em especial no Brasil, desde a chegada do processo de privatização das empresas de energia elétrica, o sistema elétrico passou a ser operado com menores reservas de geração e maiores congestionamentos na transmissão. Os sistemas de controle e proteção que hoje devem ser mais rápidos, mais confiáveis e melhores coordenados são expostos a atividades mais críticas diante destas mudanças. Para enfrentar esta nova perspectiva, novos métodos ou ações de controle e proteção tornaram-se necessários, por isso a importância da norma IEC 61850.

Os sistemas tradicionais de proteção se baseiam em unidades independentes que trabalham utilizando medidas locais para a maioria das tomadas de decisões. As redes de comunicação exercem um papel muito limitado nestes sistemas. Com o surgimento de uma nova era de controle descentralizado, associado a novos padrões de eficiência, estes métodos anteriores revelaram-se ineficientes em atender as novas exigências do sistema. Destes fatos, a indústria de potência começa a reconhecer os benefícios que a comunicação por meio de uma rede de computadores interligados pode trazer para uma melhor coordenação dos sistemas de proteção e controle, através de ações mais rápidas e precisas [2].

A expansão ou explosão da Internet que ocorreu na década de 90 ocasionou uma série de mudanças em muitas áreas, inclusive em SEP. Inicialmente induziu a um

grande interesse para o uso de redes de comunicação para as tarefas de controle e proteção de SEP. Atualmente, tem havido um grande interesse no uso de Agentes [3] (advindos da Inteligência Artificial) para trabalharem nas redes de comunicação para a melhoria da eficiência e confiabilidade dos SEP. Contudo, apesar desse crescente interesse, existem poucas oportunidades para investigação dos efeitos dos agentes em sistemas reais, devido principalmente à falta de uma plataforma computacional que possa simultaneamente simular elementos de um sistema de potência e de um sistema de comunicação. Para suprir essa deficiência foi desenvolvida então a plataforma de simulação EPOCHS (*Electric Power and Communication Synchronizing Simulator*) [4].

O EPOCHS foi desenvolvido como plataforma de simuladores de um SEP, atuando sobre uma rede de computadores, com todos os problemas de tráfego advindos de uma rede de comunicação para o sistema. Uma informação a destacar é que o *software* de segurança desenvolvido e a ser explanado nesse trabalho poderá ser aplicado sobre o EPOCHS.

1.2 Comunicação em um SEP por meio de uma Rede *Intranet*

O desenvolvimento de novas tecnologias através dos anos tem proporcionado um grande impacto econômico, e este, propiciando a continuidade de uma evolução sistemática das tecnologias. A capacidade de trocar informações aumentou extraordinariamente, sendo que um dos maiores ícones responsáveis por essa evolução na comunicação é o computador pessoal (PC). Hoje ele é tão comum quanto foi o telefone fixo a algumas décadas passadas.

O computador alcançou um rápido crescimento na capacidade de processamento, principalmente com o advento das redes de computadores [5]. Desta maneira, com toda

esta rápida mudança, há a necessidade de sistemas de comunicação confiáveis, gerenciáveis e flexíveis (escaláveis).

Na década de 80, as redes remotas (ou redes de longa distância), transportavam as informações principalmente entre terminais e *mainframes*. Durante esse período, tais redes de computadores tiveram uma intensa atividade de desenvolvimento. Na década de 90, as redes locais se tornaram o ambiente de trabalho por excelência desses novos tempos. A interligação de redes locais de computadores em um escritório permitiu um suporte a grupos de trabalho locais, enquanto que as redes [5] interligadas suportam todas as atividades da empresa.

É importante ressaltar os 03 (três) motivos que se escondem por trás do grande aumento na utilização e disponibilidade de recursos nas redes de computadores locais modernas [5]:

- **A concorrência:** as grandes empresas que fabricam *hardware* e *software* competem ferozmente para ganharem terreno no mercado de redes locais. Essa concorrência estimula a inovação e a qualidade técnica. As redes locais atuais são bastante flexíveis, expansíveis e interoperáveis.
- **Crescimento contínuo da capacidade do *Hardware* e *Software* do PC:** os computadores modernos possuem baixo custo quando comparados aos dos últimos 10 anos e estão baseados em processadores rápidos que possuem grande capacidade de processamento e de gerenciamento de memória.
- **Avanços na Arquitetura e na Execução de *Software* de Rede:** os programas processados em computadores clientes e servidores de rede utilizam uma combinação de código escrito, na sua grande maioria em linguagem C e linguagem Assembler, permitindo que os programadores

possam desenvolver novos recursos que podem ser facilmente acoplados as versões anteriores do *software*.

Atualmente existem como meios de interligação de computadores as LANS (*Local Area Network*), MANS (*Metropolitan Area Network*) e WANS (*Wide Area Network*) [5]. Conforme será caracterizado, este trabalho utiliza uma rede de comunicação baseada em uma *LAN* para um SEP.

1.3 Segurança em uma Rede de Comunicação *Intranet*

Com o passar dos tempos, a união de funcionalidades para um dispositivo como o computador, foi tornando-se uma peça comum e fundamental em residências e empresas. Um equipamento que antes era exclusivo de laboratórios em universidades tornou-se rapidamente um dos principais meios de comunicação do mundo.

As redes de comunicação baseadas em computadores favoreceram uma fascinante facilidade para a realização de atividades em um menor espaço de tempo, permitindo uma das maiores funcionalidades no mundo atual, “o compartilhamento de informações”.

Porém, a segurança dessas informações tornou-se alvo de muitos usuários não autorizados denominada de “intrusão”, que tanto podem se manifestar no âmbito externo como no interno de uma instituição, seja ela pública ou privada.

A realização de intrusão interna é na grande maioria das vezes advinda do seu próprio patrimônio funcional (funcionários) ou terceirizados (invasores internos), embora sejam eles em menor quantidade em relação á intrusão externa.

Diante dessas ameaças o administrador de segurança necessita constantemente se manter atualizado. O risco de sofrer um ataque cibernético nos dias atuais é muito elevado, mesmo que esse usuário não possua muita informação valiosa ou até mesmo

nenhuma informação. Nos casos em que o usuário ou a instituição não possua algo valioso a ser furtado, esse usuário poderá ser utilizado como meio pra atacar um outro usuário ou instituição mais importante.

Em se tratando de segurança em uma rede de comunicação por meio de uma rede de computadores, necessita-se de uma excelente equipe para realizar uma configuração segura, tanto ao nível de *hardware*, como de *software*, além de possuir um forte senso de ética pessoal e profissional.

Para ajudar os administradores de segurança de uma rede de computadores, foram criados diversos dispositivos capazes de detectar uma intrusão, sejam eles de *hardware* como também de *software*. Nesse trabalho, utiliza-se de dois sistemas de detecção de intrusão baseados em *software*, um baseado em abuso e outro baseado em anomalia. Os mesmos são respectivamente: o “Carcará”, baseado em anomalias e o *software* de detecção de intrusão SNORT [6], baseado em abuso, desenvolvido em linguagem de programação C++ [7]. Esses dois modos de atuação de Sistemas de Detecção de Intrusão (SDI), além de serem denominados de detecção por abuso e detecção por anomalia, também são chamados respectivamente de preemptivo e reacionário.

Nesta Tese de Doutorado o *software* SNORT foi configurado e adaptado para trabalhar em um SEP de subestações digitais e para isto foi criada uma rede de computadores para servir de testes de intrusão. Foi desenvolvida análises em UML do funcionamento do NS-2, do funcionamento do *software* EPOCHS e de sua base de dados, a definição das principais atividades de um operador de uma subestação digital e finalmente os casos de uso e diagramas do desenvolvimento do *software* “Carcará” na linguagem C++ para testes.

Em resumo, esta Tese de doutorado possui cinco objetivos:

- Demonstração da aplicação do *software* comercial SNORT em um SEP.
Para esta aplicação foi criado um ambiente que simulasse uma subestação digital, com o simulador EPOCHS interagindo com o NS-2, PSCAD/EMTDC, aplicação inédita em um SEP;
- Desenvolvimento e configuração de uma rede de computadores para realizarem defesas e tentativas de intrusão em um SEP;
- Análise das atividades dos operadores de uma subestação digital em UML a ser utilizado no *software* “Carcará”;
- Análise em UML do *software* de detecção por anomalia, denominado “Carcará” e suas principais atividades;
- Implementação do *software* de detecção por anomalia, denominado “Carcará” e os resultados alcançados;

Todos os cinco itens destacados acima, bem como os resultados em cada uma das etapas, são inéditos em SEP.

1.4 Organização da Tese

Devido ao caráter multidisciplinar deste trabalho, caracterizou-se na divisão dos assuntos, a necessidade de tornar a seqüência do trabalho o mais gradual possível, em seqüência e em desenvolvimento.

A abordagem do capítulo 2 foi direcionada à funcionalidade da plataforma de simulação EPOCHS. Sobre este simulador foi realizada a engenharia reversa funcional, utilizando a linguagem de modelagem unificada (UML), definindo e mostrando os principais componentes do EPOCHS e o relacionamento entre eles.

Os motivos da inserção do simulador EPOCHS se devem ao fato de que sobre este simulador é possível a utilização do *software* de perfil do usuário baseado em anomalia chamado “Carcará”.

No capítulo 3, é destacado o problema atual com relação à segurança em uma rede de comunicação que se utilizem computadores interligados, caracterizando desta maneira uma rede de computadores. Ataques locais, remotos e sistemas de detecção de intrusão e tipos de defesa são avaliados e definidos neste capítulo.

O capítulo 4 define o SNORT, o sistema de detecção de intrusão por abuso que possui código livre e poderá ser utilizado sobre os servidores em um SEP. Além disto, apresenta-se à estrutura e aplicações do SNORT. Este capítulo também especifica a metodologia a ser aplicada para a avaliação do *software* SNORT.

O capítulo 5 se dedica à criação de um perfil de usuário, conceito esse que aborda um tipo de Sistema de Detecção de Intrusão (SDI), por meio da engenharia social, bem como a estrutura do *software* que foi desenvolvido e que poderá ser utilizado no SEP para avaliar o perfil do operador e realizar a detecção de anomalias de atividades do usuário. Este *software* será de domínio público e denominado como “Carcará”. O Carcará terá o seu uso para projetistas de *software*, nas mais diversas aplicações, podendo ser utilizado em sistemas elétricos de potência ou não. A regra adotada foi deixar os parâmetros de entrada de informações o mais fracamente acoplado possível. Durante todo o desenvolvimento de análise até o desenvolvimento final será utilizada a metodologia *Dynamic CMM* [8], visando ganhar mais produtividade e eficiência no desenvolvimento desse projeto.

O capítulo 6 destaca os resultados da utilização do *software* SNORT (detecção por abuso), e do *software* “Carcará” (detecção por anomalia). No SNORT é destacado o tipo de ataques aplicados para que ele pudesse detectar atividades (Alertas) suspeitas na

rede de computadores. Cabe adiantar que os valores alcançados nos diversos tipos de ataques, aplicados em uma rede de computadores, foram satisfatórios. Os resultados dos testes com *software* “Carcará” foram favoráveis, tendo uma detecção de 100% em todas as atividades suspeitas (31 padrões de intrusão) que prontamente foram reconhecidos pelo mesmo.

Finalmente no capítulo 7 conclui-se essa etapa de trabalho, evidenciando o fechamento dos capítulos apresentados, bem como a projeção de trabalhos futuros que incluam o item segurança como fator determinante em um SEP.

2 ENGENHARIA REVERSA FUNCIONAL DO EPOCHS

UTILIZANDO A LINGUAGEM DE MODELAGEM

UNIFICADA (UML)

Este capítulo possui como objetivo mostrar a engenharia funcional do *software* EPOCHS e estudar a estrutura de *softwares* de simulação como o PSCAD/EMTDC [14] e o *Network Simulator* (NS-2) [15] por meio da *Unifield Modeling Language* (UML). Para gerenciar esses simuladores foi desenvolvido o EPOCHS. Esta é uma plataforma que incorpora os dois simuladores (PSCAD/EMTD e NS-2) e os sincroniza para alcançar os seus objetivos de simulação de um sistema elétrico sobre uma rede de computadores (*Intranet*), utilizando agentes [16] e os relés digitais do SEP. Por meio da união desses simuladores, as informações de tráfego da rede de computadores da *Intranet* podem ser analisadas para verificação de usuários não autorizados (intrusão).

2.1 Introdução

O integrador de simuladores *Electric Power Communication Synchronizing Simulator* (EPOCHS) é o resultado de pesquisas desenvolvidas pela universidade de *Cornell* (EUA) em parceria com a Escola de Engenharia de São Carlos, Departamento de Engenharia Elétrica da Universidade de São Paulo (USP).

O EPOCHS é uma plataforma que permite reunir dois grandes simuladores (Figura 01), o simulador denominado PSCAD/EMTDC, desenvolvido pelo centro de pesquisa Manitoba HVDC, e o simulador *Network Simulator* (NS-2) desenvolvido pela universidade de *Berkeley*.

Figura 01 - Estrutura do EPOCHS em forma de dependência do
UML

Na plataforma EPOCHS é possível realizar testes com diversos cenários (ambientes) de um SEP através do PSCAD/EMTDC, e a simulação de um ambiente de uma rede de comunicação baseada em computadores via NS-2.

Existem dois componentes do EPOCHS que realizam a harmonia e sincronismo dos dois simuladores (PSCAD/EMTDC e NS), são eles o *Agent Headquarter* (AgenteHQ) e o *Run-Time Infracstruture* (RTI) que serão abordados após os dois simuladores. A seguir, comenta-se do primeiro simulador, o NS-2.

2.2 Estrutura do *Network Simulator* (NS-2)

O *Network Simulator* é um *software* de simulação de redes de computadores, surgido no ano de 1989, com a sua primeira versão. Este *software* é advindo do *REAL Network Simulator*, e a partir de 1995 esse projeto de simulação de redes passou a ter o apoio do DARPA (*Defense Advanced Reseach Projects Agency*) por meio do projeto denominado VINT (*Virtual InterNetwork Testbed*), tendo como colaboradores os centros de pesquisa da UC Berkeley (*University of Califórnia Berckelely*), LBL (*Lawrence Berkeley National Laboratory*), USC/ISI (*University os Southern Califórnia / Information Sciences Institute*) e Xerox PARC (*Palo Alto Research Center*).

O NS-2 nos últimos anos passou a ser desenvolvido pela Universidade de Berkeley em C++ e OTcl (*Object TCL*) linguagem de script TCL com extensão

orientada a objeto desenvolvido no MIT (*Massachusetts Intitute of Tecnology*). A aplicação do NS é eficaz para a simulação de protocolos como o TCP (*Transmission Control Protocol*), protocolos de *multicast* em redes com ou sem fio e a simulação do roteamento de pacotes, etc. O NS possui resultados excelentes como simulador de redes locais (*LANs*) e mundiais (*WANs*).

O NS-2 é um simulador de redes que trabalha por meio de eventos que pode realizar a simulação de vários tipos de redes IP (*Internet Protocol*). O NS pode também trabalhar simulando vários protocolos de rede como, por exemplo: TCP,UDP (*User datagram Protocol*), FTP (*File Transfer Protocol*), Telnet, Web, CBR,VBR, *Drop Tail* (técnicas de gerenciamento de filas de roteadores), RED, CBQ e algoritmos de roteamento.

O NS-2 implementa *multicasting* e alguns dos protocolos da camada MAC para simulação de LAN. O projeto VINT, no qual o NS-2 está inserido, atualmente desenvolve ferramentas para a visualização de resultados de simulações, análises e conversores que possam converter topologias de redes geradas por outros *software* para o formato NS-2. A Figura 02 mostra a realização de tarefas por meio do NS.

Figura 02 - Visão Geral do NS2

Tendo como referência a figura 02, um exemplo prático e ilustrativo, seria um “Usuário” realizar um determinado planejamento e executar simulações em TCL [17], utilizando os objetos das bibliotecas OTcl. Essa agenda de eventos (seqüência de ações) e grande parte dos componentes de rede são desenvolvidos em C++ e disponíveis em OTcl pela ligação Tccl. Em outras palavras este “Usuário” poderá trabalhar com “os componentes da rede” e com a “agenda de eventos”.

Quando ocorre uma simulação por meio do NS-2, é produzido um arquivo contendo os resultados obtidos em formato de texto que podem conter dados da simulação de forma detalhada. Essas informações podem ser utilizadas para análise das simulações.

A ferramenta que produz uma saída gráfica é o NAM (*Network Animator*) que faz parte do projeto VINT, em que o NS-2 está inserido. O NAM possui uma interface gráfica intuitiva e amigável. Entre os seus principais recursos está um controlador de velocidade para a simulação.

Conforme a visualização simplificada do NS-2, por meio da Figura 03 desenvolvida em UML [18], pode-se observar que o NS-2 é na verdade um interpretador de *script* Tcl (OTcl) orientado a objeto, adicionado a uma biblioteca. Esta biblioteca contém objetos por meio da “agenda de eventos”. Tais objetos são “componentes da rede”, módulos de ajuda e especialmente de configuração de rede. Desta maneira, para utilizar o NS-2, a programação é feita em OTcl.

Figura 03 - Visão Geral do NS em UML

Por exemplo, para a realização da configuração e simulação de uma rede, o usuário precisará:

- escrever um *script* Otcl (“agenda de evento”);
- construir um novo objeto de rede de simulação (construir ou usar da biblioteca) e
- ligar o caminho de dados ao objeto.

Passo a passo a sequência seria:

- O usuário realiza um script em OTCL;
- O script em OTCL gerencia essas informações e os envia para o NS;
- O NS realiza a “agenda de eventos”, verifica quais os componentes da rede e módulos de ajuda serão necessários;
- O “resultado dessa simulação” com estes módulos podem ser utilizados pelo simulador NAM;

- Sendo que para ocorrer esse evento no NS é necessário um identificador (ID), tendo o tempo de escalonamento e ponteiro para o objeto de gerenciamento do evento.

No NS-2, uma “agenda de eventos”, sabendo o tempo de escalonamento do objeto, coloca esse objeto na fila, fazendo a chamada aos “componentes da rede” necessários no tempo pré-definido.

A comunicação dos “componentes da rede” se realiza por passagem de pacotes, contudo isto não interfere no tempo de simulação real. Com isto os “componentes da rede” que necessitarem gastar mais tempo de simulação, vão utilizar a “agenda de eventos”.

Por razões de eficiência, o NS-2 faz uma separação das implementações do caminho de dados, do caminho de controle. Isto é realizado para redução de pacotes e do tempo de processamento do evento.

Estes objetos compilados são disponibilizados para o interpretador de OTcl, que realiza uma correspondência do objeto OTcl para cada um dos objetos C++.

O objeto em C++ que não for controlado na simulação por um outro objeto, não precisa ser ligado ao OTcl. A Figura 04 mostra uma visão geral da estrutura da OTCL em UML.

Figura 04 - Visão da OTCL em UML

Uma visão estruturada de como funciona o NS-2, pode ser vista na Figura 05. Nesta pode ser observado o pacote de instalação *Ns-allinone-2* e seus respectivos pacotes hierárquicos.

As ações básicas ocorrem no pacote NS-2, sendo posteriormente criado um código similar em Otcl, tendo exemplos e testes para que a simulação possa ser validada.

Cada um dos pacotes adicionais permite fazer novas configurações, destacando que pacotes adicionais são normalmente disponibilizados via *web*, permitindo ao usuário, fazer novas experimentações em simulação de redes.

Figura 05 - Visão dos Pacotes do NS

A seguir será apresentado o segundo simulador do EPOCHS, denominado PSCAD/EMTDC, destinado às simulações do SEP. Este *software* é um dos mais sofisticados *softwares* na simulação de SEP.

2.3 Estrutura do PSCAD

O PSCAD é uma poderosa interface gráfica para o usuário e o EMTDC um simulador de projetos de engenharia elétrica. Ambas as ferramentas foram desenvolvidas pelo centro de pesquisa MANITOBA HVDC em 1976 [19].

O PSCAD possui várias bibliotecas que permitem o usuário manipular uma quantidade considerável de informações que podem ser utilizadas para criar muitas configurações de projetos elétricos. Destaca-se que na ausência de algum componente indispensável ao projeto elétrico, o usuário possui livre arbítrio para criar as suas bibliotecas. Para esta tarefa, o usuário poderá utilizar as linguagens de programação Fortran, C e C++.

Esse *software* permite a construção de circuitos, realização de simulação de linhas de força e cabos, máquinas rotativas, faltas assimétricas, sistemas eletrônicos de energia e *drives*, permitindo a análise dos resultados, bem como, um gerenciamento dos dados com uma completa integração com o ambiente gráfico.

Nos parágrafos seguintes, será utilizada a sigla PSCAD para referenciar o *software* PSCAD\EMTDC em evidência.

A Figura 06 ilustra o acesso do usuário aos componentes da “biblioteca” do PSCAD quando da formatação de um projeto. Após a disponibilização dos componentes da biblioteca, faz-se o acoplamento destes ao projeto que posteriormente é mostrado ao usuário (Visualizador do PSCAD).

Figura 06 - Visão do PSCAD em UML

A Figura 07 mostra um diagrama de estado para a criação de um projeto no PSCAD. Inicialmente o usuário escolhe o tipo de projeto a ser realizado, depois realiza a seleção da biblioteca a ser utilizada para a construção do projeto. Realizada a escolha

da biblioteca ou das bibliotecas, o projeto propriamente se inicia. Tem-se então a execução e, posteriormente, a conclusão do projeto.

Figura 07 - Diagrama de Estado para criar um Projeto em UML

Como foi utilizada a UML para o desenvolvimento do diagrama de estado (Figura 07), a primeira circunferência de cor preta no topo da figura representa o “início do estado” e a circunferência o “final do estado”, o lado direito da figura com uma circunferência maior sobrepondo a primeira circunferência, representa o “final do estado”.

No próximo item o destaque será dado para os componentes internos do EPOCHS, os componentes que realizam a ligação do NS-2 e do PSCAD.

2.4 EPOCHS

2.4.1 Trabalhos relacionados

Existem outros sistemas que combinam simuladores em grupos, que usam uma RTI como a utilizada pelo EPOCHS. Gary Bundy da MITRE *Corporation* foi o autor de uma das primeiras publicações a combinarem simuladores, criando uma plataforma de simulação de aeronaves de combate composta pelos simuladores AWSIM [20] (figura 08) e ModSAF (*Modular Semi-Automated Forces*) [21] (figura 09). A interação e

sincronização dos modelos de aeronaves foram realizadas por meio do gerenciamento conhecido como *ALSP (Aggregate Level Simulation Protocol)*. A principal motivação do projeto foi contemplar a plataforma com a união de detalhes e diferenças distintas a cada um dos simuladores (AWSIM e ModSAF). Foi verificado que a maneira mais econômica para incluir a maioria das funcionalidades dos dois, seria a junção dos simuladores. O produto resultante não foi totalmente interoperável, mas ajustes foram introduzidos para as mais importantes funcionalidades, fazendo do produto um sucesso.

Figura 08 - Simulador AWSIM

Figura 09 - Simulador ModSAF

O EPRI (*Electric Power Research Institute*) financiou uma iniciativa conjunta entre a Honeywell e a Universidade de Minnesota para criar o simulador SEPIA (*Simulator for Electric Power Industry Agents*). O SEPIA permite a pesquisadores analisarem os efeitos dos agentes quando usados em mercados de energia dentro de um ambiente desregulamentado. O projeto permite a investigação de complexos cenários de agentes e inclui uma interface intuitiva. A comunicação de rede não foi incluída nos parâmetros simulados.

2.4.2 Componentes do EPOCHS

Como dito anteriormente a plataforma EPOCHS trabalha com o PSCAD (simulação de transitórios eletromagnéticos) e o NS-2 (simulações de rede). Além de utilizar as ações desses simuladores possui os seus componentes próprios, o AgentHQ e o RTI (definidos nos próximos tópicos).

No entanto antes de falar mais detalhadamente sobre o AgentHQ é necessário definir o que é agente e sua classificação. Existem diversas definições de agentes, destacaremos 4 delas:

O Agente MuBot [22]: *“O termo agente é usado para representar dois conceitos ortogonais. O primeiro é a habilidade de execução autônoma. A segunda é a habilidade de desempenhar raciocínio orientado à domínio”*. Esta definição vem de um artigo de Sankar Virdhagriswaran da Crystaliz, Inc., onde define-se a tecnologia de agentes móveis. A execução autônoma é claramente vital para um agente.

O Agente de Maes [23]: *“Agentes autônomos são sistemas computacionais que vivem em algum meio dinâmico complexo, sentem e atuam autonomamente neste meio, e fazendo isto atingem uma série de objetivos e tarefas para as quais foram designado”*. Pattie Maes, do *Media Lab* do *Massachusetts Institute of Technology*, é uma das pioneiras da pesquisa em agentes. Ela acrescentou um elemento crucial à definição de agente: “agentes devem agir autonomamente para atingir uma série de objetivos e tarefas”.

O Agente de Hayes-Roth [24]: *“Agentes inteligentes desempenham continuamente três funções: percepção das condições dinâmicas do ambiente; atuação nas condições do ambiente; e raciocínio para interpretar as percepções, resolver problemas, fazer inferências e determinar ações”*. Bárbara Hayes-Roth do Laboratório de Sistemas de Conhecimento de Stanford insiste que os agentes devem raciocinar durante o processo de atuação. Se o raciocínio for interpretado amplamente, a arquitetura do agente permitirá ações reflexivas, assim como ações planejadas.

O Agente de Wooldridge & Jennings [16]: *“Um agente consiste em um hardware ou (mais usualmente) um programa que possui as seguintes propriedades:*

- *Autonomia: agentes operam sem a intervenção direta de humanos ou outros, e possuem algum tipo de controle sobre as suas ações e estados internos;*
- *Habilidade Social: agentes interagem com outros agentes (e possivelmente humanos) através de algum tipo de linguagem para comunicação de agentes;*
- *Reatividade: agentes percebem o seu ambiente, (que pode ser o mundo físico, a interface gráfica com usuário, um conjunto de outros agentes, a Internet, ou talvez todos estes combinados), e respondem prontamente a mudanças que neste possam ocorrer;*
- *Pró-atividade: agentes simplesmente não agem em resposta ao ambiente, eles são capazes de exibir um comportamento orientado a objetivos através da tomada de iniciativa.”*

A definição de Wooldridge e Jennings, em adição à autonomia, sensoriamento e atuação, permite uma grande, porém finita, gama de meios. Eles ainda acrescentam a necessidade de comunicação. Diante dessas definições, podemos realizar um resumo dos tipos de agentes (tabela 10).

Tabela 10 – Classificação dos agentes

	Outros nomes	Significado
Reatividade	Sensoriamento e atuação	Responde prontamente a mudanças no meio
Autonomia		Possui controle sob suas ações
Orientação a objetivos	Pró-atividade	Não age simplesmente em resposta ao ambiente
Continuidade temporal		É um processo contínuo ao longo do tempo
Inteligência	Adaptatividade	Muda o seu comportamento baseado em experiências anteriores
Mobilidade		Capaz de se transportar de uma máquina para outra
Flexibilidade		Suas ações não são predeterminadas
Caráter		Possui estado emocional e personalidade

Realizada as principais definições sobre os agentes, podemos iniciar o tema sobre um dos componentes do EPOCHS, o AgentHQ (*Agent Headquarter*). O AgentHQ possui a função de mostrar uma única visão de todos os simuladores para todos os componentes que estão envolvidos no processo de simulação. Este componente funciona como se fosse um procurador (*proxy*) para qualquer processo que ocorra durante a comunicação entre os simuladores (PSCAD e NS) e os agentes artificiais do EPOCHS.

O segundo componente, o RTI (*Run-Time Infrastructure*), possui a responsabilidade de manter consistente a sincronização entre todos os simuladores e pelo correto envio em termos de caminhos ou rotas (roteamento) das mensagens entre todos esses componentes (Figura 11).

Figura 11 - Mecanismo de funcionamento do EPOCHS

Na estrutura final do EPOCHS (Figura 12), alguns protocolos de comunicação ficam embutidos no interior do NS-2, pois foram escritos por meio da linguagem OTCL, fazendo o RTI exercer a sua função de mediador entre o NS-2 e o PSCAD. As informações são enviadas e recebidas pelo RTI que possui a importante função de sincronizar essas informações.

Figura 12 - Estruturação Funcional do EPOCHS

Na última versão do EPOCHS, o NS-2, RTI, AgentHQ e todos os seus agentes estão combinados em um único arquivo executável, sendo que cada componente é logicamente separado dentro do código-fonte e a RTI é implementada, como mencionado anteriormente, em um protocolo no interior do NS2. As razões para tal modo de implementação induzem à melhoria no desempenho do EPOCHS.

2.5 Considerações Finais

O capítulo teve como objetivo mostrar a plataforma EPOCHS, suas principais ligações, seus componentes, formato de programação dos simuladores e descrição em formato UML de seus principais componentes.

A plataforma EPOCHS é o resultado de pesquisas desenvolvidas pela universidade de *Cornell* (EUA) e a Universidade de São Paulo (USP), campus de São Carlos. A plataforma EPOCHS utiliza o PSCAD e o NS-2.

O simulador NS-2 possui uma enorme escalabilidade, tendo como centro de desenvolvimento e melhorias, uma das melhores universidades do mundo em redes de computadores, a Universidade de Berkeley.

O segundo simulador da plataforma EPOCHS, o PSCAD possui as melhores ferramentas de simulação para um sistema elétrico, considerado academicamente e profissionalmente o melhor simulador de sistemas elétricos.

Cabe frisar que o EPOCHS é uma plataforma que poderá ser utilizada para o *software* “Carcará”, o qual realiza a detecção de intrusão mediante anomalias.

No próximo capítulo estudaremos os tipos de ataques via rede de computadores, bem como os princípios de defesa. Princípios estes que podem ser utilizados em um Sistema Elétrico de Potência que utiliza como Rede de Comunicação os Microcomputadores.

3 SISTEMAS DE SEGURANÇA EM UMA REDE DE COMPUTADORES

Neste capítulo, o objetivo é mostrar os riscos existentes em computadores interligados como meio de comunicação. Investigam-se dos ataques locais até os ataques à distância (remotos).

É estudado os principais tipos de defesa para os administradores de uma rede de computadores, sendo em seguida mostrado um sistema de detecção de intrusão, que juntamente com outras ferramentas como antivírus, *firewall* etc., podem oferecer uma boa margem de segurança para a estrutura a ser protegida.

3.1 Introdução

A palavra “Segurança” pode ser definida de várias maneiras, dependendo de sua aplicação. Para os computadores, o termo “Segurança” pode ser definido como a forma de manter as informações confiáveis (sem inverdades) e sem o acesso de usuários não autorizados.

Para manter essa especificação de segurança, é necessária uma política de segurança, um envolvimento de toda a instituição para manter a segurança da empresa, o que inclui uma constante atualização e inspeção de todos os itens envolvidos com a segurança da rede de computadores para evitar invasores.

Os invasores que realizam esses ataques a redes de computadores são denominados de *hackers*. Embora o significado original para hacker tivesse a conotação de aficionado por computadores, a imprensa acabou dissipando uma outra conotação, uma imagem de invasor sem escrúpulos.

Existe uma palavra que identifica aqueles que quebram sistemas e realizam danos, esses são chamados de *crackers*, mas o termo *hacker* dominou os meios de comunicação, tomando um caminho sem volta.

O mundo atual vive cercado de máquinas que podem ser suscetíveis a ataques de diversos modos, o que ocorre por causa da automação e das interligações entre todos os serviços. Procurou-se velocidade, simplicidade, facilidade, menor custo, maior abrangência, melhores serviços, mas infelizmente faltou segurança na transmissão dessas informações.

No momento, existe um órgão da Universidade de *Carnegie Mellon*, denominado de CERT (*Computer Emergency Response Team*) [25]) dedicado a manter atualizados os administradores de redes de computadores com respeito a deficiências de *software* e meios de remediá-los, bem como os novos tipos de ataques a redes de computadores. A figura 13 mostra a quantidade de incidentes relacionados a invasões desde o ano de 1999 até dezembro de 2006. Observa-se que a cada ano os números aumentam vertiginosamente.

Valores acumulados: 1999 a 2006

Figura 13 - Incidentes em redes de Computadores (fonte CERT[25])

Essas informações servem de preparação e preocupação para os administradores de uma rede de computadores. Tais informações mostram exatamente como as invasões estão em constante aumento e que mais e mais pessoas se conectam à internet e possuem cartões de crédito e conta bancária.

Para os administradores de uma rede de computadores, haverá necessidade de uma constante atualização das defesas contra o inimigo, sendo que para isso normalmente é necessário também saber utilizar as próprias armas do inimigo (*cracker*).

A figura 14 mostra um gráfico dos principais tipos de golpes utilizados na Internet, sendo os mais utilizados a fraude e varreduras (*scans*). Tais atividades possuem forte tendência a continuar crescendo.

É interessante observar que nem todas as tentativas de invasão são enviadas ao CERT (nacional ou mundial), isto é, os valores reais dessas informações podem ser bem mais assustadores.

A razão dessas entidades não enviarem essas informações é basicamente o risco de expor a imagem da entidade aos acionistas, resultando em queda financeira e possível falência de seus empreendimentos.

Incidentes Reportados ao CERT.br -- Janeiro a Dezembro de 2006

Figura 14 – Tipos de incidentes em uma rede de computadores (fonte CERT[25])

A figura 15 mostra os protocolos e portas utilizadas para realizar invasões em uma rede de computadores. Portanto um invasor necessita saber que porta abrir e como abri-la. Isto torna o trabalho de invasão completo.

O que um bom administrador necessita fazer para complementar a segurança de seu sistema a ser protegido é fechar as portas que não são utilizadas, evitando assim surpresas desagradáveis.

Por exemplo, utilizando portas TCP, UDP e ICMP e um *software* adequado, e encontrando um microcomputador desprotegido, podem ser realizadas, desde um

simples scanner (varredura) a ações mais destrutivas, como o furto de informações, fraude utilizando cartões bancários, cartões de crédito e assim por diante.

Deverá haver ações em que todos os microcomputadores deverão estar devidamente protegidos e não somente os servidores, pois se caso algo não der certo o microcomputador individualmente poderá possuir defesas, mesmo com poucos recursos.

Incidentes Reportados ao CERT.br -- Janeiro a Dezembro de 2006

Figura 15 – Tipos de protocolos e portas mais utilizados para invasão (fonte CERT[25])

A figura 16 mostra o dia de segunda-feira, o dia da semana em que ocorrem as maiores quantidades de ataques de invasão. Uma explicação é que sendo o primeiro dia útil para as empresas, o ataque tornará o dia uma imensa confusão (muita perda de produtividade na empresa). Tais ataques podem ser advindos de vírus ou de outros *softwares* de ataque.

O segundo dia da semana mais visado é a quarta-feira, o que ocorre na metade da semana de dia útil; em terceiro lugar a sexta-feira, dia em que os funcionários estão terminando as suas atividades da semana.

Incidentes Reportados ao CERT.br -- Janeiro a Dezembro de 2006

Figura 16 – Dia da semana em que ocorrem mais incidentes (fonte CERT)

A figura 17 possui como foco os países e a quantidade de ataques de invasão que eles sofrem. Em primeiro lugar estão os Estados Unidos. Alguns explicam que uma das possíveis razões para esse primeiro lugar seja justamente a sua própria política arbitrária. Outros afirmam que o fato de os Estados Unidos da América serem a atual potência mundial, isso os colocaria como o alvo a ser destruído, o inimigo a ser derrubado.

Em segundo vem o Brasil, país da mais moderna democracia, mas que não possui uma grande quantidade de pessoal altamente capacitado para trabalhar com a segurança neste nível (em se tratando de governo).

Mas independente de quem é a culpa por haver esses números tão alarmantes é importante ver que procedimentos simples podem reverter esses gráficos.

Procedimentos estes que estão nas normas de segurança. Ações que podem ajudar o administrador de segurança. Normas de segurança que deverão ser obedecidas por toda a hierarquia da empresa.

Figura 17 – Países e a quantidade de ataques (fonte CERT)

A seguir os principais tipos de ataque, utilizados por *crackers*.

3.2 Tipos de ataques em uma Rede de Computadores

Os ataques que podem ser realizados em uma rede de comunicação baseada em computadores podem ser dos seguintes modos:

- Ataques não técnicos;
- Ataques locais;
- Ataques remotos.

Podem ser executados separadamente, ou em conjunto por meio de um ataque minuciosamente coordenado.

3.2.1 Ataques não técnicos

São os ataques nos quais os invasores na sua grande maioria não possuem grande conhecimento técnico. Tais invasores utilizam artifícios de engenharia social e segurança física para conseguir seus objetivos.

No entanto um ataque não técnico pode significar também um ataque de grandes proporções, pois pode ser executado por atacantes com alto grau de conhecimento técnico e que realizam esse ataque para conhecer melhor a vítima, sendo o seu objetivo o de realizar uma coleta de dados.

a) Engenharia Social

Uma técnica simples, mas que possui uma enorme eficiência quando bem aplicada, pois é específico no emocional e comportamental, realiza um estudo do indivíduo para depois atacar. Essa técnica não requer altos conhecimentos, basta ter poder de convencimento e um pouco de psicologia comportamental. O trabalho estudado por meio do *software* “Carcará” (capítulo 5) trabalha em parte com esse tipo de ataque.

O objetivo dos ataques que utilizam engenharia social é o de obter informações que sejam confidenciais da empresa e que possam ser utilizadas por criminosos. Tais informações podem chegar a tais pessoas mediante os próprios usuários da instituição e seus colaboradores. Observando que essas informações são obtidas mediante a confiança ou ingenuidade dos usuários, esses ataques podem ser concretizados mediante telefonemas, envio de mensagens via correio, salas de papo e até mesmo pessoalmente.

Esses ataques podem ser dos mais variados, desde uma simples ligação telefônica, por exemplo:

Invasor: pergunta por alguém que inexistente no local da empresa;

Vítima: nega a existência do funcionário com o nome mencionado;

Invasor: pergunta ao recebedor da chamada, por favor, Senhor, desculpe-me qual é o seu nome?;

Vítima: identifica-se;

Invasor: pergunta, qual é o nome desse setor Sr. X ?;

Vítima: fala o nome do setor;

Invasor: fala, o funcionário que estou procurando me disse que trabalhava no setor X, me desculpe, anotei alguma coisa errada, obrigado. Tchau.

O primeiro passo do ataque foi realizado com sucesso. O invasor tem em mãos o nome de alguém que pode ser atacado na instituição alvo. Agora o alvo está mais claro para o atacante, com o aumento de opções para o ataque.

Mas em algumas instituições não é necessária essa forma de ataque, basta ir à recepção da empresa que os nomes de funcionários e atribuições estão todos lá. É necessário apenas anotar alguns nomes e telefones e saber exatamente a hierarquia da empresa para que o ataque possa ser bem sucedido.

Quanto menor a hierarquia do funcionário, mas esse funcionário é prestativo em ajudar alguém que lhe solicita informações, não sabendo o funcionário que do outro da linha ou pessoalmente existe alguém com más intenções.

Outra situação de risco: o lixo da empresa. Os papéis de uma instituição que não forem úteis deverão ser picotados, antes de serem destinados ao lixo, pois do outro lado poderá haver alguém que poderia realizar uma busca no lixo da empresa e encontrar rascunhos de projetos, falhas do sistema, hierarquia da rede de computadores, hierarquia

de funcionários, descrição de problemas e possíveis senhas e tipos de acessos para a empresa.

Empresas terceirizadas também podem ser alvo, no caso de serem utilizadas visando um ataque maior, buscando meios de acessar a empresa alvo para a qual a empresa terceirizada presta serviços.

Em um ataque de engenharia social, o atacante busca informações da estrutura social da empresa (mercado, objetivo, tempo de existência), mapas ou plantas da estrutura física da localização da empresa (plantas baixas, plantas de engenharias, plantas de projetos de estrutura lógica de computadores) hierarquia da empresa (função e funcionários), hierarquia de computadores, tipo de segurança utilizada (física e de computadores), a estrutura pessoal dos funcionários alvos (relacionamentos amorosos, financeiros, pessoais, diversões, preferências, personalidade etc), quais as empresas colaboradoras da empresa (fornecedores e terceirizados).

Para dificultar esse tipo de ataque, algumas empresas adotam algumas medidas que podem ser bastante úteis para a manutenção de sua segurança. São elas:

- Uma política de controle de acesso físico à empresa, por meio de cartões de acesso, tanto pra funcionários como para visitantes. Uma câmera de vídeo na portaria ou recepção à vista de todo indivíduo que desejasse obter informações ou acesso à empresa, já inibe um pouco esse ataque, pois o atacante saberia que poderia ser posteriormente identificado.
- Classificação das informações, especificação do que cada funcionário pode dizer e para quem. Em outras palavras, o que pode ser divulgado e o que não pode.
- Realização de uma política de segurança sobre todas as informações da empresa, pois elas são seu maior patrimônio.

- A conscientização dos funcionários com respeito à segurança, por meio de seminários, casos de tipos de ataques, realização de cursos e simulações de ataques de engenharia reversa para comprovação dos treinamentos alcançados pela empresa.
- Uma política de senhas na qual nenhum funcionário deverá revelar a sua própria senha a nenhum outro funcionário, mesmo que trabalhe no mesmo setor.
- Criação de regras de Segurança para os Funcionários:
 - Sempre desconfie de ligações telefônicas de pessoas que possuam informações sobre a sua função e seu nome, e que possivelmente peçam informações que não esteja autorizado a fornecer. Nesses casos, peça o número de telefone para garantir a autenticidade da procedência, após isso realize o retorno para o número indicado ou disque para o setor de segurança da empresa para analisar o ocorrido.
 - Desconfie de e-mail de remetentes desconhecidos, no qual é convidado para seminários e cursos gratuitos, pois o objetivo final pode ser o de obter informações da sua empresa.
- A ocorrência de falsos fabricantes e falsos terceirizados é outro perigo de engenharia social. Diante desse tipo, o ideal é transferir ao pessoal responsável na empresa em atender e verificar se realmente as informações são verdadeiras ou não.

b) Segurança Física

Essa variação de um ataque não técnico ocorre quando o atacante necessita de mais informações que precisam ser retiradas diretamente no local. Tais informações

podem ser utilizadas para realizar o ataque direto ou posterior. Para realizar assim o ataque, as informações coletadas servirão para concretizar um ataque maior que necessita de grande investimento de tempo e de conhecimento técnico.

Para ter acesso físico à instituição (vítima) o atacante poderá utilizar crachás falsos, documentos pessoais falsos e demais recursos que lhe permitam acessar o local, como alguém devidamente legalizado e autorizado a ter acesso à instituição.

Sendo que neste tipo de ataque o maior problema fica normalmente concentrado em empresas terceirizadas que prestam serviços a grandes instituições. Estas instituições (bancárias, governamentais, de pesquisa ou até mesmo provedores de acesso etc.) contratam os serviços de empresas menores para realização de serviços como limpeza, segurança e até a central de processamento de dados da empresa. Estas grandes instituições por terem alto poder financeiro e também de informação, sofrem altos riscos de serem alvo de ataques dos mais diversos grupos de piratas virtuais. Os atacantes podem utilizar essas empresas terceirizadas como meio de alcançar e ter acesso ao local dessas empresas maiores.

3.2.2 Ataques Locais

Este ataque se concretiza quando o atacante encontra-se fisicamente no local a ser atacado. Tal ataque pode ser desferido por prestadores de serviço, estagiários, funcionários insatisfeitos, ex-funcionários com acesso físico ou virtual, sabotagem interna com fins políticos e até mesmo espionagem industrial. Esse tipo de ataque varia entre 20 a 80% dos ataques realizados a uma rede de comunicação baseada em computadores.

Esses ataques podem se concretizar por meio da utilização de algumas técnicas e ferramentas disponíveis na Internet, tais como os analisadores de tráfego (Sniffer),

forjamento de endereços, ataque do tipo homem no meio (man in the middle), quebras de senhas, ataques a bibliotecas (Libs), Rootkit e módulos de Kernel.

a) Analisadores de tráfego

São ferramentas que são utilizadas para realizar a captura de pacotes, para posterior análise. Originalmente essas ferramentas foram criadas para ajudar na administração ou gerenciamento de uma rede de computadores, mas também é utilizada por criminosos, para verificar atividades de usuários e descobrir falha em uma rede de computadores.

Essas ferramentas também são chamadas de Sniffers, pois realizam uma escuta no tráfego das informações, são semelhantes aos grampos realizados em telefones de indivíduos ou empresas que ficam sob investigação dos órgãos federais (suspeitas de crimes).

Como exemplo desses *softwares*, existe o Tcpdump ou Windump, que realizam esse serviço de análise de uma rede de computadores, sendo estes analisadores utilizados em quase 100% dos ataques para realização de coleta de senhas e informações sobre o ambiente local a ser invadido.

b) Ataque do tipo homem no meio (man in the middle)

Este tipo de ataque pode ser realizado remotamente. No entanto quando realizados em loco os recursos técnicos são bastante menores e as chances de obter sucesso são maiores. A figura 18 mostra como funciona esse tipo de ataque. Inicialmente o invasor tenta ficar entre o cliente e o servidor, desta forma ele tenta emitir um certificado falso para que a vítima possa ter a falsa ilusão de estar se comunicando com um serviço legítimo do outro lado da linha por assim dizer, esse tipo de intromissão pode ser realizado de ambos os lados (servidor ou para o cliente).

Figura 18 - Declaração de um Certificado Falso

A engenharia desse ataque é se interpor na comunicação entre um cliente e um servidor, quando que na verdade toda a comunicação estará sendo realizada por meio de uma máquina no “meio” (figura 19), que poderá forjar pacotes para ambos os lados ou simplesmente gerar tráfego. As chances de sucesso são grandes quando o invasor consegue se infiltrar na primeira conexão feita pelo cliente a um servidor, e por não ter havido nenhum vínculo anterior, este cliente poderá facilmente ser levado a crer que o servidor “do meio” é legítimo.

Figura 19 - Declaração de um Certificado Falso concluído

Tendo realizado uma conexão com o cliente servidor e tendo a conexão com o servidor fingindo-se de cliente, o ataque possivelmente será bem sucedido.

c) Quebra de Senha (*cracking*)

Existem diversas ferramentas que permitem a quebra de senha, sendo que a maioria utiliza dicionários e força bruta. Dicionários é uma base de dados composta normalmente de arquivos texto, com muitas palavras que podem ser testadas de inúmeras formas (maiúsculo, minúsculo, invertida etc.), dependendo exclusivamente do algoritmo aplicado. Estes dicionários são agrupados normalmente por idiomas, áreas, dialeto ou atividades (engenharia, biologia, economia etc.). Um exemplo desse tipo de ferramenta que utiliza dicionários e a força bruta é o *software* denominado “*Brutus*”.

Em termos de algoritmos para o desenvolvimento de *software* para obtenção de senhas, os mais utilizados são os algoritmos de *Hash*: MD4, MD5, *Blowfish* e DES.

d) Ataques a Bibliotecas (Libs)

Este tipo de ataque é sobre as bibliotecas dinâmicas. Isto ocorre por que normalmente os programas não carregam para a memória do computador todas as bibliotecas que irão necessitar durante o programa, chamando muitas delas somente quando forem necessárias. Nessa ocasião os programas carregam somente uma referência ou endereço de onde encontrar a informação (função), isto é realizado para economizar espaço na memória e no disco, mas o lado negativo é permitir que com esta ação uma alteração em uma biblioteca afete todos os programas dependentes.

e) Ataques por meio de *Rootkit*

Esses pacotes de programas denominados de rootkits são compostos de portas dos fundos (programas utilizados para realizar retorno de um servidor invadido), como cavalos de tróia que podem ser utilizados para substituir aplicativos do usuário. Esses programas também procuram se esconder apagando vestígios de sua presença na máquina vítima.

O processo ocorre da seguinte forma: assim que o atacante invade um equipamento e consegue privilégios de administrador, ele passa então a substituir os programas do sistema que poderiam identificá-lo, os programas substituídos são somente aqueles que interessam ao atacante. Em nenhum momento são substituídos todos os programas.

A possibilidade de utilização de *Backdoor* atualmente não é uso exclusivo do ambiente Unix (incluso o linux), sistemas operacionais Windows 95/98/NT e XP também podem ser vítimas, e são vulneráveis às inclusões de *Backdoors*. Tais programas são conhecidos como o *NETCAT*, *BackOrifice*, *NetBus* e *Master Paradise*.

Existem outros ataques que não serão assuntos desse trabalho como os módulos de *Kernel* (contêm as funcionalidades básicas para o funcionamento do sistema operacional), pois o atacante pode carregar um Kernel malicioso, com o propósito de esconder suas atividades das ferramentas de auditoria do sistema. A seguir os principais tipos de ataques remotos.

3.2.3 Ataques Remotos

Os ataques remotos são considerados os de maior risco, visto que o simples acesso ao servidor, via rede de computadores, já constitui uma séria ameaça ao sistema, se o usuário for alguém com objetivos maliciosos.

A dificuldade dos administradores de uma rede de computadores em detectar esse tipo de intrusão dá-se por que normalmente ele utiliza uma porta dos fundos ou *BlackDoor*, para instalar os programas com privilégios de *Root* ou que o farão se passar por um usuário legítimo. Além de eliminar os rastros deixados pela manipulação de arquivos (apagamento e instalação).

Os principais tipos de ataques remotos e ferramentas foram divididos em algumas categorias, são elas:

a) Análise do Alvo

Antes que qualquer ataque possa acontecer é realizada uma análise dos possíveis alvos, para que o atacante possa realizar a sua estratégia de ataque e assim verificar todas as possibilidades que ele (atacante) possui e qual a gravidade do ataque que poderá fazer.

Essa análise é realizada mediante a análise de portas (serviços) em uso, isso poderá ser feito, por exemplo, mediante a engenharia social (ataque local), até mesmo utilizando programas de busca como o *Google*, *Yahoo*, *Altavista Surf*, *Cadê*, *Miner* para verificar *softwares* que contenham correções e quais os problemas de segurança que eles possuem. O passo seguinte é descobrir se esses *softwares* com problemas estão em funcionamento na instituição alvo.

Isto ocorre pois, a questão da segurança é cada dia um sério problema para os desenvolvedores de *software*. E a cada dia estão mais comuns notícias de falha em CGI (*Common Gateway Interface*) e normalmente em conjunto com esta notícia, vem a forma de explorar essa falha. Uma forma bem simples e muitas vezes eficaz de conseguir essas informações é realizar uma busca pelo nome do CGI vulnerável em sites de busca.

Uma das primeiras situações a serem realizadas pelo invasor é justamente o mapeamento da rede para coleta de informações sobre particularidades do ambiente-alvo. Com isso é possível identificar a topologia, por exemplo, quais são os componentes da rede, qual o sistema operacional dos servidores e de máquinas clientes, quais os serviços estão ativos etc.

Uma ferramenta aparentemente simples que pode ser utilizada para realizar essa tarefa inicial é o PING, pois com ela é possível saber os sistemas operacionais utilizados pelo local alvo. Isto é possível por meio do campo TTL (tabela 1). Abaixo algumas especificações para se descobrir o tipo de sistema operacional mediante o uso da ferramenta PING.

Tabela 1 Sistemas Operacionais e seus campos TTL

Sistema Operacional	Campo TTL
Free/Net/OpenBSD	TTL 255
Linux	TTL 255
RedHat	TTL 64
Windows(95/98/NT/2000)	TTL 128
Roteadores (Bay/Cyclades)	TTL 30
Roteadores (Cisco)	TTL 255
Switches (3Com)	TTL 30
Impressoras HP (JetDirect/Laser Jet/etc)	TTL 60

Além da ferramenta PING e seus argumentos, podem ser utilizados outras ferramentas para verificar rotas, identificar serviços disponíveis em um equipamento remoto, bem como outras que mostram informações como tipo de produto, versão do produto etc. Na figura 20 foram utilizados somente o *Messenger* e o *Netstat* (no *Windows XP*) para descobrir o IP do outro lado da linha da rede de computadores, sendo que para isso foi solicitado um arquivo em anexo, tempo suficiente para se descobrir o IP.

Figura 20 - Amostra de identificação de IP usando o Messenger e Netstat

Abaixo algumas ferramentas para verificar rotas e um breve resumo do que cada uma dessas ferramentas é capaz de realizar.

Traceroute – existem muitas variações deste *software*, ele basicamente faz a identificação da rota entre dois equipamentos em rede.

Strobe - utilizada para identificar quais serviços estão disponíveis em um equipamento remoto. Este tipo de ataque é chamado de “ataque remoto”, pois não utiliza nenhuma espécie de técnica evasiva, sendo fácil de ser detectado por ferramentas de Detecção de Intrusão.

Glabb – visa obter informações dos servidores, isso é facilitado quando coloca nos servidores banners (informações) sobre o produto nele utilizado, a versão etc.

Nmap – é uma ferramenta para mapeamento remoto, com o objetivo de identificar o sistema operacional e a verificação de portas ativas com opções de técnicas tradicionais e evasivas, sendo que na maioria dessas ações são exigidos os privilégios de super usuário ou root.

Cprobe – esta ferramenta permite identificar remotamente determinado equipamento baseado somente em pacotes ICMP e um pacote UDP.

SATAN – desenvolvido em 1995, foi uma das primeiras ferramentas desenvolvidas para explorar as vulnerabilidades de uma rede.

Nessus – é uma ferramenta gratuita que realiza uma verificação remota de vulnerabilidades de concepção modular e com a filosofia cliente-servidor, pois mantém constantemente atualizado seu banco de dados de vulnerabilidades e ataques.

Existem também outras ferramentas bem interessantes como a *NetCat* e a *Nêmesis*, que não serão destacadas neste trabalho. Essas ferramentas também são capazes de trazer muita preocupação aos administradores de redes de computadores, pela sua capacidade de procurar falhas e portas ativas com muita facilidade.

b) Ataques em camadas

Este tipo de ataque visa despistar e dificultar a origem do ataque. Nesse tipo de ataque o invasor utiliza vários servidores (normalmente em até 3 camadas) para se esconder de uma futura busca pela rota do invasor. Nestes casos o local do possível invasor também é vítima, e isto pode conter uma lista de locais que foram manipulados, dificultando o trabalho de verdadeira origem do ataque, incapacidade em muitos casos encontrar o verdadeiro local do invasor.

O principal engano dos administradores é pensar que por sua rede não conter informações importantes que possam servir de atrativos para um invasor, negligenciam a segurança de sua rede.

No entanto, apesar dos obstáculos, qualquer ataque pode ser rastreado até a sua origem, sendo que o invasor ganha tempo devido à dificuldade de mobilização dos administradores de rede em encontrar a verdadeira origem do ataque.

Com este ataque assim como outros tipos o invasor poderá instalar cavalos de tróia e portas dos fundos, para analisar o comportamento da rede de computadores a ser invadida, como também obter acesso remoto a um equipamento (servidor, estação de trabalho, roteador, servidor de terminal etc.).

Além do ataque em camadas existem os ataques a serviços, como ataques a servidores na *web*, ataques à segurança em CGI (*Common Gateway Interface*). A aplicação de métodos PUT e POST, aplicação de negação de serviços, seqüestros de sessão e estouro de pilha (*buffer overflow*).

c) Fraudes e Falsificações

Muitas fraudes são possíveis com a atual tecnologia, desde a falsificação de e-mails, sites e até mesmo a falsificação do endereço IP de outro equipamento. Por exemplo, em termos de falsificação de e-mail pode-se falsificar sem muitas dificuldades um e-mail, destacando o remetente (*From*) do e-mail e o destinatário (*To*) e o assunto desse e-mail. O trabalho se resume a falsificar o cabeçalho do e-mail, esta técnica é muito utilizada por *SPAM* por meio de relay. O relay utiliza-se de uma má configuração em servidores de correio eletrônico, que permitem o envio de mensagens para domínios não locais, fazendo no entanto parecer que a mensagem foi enviada pelo domínio do servidor que permitiu o relay.

Mas por mais que se queira dificultar a identificação da origem do e-mail, grande parte das informações sobre a origem é descrita no cabeçalho. Em alguns casos é necessária uma investigação mais minuciosa, mas quase sempre é possível identificar a origem, algumas vezes dificultada pelo uso de *Middle on the Man* (homen no meio) ou seqüestro de sessão.

3.3 Tipos de Defesa em uma Rede de Computadores

Diante de tantas situações de ataque a uma rede de computadores, aparentemente a instituição parece estar em desvantagem, mas existem muitos passos úteis que podem servir de ajuda no momento de realizar uma defesa básica para uma Rede de Computadores.

De uma forma geral, existem ferramentas que não podem faltar para a segurança de redes, sendo elas:

a) Ferramentas de análise de Tráfego

Exemplo: *Etherman*, *netlog*, *tcpdump*, *synsniff*, *tocsin*, *clog*, *NOCOL* e *NFR*.

b) Ferramentas de Autorização e acesso remoto

Exemplo: *RADIUS*, *TACACS+*, *SSL*, *SSH* e *Kerberos*.

c) Ferramentas de Criptografia

Exemplo: *md5*, *md5check*, *PGP*, *rpem* e *UFC-crypt*.

d) Ferramentas de Gerenciamento do Sistema

Exemplo: *crack*, *localmail*, *smrsh*, *logdaemon*, *npasswd*, *op*, *passwd+*, *S4-Kit*, *Sfingerd*, *sudo*, *swatch*, *watcher*, *wuftp* e *LPRng*.

e) Ferramentas de Firewall, Filtros e Proxy

Exemplo: *fwtk*, *ipfilter*, *ipfirewall*, *portmap*, *v3*, *SOCKS*, *tcp_wrappers* e *smapped*.

f) Ferramentas de Monitoramento do Sistema

Exemplo: COPS, NSCARP, *crack*, *Tiger*, *Tripwire*, *logcheck* e *tklogger*.

g) Ferramenta de Monitoramento de Rede

Exemplo: *RealSecure*, ISS, *Satan* e *securscan*.

h) Ferramenta para Senhas

Exemplo: OPIE e S/Key.

Baseado nessas ferramentas de defesa criou-se uma lista de ações que podem amenizar a segurança de uma rede de computadores. Segue uma sequência de especificações que ajudam na criação de barreira aos invasores.

3.3.1 Atualizações e Correções

Em termos de atualizações e correções, é dever do administrador da rede de computadores, realizar constante verificação de *Bugs* em sistemas operacionais, *softwares* específicos da instituição etc. Nesta tarefa o administrador pode contar com empresas que pesquisam as falhas em *softwares* utilizados no dia a dia de uma empresa, bem como as correções para as falhas encontradas. Como exemplo de instituições que verificam o estado de segurança de vários *softwares* temos o CERT, *SecurityFocus*, *PacketStorm*, CORE, TESO Security, *AntiOnline* e *SecForum*.

3.3.2 Vírus

Dentre os tipos de invasões, os vírus são os campeões. Toda máquina, em rede ou não, nas residências ou instituições, em algum momento foram expostas a *softwares* maliciosos como os vírus de computador.

Existem vários tipos de programas tecnicamente distintos que são classificados como vírus. Desta forma, comportamentos diferentes acabam forçando os fabricantes de antivírus a incorporar novos métodos para a detecção de cavalos de tróia, vermes e outros.

O maior problema que as empresas que fabricam antivírus possuem é a forma de trabalho que conduzem, isto é, não pró-ativa. Essas empresas de *software* só trabalham diante de um novo vírus e da identificação do mesmo.

Outra questão também a ser levada em consideração é a dos administradores de uma rede de computadores ficarem expostos ao que se chama de “janela de vulnerabilidade”. Esta denominação é dada ao intervalo de tempo entre a identificação do vírus e a chegada da vacina por meio dos fabricantes de antivírus.

3.3.3 Códigos Seguros

Um dos grandes problemas hoje enfrentados pela indústria de desenvolvimento de *software* é a questão da segurança. Um exemplo disso são as atualizações que são constantes em programas de e-mails, sistemas operacionais, navegadores etc.

Os prazos e a demanda de novos recursos impedem que os *softwares* atuais sejam devidamente testados e assim seguros. Um problema muito comum é o estouro de pilha e tratamento de parâmetros. Isto se acentua, pois as equipes de desenvolvimento de *software* normalmente não prevêm todos os casos possíveis de problemas na segurança no *software*.

Quanto aos problemas de estouro de pilha, estão relacionadas a funções que não verificam a existência de espaço suficiente para o armazenamento da variável destino.

3.3.4 Segurança dos *Hosts*

Em termos de segurança de *Hosts*, essa etapa envolve muito mais do que aspectos genéricos, mas também serviços bem específicos. Algumas considerações deverão ser realizadas, como por exemplo, posicionamento no ambiente, acesso físico ao servidor, quais equipamentos devem ter acesso ao servidor, quais usuários, equipamentos e administradores podem utilizar ou acessar o servidor.

As permissões que os usuários poderão ter no servidor (tipo de leitura, gravação, remoção etc.) é responsabilidade única da política de segurança adotada pela empresa.

Por exemplo, em termos de serviços do servidor, todos os serviços desnecessários deverão ser desabilitados, sendo que a forma de desabilitá-los dependerá exclusivamente do sistema operacional em uso.

Quanto à padronização ou não de equipamentos ou *softwares*, isso vai depender muito da política de segurança da instituição. A padronização pode ter como benefícios o custo de manutenção, atualização, treinamento, perfil de administradores e demais funcionários técnicos. No entanto, a desvantagem é de que se for detectada alguma falha desse sistema, poderá haver comprometimento de toda a instalação. Isto poderá facilitar muito o trabalho de um invasor.

Quanto à aquisição de um ambiente heterogêneo, temos duas vantagens. A primeira delas relaciona-se com a vantagem de isolar determinada área se houver algum problema de segurança em alguma máquina ou serviço. Esses dispositivos poderão ser facilmente substituídos em um sistema que se encontra vulnerável. A segunda vantagem é uma questão não vinculada com segurança, mas com a performance de equipamentos para determinadas e específicas atividades da instituição. Por exemplo, alguns sistemas operacionais são melhores para atuar como servidores. Existem *softwares* que possuem

melhor desempenho para tarefas que exigem acesso rápido à rede, banco de dados que atuam melhor em determinados tipos de ambiente, etc.

3.4 Segurança em Rede

Um dos meios básicos de proteção contra intrusão em uma rede de computadores é a identificação desse usuário por meio de autenticação, bem como a criptografia dessas informações. Esses itens têm evoluído bastante nos últimos anos, especialmente a identificação digital [26]. No entanto, a área promissora será a leitura de retina em conjunto com a leitura digital. Vamos destacar nesse tópico de segurança em rede os dois modos principais, autenticação e criptografia.

a) Formas de Autenticação

O problema atual de um *software* é ter certeza de que o usuário seja realmente aquele que afirma ser. Para isso existem atualmente desde as senhas de acesso aos programas proprietários, em conjunto com os leitores digitais. Os leitores digitais são comuns em se tratando de acesso a academias, escolas e instituições que necessitam de maior segurança no seu acesso. No entanto essas formas de autenticação caminham para uma identificação de retina, e até mesmo faciais do usuário para prosseguir os seus serviços.

Algumas empresas como a *Microvision*, pesquisam a aplicação de leitura de retina (Biometria) não somente para identificação de usuários, mas também como meio de comunicação entre os usuários de uma instituição. Obviamente essa aplicação é somada às anteriores como senhas e leitura digital, dificultando com isso os meios de fraude.

b) Firewall

O *firewall* é um dispositivo que isola a rede local de sub-redes, outras redes, tipos de pacotes e Internet. Esse *software* pode funcionar em um computador ou *hardware* específico ou até mesmo, dependendo da situação, em um computador comum.

A sua utilização é de grande importância e até mesmo indispensável quando um servidor ou uma estação com arquivos compartilhados estiver ligado à Internet. Como exemplo de *Firewall*, temos os mais populares, como *Zone Alarme*, *Sygate Personal Firewall*, *Checkpoint Firewall*.

3.5 Sistemas de Detecção de Intrusão

3.5.1 Definição

Segundo Crosbie [27], uma intrusão pode ser definida como sendo “*um conjunto de ações que tentam comprometer a integridade, confidencialidade ou disponibilidade de recursos*” em um sistema computacional.

Um sistema de detecção de intrusão (SDI) realiza o processo de monitorar os eventos ocorridos em um sistema ou rede de computadores, analisando-os através de assinaturas de intrusão, ou desvio de padrão, em perfis de comportamento. As intrusões são causadas por usuários. Usuário, no contexto deste trabalho, deve ser entendido como um servidor, uma estação, um endereço ou faixa de endereço IP, um nome de domínio, ou simplesmente um indivíduo não autorizado tentando acessar um sistema através da Internet ou rede local e por usuários legítimos tentando abusar de seus privilégios, motivados pelos mais diversos valores de ganhos pessoais e financeiros. O conceito inicial sobre sistema de detecção de intrusão foi proposto pela primeira vez em 1980 por James [28].

3.5.2 Vantagens dos sistemas de Detecção de Intrusão

Os sistemas de detecção de intrusão possuem diversas características que lhes conferem papel importante dentro da infra-estrutura de segurança de uma organização. Em Zamboni [29], as seguintes características são identificadas como desejáveis para esses sistemas:

- Estar em execução contínua com a mínima supervisão humana;
- Ser tolerante a falhas;
- Resistir à subversão;
- Gerar o mínimo de *overhead* possível no ambiente onde está sendo executado, de forma a não interferir nas operações normais;
- Ter a habilidade de ser configurado de acordo com a política de segurança do ambiente que está sendo monitorado;
- Adaptar-se às mudanças do ambiente e dos comportamentos dos usuários através do tempo, uma vez que novas aplicações sendo instaladas, usuários trocando de atividade ou novos recursos sendo disponibilizados podem causar mudanças nos padrões de uso dos recursos do ambiente;
- Monitorar um grande número de *hosts* e providenciar resultados em tempo hábil e de maneira exata;
- Apresentar uma moderada degradação de serviço, no caso de algum componente do sistema ter seus serviços interrompidos por qualquer motivo;
- Permitir uma reconfiguração dinâmica, ou seja, não ser necessário reiniciar o SDI toda vez que um novo cenário for implementado.

As vantagens mais comuns dos SDI's, segundo Bace [30], são relatadas a seguir:

a) Detecção de ataques explorando vulnerabilidades que não podem ser corrigidas.

Através de vulnerabilidades conhecidas, que em alguns casos não podem ser corrigidas facilmente em tempo hábil, o invasor pode penetrar em muitas redes. A exemplo disso, existem os sistemas operacionais e *softwares* de navegação na Internet, onde constantemente ocorrem problemas envolvidos com a segurança.

b) Prevenir que intrusos examinem a rede.

Quando os intrusos atacam uma rede, eles geralmente o fazem em etapas. A primeira etapa é examinar o sistema em busca de alguma vulnerabilidade. Com um SDI, embora o iminente intruso possa continuar a analisar o sistema, ele será detectado e os devidos alertas ativados ou acionados. Sendo que a palavra “Alerta” é aplicada como sendo uma mensagem de que um evento foi detectado. Possuindo informação sobre a atividade que foi detectada, bem como as especificações das ocorrências (origem, destino, nome, tempo do evento, porta, serviço, informação de usuários e outros).

c) Documentação de ameaça de invasão.

É importante entender a frequência e as características dos ataques em um sistema computacional para medir o grau de hostilidade de um ambiente. Um SDI pode quantificar, caracterizar e verificar as ameaças de intrusão e mau uso, interna ou externamente.

d) Controle de qualidade

Com o uso contínuo de um SDI, padrões de utilização do sistema monitorado e problemas detectados tornam-se evidentes. Com essas informações, é possível tornar visíveis falhas no projeto, configuração e gerenciamento da segurança, proporcionado ao administrador a possibilidade de corrigi-las antes de ocorrer um incidente.

Essas vantagens sofrem variações de acordo com alguns aspectos básicos que devem ser considerados quando do desenvolvimento dos SDI's, tais como: os métodos de análises adotados, a escolha do tipo de monitoramento, o tempo entre a coleta e análise dos dados e o tipo de resposta a ser dada. Esses tópicos serão vistos com mais clareza nas seções seguintes.

3.6 Métodos de Análise de Detecção de Intrusão

Existem dois métodos principais de análise de detecção de intrusão, usados pelos SDI's para determinar a ocorrência de uma intrusão, conforme descritas em Bace [24]: a detecção por anomalia e a detecção por abuso.

3.6.1 Detecção por anomalia

O método de detecção de intrusão por anomalia é baseado na observação de desvios de comportamentos esperados em atividades. Essas atividades são denominadas de ocorrências detectadas como sendo de interesse do administrador, tais como: uma sessão de telnet inesperada relevantes dos usuários ou processos do sistema monitorado. Os detectores de anomalia constroem perfis (*profiles*) que representam o comportamento normal dos usuários, *hosts* ou conexões de rede. Esses perfis são gerados através de dados históricos coletados durante um período normal de operação. Os sistemas de detecção de intrusão por anomalia baseiam-se na premissa de que um ataque difere da atividade normal, podendo, dessa forma, ser identificado.

Porém, nem sempre a atividade anômala corresponde a uma atividade intrusiva. Por exemplo, um usuário que sempre trabalhou com determinada carga de uso da UCP pode necessitar de mais poder de processamento e nem por isso é uma intrusão. Em Kumar [31], são apresentadas as seguintes definições de comportamento:

- **intrusivo e anômalo:** a intrusão coincide com a anormalidade. São os verdadeiros positivos;
- **não intrusivo e não anômalo:** não há intrusão, nem anormalidade. São verdadeiros negativos.
- **intrusivo mas não anômalo:** há intrusão sem que haja comportamento anormal. São os falsos negativos.
- **não intrusivo mas anômalo:** não há intrusão, mas há comportamento anormal. São os chamados falsos positivos.

A ocorrência de um grande número de falsos negativos e falsos positivos pode se tornar um problema dos SDI's baseados em detecção por anomalia. Para reduzi-los, é preciso a definição cuidadosa dos limites que apontam a anormalidade.

Para sustentar a implementação dos SDI's baseados nesse tipo de detecção, vários métodos foram propostos. As abordagens mais comuns são: o método estatístico, o gerador de prognósticos de padrões e o uso de redes neurais.

No **método estatístico** [32], inicialmente, os perfis de comportamento dos usuários são criados. Isto é feito através de um período de treinamento onde os usuários são monitorados em suas atividades cotidianas. À medida que o sistema continua a operar normalmente, o detector cria o perfil atual, em intervalos regulares e compara-lhe com o perfil original armazenado. Assim, se o desvio for demasiadamente grande, pode-se inferir que há sinais de anormalidade, o que poderia implicar intrusão.

As principais vantagens desse método são que ele pode ser baseado em técnicas estatísticas bem conhecidas e aprender o comportamento dos usuários, adequando-se a mudanças no perfil.

É justamente devido a essa grande adaptabilidade que ocorre a principal fragilidade dessa abordagem, já que os sistemas baseados em estatística podem

gradualmente ser treinados por intrusos de tal forma que eventualmente os eventos intrusivos sejam considerados como normais.

Outra desvantagem desse método é a dificuldade em se determinar um grau de tolerância adequado para que um perfil não seja considerado intrusivo (*threshold*). Se ele for muito alto ou muito baixo, então poderá haver um grande número de falso negativo ou falso positivo.

No **gerador de prognósticos de padrões**, tenta-se prever os eventos futuros com base em eventos que já ocorreram, conforme expõe Teng [33]. Portanto, pode-se ter uma regra do tipo: $E1 - E2 \rightarrow (E3 = 80\%, E4 = 15\%, E5 = 5\%)$. Isto significa que dados os eventos E1 e E2, com E2 ocorrendo depois de E1, existe 80% de chance de que o evento E3 ocorra em seguida, 15% de chance que E4 seja o próximo e 5% que E5 ocorra. Assim, na ocorrência de um determinado evento, ou uma série de eventos, é possível saber quais os possíveis eventos subsequentes.

Da mesma forma que no método anterior, aqui é necessário primeiro haver um período de treinamento. Através do acompanhamento dos usuários em suas atividades diárias, regras temporais que caracterizam os padrões de comportamento podem ser geradas.

Há algumas vantagens nessa abordagem:

- Padrões sequenciais baseados em regras podem detectar atividades anômalas que são difíceis nos outros métodos;
- Sistemas construídos usando esse modelo são bastante adaptativos a mudanças. Isto se deve ao fato de que padrões de má qualidade são continuamente eliminados, conservando-se os padrões de alta qualidade;
- É mais fácil detectar usuários que tentam treinar o sistema durante o período de aprendizagem;

- Atividades anômalas podem ser detectadas muito rapidamente a partir do recebimento dos eventos de auditoria.

Um problema desse método é que determinados padrões de comportamento podem não ser detectados como anômalos, simplesmente porque não é possível combiná-los em nenhuma regra. Essa deficiência pode ser parcialmente resolvida acusando-se todos os eventos desconhecidos como intrusivos, correndo-se o risco de aumentar o número de falsos positivos ou como não intrusivos, aumentando-se a probabilidade de falsos negativos.

O uso de **Redes Neurais** tem sido a muitos anos objeto de pesquisa [34]. A idéia básica é o treinamento de uma rede neural para prever a próxima ação do usuário, dado o conjunto de suas ações anteriores. A rede é treinada com o conjunto das ações representativas.

Qualquer evento predito que não corresponda com a realidade, pode medir um desvio no perfil normal do usuário mediante a aplicação de redes neurais. Algumas vantagens dessa abordagem são que as redes neurais têm a capacidade de inferir dados com ruídos; o seu sucesso não depende de nenhuma suposição sobre a natureza dos dados e são mais fáceis de se modificar para uma nova comunidade de usuários.

Entretanto, esse método tem alguns problemas, pois um pequeno conjunto de informações irá resultar em muitos falsos positivos, enquanto que um conjunto grande com informações irrelevantes irá aumentar as chances de falsos negativos. Além disso, a topologia da rede só é determinada depois de muita tentativa e erro.

O ponto forte desse tipo de detecção é a capacidade de reconhecer tipos de ataques que não foram previamente cadastrados, baseado apenas nos desvios de comportamento, enquanto que os pontos fracos podem ser o grande número de alarmes

falsos gerados para uma rede treinada insuficientemente e a necessidade de um grande período de treinamento *off-line* para se construir os perfis dos usuários.

3.6.2 Detecção por abuso

A detecção de intrusão por abuso está relacionada à identificação de comportamentos intrusivos correspondentes a exploração de vulnerabilidades conhecidas, comumente chamadas de assinaturas de ataque. Análise de assinatura é a busca por padrões de configuração de sistema ou de atividades do usuário em um banco de dados de ataques conhecidos. Portanto, os SDI's devem ser programados para detectar cada tipo conhecido de ataque. Os detectores de abuso analisam a atividade do sistema, observando os eventos ou conjunto de eventos que sejam semelhantes a um padrão pré-determinado que descreva uma intrusão conhecida.

As assinaturas não servem apenas para detectar intrusões consumadas, elas são úteis, também, para identificar tentativas de ataque. Assim, quando um conjunto de eventos satisfaz apenas parcialmente uma assinatura, pode ser um indicativo de uma intrusão futura que já começou.

São conhecidas várias abordagens para implementar SDI's baseados em detecção por abuso. As mais comuns são: a utilização de sistemas especialistas; a detecção por modelo; a análise de estados de transição e o uso de redes neurais.

Os **Sistemas especialistas** são sistemas inteligentes que capturam o conhecimento de especialistas humanos em domínios limitados, geralmente na forma de regras do tipo IF-THEN. A combinação entre as fases de aquisição dos dados e a ação propriamente dita é feita de acordo com a comparação entre os eventos atuais, capturados através dos mecanismos de auditoria do sistema operacional e as regras já estabelecidas pelo engenheiro de conhecimento. As condições "IF" do sistema especialista codificam o

conhecimento sobre os ataques, declarando suas características. Então, satisfeitas tais condições, as ações "THEN" são executadas [35].

Há algumas vantagens nessa abordagem, a saber: *i)* com um conjunto de regras bem definido, é possível ter-se um sistema ágil e com pouca sobrecarga; *ii)* vários critérios podem ser analisados para se identificar um ataque e ainda manter o sistema especialista leve e eficiente; e, teoricamente, é possível deduzir simbolicamente a ocorrência de uma intrusão baseada apenas nos dados recebidos.

O método apresenta muitas desvantagens. Somente ataques conhecidos e bem definidos podem ser detectados, o que exige uma constante atualização da base de regras. Segundo Lunt [36], os sistemas especialistas são criados à semelhança de quem os programou, sendo limitado pelo conhecimento do programador ou do responsável pela segurança. Outra desvantagem é a dificuldade que há em tirar conclusões sobre situações consideradas ambíguas, devido à existência de conflitos nos fatos assumidos pelos antecedentes das regras.

De acordo com a **Detecção por Modelo**, certos cenários (cenários são seqüências de comportamento que podem caracterizar uma intrusão) de ataque podem ser inferidos através do monitoramento de determinadas atividades. Assim, se um conjunto de eventos ocorrer da forma descrita nos cenários, então é possível detectar a tentativa de ataque. Esse modelo é baseado em três importantes módulos, conforme descrito por Garvey [37].

O *ANTICIPATOR* busca no banco de dados os cenários que melhor expressam as atividades atuais do sistema. Então, ele tenta prever os próximos passos da possível tentativa de ataque de acordo com o cenário escolhido. Depois, essas informações são transmitidas ao *PLANNER*, encarregado de traduzir os passos hipotéticos em um formato compatível com sua provável representação no sistema de auditoria.

Por fim, ao *INTERPRETER* cabe procurar nos registros auditados somente os prováveis eventos. Com esse esquema, é possível supor as ações futuras que o invasor irá realizar e confirmar ou rejeitar tais suspeitas através do monitoramento e análise dos registros de *log* do sistema.

A vantagem dos SDI's baseados no método de detecção por modelo é permitir a emissão de conclusões, ainda que parciais, sobre situações ambíguas. É também possível reduzir a quantidade de processamento exigida por cada registro auditado, através de uma monitoração superficial, buscando somente aqueles eventos necessários para a confirmação do cenário de intrusão.

A desvantagem dos SDI's baseados nesse método é de não detectar situações que não tenham sido especificamente evidenciadas. Poucos são os relatos de protótipo desse modelo implementado, ficando-se dúvidas sobre a eficiência de sua execução, em função dos cálculos envolvidos.

Segundo Porras [38], a detecção por abuso baseada na análise de estado de transição considera que um sistema monitorado pode ser representado como uma sequência de transição de estados. Uma transição acontece quando alguma condição booleana é tida como verdadeira, por exemplo, a abertura de um determinado arquivo. Sucessivos estados estão conectados por arcos que representam os eventos necessários para a mudança de estado ocorrer.

Os tipos de eventos permitidos são descritos no modelo e não precisam ter uma correspondência um a um com os registros de auditoria. Para o estado final (comprometido) ser alcançado, diversas condições precisam ser satisfeitas. Assim, se todas as condições intermediárias forem verdadeiras, então é quase certo que uma tentativa de intrusão está ocorrendo. Entretanto, se alguma dessas condições é falsa, a probabilidade de uma intrusão diminui.

Algumas vantagens dessa abordagem é poder detectar ataques cooperativos, mesmo aqueles que variam entre várias sessões de usuário e prever o acontecimento de ataques iminentes, podendo-se, com isso, tomar medidas preventivas.

As desvantagens são que: *i)* os padrões de ataques podem especificar somente uma seqüência de eventos, ao invés de formas mais complexas; *ii)* não há um mecanismo eficiente que reconheça as tentativas parciais de ataque; *iii)* essa abordagem não pode detectar "negação de serviço", falhas de *login*, variações do uso normal etc, pois esses itens não são gravados pelos mecanismos de *log* e, portanto, não podem ser representados por diagramas de transição de estado.

Assim como na detecção de intrusão por anomalia, as **redes neurais** têm sido utilizadas para reconhecer padrões de ataques ou assinaturas [39]. Mas, diferentemente da primeira abordagem, aqui o mais interessante dessa tecnologia é sua capacidade de classificação e generalização.

Por exemplo, dado um ataque que não seja exatamente igual à sua descrição armazenada, a rede neural pode reconhecê-lo simplesmente pelas semelhanças existentes com a descrição. Cansian [40] desenvolveu um sistema que utiliza redes neurais para detectar padrões de intrusão em redes de computadores. Ele atua capturando e decifrando pacotes, para realizar inferências acerca do estado de segurança das sessões, utilizando um sistema de pré-filtragem e classificação. Como resultado, a rede neural fornece um número que, baseado em informações de assinaturas de ataques anteriores, indicará a gravidade de um determinado evento.

As vantagens e desvantagens do uso de redes neurais na detecção por abuso são análogas às aquelas apresentadas na detecção por anomalia.

De forma geral, a limitação principal da abordagem de detecção por abuso é que somente vulnerabilidades conhecidas são detectadas. Assim, novas técnicas de intrusão

devem ser constantemente adicionadas. Outra limitação desse método tem a ver com considerações práticas sobre o que é auditado.

Por outro lado, o método de detecção por abuso tem sido o mais utilizado pelos sistemas de detecção de intrusão, pois os estudos mostram que cerca de 93% das tentativas de intrusão ocorrem a partir de pequenas variações de padrões bem definidos de comportamento [25]. Além disso, outros fatores importantes que fazem com que esse método seja o mais preferido pelos sistemas comerciais, estão relacionados com a facilidade de configuração, custo computacional reduzido e pequeno comprometimento do desempenho. Quanto ao número de falsos positivos, a literatura disponível mostra que os sistemas de detecção de intrusão por abuso geram menos alarmes falsos do que os que usam o método de detecção por anomalia [40].

Conforme descrito anteriormente os dois métodos possuem vantagens e desvantagens quanto à sua utilização. Conseqüentemente, uma combinação de ambos pode gerar um sistema de detecção de intrusão com maior capacidade para detectar atividades maliciosas em um ambiente computacional.

3.6.3 Detectores de Intrusão baseados em Rede

A maioria dos sistemas de detecção de intrusão está classificada nessa categoria. Esses sistemas detectam intrusão através da captura e análise de pacotes que trafegam na rede, podendo, com isso, monitorar vários *hosts* conectados ao mesmo segmento de rede.

As vantagens dos SDI's baseados em rede são que: *i)* se localizados em pontos estratégicos, podem monitorar uma rede bem extensa; *ii)* tem pouco impacto na rede monitorada, exigindo um mínimo esforço de instalação, pois os sensores são dispositivos passivos; *iii)* diversas técnicas podem ser usadas para camuflar os sensores

de rede, deixando-os invisíveis aos intrusos; *iv*) é possível monitorar ataques distribuídos à rede.

Entretanto, essa categoria de SDI tem alguns pontos negativos, mostrados a seguir:

- Em longos períodos de tráfego intenso na rede, os sensores podem não conseguir capturar todos os pacotes necessários para análise, podendo, dessa forma, não reconhecer tentativas de ataque;
- Ainda não é possível inferir informação alguma sobre o conteúdo de pacotes criptografados. Enquanto as organizações optam pela criptografia para manter a confidencialidade das suas informações, os intrusos a usam para melhorar as chances de sucesso de seus ataques;
- Os sensores de rede não conseguem operar bem com os *switches* modernos, pois esses geralmente não possuem portas de monitoramento universal, o que limita bastante a capacidade de captura de pacotes;

A maioria dos SDI's baseados em rede não indica se a intrusão foi bem sucedida, só há a indicação que a tentativa de intrusão começou. Determinar se houve êxito ou não, é importante para o administrador de segurança.

3.6.4 Detectores de Intrusão baseados em *Host*

Este tipo de Detector de Intrusão examina trilhas de auditoria e arquivos de *logs* para monitorar eventos locais em um sistema de computador em particular, podendo detectar ataques que não são vistos pelo SDI baseado em rede.

Os sistemas de detecção de intrusão baseados em *host* operam analisando as atividades de algum dos computadores da rede em particular. Isto permite ao SDI coletar informações que refletem o que está acontecendo nos computadores de uma

forma bastante precisa. Para tanto, é geralmente utilizado o mecanismo de auditoria do sistema operacional. Assim, o sensor de *host* tem acesso aos diversos *logs* do sistema.

Dentre outras, as vantagens dessa abordagem são que o sistema pode: *i)* monitorar o acesso à informação em termos de "quem acessou o quê"; *ii)* mapear atividades problemáticas com um usuário específico; *iii)* rastrear mudanças de comportamento associadas com mau uso; *iv)* operar em ambientes criptografados; *v)* distribuir a carga do monitoramento ao redor dos diversos computadores da rede.

Dentre as desvantagens encontram-se: *i)* a atividade da rede não é visível pelos sensores de *host* e, dessa forma, alguns tipos de ataques não podem ser identificados; *ii)* o acionamento dos mecanismos de auditoria acarreta uma sobrecarga no sistema; *iii)* às vezes, é requerido um espaço em disco considerável para os registros; *iv)* as vulnerabilidades dos sistemas operacionais podem ser usadas para corromper os sensores de *host*; *v)* os sensores de *host* são específicos por plataforma, o que acarreta em maior custo no desenvolvimento.

3.6.5 Detectores de Intrusão baseados em Aplicação

Este Detector de Intrusão examina o comportamento de um programa de aplicação, geralmente na forma de arquivos de *log*. Esses SDI's são um subconjunto dos SDI's baseados em *host*.

Os SDI's baseados em aplicação são um subconjunto dos baseados em *host*, que analisam os eventos ocorridos nas aplicações. A maior fonte de informação são os arquivos de log das transações das aplicações.

A habilidade para interagir com a aplicação diretamente, através do conhecimento do domínio ou aplicação específica incluída na máquina de análise, permite que sejam

detectados comportamentos suspeitos, oriundos de usuários tentando exceder o uso de seus privilégios.

Essa abordagem possui algumas vantagens, tais como: monitorar a interação entre usuários e aplicação, permitindo que sejam traçadas atividades desautorizadas para usuários individuais e trabalhar em ambientes criptografados, desde que a interação seja no ponto final da transação, onde as informações são apresentadas aos usuários na forma descriptografada.

As desvantagens são que pode ser mais vulnerável a ataques do que na abordagem baseada em host, pois os logs de aplicação são menos protegidos do que as trilhas de auditoria dos sistemas operacionais. Como os eventos monitorados são ao nível de usuário, um SDI, baseado em aplicação, não pode detectar “cavalos de tróia”, ou outro tipo de ataque à aplicação.

Portanto, com base nessa exposição, pode-se concluir que é desejável um SDI que combine as vantagens apresentadas pelas três abordagens, para ter-se informação suficiente para uma análise mais precisa.

3.7 Considerações Finais

Com este capítulo procurou-se especificar os riscos que estão envolvidos em uma rede de computadores, destacando os tipos de ataques técnicos, ataques locais, ataques remotos. Destaca-se nos ataques não técnicos o uso da engenharia social como um dos maiores riscos para instituições que possuam informações sigilosas ou extremamente necessárias ou básicas para o funcionamento de serviços, especialmente serviços públicos.

Para esses e outros riscos é necessário que seja realizada uma política de segurança, desde o mais alto ao mais baixo nível de hierarquia funcional. Devem existir

procedimentos padrão de segurança, bem como a quem os funcionários deveram se dirigir diante de situações inesperadas e fora dos padrões de segurança.

Além de treinamentos, seminários, simulações e avaliações entre os funcionários para a melhoria da segurança, destaca-se que isto é necessário para que todos possam trabalhar com segurança e possam assim manter um determinado nível de produtividade. A seguir foram destacados os principais tipos de ataques como *middle on the man*, analisadores de tráfego (*sniffers*) para espionagens, quebras de senha, ataques com cavalos de tróia, ataques a bibliotecas, ataques em camadas etc.

Após uma longa lista de ferramentas que podem ser utilizadas para invasão em uma rede de computadores, foi destacado o tipo de ferramentas voltadas para a defesa dessa rede de computadores, destacando as principais ferramentas utilizadas pelos administradores de uma rede de computadores (atualizações, correções, vírus, *firewall* etc.).

Foi dado um enfoque especial aos Sistemas de Detecção de Intrusão (SDIs), os seus principais tipos (baseado em *host*, baseado em rede e aplicação) e a sua aplicação.

O próximo capítulo vai destacar o sistema de detecção de intrusão SNORT, um SDI baseado em Rede, mostrando seus principais componentes, como eles atuam isoladamente e como atuam em conjunto com os demais componentes do sistema.

4 DEFINIÇÃO, ESTRUTURA E METODOLOGIA DE AVALIAÇÃO DO *SOFTWARE* DE DETECÇÃO DE INTRUSÃO SNORT

Este capítulo destaca o sistema de detecção de intrusos (SDI) denominado SNORT [41]. Este *software* participante do projeto GNU do inglês (Fundação de *Software* Livre) [42] é um *software* de código aberto de domínio publico, possui grandes potencialidades e uma grande quantidade de adeptos ao redor do mundo. Ele é utilizado em servidores de bancos de dados, provedor de acesso à Internet e em grandes e pequenas instituições que estejam interligadas via rede de computadores. Neste capítulo é mostrado a sua estrutura e funcionamento de seus principais componentes, os seus concorrentes e o futuro dos SDI. A existência do capítulo (SNORT) possui como justificativa a aplicação em um sistema elétrico de potência (SEP) que utilize uma rede de computadores, tendo o objetivo de tratar os ataques sobre o ponto de vista denominado de “abuso”.

Este capítulo também abordará os métodos utilizados para realizar testes para detectar, através do *software* SNORT (detecção por assinatura ou por abuso), as possíveis evidências de tentativa de invasão à rede de computadores e/ou equipamentos microprocessados.

Os sistemas de detecção de intrusos são avaliados quanto às seguintes características: (a) capacidade do Sistema de Detecção de Intrusos (SDI), (b) escalabilidade e (c) taxa de falsos positivos que são gerados. A metodologia utilizada é composta por cinco etapas: (a) seleção dos ataques, (b) seleção de ferramentas, (c) geração do tráfego dos cenários de avaliação, (d) montagem do ambiente de avaliação e (e) análise dos SDIs.

Serão destacadas nesse capítulo as definições das etapas da metodologia de avaliação aplicada no SDI por abuso (SNORT).

4.1 Definição do SNORT

Dentre os mais diversos tipos de sistemas de detecção de intrusão (SDI), o SNORT é um dos mais conhecidos, por diversos fatores. O primeiro é o fato de ser um *software* livre. Segundo, o de ser eficiente e terceiro, por ser simples.

O SNORT foi desenvolvido originalmente por Marty Roesch em 1998 e licenciado pelo GPL (*General Public Licence*). Este *software* inicialmente era denominado de APE. O objetivo inicial era produzir um programa de captura de pacotes de rede, denominado popularmente de *Sniffer*. Com o passar do tempo, Marty decidiu incorporar algumas funcionalidades ao seu *software*, sendo algumas delas:

- Portabilidade, possuir a capacidade de funcionar ou iniciar em diversos Sistemas Operacionais, como por exemplo, Linux, FreeBSD, NetBSD, OpenBSD, Windows etc.;
- Mecanismo de hexdump (Plugin de visualização de arquivo realizada por meio de uma verificação hexadecimal do conteúdo do arquivo) no *Payload* (segurança do encapsulamento IP) dos pacotes;
- Mecanismo para tratamento de pacotes genérico.

No final de 1998 Marty disponibilizou o seu *software*, agora denominado de SNORT, tendo basicamente a função de *sniffer* (farejador). O seu projeto passou por algumas reformulações e em 1999 o SNORT passou a ser um sistema SDI, para uma análise baseada em assinaturas.

O significado adotado neste trabalho para o termo “assinatura” é o de um padrão que deverá ser verificado em um pacote de rede. Se este padrão for identificado ou

encontrado no pacote de assinaturas, um evento é gerado. Com o disparo desse evento, essa base de alertas pode ser registrada em um banco de dados como, por exemplo, no MYSQL ou até mesmo em um arquivo texto. A seguir é destacada a estrutura do SNORT com os seus principais componentes.

4.2 Componentes do SNORT

A estrutura básica do SNORT é composta de quatro componentes básicos:

- O sniffer;
- O pré-processador;
- O Motor de Detecção;
- A saída.

Na sua forma básica o SNORT é um pacote *sniffer*. As informações provenientes desse sniffer são enviadas ao pré-processador que em seguida envia essas informações ao módulo de engenharia de detecção; nesse módulo é realizada a checagem desses pacotes através das regras (assinaturas).

Na figura 21 é mostrada uma visão geral de como o SNORT integra e trabalha com os seus componentes. Inicia-se com os dados a serem analisados por meio da rede *Backbone*, os pacotes são capturados pelo *sniffer* que os envia ao pré-processador que é analisado pela engenharia de detecção, utilizando para isso uma base de regras ou conjunto de regras. Estas regras podem, por exemplo, estar armazenadas em um banco de dados livre como o MYSQL ou algum outro banco de dados relacional ou orientado a objetos.

Figura 21 – Arquitetura do SNORT (adaptado de Ryan Russel ([41]))

De acordo com a figura 21, vamos examinar cada um desses componentes do SNORT. Iniciemos pelo *Sniffer*.

4.3 *Sniffer* do SNORT

O *Sniffer* funciona como uma espécie de escuta das informações que estão trafegando pela rede de computadores. Uma analogia com a tarefa que o *sniffer* realiza é um grampo telefônico no qual se podem escutar todas as conversas que estão em trânsito, partindo ou chegando ao telefone investigado.

Essa captura de pacotes pode ser realizada de tempos em tempos ou em tempo real juntamente com outros *sniffers* que podem em conjunto realizar as armadilhas em vários pontos da rede de computadores. Isto vai depender muito da quantidade de informação que está em trânsito, bem como o limite de informações que podem ser analisadas.

Destaca-se que podem ser utilizados vários *sniffers*, alguns deles podendo analisar defeitos na rede, outros o desempenho da rede.

A forma de não permitir o acesso às informações de uma rede de computadores com alta quantidade de informação e de alto sigilo é realizar a encriptação de informações que estão no tráfego da rede. Outra forma é a instalação de *firewall* entre as sub-redes e assim conseguir um nível maior de segurança.

Na figura 22 pode-se verificar a atuação funcional do pacote sniffer. O usuário comum somente possui acesso à interface visível, tendo sobre a camada de aplicação os principais protocolos de comunicação como: SSH, HTTPS, SQL, SMB, SNMP etc. As interfaces maliciosas são tratadas e investigadas pelo *Sniffer* do SNORT.

Figura 22 – Funcionalidade do Sniffer do SNORT (adaptado de Ryan Russel SNORT [41])

Isto ocorre por que o *Sniffer* pode salvar as informações de um pacote e depois avaliar o seu grau de risco para a rede de computadores. Esses pacotes são gravados em arquivos de *log* para posterior análise.

4.4 Pré-Processador do SNORT

No pré-processador do SNORT as informações são ordenadas para que possam ser absorvidas pelos outros módulos do SNORT. Após a coleta realizada pelo *Sniffer*, o pré-processador entra em ação com o objetivo de pegar os pacotes e verificá-los por meio de pacotes. Esse pacote pode ser um encaixe ou modulo por meio de um RPC ou mesmo um Varredor de plug-in. (figura 23).

Figura 23 – Pré-processador do SNORT(adaptado de Ryan Russel

SNORT [41])

O pré-processador verifica algum tipo de comportamento anormal do pacote, pois cada pacote é determinado para ter um tipo particular de "comportamento". O pacote é enviado ao motor da detecção. Na figura 23 pode-se visualizar como o pré-processador usa seus módulos para verificar um pacote.

Esta é uma excelente habilidade para um SDI, pois outros módulos podem ser encaixados conforme a necessidade do usuário (administrador) em manter a sua rede mais segura.

A utilização de módulos permite que sejam habilitadas ou desabilitadas essas funções conforme a necessidade a ser adquirida pelo pré-processador. Como exemplo, o usuário (administrador) poderá desabilitar o módulo ou encaixe do RPC (chamada de procedimento remoto) de sua rede e habilitar outros módulos necessários para a segurança de sua rede.

4.5 Motor de Detecção

O motor da detecção do SNORT é o centro do SDI. Os dados que vêm do pré-processador são verificados através de um conjunto de regras. Se as regras combinarem com os dados no pacote, é enviado ao processador um alerta.

Em razão de o SNORT ser um IDS baseado em assinaturas, ele utiliza vários conjuntos de regras (assinaturas). Essas assinaturas estão agrupadas pela categoria, como por exemplo, os cavalos de Tróia, *Buffers overflows* etc. Este conjunto de regras é atualizado regularmente.

Estas regras consistem de duas partes:

- **O cabeçalho da regra**

O cabeçalho da regra é basicamente uma ação a ser realizada (*Log* ou *Alerta*), tipo de pacote (TCP, UDP, ICMP etc.), origem e destino do endereço IP e portas.

- **A opção da regra**

A opção é índice, permite que o pacote faça a combinação com a regra.

O motor da detecção e suas regras exigem um esforço maior de aprendizagem e compreensão do SNORT, já que a curva de aprendizagem é mais íngreme. O SNORT possui uma sintaxe particular que é utilizada para a criação de suas regras. A sintaxe das regras pode envolver o tipo de protocolo, índice, tamanho, cabeçalho e de vários outros elementos, incluindo caracteres com lixo etc.

Aprendendo a escrever as regras do SNORT, o usuário pode otimizar as regras do SDI, melhorando a sua funcionalidade. As regras são criadas de acordo com as necessidades do administrador da rede (as regras podem ser particulares ao ambiente em que o SNORT está instalado).

A figura 24 mostra a situação do motor de detecção para realizar um alerta ou não de um pacote suspeito capturado.

Figura 24 – Motor de Detecção do SNORT (adaptado de Ryan Russel (SNORT [41]))

4.6 Alertas

Como mostrado na figura 24, o SNORT quando recebe um pacote suspeito, realiza a leitura do pacote, tendo alguma confirmação de ameaça por meio da verificação das regras da base de assinaturas, é realizado o alerta. Caso não seja verificada nenhuma anormalidade, o pacote é descartado, não causando nenhum alerta no sistema.

Esses alertas podem ser enviados para o arquivo de Log, através da conexão de rede em arquivos do banco de dados SQL como o *MySQL*.

Nesses arquivos de *Log*, podem ser posteriormente utilizados vários *plugins*, como por exemplo, *plugins* para Perl e PHP, para mostrar os dados por meio de uma interface para a Web ou interface gráfica para aplicativos ou linguagens visuais.

Normalmente os dados de *Log* são armazenados em formato texto ou em banco de dados.

Os alertas podem ser enviados por meio de protocolos do sistema operacional que estão sendo utilizados na máquina que está servindo como base para disparar os alarmes em uma rede de computadores. Por exemplo, podem ser utilizados o SNMP (*Simple Network Management Protocol*) e mensagens do Winpopup. A figura 25 mostra como isto pode ocorrer em uma rede de computadores com o SNORT instalado.

Figura 25 – Componentes de Alerta do SNORT

(adaptado de Ryan Russel (SNORT [41]))

Esses alertas, além de serem enviados para serem armazenados em um banco de dados, também são enviados em tempo real ao administrador responsável pela segurança da rede de computadores. Esse aviso pode ser via terminal ou monitor de vídeo, telefone fixo ou celular, bem como por e-mail, independente da hora ou dia, podendo disparar esse modo de alerta para o administrador a qualquer momento, funcionando 24 horas por dia, os sete dias da semana.

Abaixo, a Tabela 2 contém alguns produtos do SNORT que podem ser adicionados à base já instalada para melhor visualizar ou tratar os dados juntamente com os alertas obtidos.

Tabela 2 – Descrição de produtos adicionais para o SNORT

Produto	Descrição
SNORTSnarf	Um analisador do SNORT da Silicon Defense utilizada para diagnósticos cuja saída é em HTML.
SNORTplot	Um script Perl que faz graficamente a impressão dos ataques.
Swatch	Um monitor syslog em tempo real que também providencia alertas em tempo real via e-mail.
ACID	Um console analisador para intrusão de banco de dados. Possui logins para análise pelo SNORT. Necessita do plugin para Banco de dados.

Além desses produtos mencionados na tabela, também podem ser incluídos o *Demarc*, *Razorback*, *Incident.pl*, *Loghog*, *Oinkmaster*, *Sneakyman*, *SNORTReport* etc. Todos eles possuem maneiras de trabalhar muito bem com o SNORT. Recursos para o SNORT variam somente a plataforma de instalação, tipos de relatórios e gráficos, bem como os meios dos dados a serem importados para outras ferramentas que determinam uma análise mais detalhada dos *logins* e mensagens produzidas pelo SNORT.

A seguir, as aplicações do SNORT, mostrando a combinação de equipamentos que podem aumentar o nível de segurança de uma rede de computadores utilizando um SDI.

4.7 Aplicações do SNORT

A questão de ter um sistema de detecção de intrusão não conclui que a rede de computadores com as informações esteja eternamente segura. Esse é um grave erro cometido por muitos.

Uma instituição, que realmente queira ter um sistema com níveis de segurança, necessita de toda uma estrutura para que essa segurança realmente possa se manter em alto grau. Por isso um sistema de detecção de intrusão nunca deverá trabalhar sozinho, o que inclui *hardware* e um planejamento da rede de computadores que permitam que esses componentes ao nível de *software* e *hardware* possam trabalhar em harmonia e plenitude.

Como exemplo desta situação mencionada, temos a integração de um *firewall* juntamente com o SDI (figura 26). No primeiro nível de defesa estaria o SDI (SNORT) para filtrar os pacotes que estariam saindo ou chegando à rede interna. Como segundo nível de defesa estaria o *firewall* impedindo por meio de uma base de assinatura os tipos de informações que não poderão sair ou chegar à rede interna de computadores (*LAN*).

Figura 26 – SDI com Roteador

(adaptado de Ryan Russel (SNORT [41]))

Tanto o SDI SNORT como o *firewall* necessitam ser constantemente atualizados ao nível de base de assinaturas, caso contrário com o tempo novos tipos de ataques serão desenvolvidos e em pouco tempo, aquela que poderia ser considerada segura, pode ser considerada uma rede de baixíssimo nível de segurança.

Uma estrutura mais segura seria a colocação de dois níveis de um SDI, o que é realizado mediante a inclusão de duas máquinas contendo o SNORT realizando a captura de pacotes, sendo que no centro desses dois SDI, ficaria o *firewall*. A figura 27 mostra exatamente esta situação. Os custos obviamente aumentam, mas também aumenta o nível desejado de segurança em uma rede de computadores.

Figura 27 – Firewall com SNORT

(adaptado de Ryan Russel (SNORT [41]))

O problema fica para a instituição do custo/benefício a investir em segurança em sua rede de computadores que alcance o grau de segurança desejado. Esse investimento, além de incluir *software* e *hardware*, inclui treinamentos, palestras e seminários em todos os níveis da empresa. Ocorrendo desde o alto ao mais baixo nível da empresa, incluindo todas as empresas terceirizadas, pois elas também podem em alguns casos ter

acesso a informações muito sigilosas da empresa, como exemplo o setor contábil ou de projetos da instituição.

Em casos que exijam uma segurança extra, é interessante que haja um triângulo de sistemas de detecção de intrusão, desde que as informações sejam altamente importantes e que necessitem um alto grau de segurança.

A figura 28 mostra exatamente a situação mencionada anteriormente, o *firewall* fica no centro dos SDI (SNORT). Este projeto de rede de computadores com um trio de SDI é o mais recomendado quando se necessita de uma maior segurança para usuários e clientes. A sigla DMZ (*Demilitarized Zone*) vem do inglês que significa em português, “zona delimitada” (figura 28). Este tipo de arranjo permite delimitar as zonas de atuação sobre os servidores, delimita e protege os servidores (HHTTP, WEB, Aplicação).

Figura 28 – Firewall DMZ e SNORT (adaptado de Ryan Russel SNORT [41])

4.8 Concorrentes do SNORT

Excluindo o SNORT, os outros SDI embora também sejam bastante eficazes no que fazem, eles não são gratuitos, nem possuem uma grande quantidade de adeptos, como o SNORT.

Existem muitas ferramentas de detecção de intrusão. Como concorrentes do SNORT existem SDI comerciais com grande poder de atuação no mercado de *softwares*, destacando-se o SDI da Cisco, denominado “Cisco Netranger”, seguem abaixo outros SDI que fazem uma rede de computadores mais segura.

Cisco Netranger

Esse é um sistema de detecção de intrusão em tempo real que, além de muito fácil de instalar e administrar. O produto é escalável e confiável como o SNORT, possui características necessárias para a operação eficaz de soluções de segurança tanto em redes de Internet quanto de intranet.

Esse SDI da Cisco, o “Cisco NetRanger”, inclui dois componentes: *Director*, um sistema escalável de administração remota, e um Sensor, um dispositivo de alto desempenho e fácil instalação. Quando o sistema detecta uma atividade não autorizada, os dispositivos Sensor podem interromper automaticamente uma conexão específica, bloqueando permanentemente o ataque, além de registrar o incidente e enviar um alarme ao *NetRanger Director*.

Além desse forte concorrente do SNORT existem outros como: *DoD Shadown*, *ISS RealSecure*, *NetWork Flight Recorder*, *TCP Wrappers*, *TriWire*, *Nuke Nabber*.

As opções para SDI são muitas, sendo que o que determina a escolha é o administrador da rede de computadores. No entanto, em termos de custo/benefício o SNORT continua imbatível.

A seguir a metodologia de avaliação do *software* de detecção de intruso SNORT [43].

4.9 Etapas da Metodologia

4.9.1 Seleção dos ataques

Com este item, é realizada a seleção de ataques que possa explorar especificações técnicas únicas entre si. Ao invés de simplesmente reunir um conjunto de ataques, o que se busca, ao finalizar esta etapa, é selecionar ataques cuja detecção seja possível a partir de diferentes mecanismos de detecção existentes em um SDI. Por exemplo, para que um SDI seja capaz de detectar um ataque de inserção, o *URL Encoding*, necessita mais do que simplesmente a capacidade de análise de um pacote HTTP, pois é necessário, ainda, um mecanismo de decodificação do conteúdo do pacote. Já o processo de detecção de um ataque de negação de serviço, como o *teardrop*, requer capacidades como remontar pacotes IP fragmentados. Dessa forma, os ataques selecionados nessa etapa, representam um conjunto de características ímpares que permitem avaliar as diferentes capacidades de detecção dos SDIs e não simplesmente a base de assinaturas dessas ferramentas. Essa forma de classificação denomina-se descrição técnica do ataque. Para que essa forma de seleção de ataques seja colocada em prática é necessário definir um conjunto inicial de ataques [43]. Essa atividade está descrita na seção a seguir.

a) Ataques Propostos

A primeira atividade prevista para essa etapa de seleção de ataques é definir quais os tipos de ataques que serão utilizados na avaliação. Nessa Tese foram considerados os seguintes tipos de ataques:

Evasão: Através de um ataque de evasão é possível obter desde o tipo e a versão de servidor *web* utilizados na estação alvo até a executar *scripts* que possam colocar em risco a segurança de tal servidor. Independente do tipo de ação executada, o objetivo ao realizar este tipo de ataque é primeiramente evitar a detecção do mesmo, fazendo com

que no processo de análise do conteúdo de um pacote os SDIs sofram perdas de informações vitais para a detecção. Já as aplicações alvo recebem todas as informações contidas nos pacotes enviados e, portanto, serão vítimas do ataque.

Varredura de portas ou *port scan*: Normalmente essa é uma das primeiras atividades realizadas por um intruso e consiste basicamente em uma coleta de dados, cujo objetivo é reunir o maior número possível de informações sobre a rede ou o servidor alvo. Essas informações podem variar desde o tipo de sistema operacional instalado em uma determinada estação até quais serviços estão em execução e as suas respectivas versões. As varreduras de portas estão entre as maiores vulnerabilidades críticas de segurança na Internet [25].

Negação de serviço ou *Deny of Services (DoS)*: Este ataque tem o objetivo de esgotar os recursos de um serviço ou rede tornando-o inacessível ou com respostas muito lentas. Os ataques de negação de serviços são normalmente executados usando ferramentas que enviam de forma indiscriminada requisições a um determinado servidor, sobrecarregando os recursos do mesmo e, por vezes, tornando o sistema inoperável. Na grande maioria desses ataques o endereço de origem é forjado (*spoofing*) e, portanto, dificultam o processo de auditoria. Todos os sistemas conectados à Internet e que estejam executando serviços de rede baseados no protocolo TCP estão sujeitos a ataques de negação de serviços [44]. Esse tipo de ataque possui uma variante denominada negação de serviço distribuída, *Distributed Denial of Service (DDoS)*, que corresponde a ataques de DoS realizados em larga escala, partindo de várias estações e disparados simultaneamente de forma coordenada sobre um ou mais alvos.

Uma vez definidos os tipos de ataques a serem utilizados na avaliação é o momento de selecionar um conjunto de atividades intrusivas que irá representar cada

um desses tipos de ataques. A seguir serão descritos os ataques que compõem os cenários de ataque propostos nessa Tese.

b) Ataques selecionados

Evasão

Case Sensitivity

Os sistemas operacionais tais como o Windows 98 e Windows 2000 Server não diferem letras maiúsculas de letras minúsculas (não são *case sensitivity*), ou seja, o arquivo *phf.cgi* pode ser referenciado tanto como *PHF.cgi* quanto como *PHF.CGI*. Logo os Sistemas de Detecção de Intrusos devem ser capazes de detectar ambas as requisições, caso contrário o ataque será bem sucedido.

Method Matching

No intuito de explorar as vulnerabilidades de um *script* e, ainda, tentar inviabilizar a detecção de tal ataque, pode-se utilizar métodos alternativos de solicitações tais como *Put*, *Head* e *Post*. Dessa forma, embora alguns SDIs identifiquem a requisição "Get /cgi29 bin/phf.cgi HTTP/1.0", podem vir a não identificar uma requisição do tipo "Put /cgi-bin/phf.cgi HTTP/1.0".

Session Splicing

Diferentemente dos ataques de fragmentação, este ataque consiste no envio de diversos pacotes. Por exemplo, a solicitação "Get /cgi-bin/phf.cgi HTTP/1.0" pode ser dividida em múltiplos pacotes "Ge", "t", "/", "cgi", "-bin", "p", "hf.c", "gi", "H", "T", "TP", "/", ".0". Sendo assim, o processo de detecção é ainda mais difícil e para que seja possível os SDIs devem ser capazes de analisar uma sequência de pacotes.

HTTP Mis-formatting

Conforme a RFC 2616 a estrutura de uma requisição a um servidor *web* deve seguir o seguinte formato: método <espaço> URL <espaço> versão CRFL CRFL; onde CRFLs corresponde a uma linha em branco obrigatória. No entanto, muitos servidores *web* aceitam requisições que não estejam plenamente em conformidade com essas especificações, por exemplo: método <tabulação> URL <tabulação> versão CRFL CRFL. Portanto, existe a possibilidade de iludir alguns SDIs, pois ao ser realizada a comparação entre o pacote e a base de assinaturas de ataques não haverá nenhuma relação, logo o pacote será considerado uma solicitação dentro dos padrões de normalidade e não um ataque.

DOS Directory Syntax

Em plataformas Windows o separador de diretórios é representado por '\', enquanto que a especificação do protocolo HTTP utiliza o separador de diretórios *web* '/'. Isso ocorre toda vez que uma requisição do tipo "Get /cgi-bin/phf.cgi" é enviada a um servidor web da Microsoft, esse tenha que converter '/' para '\', de forma que a aplicação em questão traduza a requisição como "Get \cgi-bin\phf.cgi". Portanto, este servidor aceita requisições como, "\cgi-bin\phf.cgi", o que faz com que o SDI, ao analisar o pacote HTTP, não detecte um ataque conhecido.

- Inserção

Long URLs

Existem várias técnicas que visam melhorar o desempenho dos SDIs. Uma dessas técnicas limita a quantidade de informações de uma requisição HTTP a serem analisadas. Dessa forma, é possível a inserção de uma quantidade suficiente de caracteres para mover o código de ataque para além do escopo da análise do SDI, o que faz com que o ataque não seja detectado.

Self Reference

Os caracteres “..” quando utilizados para acessar diretórios conduzem o usuário para um diretório superior (diretório pai) ao diretório atual. Já o caracter “.” faz referência ao diretório atual. Sendo assim, então “c:\temp\..\..\.” é equivalente a “c:\temp\”. O objetivo da técnica denominada *Self Reference* é confundir os mecanismos de análise de assinaturas dos SDIs enviando a requisição “Get /./cgibin/./phf”.

URL Encoding

Conforme a RFC 2616, caracteres binários arbitrários podem ser passados em uma requisição HTTP desde que estejam na seguinte notação: %xx, onde xx corresponde ao valor hexadecimal do caracter. Uma vez que a requisição “Get /cgi-bin/teste.cgi HTTP/1.0” seja 30 codificada torna-se “Get /cg%69-b%69n/t%65st%65.cg%69 HTTP/1.0”. Portanto, os SDIs, antes de analisarem qualquer *string* devem decodificar a mesma.

Multiple Slashes

Os servidores *web* aceitam requisições contendo múltiplas barras, *slashes*, como em “Get /cgi-bin//scripts//phf.cgi HTTP/1.0”. Contudo, existe a possibilidade que alguns SDIs ao analisarem esses tipos de requisições falhem devido ao fato de que a assinatura existente contenha apenas uma barra.

Parameter Hiding

Uma requisição de página *web* pode conter informações adicionais, parâmetros, que serão utilizadas para construir o conteúdo de páginas dinâmicas. Esses parâmetros são determinados após um sinal de interrogação no identificador uniforme de recursos, *Uniform Resource Locator* (URL), como em “Get /index.htm?user=normal HTTP/1.0”. Alguns SDIs, a fim de otimizar o processo de análise de pacotes ignoram todas as informações

após a indicação de parâmetros. Sendo assim, há a possibilidade da inserção de um código malicioso após essa parte da requisição.

Reverse Traversal

Consiste em uma tentativa de iludir os SDIs através de uma requisição na qual existam referências a outros diretórios que não estejam especificados na base de assinaturas dos SDIs. Por exemplo, a requisição "Get /cgi-bin/phf.cgi HTTP/1.0", é facilmente detectada. Alguns SDIs, porém, podem desconsiderar esta mesma requisição quando requerida da seguinte forma: "Get /cgi-bin/scripts/./phf.cgi HTTP/1.0".

- **Varredura de Portas e Serviços**

TCP connect

A chamada de sistema *connect* (), provida pelo sistema operacional, é usada para estabelecer conexão com um conjunto de portas na máquina alvo. Caso a porta esteja no estado *listening*, *connect* irá estabelecer uma conexão. Caso contrário, o usuário receberá a mensagem de porta inalcançável.

SYN Scan

Técnica conhecida como "*half-open scanning*" por não estabelecer uma conexão TCP completa. O primeiro pacote a ser enviado está com o *flag* SYN configurado para estabelecer uma conexão real e, portanto, deve aguardar uma resposta da estação para a qual o pacote foi enviado. Ao receber uma resposta com os *flags* SYN/ACK ligados isto indica que a porta está no estado *listening*. Já uma resposta com o *flag* RST ligado é uma indicação que a porta está fechada.

ACK Scan

Este tipo de sondagem envia pacotes com o *flag* ACK para uma porta específica. Caso seja retornado um pacote TCP com *flag* RST ligado, a porta é classificada como

"não filtrada"; caso seja retornado um ICMP *unreachable*, a porta é classificada como "filtrada".

Window Scan

Este tipo de *scan* é muito similar ao ACK *scan*. No entanto, é possível detectar portas abertas mesmo quando essas estão sendo filtradas por um *firewall*. Isso ocorre devido ao tamanho da janela TCP existentes em diversos sistemas operacionais (por exemplo: FreeBSD, SunOS e OpenVMS).

FIN Scan

Esta técnica consiste em enviar um pacote com o *flag* FIN habilitado para uma determinada porta. Segundo a RFC 793 as portas que estiverem fechadas devem responder com um pacote TCP com o *flag* RST ligado, enquanto que as portas que estiverem abertas devem ignorar o pacote em questão.

UDP Scan

Este método é usado para determinar quais as portas UDP (*User Datagram Protocol*) estão abertas. A técnica implica em enviar 0 bytes de dados de pacotes UDP para cada porta da estação alvo. Caso a resposta seja uma mensagem ICMP *port unreachable* então a porta está fechada.

Null Scan

Esta técnica consiste em enviar um pacote TCP com todos os *flags* desabilitados para uma determinada porta na estação alvo, sendo que as portas que estiverem fechadas devem responder com um pacote TCP com o *flag* RST habilitado, enquanto que as demais devem ignorar o pacote em questão.

Xmas

Ao contrário do método denominado *null scan* esta técnica envia um pacote para cada porta da estação alvo a ser sondada com todos os *flags* habilitados exceto o *flag* SYN.

TCP Ping

Através desta técnica é possível determinar quais as estações que estão ativas no momento. Para tal, ao invés de enviar pacotes ICMP *echo request* e aguardar pelas respostas são enviados pacotes com *flag* ACK habilitado por toda a rede. Estações que estiverem ligadas devem responder com um pacote TCP com o *flag* RST habilitado.

TCP fragmentation scanning

Esta forma de sondagem utiliza várias outras técnicas de varredura de portas tais como *SYN scan*, *FIN scan*, *Xmas* e *Null scan*. Os pacotes enviados a estação alvo são fragmentados, ou seja, o cabeçalho TCP é dividido em vários pacotes. Com isso, os SDIs que não possuem mecanismos eficientes de remontagem de pacotes não conseguem identificar essa forma de ataque, pois os diversos datagramas enviados individualmente não correspondem a uma ameaça.

Sondagem do protocolo IP

Este método determina quais protocolos da família TCP/IP estão sendo utilizados na estação alvo. A técnica consiste em enviar pacotes IP *raw* sem nenhum cabeçalho para cada porta na estação alvo. Caso a resposta seja ICMP *unreachable*, o protocolo não está sendo utilizado.

Identificação Remota de Sistemas Operacionais (*fingerprinting*)

A fim de identificar o sistema operacional instalado em uma determinada estação, se utiliza um conjunto de técnicas que detectam características da implementação do protocolo TCP/IP do sistema operacional que está instalado na estação alvo. Uma vez

que essas características tenham sido identificadas é realizada uma comparação dessas informações com a base de dados da ferramenta de ataque, a fim de definir qual o sistema operacional da estação em questão. Atualmente as técnicas mais utilizadas para determinar o tipo de sistema operacional são: sondagem FIN (*FIN probe*), identificação de padrões do número inicial de sequência (*Initial Sequence Number* - ISN) escolhido pelo TCP ao responder um pedido de conexão, verificação da existência ou não do bit de não fragmentação, análise do tamanho da janela entregue pelos pacotes de retorno (*TCP Initial Window*), valor do *flag* ACK, tamanho da mensagem ICMP de erro, verificação do valor do tipo de serviço retornado pelas mensagens de ICMP *port unreachable* e, ainda, análise da forma como é feito o controle de fragmentação e das informações existentes no campo de opções do cabeçalho TCP.

IDENT Reverso TCP

O protocolo IDENT (RFC 1413) retorna nomes de usuários válidos e é consultado por diversos serviços (IRC, FTP, SMTP, etc.), além de servir como mecanismo de restrição de acesso baseado na relação usuário e endereço IP. Porém, conforme notificado por Dave Goldsmith em 1996, o rastreamento do protocolo IDENT permite revelar o nome dos usuários donos dos processos conectados via TCP.

- **Negação de Serviços**

Smurf

Esse ataque utiliza-se de redes que permitem tráfego na interface de *broadcast*. O ataque consiste na falsificação do endereço de origem de um pacote ICMP *echo request*, fazendo com que uma grande quantidade de respostas, pacotes ICMP *echo reply*, sejam direcionadas ao endereço que foi falsificado.

UDP Storm

A exemplo do ataque anterior o objetivo é congestionar a rede e, por conseguinte, diminuir a largura de banda da mesma. Ao se estabelecer uma conexão entre dois serviços UDP, por exemplo, *echo*/UDP e *chargen*/UDP, serão gerados uma grande quantidade de pacotes na rede até que ocorra uma intervenção externa, como, por exemplo, reiniciar o serviço *inetd*.

SYN Flood

Este ataque explora as limitações do processo inicial de uma conexão denominado *handshake*, procedimento que, através do envio de pacotes TCP com os *flags* SYN e ACK habilitados entre cliente e servidor, permite efetuar o início de uma conexão a um serviço. O objetivo é exceder os limites definidos para o número de conexões que podem ser estabelecidas à um determinado serviço. Isso faz com que não seja possível estabelecer quaisquer outras conexões a esse serviço até que o número de conexões em espera seja reduzido. O problema mais crítico que envolve esse tipo de ataque e os SDIs é a alta probabilidade de falsos positivos, uma vez que nem todas as tentativas de conexões em um pequeno intervalo de tempo, podem ser consideradas tentativas de *SYN Flood*.

Teardrop

Ao contrário do ataque denominado *Smurf*, que utiliza a “força bruta” para gerar o ataque, o *teardrop* executa um ataque de DoS utilizando-se de falhas em diferentes implementações da pilha TCP/IP. Este ataque explora a incapacidade de alguns sistemas operacionais de reconstituir pacotes IP fragmentados. Como resultado, os sistemas suscetíveis a esse ataque têm o seu funcionamento prejudicado, podendo inclusive travar o sistema operacional.

ICMP Fragmentation

Para ser transmitido entre redes locais, um pacote IP deve ser fragmentado toda vez que exceder o limite do maior quadro que uma determinada rede local é capaz de transmitir (*Maximum Transfer Unit* - MTU). Nesse caso, é necessário dividir o pacote IP em fragmentos menores que a MTU. O protocolo ICMP é um protocolo auxiliar ao IP, que carrega informações de controle e diagnóstico, informando falhas como TTL do pacote IP excedido, erros de fragmentação e roteadores congestionados. O ataque em questão consiste no envio de um pacote ICMP mal formado para uma determinada estação, fazendo com que a estação de destino ao receber este pacote reduza a MTU desnecessariamente. Isto faz com que a conexão entre essas duas estações fique extremamente lenta. Além disso, em função da quantidade de pacotes enviados uma razoável quantidade da largura de banda é consumida.

4.9.2 Descrição técnica do ataque

Para realizar a descrição técnica do ataque proposto é de fundamental importância o conhecimento das características técnicas do ataque. Essas características foram obtidas através do estudo dos ataques descritos na seção anterior. A partir desse estudo identificaram-se as seguintes características: i) tipo e quantidade de pacotes enviados, ii) campos do cabeçalho desse pacote que são importantes para que o ataque seja realizado, iii) a quantidade e o tamanho dos pacotes a serem gerados. Sendo assim, uma vez que os ataques propostos tenham sido estudados e as suas características forem conhecidas é o momento de classificar os ataques em função das suas características técnicas. Isto deve reduzir o cenário inicial de ataques a uma quantidade que represente apenas ataques que explorem características diferentes uns dos outros. As tabelas (3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 e 14) que seguem, organizadas da seguinte forma: nas linhas foram representados os ataques propostos no cenário de ataques. Já nas colunas aparecem às características

exploradas em cada ataque. Estas características estão agrupadas em função dos tipos de pacotes utilizados (HTTP, IP, TCP, ICMP e UDP) e de aspectos relevantes para a realização do ataque tais como: a quantidade de pacotes utilizados e o estabelecimento ou não de conexões com a estação a ser atacada. Assim que todos os ataques e todas as características forem colocados nessa tabela é o momento de relacionar os ataques com as suas respectivas características. Como resultado, tem-se a visão detalhada de cada ataque na etapa de seleção e quais ataques devem compor o cenário de avaliação.

Tabela 3 – Descrição técnica de ataques de evasão HTTP

Evasão	HTTP
Method Matching	(Conexão - Estabelece Conexão), Linhas de requisição
Session Splicing	(Conexão - Múltiplos pacotes, Estabelece Conexão), Linhas de requisição

Tabela 4 – Descrição técnica de ataques de Inserção HTTP

Inserção	HTTP
Long URLs	(Conexão - Estabelece Conexão), Linhas de requisição
Self Reference	(Conexão - Estabelece Conexão), Linhas de requisição
URL Enconding	(Conexão - Estabelece Conexão), Tamanho da requisição, Codificação da Requisição, Linhas de requisição

Tabela 5 – Descrição técnica de ataques de Varredura de Portas - Conexão

Varredura de Portas	Conexão
UDP Scan	Múltiplos pacotes
Xmas	Múltiplos pacotes
TCP Fragmentation	Múltiplos pacotes
Varredura do protocolo IP	Múltiplos pacotes
FingerPriting	Múltiplos pacotes
Ident Reverso TCP	Múltiplos pacotes, Estabelecer Conexão

Tabela 6 – Descrição técnica de ataques de Varredura de Portas IP

Varredura de Portas	IP
TCP Fragmentation	Off Set de Fragmento, Identificação de Fragmento, FLAGS de controle de Fragmento
Varredura do protocolo IP	Pacote IP Raw
FingerPriting	Pacote IP com DF bit habilitação

Tabela 7 – Descrição técnica de ataques de Varredura de Portas TCP

Varredura de Portas	TCP
Xmas	Pacote com <i>Flag</i> FIN, Pacote com <i>Flag</i> ACK, Pacote com <i>Flag</i> URG, Pacote com <i>Flag</i> PUSH
TCP Fragmentation	Pacote com <i>Flag</i> FIN, Pacote com <i>Flag</i> ACK, Pacote com <i>Flag</i> URG, Pacote com <i>Flag</i> PUSH
FingerPriting	Opções de controle da fragmentação, Campo: tipo de serviço, Número da Sequência, TCP Initial Window, Pacote com <i>Flag</i> FIN, Pacote com <i>Flag</i> ACK
Ident Reverso TCP	Pacote com <i>Flag</i> ACK

Tabela 8 – Descrição técnica de ataques de Varredura de Portas ICMP

Varredura de Portas	ICMP
FingerPriting	Tamanho da mensagem ICMP de Erro

Tabela 9 – Descrição técnica de ataques de Varredura de Portas UDP

Varredura de Portas	UDP
UDP Scan	Pacote UDP de zero bytes

Tabela 10 – Descrição técnica de ataques de Negação de Serviço - Conexão

Negação de Serviço	Conexão
Smurf	Múltiplos serviços
UDP Storm	Múltiplos serviços
Syn Flood	Múltiplos serviços, Não estabelece conexão (Half open)
Teardrop	Múltiplos serviços
ICMP Fragmentation	Múltiplos serviços

Tabela 11 – Descrição técnica de ataques de Negação de Serviço IP

Negação de Serviço	IP
Teardrop	Off set de Fragmento, Pacote IP com DF bit habilitado, FLAGS de controle da fragmentação.
ICMP Fragmentation	Off set de Fragmento, Pacote IP com DF bit habilitado, FLAGS de controle da fragmentação.

Tabela 12 – Descrição técnica de ataques de Negação de Serviço TCP

Negação de Serviço	TCP
Syn Flood	Pacote com <i>Flag</i> SYN

Tabela 13 – Descrição técnica de ataques de Negação de Serviço ICMP

Negação de Serviço	ICMP
Smurf	Pacote ICMP
ICMP fragmentation	Pacote ICMP

Tabela 14 – Descrição técnica de ataques de Negação de Serviço UDP

Negação de Serviço	UDP
UDP Storm	Pacote UDP de zero bytes

A seguir serão apresentados os motivos que conduziram à seleção dos ataques nas tabelas acima destacadas.

Evasão

O ataque *Session Splicing* é o único entre os ataques de evasão que utiliza múltiplos pacotes. Todos os demais utilizam somente um pacote HTTP. Entre os ataques que apresentam as mesmas características, o *Method Mathing* foi arbitrariamente selecionado.

Inserção

Os ataques selecionados, *Long URLs* e *URL Enconding*, apresentam como características exclusivas em relação aos demais ataques, respectivamente, uma requisição HTTP com uma grande quantidade de caracteres e a presença de um padrão de codificação da requisição. Entre os ataques *Self Reference*, *Multiple Slashes* e *Reverse Traversal* que exploram as mesmas características foi escolhido arbitrariamente o *Self Reference*.

Varredura de portas

Os ataques *UDP scan*, varredura do protocolo IP e *TCP Fragmentation* foram selecionados, pois geram, respectivamente, pacotes UDP, IP *raw* e IP fragmentados. Entre os ataques que estabelecem conexão (*TCP connect* e *Ident Reverso TCP*) ambos utilizam pacotes TCP, no entanto *Ident Reverso TCP* foi o escolhido para representar este tipo de técnica de varredura de portas. O último ataque selecionado *Fingerprinting* utiliza três tipos diferentes de pacotes em uma mesma seção de ataque (IP, TCP e ICMP). Já *Xmas* foi o ataque selecionado para representar todos os ataques de varredura de portas que utilizam somente pacotes TCP e que não estabelecem conexões.

Negação de serviço

Os ataques *Smurf*, *UDP Storm*, *Syn Flood* e *Teardrop* foram selecionados, pois diferem um dos outros quanto aos tipos de pacotes utilizados, ou seja, ICMP, UDP, TCP e IP, respectivamente. Já o ataque identificado como *ICMP Fragmentation* foi incluso nessa seleção, devido ao fato de que entre esses ataques é o único que utiliza pacotes do tipo ICMP fragmentados.

4.9.3 Seleção de Ferramentas

Essa etapa foi dedicada à obtenção de ferramentas que permitissem reproduzir o cenário de avaliação proposto na etapa anterior. Essa etapa pôde ser realizada num curto espaço de tempo devido à facilidade com que atualmente se pode encontrar e utilizar tais ferramentas pela *Internet* (Tabela 15).

Tabela 15 – Seleção das Ferramentas do Cenário de Avaliação

Ataques	Ferramentas	Pré-requisitos
Evasão		
Method Mathing	Whisker 1.4	GNU/Linux, Perl 5.0 e Libwhisker
Session Splicing		
Inserção		
Long URLs	Whisker 1.4	GNU/Linux, Perl 5.0 e Libwhisker
Self Reference		
URL Enconding		
Varredura de portas		
TCP Fragmentation	Nmap 2.54	GNU/Linux
UDP Scan		
TCP Connect		
Varredura do protocolo IP		
Fingerprinting		
Ident Reverso TCP		
Negação de serviços		
Smurf	Smurf 4.0	GNU/Linux e gcc 2.9.6
UDP Storm	Udpstorm 1.0	GNU/Linux e gcc 2.9.6
SYN Flood	Synk4	GNU/Linux e gcc 2.9.6
Teardrop	Teardrop 2.0	GNU/Linux e gcc 2.9.6
ICMP Fragmentation	Jolt2	GNU/Linux e gcc 2.9.6

4.9.4 Geração do tráfego do Cenário de Avaliação

O cenário de avaliação é formado pelos ataques selecionados na seção 4.9.2 e pelo tráfego de fundo necessário para a análise de escalabilidade. A seguir serão descritas as formas propostas nessa metodologia para armazenar o tráfego de ataque e gerar o tráfego de fundo.

4.9.4.1 Coleta do tráfego de ataque

Os testes previstos na etapa de análises dos SDIs são realizados a partir da reprodução do tráfego de ataque, fazendo com que não seja necessária a utilização de cada uma das ferramentas de ataque cada vez que os testes tiverem que ser reproduzidos.

A primeira atividade prevista nessa etapa corresponde à montagem do ambiente de rede, conforme a figura 29. Inicialmente esta rede de computadores é composta por apenas três estações, todas com sistema operacional Linux. Na estação *Ataque* estão instaladas todas as ferramentas de ataque selecionadas na etapa de seleção de ferramentas. Já na estação *Vítima* estão instalados todos os serviços a serem atacados, por exemplo, o servidor *web Apache* para ataques de evasão e inserção. A estação *sniffer* possui como atribuição coletar o tráfego gerado pelas ferramentas de ataque. Para tal, o *sniffer tcpdump* [45] precisa ser instalado nessa estação.

Figura 29 - Ambiente de rede para geração do tráfego dos cenários de teste

O *tcpdump* é uma ferramenta do Unix utilizada para coletar dados de uma interface de rede. Todo o tráfego coletado é armazenado para, posteriormente, ser reproduzido. A fim de realizar essa coleta de dados a linha de comando a seguir deve ser utilizada.

```
$:tcpdump -w nomearquivo
```

O parâmetro *w* indica que os registros estão gravados em formato binário e *nomearquivo* refere-se ao nome do arquivo gravado. Para ler esse arquivo é necessário executar a seguinte linha de comando:

```
$:tcpdump -r nomearquivo
```

```
20:12:06:920000 10.10.10.2:1173 > 10.10.10.8:21: S 72797701: 72797701(0) win 512
```

A saída gerada pelo *tcpdump* segue o seguinte formato: dois dígitos para hora, dois dígitos para minuto, dois dígitos para segundos e seis dígitos para a parte fracionária de um segundo. Em seguida são exibidos os endereços IP e as portas de origem e destino, separados pelo caracter “>” que indica o sentido do fluxo de dados. O *flag* TCP indicado pela letra “S” (SYN) representa uma requisição de conexão e os números (72797701:72797701(0)) representam, respectivamente, o número de

seqüência TCP inicial; e o número de seqüência TCP final. O valor zero entre parênteses corresponde ao número de bytes enviados para uma requisição de estabelecimento de conexão. O último dado fornecido (win 512) é o tamanho do *buffer* TCP da estação destino. Uma vez que todas as estações foram configuradas iniciaram-se as atividades de coleta do tráfego de ataque. O fluxograma abaixo, ilustrado na figura 30, representa a seqüência em que essas atividades foram executadas para cada um dos ataques selecionados.

Figura 30 - Seqüência de atividades a serem realizadas para coletar o tráfego de ataque

Antes de reproduzir um determinado ataque, o *tcpdump* deve ser iniciado, conforme descrito anteriormente. Só então o ataque é executado. Assim que o ataque estiver concluído o *tcpdump* é parado e o arquivo gerado por esse *sniffer* é armazenado para que posteriormente possa ser reproduzido. Essa seqüência de atividades é realizada até que o último ataque tenha sido reproduzido.

4.9.4.2 Geração do tráfego de fundo

Conforme já mencionado, o tráfego de fundo é necessário para realizar a análise de escalabilidade dos SDIs. Inicialmente, foi considerada a hipótese de compor o tráfego de fundo por pacotes UDP de tamanhos (*payload*) variados (256, 512 e 1024 bytes). No entanto, ao realizar os testes de escalabilidade, verificou-se que a variação no tamanho dos pacotes praticamente não influenciou nos resultados obtidos pelos SDIs. Sendo assim, optou-se por utilizar um tráfego de fundo composto somente por pacotes UDP de 256 bytes. Esse tráfego é reproduzido em diferentes taxas de transmissão. A primeira dessas taxas corresponde a 4 (Mbps), taxa na qual os SDI geralmente ainda não descartam pacotes. Colocou-se a reprodução das taxas em uma seqüência linear, por exemplo: 6, 8, 10 e 12 Mbps. Esta metodologia não determina uma taxa limite para reprodução desse tráfego. No entanto, o gerador de pacotes UDP utilizado nos experimentos tem capacidade para reproduzir até 100 Mbps. A ferramenta utilizada para geração do tráfego de fundo é denominada *gerador_udp* e foi desenvolvida pelo Laboratório de Modelagem/Análise e Desenvolvimento de Sistemas de Computação (LAND) da Universidade Federal do Rio de Janeiro (UFRJ). Através dessa ferramenta é possível determinar (a) o endereço IP da estação destino, (b) o tamanho do pacote a ser enviado, (c) a taxa de transmissão em Kbps e (d) o tempo de geração do tráfego.

4.10 Montagem do ambiente de avaliação

Esse ambiente foi composto pelos SDIs a serem avaliados, pelas estações-alvo (*vítimas*) que sofreram os ataques, além de uma estação para reproduzir o tráfego de ataque (*Trf_ataque*) e outra para reproduzir o tráfego de fundo (*Trf_fundo*) ao longo dos testes de escalabilidade.

Figura 31 - Ambiente de rede para o cenário de avaliação

A figura acima representa o ambiente utilizado nessa avaliação, cujos resultados serão apresentados no capítulo 6. Para a avaliação proposta nessa Tese, esse ambiente é adequado. No entanto, a quantidade de estações vítimas e tipo de sistema operacional instalado nessas estações podem variar conforme os ataques selecionados para compor o cenário de avaliação (validação de ataque e defesa). Por exemplo, caso o cenário de avaliação seja constituído por ataques a estações *Solaris* e *Windows 2000 Server*, o ambiente de rede representado na figura acima deverá contar com mais duas máquinas vítimas, nas quais esses sistemas devem estar instalados e devidamente configurados. Além disso, a quantidade de SDIs avaliados também pode variar, por conseguinte o número de estações para esses sistemas deverá ser maior do que ilustrado na figura 31. As estações responsáveis pela geração dos tráfegos de ataque e de fundo devem conter, respectivamente, a ferramenta *tcpreplay* [46] e o gerador de pacotes UDP. O *tcpreplay* é uma ferramenta que possibilita a reprodução dos pacotes capturados, *via tcpdump*. A linha de comando que segue demonstra como proceder para reproduzir um determinado tráfego na mesma taxa em que foi capturado.

4.11 Considerações Finais

Este capítulo foi desenvolvido para realizar um estudo de um dos principais e melhores sistemas de Detecção de Intrusos (SDI), denominado de SNORT. Este *software* foi desenvolvido por Marty no final da década de 90 e hoje é um dos mais populares *softwares* de detecção de intrusos do mundo.

Foram mostrados os principais componentes que compõem a sua arquitetura, como o *Sniffer*, o pré-processador, o motor de detecção e o módulo de alertas e Login do SNORT.

A importância desse tipo de Sistemas de Detecção de Intrusos é grande em razão do avanço da engenharia de eletricidade e os riscos envolvidos quando as informações se concentram em uma rede de computadores.

O SNORT trabalha como um detector de intrusos por meio de detecção por abuso. Isto é caracterizado por determinado tipo de ataque ou padrão de ataque. Para isto, o SNORT contém uma base de assinaturas que constantemente é atualizada.

As atuais pesquisas de detecção de intrusos caminham para a criação de um novo SDI com recursos capazes de lidar com expressões regulares e assim ter maiores chances de identificar um ataque.

Com este capítulo também foi destacada a metodologia para a aplicação dos testes com o *software* SNORT, utilizando para isso máquinas com o Linux.

Na metodologia são definidos os testes a serem realizados no SNORT, testando a capacidade de detecção e a escalabilidade.

Tais atividades de configuração e aplicação de um SDI, como o SNORT que trabalha com a detecção por abuso, operando sobre uma rede de computadores para simular um SEP e realizar ataques de intrusão, são inéditos. A questão da segurança em

SEP ainda é assunto de estudos e pesquisas, este trabalho de doutorado possui o objetivo de tratar da segurança de SEP, usando o SDI SNORT e o “Carcará”.

Foi realizada a interpretação da base de dados dos operadores da subestação, definidos os turnos, as atividades e sub-atividades de cada perfil do usuário. Tais atividades do operador foram desenvolvidas em UML para os casos de uso e demais diagramas que constituem o *software* “Carcará”

O próximo capítulo pretende mostrar como se pode realizar a criação de um perfil de usuário para posterior utilização em um sistema de detecção de intruso utilizando a técnica de detecção por anomalia.

No capítulo 5 está definido o perfil do usuário da subestação de um Sistema Elétrico de Potência (SEP) com os seus casos de uso descrito na linguagem de modelagem unificada (UML), também está a descrição da modelagem em UML do *software* de detecção por intruso baseado em anomalia, o *software* “Carcará”.

5 ESTRUTURA DE UM PERFIL DE UM USUÁRIO DO SEP E O *SOFTWARE* “CARCARÁ”

Este capítulo possui o objetivo de realizar a criação de um perfil de um usuário para um Sistema Elétrico de Potência (SEP), suas principais atividades e o meio de realização destas atividades. Para tanto se utiliza a linguagem de modelagem unificada (UML), descrevendo todos os cenários criados para o operador de um SEP.

Após a descrição das atividades do operador de um SEP é desenvolvida a análise do *software* de detecção de intrusão por meio de anomalia denominado “Carcará”, analisando suas atividades e fazendo a construção das ações e cenários.

5.1 Introdução

Segundo o dicionário universal da língua portuguesa, a palavra usuário é definida como “Aquele que por direito proveniente do uso, possui ou usufrui alguma coisa”. Uma definição abrangente, mas que satisfaz as exigências utilizadas e aplicadas para um usuário em se tratando de um SEP, pois o termo deverá ser utilizado por operadores de uma subestação, pelo administrador do SEP ou da rede de computadores, em que todos de uma maneira ou outra desempenham em algum momento o papel de usuário. A seqüência desse capítulo é desenvolver uma modelagem das tarefas executadas pelo usuário. Em seguida será destacado um *software* que poderá realizar o acompanhamento desses usuários por meio de uma rede neural, sendo que ao realizar alguma ação de anormalidade no sistema de atividades de uma subestação, o *software* que é um SDI por anomalia irá disparar um alarme ao administrador de rede ou ao responsável pela segurança da subestação.

5.2 Alguns Trabalhos de Modelagem do Usuário

O Modelo do Usuário representa características próprias e típicas de um usuário individual armazenado no sistema (log ou histórico). É um dos componentes mais importantes na implementação de um SDI onde se quer implantar interação otimizada e padronizada de atividades voltadas para detecção de intrusos via anomalia. Será destacado aqui o principal aspecto genérico dos Sistemas de Modelagem do Usuário, bem como seu Histórico, sua Evolução e sua importância para a segurança de uma rede de computadores.

Iniciamos com a modelagem do usuário que começou com os trabalhos de Allen, Cohen e Perraul [47], [48], [49] e Elaine Rich [50]. Por uma década foram desenvolvidos sistemas embrionários que coletam diferentes tipos de informação e diferentes espécies de adaptação aos seus atuais usuários [51], [52].

No ano de 1986, Tim Finin publicou o “*General User Modeling System – GUMS*” (Sistema Universal de Modelagem do Usuário) [53]. A abordagem desenvolvida nesse trabalho permitiu aos programadores de aplicações adaptáveis ao usuário a definição de simples hierarquias estereotipadas para cada estereótipo, fatos estes que poderiam ser descritos em PROLOG (estereótipos e regras).

Aparentemente, Kobsa [54] é o primeiro autor a utilizar o termo “Sistema *Shell* de Modelagem do Usuário”, para distintas espécies de ferramentas de *software*. O termo “Sistema *Shell*”, ou “*Shell*” abreviado, advindo da área de Sistemas Especialistas, onde Van Melle [55] concentrou as experiências feitas com os sistemas especialistas na área médica, Mycin [56]. Um sistema especialista “vazio” que tinha de ser preenchido com regras específicas do domínio para ser entendido como um sistema especialista “real”.

O objetivo geral que fundamenta a tendência de *Shells* de modelagem do usuário é a decomposição do *software* e a abstração para sustentar a capacidade de modificação e reusabilidade.

5.3 Exemplos de *Shells* de Modelagem do Usuário

Quando teve início a década de 90, muitos grupos de pesquisa em vários lugares começaram a construir estruturas básicas e processos em *Shells* de modelagem do usuário que achavam ser importante para sistemas adaptáveis. Como exemplo, podemos citar alguns modelos criados para a modelagem do usuário:

- a) BGP-MS [57] é o sistema mais antigo. Este sistema permite a realização de deduções sobre o usuário e deduções estereotípicas do grupo de usuários a serem representadas em um predicado de lógica de primeira ordem. Um subconjunto dessas deduções é armazenado em uma lógica terminológica. O sistema pode ser usado como um servidor de rede com capacidades multi-usuários e multi-aplicações;
- b) UMT-MS [58] permite ao desenvolvedor do modelo do usuário a definição de estereótipos do usuário ordenados hierarquicamente, e de regras para inferências do modelo do usuário, assim como a detecção de contradições. A Informação sobre o usuário é classificada no sistema como premissas invariáveis ou deduções. Logo após o disparo de todas as regras de inferência aplicáveis e aplicação de todos os estereótipos aplicáveis, contradições entre deduções são procuradas e várias estratégias de resolução aplicadas. Em outras palavras, ocorre o que se denomina de “manutenção da verdade”;
- c) DOPPELGANGER [59] também é um servidor de modelagem de usuário que aceita a informação sobre o usuário através de sensores de *hardware* e *software*.

- Ele utiliza técnicas para dedução e inferência de dados dos sensores (predição linear, modelos de Markov, e agrupamento não supervisionado para a criação de estereótipos). Os usuários podem inspecionar e editar seus modelos de usuário;
- d) Em TAGUS [60], o modelo representa deduções sobre o usuário fórmulas de primeira ordem, com operadores meta expressando os tipos de deduções. Este sistema permite a definição de uma hierarquia de estereótipos, tendo um mecanismo de inferência, um sistema de manutenção da verdade, e um subsistema de diagnóstico que incluem uma biblioteca de conceitos denominados errados ou equivocados. Ele permite ter suporte à “simulações do usuário” através de inferências direcionadas a frente (*forward-directed*) com bases no modelo do usuário, e diagnóstico de comportamentos inesperados do usuário;
- e) Em UM [61], este modelo possui um conjunto de ferramentas para modelagem do usuário que representa deduções sobre o conhecimento, crença, preferências do usuário e características de outros usuários em pares de atributos-valor. Cada componente de informação é acompanhado por uma lista de evidências para sua verdade e falsidade. A fonte de cada pedaço de evidência, seu tipo (observação, ativação de estereótipo, invocação de regra, entrada do usuário) e uma marca de tempo (*time stamp*) também são gravados.

5.4 Características Importantes para um Sistema de Modelagem do Usuário

As decisões, assim como importantes estruturas e processos, devem fazer parte dos sistemas *Shell* de modelagem do usuário, em grande parte foi baseada em intuições e/ou experiência dos desenvolvedores de *Shell*, mediante trabalhos anteriores construídos em sistemas adaptáveis para o usuário [62], [63].

Em uma tentativa de estender a definição de fato dos sistemas *Shells* de modelagem do usuário introduzidas pelo GUMS e evitar a caracterização de sistemas *Shell* de modelagem do usuário em processos e estruturas internas, procurou-se os serviços freqüentemente encontrados nesses sistemas:

- a) A representação de deduções sobre um ou mais tipos das características do usuário em modelos de usuários individuais (deduções sobre seus conhecimentos, concepções erradas, objetivos planos, preferências, tarefas, habilidades etc.);
- b) A representação de características comuns relevantes dos usuários pertencentes a subgrupos de usuários específicos do sistema (estereótipos);
- c) A classificação dos usuários como pertencentes a um ou mais destes subgrupos, e a integração de características típicas destes subgrupos no modelo individual do usuário atual;
- d) A gravação do comportamento dos usuários, particularmente suas interações passadas com o sistema;
- e) A formação de deduções sobre o usuário baseadas no histórico de interações;
- f) A generalização dos históricos de interações de vários usuários em estereótipos;
- g) O esboço de deduções adicionais sobre o usuário atual;
- h) Manutenção da consistência no modelo do usuário;
- i) A provisão das atuais deduções sobre o usuário, assim como justificativas para estas deduções;
- j) A avaliação das entradas no modelo do usuário atual, e a comparação com padrões fornecidos;

Vários requisitos para sistemas *Shell* de modelagem do usuário foram observados como importantes:

- a) Generalidade, incluindo independência de domínio – com o aumento nas aplicações dos Sistemas *Shell*, abrangendo vários domínios de satisfação e dentro destes domínios para um maior número possível de tarefas de modelagem de usuário. Desta forma, eles passaram a fornecer um maior número possível de serviços. “Concessões” a esse respeito foram apenas feitas para sistema *Shell* em sistemas tutoriais adaptáveis a estudantes, sendo que a expectativa inicial era de aplicação somente no ensino de assuntos, mas não para aplicações de domínios adicionais além dos educacionais;
- b) Expressividade – Esperou-se que os sistemas *Shell* fossem capazes de expressar o maior número possível de deduções sobre os usuários possíveis ao mesmo tempo. Isto não apenas estaria incluindo os diferentes tipos de atitudes proposicionais anteriormente citadas, mas também os tipos de deduções reflexivas para o usuário e ao sistema [64], e ainda incerteza e imprecisão com as deduções;
- c) Fortes Habilidades de Inferência – A esperança era que os sistemas *Shell* fossem capazes de realizar todos os tipos de raciocínio que são normalmente distinguidos na inteligência artificial e lógica modal, assim como o raciocínio no predicado de lógica de primeira ordem, raciocínio complexo modal, raciocínio com incerteza, raciocínio razoável quando não se possui informação completa, e resolução de conflitos quando deduções contraditórias são encontradas.

Em termos gerais havia uma expectativa que os *Shells* de modelagem do usuário pudessem dar suporte a deduções e raciocínios complexos sobre usuários que tenham sido identificados nesses domínios e que em outro momento ou posteriormente pudesse ser adicionado para ser aplicada em uma maior variedade de áreas.

Quando, na metade dos anos 90, houve uma mudança dos sistemas adaptáveis ao usuário para diferentes domínios com modelagens de usuários com requisitos menos exigentes com ambientes de aprendizagem adaptáveis ao usuário [65] e *web sites* personalizados, tais modelagens complexas de usuário e habilidades de raciocínio se tornaram redundantes. Um agravante a mais seria o de que as aplicações comerciais necessitavam de serviços e requisitos adicionais que estavam desprovidos na pesquisa orientada a *Shells*.

Algumas das idéias que foram inicialmente exploradas nestes protótipos de sistemas (particularmente a abordagem de estereótipos e arquitetura cliente-servidor) conseguiram chegar ao *software* comercial de modelagem de usuário.

De agora em diante, usaremos o termo “servidor de modelagem do usuário” para nos referenciarmos a estes sistemas comerciais de modelagem do usuário. Já que o termo *Shell* se tornou ultrapassado, nós utilizaremos o termo “sistema da modelagem do usuário (genérico)” para referenciarmos a qualquer sistema genérico que ofereça serviços de modelagem do usuário em tempo de execução que possam ser configurados em tempo de desenvolvimento.

Servidores comerciais de modelagem do usuário devem dar suporte a serviços que de certa forma são diferentes daqueles que se esperavam dos *Shells* acadêmicos de modelagem do usuário. Abaixo estão alguns exemplos destes novos serviços de modelagem do usuário:

- a) Comparação de ações seletivas de diferentes usuários. Em certas áreas de aplicação as escolhas do usuário não podem ser muito bem reconstruídas por processos *step-wise* de raciocínio e sim apenas por referências a conceitos vagos como o gosto do usuário, sua personalidade e estilo de vida (ex. incluem a seleção de músicas, livros, roupas e restaurantes). Em tais domínios, achou-se

útil emparelhar ações seletivas dos usuários (compra de itens, sua visualização, colocá-los em carrinhos de compra, classificá-los com alta qualidade) com as de outros usuários, e prever futuras ações seletivas dos usuários com base nas de maior similaridade dos outros usuários. Vários servidores comerciais de modelagem do usuário atuais, conseqüentemente dão suporte à comparação de diferentes padrões de ações dos usuários, utilizando algoritmos de filtragem “colaborativos”;

- b) Importação da informação externa referente ao usuário - Muitos negócios já possuem dados referentes ao *marketing* e a clientes, e geralmente é desejado que sejam integrados a sistemas de modelagem do usuário, quando dá início a personalização do comércio eletrônico. Para acessar dados externos, interfaces ou suportes nativos para uma grande variedade de bancos de dados são necessários. Devido ao legado dos processos de negócio e *software*, informações externas relacionadas ao usuário, freqüentemente continuam sendo atualizadas em paralelo à aplicação de comércio eletrônico, por isso, necessitam ser continuamente integradas a custos razoáveis e sem atrasar o tempo de resposta;
- c) Suporte à privacidade. As políticas e normas de privacidade de empresas e indústrias, legislação de privacidade nacional e internacional, convenções, e ferramentas de *software* de suporte à privacidade e provedores de serviços estão emergindo lentamente. Enquanto isto não é o caso, servidores de modelagem do usuário devem dar suporte a qualquer política de privacidade que obedeça a estas restrições e tornar possível a obtenção de vantagens dos principais *softwares* e serviços de privacidade que estão disponíveis no mercado.

Os atuais servidores de modelagem do usuário são em sua maioria orientados a comportamento. As ações observadas dos usuários ou padrões de ações freqüentemente

levam a adaptações, sem uma representação explícita das características do usuário (interesses, conhecimento, planos etc.), o que provavelmente causa este comportamento e justifica estas adaptações. Ao se fazer estas deduções, explicitamente se permite que o sistema de modelagem do usuário as empregue em outros propósitos além dos quais foram gravados, como foi o caso dos *Shells* de modelagem dos usuários clássicos.

Os atuais servidores de modelagem do usuário possuem uma fraca avaliação de características, como dimensões de generalização, expressividade e capacidades de inferência, das quais todas foram consideradas como sendo importantes para os *Shells* acadêmicos de modelagem do usuário. Em vários casos, eles são dependentes de domínio, sua representação do modelo do usuário é mesclada com considerações referentes ao processamento, e podem apenas ser utilizados para limitados propósitos de personalização. De qualquer maneira, estas características clássicas de qualidade não são mais consideradas como sendo importantes para servidores comerciais de modelagem do usuário.

As suas características atualmente mais desejadas são apresentadas a seguir:

- a) Adaptação rápida - Vários sistemas comerciais de modelagem do usuário podem selecionar mais de uma modelagem e métodos de personalização com diferentes graus de complexidade, dependendo da quantidade de dados que já se tem disponível do usuário;
- b) Capacidade de extensão - Os atuais servidores de modelagem do usuário dão suporte a um número de aquisições de modelos de usuário e métodos de personalização, mas empresas podem querer integrar seus próprios métodos ou ferramentas de terceiros;
- c) Equilíbrio de carga - Sob as condições do mundo real, servidores de modelagem do usuário passarão por mudanças dramáticas em sua carga normal. Notáveis

atrasos de resposta ou até a negação de requisições devem ocorrer apenas em situações de emergência. Os servidores de modelagem do usuário devem poder reagir a aumentos na carga através da distribuição de carga, que possam ser distribuídos pela rede de computadores e possivelmente através de uma menos completa análise do modelo;

- d) Estratégias de falhas - Arquiteturas centralizadas necessitam prover mecanismos de recuo em caso de danos;
- e) Consistência transacional - A leitura escrita no modelo do usuário e um fim anormal de processos podem levar a inconsistências que devem ser evitadas através de uma escolha cuidadosa de estratégias de gerenciamento de transações.

5.5 Tendências nos Sistemas de Modelagem do Usuário

É importante salientar que as predições referentes ao futuro dos sistemas de modelagem do usuário são especulativas, devido às rápidas mudanças que ocorrem na computação. Já que a personalização demonstrou ser capaz de beneficiar ambos usuários e provedores de serviços personalizados, é praticamente certo que os sistemas de ferramentas genéricos que permitam ou facilitem o desenvolvimento e manutenção de sistemas personalizados também serão necessários no futuro.

A maneira exata pela qual os sistemas de modelagem do usuário, um dia podem se tornar um padrão, é provável que seja por influencia de muitas características. Aqui estão listadas algumas considerações referentes a prováveis caminhos futuros.

- a) Modelos móveis do usuário - A computação está cada vez mais se tornando móvel, mas em um futuro próximo a confiança em redes móveis não atenderá às demandas impostas pelas arquiteturas cliente-servidor para sistemas de modelagem de usuário, os quais requerem conectividade permanente. É provável

que a computação e cenários de informação ambíguos levem em consideração os modelos de usuários móveis. Estes agentes de modelo do usuário podem residir do lado do servidor e serem replicados no início de cada interação, ou eles podem ser verdadeiros agentes móveis e permanecerem com o usuário o tempo todo;

- b) Modelos do usuário para dispositivos inteligentes - Recentemente estão sendo oferecidos dispositivos que possibilitam uma personalização limitada. Alguns exemplos incluem rádios automotivos que possuem um cartão *chip* que contém um código de segurança e também armazena as preferências do motorista, levando em consideração as estações de rádio sintonizadas, volume e tom, e notícias de tráfego. Existem chaves eletrônicas de carros que ajustam o banco do carro, os espelhos e o sistema GPS (*Global Positioning Systems*) às preferências individuais do motorista, quando colocadas na ignição do carro;
- c) *Multiple-purpose usage* - Informações sobre as características individuais dos usuários podem ser interessantes não apenas para propósitos de personalizações. Outras possíveis aplicações incluem serviços organizacionais dirigidos, sistemas de habilidades criativas e aplicações de procura de especialistas globais ou organizacionais. Considerações referentes a um servidor de modelo do usuário central versus um agente de modelo do usuário local também podem ser feitas a respeito destes tipos aplicações de YIMAM [66].

Como consequência desta quantidade de possíveis aplicações das informações sobre os usuários, não é provável que em um futuro próximo haja um único ou pequeno número de sistemas de modelagem do usuário apropriados para um grande número de tarefas de modelagem do usuário. Em vez disso, pode-se esperar encontrar uma grande variedade de sistemas genéricos de modelagem do usuário, dos quais somente darão

suporte a algumas das diferentes manifestações da personalização e outras aplicações sobre usuários.

A seguir temos as atividades do operador de uma subestação que terá suas atividades analisadas por meio de uma Rede Neural Artificial, que verificará se existe alguma anomalia ou não nas ações do operador.

5.6 Definição de Atividades do Operador de uma Subestação de um SEP

Este *software* possui como premissa de que o sistema de detecção de Intrusão por anomalia, denominado aqui de “Carcará”, será utilizado em uma subestação totalmente digital. Com isso a função do operador limitar-se-á a observar o desenvolvimento das atividades do sistema elétrico de potência, intervindo somente quando for essencialmente necessário. Os diagramas a seguir do *software* foram desenvolvidos em UML.

Figura 32 – Caso de Uso de Atividades do Operador

Baseado nessas premissas chegou-se à conclusão de que o operador possui cinco casos de uso (figura 32), são eles:

- a) Realizar a manipulação de Eventos Sequenciais;
- b) Leitura de Parâmetros e Configuração;
- c) Realizar a Indicação de *Status*;
- d) Avaliação de Valores e Checagem de Limites;
- e) Operação de Equipamentos (alta tensão e controle).

Vamos especificar o que ocorre em cada um dos casos de uso, para compreendermos melhor o papel do operador.

a) Realizar a manipulação de Eventos Sequenciais

Neste caso de uso o operador realiza a manipulação de eventos sequenciais do Sistema Elétrico de Potência (SEP) através de equipamentos digitais. Tais ações podem ser para manter a normalidade do SEP ou para que possam retornar a sua normalidade do SEP. O cenário em que este operador está realizando essas atividades está descrito a seguir.

Cenário Básico

1. Iniciar o caso de uso;
2. Iniciar a Manipulação de eventos sequenciais;
3. Efetivar as ações dos eventos sequenciais;
4. Verificada alguma anormalidade no caso de uso “Leitura de Parâmetros e Configuração”, fazer passo 6, caso contrário passo 5;
5. Efetuar ações para restaurar das operações normais;
6. Realizar ações padrão para continuidade da normalidade;

7. Encerrar a “Realizar a Manipulação de Eventos Seqüenciais”;
8. O caso de uso encerra com sucesso.

Esse tipo de cenário é chamado de cenário básico, pois nele tudo funciona, isto é, as atividades funcionam de maneira perfeita. Porém, é necessário ressaltar que obviamente existem os cenários alternativos que constroem todo o caso de uso, isto se repetirá em todos os casos de uso que aqui serão mencionados.

b) Leitura de Parâmetros e Configuração

Neste caso de uso o operador realiza a leitura de parâmetros dos equipamentos como também realiza configurações no Sistema Elétrico de Potência (SEP). O contexto envolvido é o do operador realizar a leitura de valores (*Status*) dos equipamentos e realizar configurações se for necessário. O cenário em que este operador está realizando as suas atividades está descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Verificar os valores de fluxo de carga;
3. Realizar o caso de uso “Realizar a Indicação de *Status*”;
4. Efetivar a leitura dos equipamentos de detecção de Falta;
5. Realizar o caso de uso “Realizar a Indicação de *Status*”;
6. Encerrar a “Leitura de Parâmetros e Configuração” (Relés de Proteção);
7. O caso de uso encerra com sucesso.

c) Realizar a Indicação de *Status*

Neste caso de uso o operador realiza a avaliação de *Status* do Sistema Elétrico de Potência (SEP) e caso haja alguma anormalidade, realiza ações que possam retornar a

normalidade do sistema ou envia as informações de anormalidade ao responsável em realizar ações para a normalidade do Sistema Elétrico de Potência.

O cenário em que este operador está realizando as suas atividades está descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Verificar o relatório de um ou de um grupo de equipamentos;
3. Observar o *Status* desses equipamentos;
4. Fazer o caso de uso “Avaliação de valores e Checagem de Limites”;
5. Enviar os resultados obtidos para o responsável para a tomada de alguma ação;
6. Realizar a ação (operador);
7. O caso de uso encerra com sucesso.

d) Avaliação de Valores e Checagem de Limites

Neste caso de uso o operador realiza a avaliação de valores numéricos dos equipamentos por meio da checagem de limites desses equipamentos. O operador escolhe o equipamento (ou grupo) a ser avaliado e verifica os valores para autenticação de normalidade. O cenário em que este operador está realizando as suas atividades está descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Escolher um equipamento para realizar a checagem de limites;
3. Avaliar os valores lidos do equipamento;
4. Realizar a tomada de decisão baseado nos valores obtidos no equipamento;

5. Observado que os valores são normais e seguros, ir para o passo 6;
6. Encerrar a avaliação de valores e checagem de limites do equipamento;
7. O caso de uso encerra com sucesso.

e) Operação de Equipamentos (alta tensão e controle)

Neste caso de uso o operador realiza a operação de Equipamentos sejam eles de Alta Tensão ou Controle; para isto o equipamento necessita estar em perfeito funcionamento. O operador escolhe o equipamento a ser realizada alguma atividade, em seguida realiza a leitura desse equipamento para depois iniciar a operação desse equipamento. O cenário em que este operador está realizando as suas atividades está descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Escolher um equipamento para realizar a operação;
3. Verificar os valores quantitativos do equipamento escolhido;
4. Iniciar a ação de operação do equipamento;
5. Observar o efeito da ação realizada no equipamento;
6. Continuar a ação de Operação do equipamento;
7. Enquanto a operação do passo 6 não finaliza realizar o passo 5;
8. Encerrar a operação do equipamento;
9. O caso de uso encerra com sucesso.

A análise dos casos de uso pode ser feita por meio do Diagrama de Colaboração “Avaliação de Valores e Checagem de Limites”. Nesse diagrama (figura 33), o operador

inicia o caso de uso escolhendo o equipamento a ser avaliado para checagem de limites, o equipamento é escolhido de uma base de equipamentos. A base de equipamentos solicita todos os dados do equipamento na base de dados do equipamento, este por sua vez as envia a central de controle do sistema que os mostra ao operador. Diante dessas informações esse operador realiza a tomada de decisão com relação ao equipamento.

Nesta ocasião poderia haver vários cenários alternativos para a atuação do operador neste caso de uso. Pode ocorrer a situação de não se encontrar dados suficientes dos equipamentos até a situação extrema de não encontrar nem o equipamento cadastrado na base de equipamentos (banco de dados).

Figura 33 – Diagrama de Colaboração “Avaliação de Valores e Checagem de Limites”

O segundo caso de uso em estudo é “Leitura de Parâmetros e Configurações (Relés de Proteção)” (figura 34). Nele o operador inicia verificando os valores de fluxo de carga, o sistema de controle solicita essas informações ao caso de uso “realizar Indicação de *Status*”.

Figura 34 – Diagrama de Colaboração “Leitura de Parâmetros e Configurações (Relés de Proteção)”

Este mesmo caso de uso reporta as informações ao sistema de controle, o operador tendo observado essas informações por meio do sistema de controle, realiza a leitura de equipamentos de detecção de falta.

No caso de uso seguinte “Operação de Equipamentos (Alta Tensão e Controle)” (figura 35), o operador inicialmente escolhe o equipamento. Escolhido o equipamento, o passo seguinte é avaliar os dados do equipamento e o sistema de controle busca as informações na base de Dados. Agora a base de Dados realiza uma busca e os envia para o controle, que os mostra ao operador. O operador depois então inicia o manuseio do equipamento, sendo que enquanto o equipamento está sendo analisado, os números desse manuseio estão sendo transmitidos e armazenados na base de Dados do equipamento.

Figura 35 – Diagrama de Colaboração “Operação de Equipamentos (Alta Tensão e Controle)”

O diagrama de Colaboração “Realizar a Indicação de *Status*” (figura 36), mostra o operador iniciando a leitura de um equipamento ou de um grupo de equipamentos similares. O sistema de controle recebe as ordens e então começa a leitura das informações provenientes da base de Dados.

Figura 36 – Diagrama de Colaboração “Realizar a Indicação de *Status*”

O operador avalia os valores, e nesse momento o caso de uso “Avaliação de valores e Checagem de Limites” é solicitado. O sistema de controle recebe as informações e os remete ao caso de uso “Avaliação de Valores”. O caso de uso “Avaliação de valores” envia ao sistema de controle o resultado. O sistema de controle

mostra esses valores ao operador, que toma as ações necessárias conforme os valores mostrados pelo sistema de controle.

A figura 37 nos mostra o diagrama de colaboração do caso de uso “Realizar a Manipulação de Eventos Sequenciais”. Inicialmente o operador escolhe a manipulação de eventos sequenciais. Com isso o passo seguinte é a tarefa de realizar as ações de eventos sequenciais. O sistema de controle verifica então a normalidade por meio do caso de uso “Leitura de parâmetros”. Caso haja alguma anormalidade, o sistema de controle inicia o processo de atividade da base de “Ações para retornar Normalidade” e caso tudo esteja bem, realiza a base “Ações para continuar Normalidade”.

Figura 37 – Diagrama de Colaboração “Realizar a Manipulação de Eventos Sequenciais”

Esses diagramas poderiam ser outros, como os diagramas de seqüência, estado ou de classes. No entanto, optou-se pelo diagrama de colaboração por serem simples e de fácil compreensão.

A seguir a especificação da base de atividades e sub atividades dos operadores da subestação.

5.7 Base de dados do operador do *Software* “Carcará”

A base de dados possui diversos itens para identificar o turno de trabalho (Figura 38) do operador, o perfil de saída do operador, as atividades realizadas e as suas sub-atividades desenvolvidas por ele [67].

Figura 38 – Os quatro turnos que os operadores trabalham (cada turno de 6 horas)

As figuras 38 e 39 mostram uma adequação a este tipo de base de operador. Os turnos, cada um deles de 06 horas, ficaram divididos em 04 turnos (Turno 01, 02, 03, 04).

Optou-se por definir 04 tipos de usuários, sem destacar nenhum deles propriamente por nomes e sim por atividades que cada perfil exerceria no decorrer do uso do *software* carcará. Os 04 tipos de usuários (Figura 39) em questão são:

Usuário **A1** – Avaliação e checagem de valores (Atividade **80**)

Usuário **A1** – Realizar a Indicação de *Status* (Atividade **90**)

Usuário **B1** – Realizar a manipulação de eventos sequenciais (Atividade **50**)

Usuário **C1** – Leitura de parâmetros e Configuração (Atividade **70**)

Usuário **D1** – Operação de equipamentos (Alta tensão e controle) (Atividade **60**)

Saída do Usuário

Figura 39 – Os quatro tipos de perfis de usuários (operadores)

Os pesos da rede neural ficam associados a cada tipo de usuário, sendo eles **A1**, **B1**, **C1**, **D1**. Por exemplo, a cada usuário “X” é especificado o perfil a ser aplicado para este usuário. Quando necessário depois este usuário é promovido a outros perfis já determinados e especificados para as operações da subestação.

A quantidade de perfis é independente da quantidade de usuários que estejam realizando operações e serviços na subestação, ou seja, permanece 04 perfis, mesmo tendo 1 a N operadores para cada perfil. Obviamente a carga da subestação e a quantidade de operações será o fator determinante na quantidade de operadores, ações e serviços (Tabela 16).

Tabela 16 – Operador (usuário) suas atividades e código da atividade

Usuário	Atividade	Código de Atividade
A1	- Realizar a Indicação de <i>Status</i> ;	80
	- Avaliação de valores e Checagem de Limites;	90
B1	- Realizar a manipulação de Eventos Sequenciais;	50
C1	- Leitura de Parâmetros e Configuração;	70
D1	- Operação de equipamentos (Alta Tensão e Controle)	60

As atividades de cada perfil obedecem aos casos de uso mencionados e especificados no início deste tópico sobre as atividades do operador por meio da UML (figura 40).

Figura 40 – As cinco atividades Principais dos usuários (operadores)

Existe uma informação a destacar mediante a utilização dessas atividades principais pelo operador da subestação. Essas atividades do operador contêm o limite máximo de 9.999 subatividades que podem ser cadastradas (figura 41). Obviamente os casos de uso das atividades podem ser adicionados de acordo com a expansão da tecnologia utilizada para o controle e operação das subestações.

Figura 41 – As subatividades dos usuários (operadores)

O *log* das atividades do operador da subestação fica, portanto, com a seguinte configuração. Uma ação composta de 8 (oito) dígitos e com uma saída de algum dos perfis calculados, A1, B1, C1 ou D1.

A figura 42 ilustra esta situação, tendo a base de dados os nomes dos operadores que estão cadastrados para fazerem a saída A1 (perfil do operador).

Figura 42 – A ação do usuário e a saída do usuário (operador)

Nesta situação pode-se verificar que os dois primeiros dígitos (figura 43) representam o turno do operador, o terceiro e quarto dígito identificam a atividade que o operador está realizando, e os quatro últimos dígitos identificam a sub atividade que este operador selecionou para operar no sistema da subestação.

Figura 43 – A ação do usuário (operador)

O desenvolvimento do *software* “Carcará” foi desenvolvido por meio da linguagem C++.

5.8 A estrutura do *Software* “Carcará”

O *software* “Carcará” é um *software* para identificar os pesos do perfil do operador, por meio de uma rede neural. A utilização da rede neural se dá através do “Joone”, *software free* que permite calcular o padrão de comportamento do usuário. Essa base de dados é estruturada originalmente na base de dados do *software* EPOCHS, sendo que o *Log* de atividades original do EPOCHS não possuía as ações do operador e foram inseridas posteriormente manualmente.

Figura 44 – Estruturação da ligação do Programa

O *log* para teste do *software* estará armazenado em um banco de dados em MYSQL (figura 44). O sistema operacional Linux será a plataforma inicial de testes do *software*, sendo que o mesmo foi também testado no Windows.

Figura 45 – Estruturação da ligação do Programa II

O *software* Joone (figura 45) possui a missão de realizar o teste no log das atividades do operador e descobrir os pesos associados à rede neural. O sistema “Carcará” de posse dos pesos realiza a identificação e faz o processamento do perfil do usuário.

A seguir serão descritas as atividades relacionadas com o sistema de detecção de Intrusos por Anomalia por meio do *software*, “Carcará”, utilizando o UML.

5.9 Definição de casos de uso do Sistema de Detecção de Intruso “Carcará”

Inicialmente foram divididas em casos de uso, as principais atividades do operador de uma subestação digital, num total de cinco casos de uso (figura 46). Os casos de uso foram:

- a) Realizar Auditoria;
- b) Sinalizar Alertas;
- c) Operação de atividades;
- d) Configurar Direitos;
- e) Imprimir Relatórios.

Figura 46 – Caso de uso do “Sistema de Atividades do Operador”

O ator principal denominado de “Sistema” depende dos casos de uso, para que possa funcionar. Isso também inclui o *software* Joone (RNA para o perfil) indicado na figura 46 como *software* externo. As setas tracejadas indicam a dependência do sistema em relação aos casos de uso e do ator externo.

Será citado cada um desses casos de uso e a sua finalidade, mostrando o cenário em que cada um deles ocorre. Em seguida serão inseridos os diagramas de colaboração dessas ações.

É importante destacar inicialmente que os operadores foram divididos em 3 níveis de acesso. Sobre cada uma das atividades desempenhadas por cada um deles, existem várias subatividades. Na prática, existem somente dois níveis, tendo a subestação dois operadores, um deles podendo assumir o papel de líder (nível 3), sendo que para isso, ele deverá assumir o *login* de líder ou gerente (nível 3) das ações ali executadas e o outro como operador comum (nível 1 ou 2). Posteriormente, poderá ser adicionada no *software*, a necessidade de executar ações anormais para as atividades da subestação, haverá a necessidade da colocação da senha do líder (gerente ou responsável pelo turno) das ações para que a operação seja executada, o que é comumente chamado de duas senhas.

Vamos à explicação (coloquial) dos casos de uso do operador.

a) Caso de uso Realizar Auditoria

O caso de uso por meio do operador de nível 3 realiza a auditoria de operadores de nível 1 ou 2, bem como auditoria de equipamentos e atividades relacionadas às atividades realizadas no SEP. O contexto envolvido nesse caso de uso é o operador de nível 3 ao realizar a Auditoria dos equipamentos (base de dados), atividades e operadores do SEP. O cenário de ações ideais é descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Realizar o login no sistema com o ID e senha de operador de nível 3;
3. Escolher a atividade, equipamento ou operador para realizar uma auditoria;
4. Análise do histórico das atividades, realizada pelo operador de nível 3;
5. Concluir um relatório ou laudo sobre as atividades realizadas (Realizada pelo operador de nível 3);
6. Executar ações baseadas nas atividades analisadas pelo operador de nível 3;
7. O caso de uso encerra com sucesso.

b) Caso de uso Sinalizar Alertas

Neste caso de uso o sistema realiza alerta para o operador de nível 3 sobre atividades não comuns realizadas pelos operadores de nível 1 e 2. No contexto envolvido, o sistema dispara alerta para o operador de nível 3, quando ocorrem atividades que fogem do padrão de atividades dos operadores 1 e 2. O cenário de ações ideais é descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Operação de alguma atividade pelo operador que não foi detectada pelo sistema e que necessita ser concluída. Para esta situação, o Sistema necessita de duas chaves (ID e Senha dos dois operadores para realizar ou confirmar a atividade);
3. Aceitação do sistema da atividade baseada na confirmação das duas chaves (dois operadores). No entanto, a atividade foi monitorada pelo sistema causando um sinal de alerta ao administrador da subestação;

4. Imprimir um relatório da atividade anormal, detalhando o início, a finalidade, a utilização das chaves (duas) e o efeito causado pela ação não detectada pelo sistema;
5. Encerramento da atividade anormal;
6. O caso de uso encerra com sucesso.

c) Caso de uso Operação de atividades

Neste caso de uso, o operador de nível 1 e 2 realiza as atividades relacionadas com o SEP, tendo cada operador direitos e um perfil a ser analisado pelo Sistema. O contexto do caso de uso é quando o operador realiza as atividades inerentes ao seu perfil no SEP digital. O cenário de ações ideais é descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Realizar uma atividade a ser escolhida pelo operador;
3. Escolher uma subatividade;
4. Fazer a confirmação das subatividades e atividades pelo operador do SEP;
5. Aguardar o *Status* de sucesso ou falha das ações;
6. O caso de uso encerra com sucesso.

d) Caso de uso Configurar Direitos

Neste caso de uso o operador de nível 3 realiza a configuração dos direitos exercidos e gerenciados por ele, para cadastrar, alterar ou eliminar os direitos de outros operadores. Este nível de operador (3) pode ser considerado o administrador do sistema para verificação de atividades e segurança do SEP. O contexto do caso de uso é a do

operador gerenciar os direitos dos outros operadores do SEP digital. O cenário de ações ideais é descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Iniciar a operação do Sistema por meio do operador de nível 3 para cadastrar ou alterar os dados de um usuário;
3. Realizar a confirmação ou não por meio do operador de nível 3 no cadastramento ou alteração dos dados do usuário;
4. Efetivar a finalização do módulo do sistema por meio do operador de nível 3;
5. O caso de uso encerra com sucesso.

e) Caso de uso Imprimir Relatórios

Neste caso de uso o operador realiza a impressão de dados referentes às atividades realizadas automaticamente pelo sistema da subestação, atividades realizadas pelos operadores da subestação ou simplesmente atividades de um determinado equipamento do SEP. O contexto desse caso de uso é quando o operador escolhe o equipamento (ou grupo), operador ou atividades a serem mostradas na tela do computador ou por meio de impressão. Como observação, somente o operador de nível 3 pode realizar auditorias sobre os outros operadores. O cenário de ações ideais para esse caso de uso é descrito abaixo.

Cenário Básico

1. Iniciar o caso de uso;
2. Acessar o módulo de impressão (operadores de nível 1, 2 e 3);

3. Escolher o equipamento, atividade ou operador (somente o operador nível 3, pode acessar este modo) a ser impressa, especificando ou não a data, hora para o relatório (caso não seja especificada a hora, será impresso o relatório das últimas 24 horas);
4. Aguarda o término da impressão do relatório;
5. Analisar os dados do relatório e realizar ações de reação aos dados do relatório;
6. O caso de uso encerra com sucesso.

5.10 Diagramas de Colaboração dos casos de uso do Sistema de Detecção de Intrusão “Carará”

Para os casos de uso do operador, foi criado um diagrama de colaboração para cada um desses casos de uso, verificando-se cada um dos passos executados pelo sistema e pelo operador.

A figura 47 mostra o diagrama de colaboração “Operação de Atividades”. A primeira ação tomada é escolher o módulo de atividades na interface. Depois disso, a interface transmite para a central de controle que realiza uma busca na base de atividades (banco de dados) o módulo de atividade em funcionamento. Uma vez carregadas as atividades, o operador agora escolhe a atividade e a subatividade que ele deseja realizar. No entanto, é necessário especificar que nem todas as atividades e conseqüentes subatividades estarão disponíveis ao operador. O que vai definir uma atividade ou outra será o nível desse operador (1, 2 e 3) no sistema.

Figura 47 – Diagrama de Colaboração “Operação de Atividades”

A figura 48 mostra o diagrama de colaboração do caso de uso “Configuração de direitos”. Inicialmente o operador administrador ou líder cadastra os dados do usuário (outro operador de menor ou igual nível), depois a interface recebe essas informações e as transmite ao sistema de controle que possui a responsabilidade de armazená-los na base de Dados do Usuário.

Figura 48 – Diagrama de Colaboração “Configuração de Atividades”

A figura 49 destaca o diagrama de colaboração “Sinalizar Alertas”. Nesse caso de uso, o sistema realiza um alerta de atividade anormal, realizada pelo operador ou pelo sistema. Com este alarme é destacado o início da anormalidade, as consequências das ações no sistema. Sobre essa situação totalmente anormal, o operador de nível mais baixo deverá ter também a autorização do operador de maior nível para realizar as ações que farão o sistema retornar à normalidade. Com isto ocorre a ação das duas chaves, semelhante à de bancos que para abrirem determinados cofres necessitam de duas chaves.

Figura 49 – Diagrama de Colaboração “Sinalizar Alertas”

A figura 50 se dedica a mostrar os passos que ocorrem no diagrama de colaboração quando o operador decide realizar a impressão de algum relatório. Para isso foram criados vários estereótipos para trabalhar no processo de impressão. Alguns já estavam criados devido aos casos de uso anteriores.

Uma questão importante desse diagrama é que somente o operador, com privilégios de administrador ou líder, poderá imprimir relatórios (tela ou papel) com as

atividades dos outros operadores. Os outros operadores (nível 1 e 2) somente poderão imprimir relatórios de equipamentos e outros dispositivos. Outra questão importante é que nenhum dos operadores, não importando o nível de operação, jamais poderá apagar, alterar ou inserir informações no histórico ou *log* de atividades dos operadores.

Figura 50 – Diagrama de Colaboração “Imprimir Relatórios”

A figura 51 complementa o ciclo final de atividades. Neste caso o operador de nível 3 (administrador ou líder) deseja realizar uma auditoria. Para isso é necessário a ID (identificação) e a senha (podendo ser completada com a leitura digital por meio do mouse). A partir dessa escolha o administrador pode especificar qual o tipo de auditoria, podendo ela ser sobre atividades em geral, equipamentos ou sobre o usuário.

Figura 51 – Diagrama de Colaboração “Realizar Auditoria”

Estes foram os diagramas de colaboração dos principais casos de uso do operador. Outros casos de uso poderão ser inseridos, retirados ou inseridos em outros casos de uso mais abrangentes. A seguir as considerações finais do capítulo.

5.11 Considerações Finais

No capítulo foi destaque às características ditas importantes para o sistema de modelagem do usuário, enfatizada por vários especialistas. Esses modelos que antes eram futuristas, hoje caminham para a realidade, como por exemplo, os modelos móveis do usuário e os modelos do usuário para dispositivos astutos.

Foram definidas as atividades desempenhadas por um operador de uma subestação em um SEP, sendo cinco casos de uso encontrados. São eles:

- Realizar a manipulação de Eventos Seqüenciais;
- Leitura de Parâmetros e Configuração;
- Realizar a Indicação de *Status*;
- Avaliação de Valores e Checagem de Limites;
- Operação de Equipamentos (alta tensão e controle).

Foi realizada a interpretação da base de dados dos operadores da subestação, definido os turnos, as atividades e subatividades de cada perfil do usuário. Tais atividades do operador foram desenvolvidas em UML para os casos de uso e demais diagramas que constituem o *software* “Carcará”.

Foram definidos os casos de uso do Sistema de Detecção de Intrusos por Anomalia, batizado de “Carcará”. São eles:

- a) Realizar Auditoria;
- b) Sinalizar Alertas;
- c) Operação de atividades;
- d) Configurar Direitos;
- e) Imprimir Relatórios.

Neste capítulo foi finalizado com a análise completa do *software* proposto (ao nível de análise de engenharia de *software* orientada a objetos).

No capítulo 6 estão inclusos os testes finais desenvolvidos com o *software* SNORT, *software* utilizado para detecção por abuso. Estão também os testes finais desenvolvidos com o *software* “Carcará”, *software* de detecção por anomalia, e os resultados obtidos com cada um deles.

Os resultados alcançados com o *software* de detecção de Intrusão de anomalia “Carcará” foi inteiramente desenvolvido neste projeto de doutorado e são inéditos em SEP. Este e outros itens de segurança ainda estão em desenvolvimento mediante a norma IEC 61850.

6 RESULTADOS DA APLICAÇÃO DO *SOFTWARE* DE DETECÇÃO DE INTRUSÃO SNORT E DO *SOFTWARE* “CARCARÁ”

Este capítulo abordará os testes realizados sobre um computador de teste, cuja missão é detectar ações que levassem o *software* SNORT a concluir que haveria evidências de tentativa de invasão à rede de computadores da subestação, sendo esse um dos objetivos do projeto de Segurança delineado neste trabalho.

Como primeira etapa desse projeto, o SNORT (Detecção por abuso) foi configurado em um ambiente operacional Linux [68], com poucas regras, mas necessárias para servir como última barreira de segurança (externa) para proteger o sistema elétrico, aqui representado pelo sistema EPOCHS (Simulador de um Sistema Elétrico de Potência).

Neste capítulo também são apresentados os resultados do *software* “Carcará” (detecção por anomalia), desenvolvido em C++ e que por meio da base de atividades e sub-atividades dos operadores da rede, são calculados pesos que representam o padrão de atividades para cada perfil de operador.

O *software* “Carcará” possui o objetivo de detectar ações que representam desvios de padrões de atividades de operadores das subestações, permitindo assim realizar posteriormente eventuais análises de atividades suspeitas ou equivocadas.

6.1 Resultados com o *Software* SNORT

6.1.1 Introdução

A estrutura ideal de um SEP utilizando uma Rede de computadores envolve *Firewalls*, *Antivírus*, *Honeypots*, DMZ (isolamento dos servidores da rede), Biometria, senhas e SDI. Isto é, todo meio conhecido para identificar um usuário autorizado a utilizar os serviços da rede de computadores. Isto tudo é implementado para que o possível invasor não possua muitas chances de realizar tentativas de *scanners* ou até mesmo de ações mais destrutivas, sem passar despercebido pelo sistema.

Atualmente existem muitas falhas de segurança em *softwares*, havendo a indução dos administradores a tratar cada alerta ao nível máximo de ataque, embora a maioria seja de simples intrusão ou falsos alertas.

Ativados os dois tipos básicos de SDI (abuso e anormalidade), as chances de invasão se reduzem de forma drástica. No entanto, é importante ressaltar que nenhum programa de segurança será eficaz se não houver uma política de segurança (Apêndice B) a ser realizada por todos os funcionários da empresa, desde os que ocupam os cargos mais importantes na empresa, aos menos favorecidos hierarquicamente.

Esse é um dos passos fundamentais, pois não adiantará um super sistema de segurança, se os funcionários não souberem agir diante do inesperado ou de ações não especificadas nos treinamentos da empresa.

Esses funcionários determinarão o sucesso ou fracasso do sistema, tempo e dos valores monetários investidos em prol da segurança da empresa e do seu patrimônio (Funcionários e valores móveis e imóveis).

A figura 52 [69] abaixo mostra uma situação em que o possível invasor tentará inicialmente realizar um *scanner*, para depois escolher seu possível alvo. Assim como um predador ataca o mais vulnerável de uma lista de possíveis alvos, do mesmo modo este invasor estará agindo.

Figura 52 – Invasor realizando *Scanner* (adaptado de Antonio Brandão [69])

Todas estas possibilidades serão analisadas pelo invasor para um posterior ataque. Por esta razão, o administrador de Segurança de uma rede de computadores será o responsável em providenciar treinamento para os demais funcionários em termos de ações que possam contra-atacar e frustrar as tentativas do invasor. Um exemplo de tais procedimentos seria a regra do não recebimento de e-mails com anexos em computadores da empresa. A seguir as configurações utilizadas para os testes no SNORT são apresentadas.

6.1.2 Planejamento Ideal para instalação de um SDI para um SEP

O sistema ideal para um Sistema Elétrico de Potência (SEP) é ilustrado na figura 53. Ele consiste de um DMZ, tendo dois níveis de defesa do SNORT para melhorar a defesa da rede de computadores contra intrusão.

Figura 53 – Firewall com DMZ (adaptado de Antônio Brandão)

Neste caso de uso da figura 53, existe a possibilidade de haver vários microcomputadores farejadores (por meio do SNORT) que estariam fazendo a coleta de dados da rede de computadores em termos de pacotes, para averiguação de pacotes suspeitos. A situação ideal seria o não compartilhamento do SEP com os outros serviços da rede de computadores. Por exemplo, um sinal de disparo de um relé nunca poderá competir no canal com um e-mail. O relé deverá sempre ter prioridade máxima, ocupando quanto espaço (canal) for necessário para resolver suas tarefas. Desta forma o SEP teria um canal específico sempre disponível, (Figura 54 parte a) no qual poderia haver um aumento do seu canal caso fosse necessário (Figura 54 parte b). Assim, durante a operação o canal do SEP, o mesmo poderá ocupar uma área que inicialmente

não lhe pertencia, visto que a demanda ou prioridades podem tornar necessária esta ação.

Figura 54 – Canal dedicado do SEP e expandindo se necessário

Para definir esta e outras situações, foi criada a norma IEC 61850 (IEEE) que visa uma padronização em projetos, equipamentos e ações pra o SEP que envolve subestações digitais. As principais vantagens desta norma são[1]:

- Definição de um protocolo para toda subestação;
- Dar suporte total à subestação, compreendendo automação de funções, proteção e monitoramento;
- A arquitetura permite futuras extensões, protegendo o investimento;
- Possui aplicação com padrões definidos, a solução para interoperabilidade;
- Definição de requisitos de qualidade (integração de dados, segurança etc.), condições de ambiente e serviços auxiliares do sistema;
- Especificação do processo de engenharia e suporte de ferramentas, sistema de ciclo de vida e requisitos na qualidade da segurança e manutenção para todo o sistema automático da subestação.
- *Status* de testes e desempenho de saída dos dispositivos;
- É flexível e aceita otimização para a arquitetura do sistema (tecnologia escalável);
- Utiliza leitura de avaliação indústria *Ethernet* e permite a comunicação entre componentes;
- Facilidade de comunicação em termos de infraestrutura do centro de controle para as demais áreas de controle.

Com uma subestação tendo esta especificação da norma IEC 61850, haverá um ganho muito elevado de facilidades, tanto para o operador quanto aos serviços disponibilizados pela subestação aos usuários finais.

Esta tese possui como premissa que a subestação que utilizar um SDI (abuso e anomalia), bem como a norma padrão de segurança (ações e comportamento), deverá

obedecer à Norma IEC 61850 para alcançar os ganhos reais de qualidade e segurança em um SEP.

A seguir, os resultados obtidos com o SNORT para a detecção de intrusão em uma subestação que estará funcionando em uma rede de comunicação. A configuração do sistema atacado foi destacada no capítulo 4 dessa tese de doutorado e está melhor detalhada na Fig. 58 a seguir.

6.1.3 Testes com o SNORT

O cenário criado para esta estrutura de testes com o SNORT, inicia com a escolha das regras a serem executadas durante a atuação do SNORT. As regras aplicadas foram às necessárias para a detecção dos ataques sugeridos no capítulo anterior para a avaliação do SNORT como Sistema de Detecção de Intrusão confiável.

6.1.3.1 Regras Básicas Utilizadas pelo SNORT

A página oficial na Internet do SNORT disponibiliza um pacote de regras para usuários registrados e para usuários não registrados. O pacote de regras utilizado para testes foi o de usuários não registrados, juntamente com regras de terceiros advindos de *sites* de administradores de redes que utilizam o SNORT.

No interior do arquivo SNORT.conf (arquivo de configuração do SNORT), foram ativadas dentre outras regras, 07 (sete) regras que servirão como última barreira para o SDI por abuso SNORT. São elas:

- *Local.rules;*
- *Bad_traffic.rules;*
- *Finger.rules;*
- *Community_ftp.rules;*
- *Rpc.rules;*
- *Rservices.rules;*
- *Tftp.rules.*

Na figura 55 observa-se um pequeno trecho da sintaxe da regra *Bad-traffic*.

```
# $Id: bad-traffic.rules,v 1.31.2.3.2.1 2005/05/16 22:17:51 mwatchinski Exp $
#-----
# BAD TRAFFIC RULES
#-----
# These signatures are representative of traffic that should never be seen on
# any network. None of these signatures include datagram content checking
# and are extremely quick signatures
#
alert tcp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD-TRAFFIC tcp port 0 traffic";
alert tcp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD-TRAFFIC tcp port 0 traffic";
```

Figura 55 – Sintaxe da regra *bad-traffic*

Em parte dos testes realizados, foram utilizados dois *softwares* conhecidos no mercado de segurança. Foram escolhidos os *softwares Ethreal* e *Languard*. A razão da escolha desses *softwares* foi a facilidade de instalação e aprendizagem para os testes.

Brevemente, falaremos de cada um desses *softwares* que foram utilizados na fase de teste para o SNORT.

a) **Ethreal**

O Ethreal [70] é um *sniffer* que funciona de forma bem simples, fazendo a leitura das informações que passam pela placa de rede e gravando em um arquivo de *log*, para imediata ou posterior avaliação. Para isto ocorrer é necessário estar conectado em uma Rede de Computadores, seja ela a Internet ou uma rede local.

Por meio do Ethreal foram realizados testes no microcomputador “Cobaia I”, por meio da rede de computador, usando para isso o microcomputador “X1”, “X2”, “X3” e “X4” para a realização de ataques.

A seguir o *software* que foi plenamente utilizado para fazer todos os testes desta primeira etapa que foi destacada como “*Scanner*”, o *software Languard*.

b) **Languard**

O *Languard* [71] é um dos *softwares* mais sofisticados em termos de verificação de portas, pendências, atualizações, varredura etc. Possui uma versão demonstrativa válida por alguns dias, sendo comercializada sua versão definitiva com todos os módulos ativados. Ele foi utilizado para fazer ataques de *scanner* e assim observarmos a ativação das regras no computador “Cobaia I”. A figura 57 mostra a varredura do *Languard* sobre todos os computadores da Escola de Engenharia de São Carlos (EESC).

Nesta figura 56, observamos que o *software* detectou uma máquina ou *host*, com 04 (quatro) portas abertas e especificou as portas para maior facilidade do administrador de redes.

Para fins de privacidade do *host* em questão, foi apagada a identificação da máquina em que foram encontradas as portas abertas.

Este *software* é ideal para administradores de redes, visto que se pode ter uma avaliação dos riscos que podem ocorrer a partir do momento que portas são abertas por usuários, que em muitos casos coloca em risco toda a estrutura da empresa. Tais riscos colocam servidores, outros usuários, *softwares* e todo o projeto de segurança em processo de alta possibilidade de invasão.

Figura 56 – *Software* de teste GFI Languard

Outras ferramentas podem ser utilizadas para a realização de testes, pois o que muda é basicamente o fabricante, a interface e a otimização do algoritmo utilizada para realizar os testes de invasão.

Basicamente, muitos dos problemas encontrados em termos de segurança podem ser amenizados por meio de um excelente *firewall* e atualizações dos *softwares* instalados no computador a ser protegido.

6.1.3.2 Ataque de “X1”, “X2”, “X3” e “X4” e resposta do “Cobaia I”

Um total de 1.208 tentativas de invasão foi realizado por meio dos *softwares* *Ethreal* e *Languard* para fins de testes iniciais, não contabilizados nos resultados dos testes finais descritos nas tabelas de resultados. Foi inserida nesse texto uma dessas tentativas de invasão, cuja detecção foi confirmada pelo SNORT. Ressaltamos que todas as tentativas de invasão tiveram 99% de detecção. O 1% restante deu-se em razão de má configuração das regras no SNORT, provenientes das ações equivocadas do administrador de segurança. Segue abaixo um resumo do que foi realizado, passo a passo, em um desses testes de invasão e a sua detecção pelo SDI SNORT.

A visão da estrutura de ataque e defesa dos microcomputadores é mostrada na figura 57.

Figura 57 – Estrutura da Rede de Computadores para os testes do SNORT

Os Computadores de ataque foram os computadores identificados como “X1”, “X2”, “X3” e “X4”. Utilizou-se para os testes preliminares o *software* *Languard*. No mesmo instante da varredura, o *software* SNORT, instalado no “Cobaia I”, detectou a atividade suspeita (varredura) do microcomputador “X1”, “X2”, “X3” e “X4”, disparando o alarme no arquivo de alerta (*alert.log*).

a) Inicialização do SNORT no Linux Kurumin

Primeiramente, foi iniciado o *software* SNORT no microcomputador “Cobaia I” para depois realizar qualquer tentativa de *scanner*. A figura 58 mostra o comando do

SNORT para que o mesmo possa carregar as regras que atuaram para o SNORT funcionar.

```

root@kurumin:/home/kurumin# snort -c /etc/snort/snort.conf -i eth0 &
[2] 3162
root@kurumin:/home/kurumin# Running in IDS mode
Log directory = /var/log/snort

Initializing Network Interface eth0

--== Initializing Snort ==--
Initializing Output Plugins!
Decoding Ethernet on interface eth0
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file /etc/snort/snort.conf

+++++
Initializing rule chains...
181 Snort rules read...
181 Option Chains linked into 24 Chain Headers
0 Dynamic rules
+++++

-----[thresholding-config]-----
| memory-cap : 1048576 bytes
-----[thresholding-global]-----
| none
-----[thresholding-local]-----
| none
-----[suppression]-----
| none
-----

Rule application order: ->activation->dynamic->drop->alert->pass->log
Log directory = /var/log/snort

--== Initialization Complete ==--

-*> Snort! <*-
o"  )~
+++
Version 2.4.0 (Build 18)
By Martin Roesch & The Snort Team: http://www.snort.org/team.html
(C) Copyright 1998-2005 Sourcefire Inc., et al.

root@kurumin:/home/kurumin#

```

Figura 58 – Iniciando o *software* SNORT no computador “Cobaia I”

O comando “SNORT -c /etc/snort/snort.conf -i eth0 &” possui as seguintes singularidades: a opção “-c” ordena que sejam lidas as configurações realizadas no snort.conf (arquivo de configuração do SNORT); com a opção “-i”, ordena-se que ele fique escutando as conexões realizadas na interface eth0 e o “&” ordena que o SNORT fique em funcionamento no modo *background*.

b) Início do ataque por meio do *software* Languard

Após o acionamento do comando, o arquivo snort.conf, o modo SDI é iniciado e procede ao arquivamento das atividades ou ações no arquivo de log no diretório /var/log/snort, etapa esta em que as regras são carregadas. Nos testes as regras dinâmicas não são carregadas.

A figura 59 mostra o ataque de varredura do microcomputador “X1”, sendo que os outros computadores de ataque “X2”, “X3” e “X4”, neste mesmo instante realizam esta etapa também de varredura sobre o microcomputador “Cobaia I” (143.107.235.237).

Figura 59 - Microcomputador sendo encontrado pelo Microcomputador “X”

Uma observação a ser destacada sobre o microcomputador “Cobaia I”: este computador, com o *firewall* ativado, não permitia que os microcomputadores “X1”, “X2”, “X3” e “X4” realizassem o *scanner*.

Figura 60 – *Software* de teste GFI Languard iniciado

Na figura 60, observa-se a verificação de portas e vulnerabilidades do computador “Cobaia I”. Com esta etapa, verifica-se a importância das atualizações de *software* por meio dos *patches* provenientes dos seus fabricantes.

Este tipo de teste é importante para um administrador de redes poder avaliar o estado da sua rede de computadores. Focado que esta função deve ser contínua,

preferencialmente pelo menos uma vez a cada semana. Especialmente nos dias de hoje, em que os *patches* são bem comuns e os fabricantes os disponibilizam para que possam ser corrigidos os seus *softwares* para a segurança dos dados dos seus clientes.

```
=====
Completed security scan for [143.107.235.237]: 14:40:10.
Scan time: 27 seconds
=====

COMPLETED SECURITY SCAN FOR MACHINE/RANGE: 143.107.235.237
Scan Start Time: 14:39:39
Scan Duration: 31 seconds
=====
```

Figura 61 – *Software* de teste GFI Language com resultados

A figura 61 mostra o tempo que foi gasto para concluir os resultados obtidos com o *scanner* do computador “X1” sobre o computador “Cobaia I”, os tempos desenvolvidos pelos computadores “X2”, “X3” e “X4” foram na ordem de diferença de 1 segundo para mais ou mais menos de 31 segundos (tempo do computador “X1”).

c) Detecção do ataque por meio do SNORT

No momento em que foi iniciado o teste de *scanner* pelos computadores de ataque “X1”, “X2”, “X3” e “X4”, imediatamente o Sistema de detecção de Intrusão por abuso SNORT instalado no computador de defesa “Cobaia I”, detectou a atividade de *scanner*, mostrando a classificação da ação, o horário e o IP da origem da varredura (*scanner*), as mesmas telas de detecção foram mostradas sobre a identificação dos ataques dos outros computadores de ataque “X2”, “X3” e “X4” (figura 62).

```
/var/log/snort/alert - KEdit
Arquivo  Editar  Ir  Ferramentas  Configurações  Ajuda

[**] [1:525:9] BAD-TRAFFIC udp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
10/26-14:56:06.689367 143.107.235.32:3757 -> 143.107.235.237:0
UDP TTL:128 TOS:0x0 ID:4868 Iplen:20 Dgmlen:46
Len: 18
[Xref -> nessus 10074][Xref -> cve 1999-0675][Xref -> bugtraq 576]

[**] [1:525:9] BAD-TRAFFIC udp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
10/26-14:56:06.689367 143.107.235.32:3757 -> 143.107.235.237:0
UDP TTL:128 TOS:0x0 ID:4868 Iplen:20 Dgmlen:46
Len: 18
[Xref -> nessus 10074][Xref -> cve 1999-0675][Xref -> bugtraq 576]

[**] [1:1923:6] RPC portmap proxy attempt UDP [**]
[Classification: Decode of an RPC Query] [Priority: 2]
10/26-14:56:11.893844 143.107.235.247:3862 -> 143.107.235.255:111
UDP TTL:64 TOS:0x0 ID:0 Iplen:20 Dgmlen:132 DF
Len: 104
```

Figura 62 – *Software* SNORT detectando o scanner do computador “X1”

O teste serviu para mostrar que diante de uma rede de microcomputadores, existem muitos riscos envolvidos, especialmente se estes microcomputadores estiverem expostos (conectados) a uma rede maior como a Internet.

Mostramos, também, que existem formas simples (ações de segurança) e *softwares* gratuitos e outros pagos que podem proteger uma rede de micromputadores.

Ratifique-se, por oportuno, a necessidade de treinamento a todos os funcionários para agirem de forma segura diante dos mais estranhos acontecimentos que envolvam riscos a segurança (Apêndice B).

a) Escalabilidade – Ataques de Inserção

O primeiro teste realizado com o SNORT foi o teste de escalabilidade para verificação de expansão de capacidade de detecção variando a taxa de *Megabytes*. A Tabela 17 representa exatamente como foi realizada esta etapa e os percentuais de acerto de cada etapa.

Tabela 17 – Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de Inserção

Taxa (Mbps)	Long URL	Self Reference	URL Encoding	Total de Ataques	Total de Alertas	Alertas
4	634	94	107	835	835	100%
6	629	88	98	835	815	97,60%
8	535	75	85	835	695	83,23%

Conforme a Tabela 17, a diferença entre o número total de alertas gerados nas duas primeiras taxas é de 2,4%. Sendo que a 8 Mbps a quantidade de alertas gerados reduz aproximadamente 16,7% em relação à primeira taxa de transmissão(4 Mbps). A figura 63 mostra a mesma análise em termos de gráfico de barras para os ataques de inserção.

Figura 63 – Análise de escalabilidade do SNORT em relação aos ataques de inserção

b) Escalabilidade – Ataques de Evasão

Os ataques de evasão a uma taxa de 6 Mbps em comparação com a taxa de 4 Mbps gerou uma taxa de 2,28% a menos de alertas (tabela 18). A figura 64 representa gráfico de barras esses números de alertas gerados.

Tabela 18 – Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de Evasão

Taxa (Mbps)	Method Matching	Session Splice	Total de Ataques	Total de Alertas	Alertas
4	105	71	176	176	100%
6	103	69	176	172	97,72%
8	95	61	176	156	88,63%

Figura 64 – Análise de escalabilidade do SNORT em relação aos ataques de evasão

c) Escalabilidade – Ataques de Varredura de Portas

Quanto aos ataques de varredura de portas, cujos percentuais de alertas gerados a cada taxa de transmissão constam na tabela 19, constatou-se que o SNORT gerou praticamente o mesmo número de alertas nas duas primeiras taxas. Sendo que somente a 8 Mbps a diferença entre o número de alertas gerados, em relação a primeira taxa, pode ser considerada significativa como mostra a figura 65.

Tabela 19 - Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de Varredura de Portas

Taxa (Mbps)	Fragmentation	Fingerprinting	Ident Reverse	Protocolo IP	Xmas	UDP Scan	Total de Ataques	Total de Alertas	Alertas
4	1105	15	9	241	912	35	2317	2317	100%
6	1101	11	9	239	916	35	2317	2311	99,74%
8	1084	8	9	228	859	29	2317	2217	95,68%

Figura 65 - Análise de escalabilidade do SNORT em relação aos ataques de Varredura de Portas

c) Escalabilidade – Ataques de Negação de Serviço

Na análise de escalabilidade em relação aos ataques de negação de serviços, novamente o SNORT apresentou resultados bastante próximos nas duas primeiras taxas de transmissão, como pode ser observado na tabela 20.

Tabela 20 - Resultados obtidos na análise de escalabilidade do SNORT em relação aos ataques de negação de serviços

Taxa (Mbps)	Smurf	Teardrop	UDP	ICMP	Syn Flood	Total de Ataques	Total de Alertas	Alertas
4	2050	105	215	2015	989	5374	5374	100%
6	1990	101	215	2005	993	5374	5304	98,69%
8	1825	91	212	1901	941	5374	4970	92,48%

Os dados da figura acima estão representados na figura 66. A partir desse gráfico percebe-se que há uma queda de aproximadamente 7,5% na quantidade de alertas gerados entre a primeira e a última taxa de transmissão.

Figura 66 - Análise de escalabilidade do SNORT em relação aos ataques de negação de serviços

6.2 Resultados com o *Software* “Carcará”

As redes neurais [72], [73], [74], [75] foram inicialmente estudadas com maior ênfase a partir da invenção de Roseblatt em 1962 do *perceptron*. A invenção chamada de *perceptron* modela um neurônio, realizando uma soma ponderada de suas entradas e enviando o resultado 1 se a soma for maior que algum valor inicial ajustável.

A arquitetura de redes neurais também intituladas arquiteturas “conexionistas” utilizada pelo *software* “Carcará” para detecção de comportamentos de intrusão, foi escolhida pelas seguintes razões:

- Um grande número de elementos de processamento muito simples, parecidos com os neurônios humanos;
- Um grande número de conexões ponderadas entre os elementos. Os pesos das conexões codificam o conhecimento de uma rede;
- Ênfase na aprendizagem automática de representações internas.

A rede neural desenvolvida e utilizada pelo *software* “Carcará” no princípio foi projetada para trabalhar com somente um neurônio, uma solução simples para o treinamento e testes da rede neural. No entanto, os resultados alcançados não foram relevantes, as informações contidas na base de atividades dos operadores não eram totalmente assimiladas para a definição de um padrão de atividades dos operadores. A opção seguinte foi implementar a rede neural com múltiplas camadas, isto é com oito (8) camadas de entrada, contendo todos os itens para avaliação da rede neural. Desta

forma todos os dados da base de atividades dos operadores, que é extremamente relevante, poderia ser analisada e calculada para a definição do padrão do operador.

A opção por uma rede de múltiplas camadas se baseia no fato dos neurônios da mesma serem independentes uns dos outros, podendo eles ser treinados separadamente.

Na rede de multicamadas do *software* “Carcará” [67], foi aplicado o método de retropropagação (também chamado de *crossvalidation*) que inicialmente começa com um conjunto de pesos aleatórios. A rede ajusta seus pesos toda vez que recebe um par entrada-saída. Cada par requer dois estágios: uma passagem para frente e uma para trás. A passagem para frente envolve apresentar um exemplo da entrada para a rede e deixar as ativações fluírem até chegarem à camada de saída. Durante a passagem para trás, o produto da rede (obtido na passagem para frente) é comparado com a saída-objeto, e estimativas de erro são computadas para as unidades de saída. Os pesos conectados às unidades de saída podem ser ajustados para reduzir esses erros. Depois essas estimativas de erros das unidades de saída podem ser utilizadas para derivar estimativas de erros para as unidades de camadas ocultas, com isto os erros são propagados para trás até a conexão cuja está nas unidades de entrada.

O algoritmo de retropropagação atualiza seus pesos incrementalmente, depois de ver cada par entrada-saída. Depois de visto todos os pares entrada-saída e ajustando seus pesos todas às vezes, pode-se afirmar que uma época foi concluída. O treinamento de uma rede de retropropagação em geral requer muitas épocas. No *software* “Carcará”, foram utilizadas 500 épocas para calcular e alcançar os pesos de cada entrada. Destaca-se que para esta etapa de configuração da rede neural foram encontrados empiricamente a quantidade de neurônios da camada intermediária ou escondida, perfazendo um total de seis (6) neurônios nesta camada e quatro (4) neurônios na camada de saída.

Antes de mostrar os resultados alcançados com o *software* “Carcará”, será apresentado um exemplo de atuação do *software* “Carcará” dentro de um ambiente de SEP, especificamente atuando em uma subestação digital. A seguir o exemplo:

“Um operador de nome Raimundo João possui específicos privilégios. Ele e outros operadores possuem este perfil para poderem realizar as suas atividades de manutenção da Subestação Elétrica. Esse operador Raimundo João (figura 67) possui uma tabela de horários e turnos nos quais vai atuar durante os próximos dois meses de trabalho. Desta forma todos sabem os seus horários de atuação e na pior das hipóteses é realizado uma troca de horários entre operadores de mesmo perfil. Esta atividade é confirmada pelo

gerente do setor e por todos os operadores que estiverem presentes no momento da troca de operador (substituição), mediante uso de senha de confirmação de troca de operadores. Se em algum momento um intruso interno tiver acesso a alguma senha de um usuário devidamente cadastrado e autorizado, ele será identificado pelas ações não condizentes com o perfil do usuário original. Também será checado o horário e turno desse operador e a assinatura de confirmação de autorização dos outros operadores do turno para o operador que estiver trabalhando, inclusive a do gerente de operação que estiver de plantão”. Desta forma a possibilidade de intrusão interna é mínima no SEP. A seguir os resultados alcançados com o *software* “Carcará”.

Figuras 67 – Tentativas de realização de atividades de um perfil autorizado em um SEP

O administrador da subestação do SEP receberá um e-mail correspondendo a atividades suspeitas de operadores da subestação, destacando que toda as atividades dos operadores estão sendo registradas em um arquivo de *log*, que posteriormente poderá ser analisado.

No *software* “Carcará” foram utilizados 1.324 (Um mil trezentos e vinte e quatro) padrões de atividades e sub atividades dos operadores de uma subestação. Deste total 662 (seiscentos e secenta e dois) foram colocados para treinamento e o restante, os outros 662 padrões, foram colocados para a realização de testes, sendo que os testes foram divididos em duas etapas: A primeira etapa foi realizada com 662 padrões sem intrusão ou padrão não comum de algum operador da subestação. Na segunda etapa foram inseridos 31 padrões de intrusão nos 662 padrões anteriormente testados sem nenhuma intrusão (Tabela 21).

Tabela 21 – Quantidade de Padrões dos Operadores, avaliados

Quantidade de Casos	1324
Treinamento	662
Testes sem Intrusão	662
Testes com Intrusão	31

A rede neural do *software* “Carcará” para cálculo dos pesos dos padrões dos operadores da subestação é uma rede de várias camadas (*Multi Layer perceptron*). As saídas esperadas são os perfis **A1**, **B1**, **C1** e **D1** (figura 68).

Figura 68 – Rede neural de várias Camadas do *Software* Carcará

A rede neural de várias camadas ou rede neural Multiperceptron está configurada para receber 8 (oito) valores de entrada de padrões do operador (exemplo: 0250890) valores que são compostos de turno, atividade e sub atividade do operador. A camada oculta ou escondida é composta de 6 (seis) neurônios e a camada de saída é 4 (quatro) valores de saída (os perfis dos operadores).

Os valores da rede neural:

- A taxa de aprendizagem de 0,3;
- Valor do Momentum de 0,2;
- Um total de 500 interações ou épocas para calcular os pesos da rede.

Figura 69 – Rede neural de várias Camadas do *Software* Carcará

A figura 69 mostra a tela do *software* “Carcará”, que utiliza uma rede neural para realização do cálculo dos pesos das atividades dos operadores de uma subestação.

A figura 70 mostra na tela do “Carcará” o perfil do primeiro usuário e o seu teste da rede neural, em um total de 254 (duzentos e cinquenta e quatro) padrões. Havia 248 padrões corretos de saída **A1**, o restante estava como intrusão, isto é atividade suspeita. O valor de acerto foi de 248 (97,64% do total de 254 padrões) padrões do perfil **A1**, isto corresponde a 100% de acerto, sendo 6 (2,36% do total de 254 padrões) padrões identificados como possível intrusão, inseridos propositalmente para testes de identificação, isto é tinham padrão de outras saídas (**B1**, **C1**, **D1**). Os percentuais mostrados na figura 70 correspondem à quantidade de padrões testados pela rede neural (254 padrões).

Figura 70 – Teste da Classe A1 (perfil do Operador)

A figura 71 mostra o teste da classe **B1** e os seus percentuais de acerto. Em um total de 135 (cento e trinta e cinco) padrões, 125 (92,59% do total de padrões) estavam corretos como sendo da saída **B1**, isto corresponde a 100% de acerto, e o restante 10 padrões (7,41% do total de padrões) foram inseridos propositalmente para testes de identificação de atividades suspeitas, isto é, possível intrusão. Os percentuais mostrados na figura 71 correspondem à quantidade total de padrões testados pela rede neural (135 padrões).

Figura 71 – Teste da Classe B1 (perfil do Operador)

A figura 72 mostra o teste da classe **C1** e os seus percentuais de acerto. Em um total de 135 (cento e trinta e cinco) padrões, 127 (94,07% do total de padrões) estavam corretos como saída **C1**, isto corresponde a 100% de acerto, e o restante 08 padrões (5,93% do total de padrões) foram inseridos propositalmente para testes de identificação de atividades suspeitas, isto é, possível intrusão. Os percentuais mostrados na figura 72 correspondem à quantidade de padrões testados pela rede neural (135 padrões).

Figura 72 – Teste da Classe C1 (perfil do Operador)

A figura 73 mostra o teste da classe **D1** e os seus percentuais de acerto. Em um total de 138 (cento e trinta e oito) padrões, 131 (95% do total de padrões) estavam corretos como saída **D1**, isto corresponde a 100% de acerto correto, e o restante 07 padrões (5% do total de padrões) foram inseridos propositalmente para testes de identificação de atividades suspeitas, isto é, possível intrusão.

Figura 73 – Teste da Classe D1 (perfil do Operador)

Nos padrões de teste (662 padrões) foram inseridas 31 (trinta e uma) intrusões. Sendo que existiam 631 padrões correspondentes a 95% dos padrões de testes, reconhecidos pelo “Carcará”, isto corresponde a 100% de acerto correto. Como saída correta (A1, B1, C1, D1). Outros 31 padrões (correspondem a 5% dos padrões de teste) foram inseridos propositalmente e reconhecidos como possíveis atos de intrusão (figura 74).

Figura 74 – Rede neural de várias Camadas do Software Carcará

Foi utilizada a apresentação final dos acertos do *software* (detecção de intrusão por anomalia) contendo padrões de intrusão em formato de “Matriz de Confusão”. A definição utilizada para matriz de confusão[76] é:

“A matriz de confusão de uma hipótese h oferece uma medida efetiva do modelo de classificação, ao mostrar o número de classificações corretas versus as classificações preditas para cada classe, sobre um conjunto de exemplos T ”.

O número de acertos, para cada classe, se localiza na diagonal principal $M(C_i, C_i)$ da matriz. Os demais elementos $M(C_i, C_j)$, para $i \neq j$, representam erros na classificação. A matriz de confusão de um classificador ideal possui todos esses elementos iguais a zero uma vez que ele não comete erros.

A figura 75 mostra a matriz de confusão dos testes realizados com 662 padrões da rede neural contendo tentativas de intrusão. Os valores fora da diagonal principal são exatamente os valores de tentativas de intrusão ou invasão.

Figura 75 – Rede neural de várias Camadas do *Software* Carcará

A1	B1	C1	D1	Total
248	1	4	1	254
8	125	1	1	135
6	2	127	0	135
3	4	0	131	138
Total				662

6.3 Considerações Finais

Com este capítulo destacou-se a etapa de testes com o *software* SNORT, utilizando para isso várias máquinas Linux. Os testes de *scanner* e demais testes de escalabilidade (evasão, inserção, varredura de portas e negação de serviços) foram realizados por máquinas *Windows* e *Linux*, em um ambiente gráfico e não gráfico.

Os testes foram os esperados, visto que a primeira etapa de uma invasão é justamente a varredura da máquina alvo. Nenhum invasor tenta realizar uma invasão sem saber a porta a ser aberta.

Foram inclusos também neste capítulo, os testes com o *software* “Carcará” que foi desenvolvido em linguagem C++. Para este *software*, foi desenvolvida uma análise de todas as principais atividades de um operador de uma subestação, em um total de cinco principais atividades analisadas por meio da linguagem UML. Baseando-se nessas cinco principais atividades foi criada uma base de atividades para os operadores (usuários) da subestação. O *software* desenvolvido em C++ alcançou o objetivo de detectar atividades suspeitas em um *Login* de atividades desses operadores.

No próximo capítulo é realizada a conclusão desta tese de doutorado, resumindo os objetivos alcançados com a utilização do *software* SNORT e com o *software* “Carcará”, e as perspectivas futuras para os próximos trabalhos que almejem trabalhar com os SDI de detecção por abuso e por anomalia.

7 CONCLUSÕES FINAIS E CONTINUIDADE DO TRABALHO

Este trabalho teve como foco principal a pesquisa em Sistemas de Detecção de Intruso por anomalia, tendo como objetivo específico mostrar como pode ser realizada uma análise para o desenvolvimento e implementação de um *software* que utilizasse redes neurais para a criação de um perfil de atividades de um usuário em uma subestação de um SEP e que trouxesse valores satisfatórios de funcionamento. Para esta tarefa foi desenvolvido o *software* “Carcará”.

A base de dados das atividades de um operador de uma subestação digital analisada pela rede neural possui como referência o simulador EPOCHS, desenvolvido pela Universidade de *Cornell* (EUA) com a participação da USP - EESC de São Carlos.

O EPOCHS, assunto do capítulo 2, trabalha com agentes (advindos da Inteligência Artificial). Esse tipo de definição denominada agentes, por longo tempo, vem sendo muito utilizada e aplicada em pesquisas na área de Ciências da Computação, e vem há pouco tempo sendo também aplicada na área de Engenharia Elétrica. Para um melhor entendimento do EPOCHS, foi utilizado o UML para especificar o relacionamento entre as classes (estereótipos), bem como em alguns casos mostrar classes e diagramas de estado do funcionamento do *software*.

O capítulo 3 destacou os tipos de ataques via rede de computadores, bem como os princípios de defesa. Estes princípios podem ser aplicados em um Sistema Elétrico de Potência que utiliza uma rede para equipamentos microprocessados e computadores. Foram especificados os riscos que estão envolvidos em uma rede de computadores, os tipos de ataques técnicos, ataques locais, ataques remotos, os ataques não técnicos, o uso da engenharia social e os riscos para instituições que possuam informações sigilosas

ou extremamente básicas para o funcionamento de serviços, especialmente serviços públicos. Finalizando o capítulo 3, foi invocado o que é necessário à realização de uma política de segurança, do mais alto ao mais baixo nível de hierarquia funcional, o que envolve procedimentos específicos para as situações plausíveis de acontecer (inclui a quem se dirigir diante de situações inesperadas). Além disso, devem ser citados treinamentos, seminários, simulações e avaliações entre os funcionários para a melhoria da segurança, que são necessários para que todos possam trabalhar e manter assim o nível de produtividade. No capítulo 3, também foram colocados os principais tipos de ataques, como *man on the middle*, analisadores de tráfego (*sniffers*) para espionagens, quebras de senha, ataques com cavalos de tróia, ataques a bibliotecas, ataques em camadas etc. Após a lista de ferramentas de invasão, foi criada uma lista de ferramentas voltada para a defesa da rede de computadores, destacando as principais ferramentas utilizadas pelos administradores de uma rede de computadores (atualizações, correções, vírus, *firewall* etc.). Nessa lista, foi dado um enfoque especial aos Sistemas de Detecção de Intrusos (SDIs), os seus principais tipos (baseado em *host*, baseado em rede e aplicação) e a sua aplicação. As vantagens e desvantagens dos sistemas, bem como os métodos utilizados para a análise de detecção de intrusão foram abordados.

Com o capítulo 4 foi desenvolvido um estudo de um dos principais e melhores sistemas de Detecção de Intrusão (SDI), o *software* de código aberto denominado SNORT. Foram mostrados os principais componentes de sua arquitetura, como o *Sniffer*, o pré-processador, a engenharia de detecção e o módulo de alertas e *Login* do SNORT. Foi dada a explicação de cada um dos módulos e as ações perante todo o conjunto de componentes do SNORT.

A importância desse tipo de Sistemas de Detecção de Intrusos é grande em razão do avanço da engenharia elétrica e os riscos envolvidos quando as informações se

concentram em uma rede de computadores. O SNORT trabalha como um detector de intrusos por meio de detecção por abuso, isto é caracterizado por determinado tipo de ataque ou padrão de ataque. Para isto o SNORT contém uma base de assinaturas que constantemente são atualizadas. As atuais pesquisas de detecção de intrusos caminham para a criação de um novo SDI com recursos capazes de lidar com expressões regulares e assim ter maiores chances de identificar um ataque.

Também foi destacado o método utilizado, para realizar testes sobre uma rede de computadores, cuja missão é detectar ações que levassem o *software* SNORT (detecção por assinatura ou por abuso) a concluir que haveria possíveis evidências de tentativa de invasão à rede de computadores.

Os sistemas de detecção de intrusos são avaliados quanto às seguintes características: (a) capacidade do Sistema de Detecção de Intruso (SDI); (b) escalabilidade e (c) taxa de falsos positivos que são gerados. A metodologia utilizada é composta por cinco etapas: (a) seleção dos ataques; (b) seleção de ferramentas; (c) geração do tráfego dos cenários de avaliação; (d) montagem do ambiente de avaliação e (e) análise dos SDIs. Foram destacadas as definições das etapas da metodologia de avaliação aplicada no SDI por abuso (SNORT).

O capítulo 5 mostrou como se pode realizar a criação de um perfil de usuário para posterior utilização em um sistema de detecção de intruso, utilizando para isso a detecção por anomalia. Neste capítulo, foram vistos o modelo do usuário e os tipos de modelos criados com o decorrer do tempo por meio de estudos no comportamento. A sua evolução por meio dos *Shells* e depois por meio da modelagem do usuário. Não foram esquecidas as características ditas importantes para o sistema de modelagem do usuário, destacadas por meio de vários especialistas. Foram definidas as atividades

desempenhadas por um operador de uma subestação, sendo cinco casos de uso encontrados.

- a) Realizar a manipulação de Eventos Sequenciais;
- b) Leitura de Parâmetros e Configuração;
- c) Realizar a Indicação de Status;
- d) Avaliação de Valores e Checagem de Limites;
- e) Operação de Equipamentos (alta tensão e controle).

Concluída esta etapa, foi feito um breve resumo da linguagem utilizada na criação do *software* “Carcará” e o relacionamento interno com outras unidades do *software*. Foram definidos os casos de uso do Sistema de Detecção de Intrusos por Anomalia, batizado de “Carcará”.

- a) Realizar Auditoria;
- b) Sinalizar Alertas;
- c) Operação de atividades;
- d) Configurar Direitos;
- e) Imprimir Relatórios.

O capítulo 6 mostrou resultados da utilização do *software* SNORT (detecção por abuso), destacando os tipos de ataques utilizados para que o SNORT pudesse detectar atividades (Alertas) suspeitas na rede de computadores. Os valores alcançados nos diversos tipos de ataques, como os de negação de serviços, inserção, evasão e varreduras de portas, foram altamente satisfatórios. Os problemas encontrados foram:

- Quando ocorreu o aumento da taxa de tráfego na rede, a quantidade de alertas de intrusão foi reduzida.

- Houve a necessidade de trocar da linguagem de implementação, pois a linguagem Java trazia desvantagens de processamento em relação à linguagem C++ no *software* “Carcará”.

Os testes com *software* “Carcará”, um sistema de detecção de intrusos baseado em anomalia foram altamente favoráveis, tendo uma detecção de 100% em todas as atividades suspeitas. Utilizou-se uma quantidade razoável de padrões normais de atividades dos operadores para serem treinados e testados (1.324 padrões no total). Posteriormente na fase de teste foram inclusos 31 padrões de intrusão que prontamente foram reconhecidos pelo *software* “Carcará”.

Desta forma as contribuições inéditas desta tese de doutorado foram:

- A engenharia reversa funcional do EPOCHS;
- A análise em UML das atividades de um operador de uma subestação digital;
- A análise em UML do *software* “Carcará”;
- O planejamento e implantação de uma rede de computadores para simular um SEP;
- O planejamento, configuração e implantação do SNORT e do “Carcará” SDI para o SEP em uma rede de computadores;
- A configuração dos *softwares* de ataque de uma rede de computadores personalizadas para um SEP;
- A completa implementação do *software* “Carcará”;
- Os resultados dos testes dos *softwares* avaliados, “SNORT” e “Carcará”.

Um próximo passo para o desenvolvimento deste trabalho seria a implementação prática em laboratório de subestações digitalizadas com uma rede de comunicação, onde os agentes artificiais pudessem auxiliar na proteção da mesma, conjuntamente com o

protocolo IEC 61850 e as questões envolvendo a segurança em um sistema elétrico de potência, destacadas nesta tese de doutorado. Este seria um passo intermediário para a implantação da metodologia proposta em uma subestação real.

8. REFERÊNCIAS BIBLIOGRAFICAS

- [1] TECHNICAL REPORT – IEC 61850-1 / IEC 61850-10, primeira edição, 2005.
- [2] THORP, J. S.; COURY, D.; GIOVANINI, R. et al. (2003). Agent Technology Applied to the Protection of Power Systems. In: *Autonomous Systems and Intelligent Agents in Power System Control and Operation*. 1a. ed. Baden,2003, Suíça: Springer-Verlag.
- [3] WOOLDRIDGE, MICHAEL, *An Introduction to Multiagent System*, John Wiley & Sons, Ltd, 2000, paginas 15-42.
- [4] GIOVANINI, RENAN, “Aplicação de Novas Tecnologias Baseadas em Agentes para Proteção de Sistemas Elétricos de Potência”, Tese de Doutorado, 27 de junho, São Carlos-SP, 2005, 190 p.
- [5] TANENBAUM, ANDREW S., “Rede de Computadores”, Editora Campus, 2003, paginas 3-28.
- [6] SNORT – Sistema de Detecção de Intruso. Disponível em:<<http://www.snort.org/>>. Acesso em 10 de fev.2004.
- [7] JANSÁ, KRIS, “Aprendendo C++”, Editora Makron Books, 1999, 271 p.
- [8] COSTA, NILSON S., PANSANATO, TADEU E., SANCHES, ROSELY: “Uma visão do *Dynamic CMM* para pequenas empresas”- Relatórios Técnicos - ICMC – USP, São Carlos, Jul.2004.
- [9] MENEZES, PAULO BLAETH, “Linguagens Formais e Autômatos”, 4 Edição, Série Livros Didáticos, Número 3, Instituto de Informática da UFRGS, Editora Sagra Luzzato, 2002, 165 p.

- [10] PRINCE, ANA M. A.; TOSCANI, SIMÃO S.; 2 Edição, Série Livros Didáticos, Número 9, Instituto de Informática da UFRGS, Editora Sagra Luzzato, 2001, 195 p.
- [11] ALFRED, V. AHO; SETHI, RAVI; ULLMAN, JEFFREY D.; “*Compilers Principles, Techniques, and Tools*”, Editora Addison Wesley, 1986, 500 p.
- [12] CANDIDO, JOSÉ R. R.; ARAÚJO, CARLOS A. S.; SOUZA, FLÁVIO C., “Proteção de Sistemas Elétricos”, Editora Interciência, 2002, 258 p.
- [13] COURY, DENIS VINICIUS; “SEL 308 – Introdução aos Sistemas Elétricos de Potência”, Notas de aula, 2006, EESC-USP, 48 p.
- [14] MANITOBA HVDC RESEARCH CENTRE (2002). *PSCAD/EMTDC Manual Getting Started*. Winnipeg, Manitoba, Canadá.
- [15] THE NETWORK SIMULATOR NS-2 (2003). NS-2 Manual. Disponível em: <<http://www.isi.edu/nsnam/ns/>>. Acesso em março de 2004.
- [16] WOOLDRIDGE, M.; JENNINGS, N. R. (1995). Agent Theories, Architectures and Languages: a Survey. In: *Intelligent Agents*. Berlin: Springer-Verlag, p. 1-22.
- [17] TCL (2005). Manual Disponível em: <<http://tcl.sourceforge.net/>>. Acesso em abril de 2004.
- [18] BOOCH, GRADY; RUMBAUGH, JAMES; JACOBSON, IVAR. UML Guia do Usuário. 2 ed. Rio de Janeiro: Campus, 2000. 496 p.
- [19] MANITOBA HVDC (1998). *PSCAD/EMTDC Manual Getting Started*. Winnipeg, Manitoba, Canadá.
- [20] AWSIM-AEROSPACE. Disponível em: <<http://www.freeflight.com/aerospace/products/awsim/index.html>> . Acesso em 07 Fev. 2005.

- [21] MODSAF - U.S. ARMY PEO STRI PROGRAM EXECUTIVE OFFICE for SIMULATION, TRAINING, & INSTRUMENTATION . Disponível em: <http://www.peostri.army.mil/PRODUCTS/MODSAF-PMITS/> . Acesso em 15 jun. 2005.
- [22] VIRDHAGRISWARAN, S. (1998). *The MuBot Agent*. Disponível em: <<http://www.crystaliz.com/logicware/mubot.html>>. Acesso em: setembro/2000.
- [23] MAES, P. (1995). *Artificial Life Meets Entertainment: Life like Autonomous Agents*. *Communications of the ACM*, v. 11, p. 108-14.
- [24] HAYES-ROTH, B. (1995). An Architecture for Adaptative Intelligent Systems. *Artificial Intelligence: Special Issue on Agents And Interactivity*, v. 72, p. 329-65.
- [25] CERT - *COMPUTER EMERGENCY RESPONSE TEAM*. Disponível em: <<http://www.cert.org/>>. Acesso em 20 out. 2005.
- [26] SFOCUS - *SECURITY FOCUS*. Disponível em: <<http://www.securityfocus.com/>> Acesso em 10 Fev. 2004.
- [27] CROSBIE, M; SPAFFORD, E.H. *Active Defense of a Computer System using Autonomous Agents*. *Department of Computer Sciences*, Purdue University. In Proceedings of the 18th National Information Security Conference, October 1995.
- [28] ANDERSON, JAMES P. *Computer Security Threat Monitoring and Surveillance*, Technical Report, James P. Anderson Co., Fort Washington, PA, abr. 1980.
- [29] ZAMBONI, DIEGO; BALASUBRAMANIYAN, JAI; GARCIA-FERNANDEZ, JOSE OMAR; ISACOFF, DAVID; SPAFFORD, E. H. *An Architecture for Intrusion Detection using Autonomous Agents*. *Department of Computer Sciences*, Purdue University, Coast TR 98-05, 1998.

- [30] BACE, REBECCA; MELL, PETTER. *“Intrusion Detection Systems”*, NIST Special Publication, Nov. 2001.
- [31] KUMAR, SANDEEP. *Classification and Detection of Computer Intrusions*. USA, 1995. Tese (Doutorado em Filosofia), Purdue University, 1995.
- [32] LUNT, T. F.; TAMARU, A.; GILHAM, F.; JAGANNATHAN, R.; NEUMANN, P. G.; JAVITZ, H. S.; VALDES, A.; GARVEY, T. D. *A Real-Time Intrusion Detection Expert System (IDES)*. Final Technical Report. Computer Science Laboratory, SRI International, Menlo Park, California, fev. 1992.
- [33] TENG, H.S.; CHENG, K. *Security Audit Trail Analysis Using Inductively Generated Predictive Rules*. New Jersey: Sixth Conference on Artificial Intelligence Applications, 1990.
- [34] FOX, KEVIN L.; HENNING, RONDA R.; REED, JONATHAN H.; SIMONIAN, Richard. *A Neural Network Approach Towards Intrusion Detection*. In Proceedings of the 13th National Computer Security Conference, p. 125 -134, Washington, DC, out. 1990.
- [35] SNAPP, STEVEN R.; SMAHA STEPHEN E. *Signature Analysis Model Definition and Formalism*. In Proceedings of the Fourth Workshop on Computer Security Incident Handling, Denver, Colorado, ago. 1992.
- [36] LUNT, TERESA F. *A Survey of Intrusion Detection Techniques*. Computer Security, USA, v.12, n.04, p. 405-418, Jun. 1993.
- [37] GARVEY, T. D.; LUNT, T. F. *Model based Intrusion Detection*. USA: 14th National Computer Security Conference, 1991.

- [38] PORRAS, PHILIP A.; KEMMERER, RICHARD. A. *Penetration State Transition Analysis: a Rule-Based Intrusion Detection Approach*. USA: Eighth Annual Computer Security Applications Conference, 1992.
- [39] BONIFÁCIO Jr., J. M.; CANSIAN, A.M.; CARVALHO, A.C.P.L; MOREIRA, E.S. *Neural Networks Applied in Intrusion Detection Systems*. In: Proceedings of the IEEE IJCNN '98 International Joint Conference on *Neural Networks*, Anchorage, Alaska, maio 1998.
- [40] CANSIAN, A.M.; MOREIRA, E.M.; CARVALHO, A.C.P.L.; BONIFÁCIO Jr., J.M. *Network Intrusion Detection Using Neural Networks*. In: Proceedings of International Conference on Computational Intelligence and Multimedia Applications, ICCIMA'97, Gold Coast, Australia, p.276- 280, fev. 1997.
- [41] CASWELL, BRIAN, “Snort 2 – Sistema de Detecção de Intruso”, Editora Altabooks, 2003, 383 p.
- [42] GNU - Fundação de Software Livre. Disponível em: <<http://www.gnu.org/>>. Acesso em 21 de abril. 2004
- [43] FAGUNDES, L.L. “Metodologia para a avaliação de sistemas de detecção de intrusão”. Monografia de Graduação para obtenção do título de Bacharel em Informática, 15 de Novembro de 2002, São Leopoldo-SP, 2002, 68 p.
- [44] PTACEK, THOMAS H.; NEWSHAM, TIMOTHY N. Insertion, Evasion, and deny of service: Eluding network intrusion detection, 1998. Disponível em http://www.clark.net/pub/roesh/public_html/IDSpaper.pdf.
- [45] TCPDUMP. TCPDUMP public repository. [online], 2006. Disponível em <http://tcpdump.org>.

[46] TCREPLAY. Tool to replay captured network traffic. [online], 2002.

Disponível em <http://sourceforge.net/projects/tcp Replay/>.

[47] PERRAULT, C. R., ALLEN, J. F. e COHEN, P. RSpeech acts as a basis for understanding dialogue coherence. Report 78^5, Department of Computer Science, University of Toronto, Canada, 1978.

[48] COHEN, P. R., PERRAULT, C. R. Elements of a Plan-based theory of speech acts. Cognitive Science 3, 177-212, 1979.

[49] ALLEN, J. F.: A Plan-based approach to speech act recognition. Technical Report 131/79, Dept. of Computer Science, University of Toronto, Canada, 1979.

[50] RICH, E. Stereotypes and user modeling. In: A. Kobsa, and W. Wahlster (eds.), User Models in Dialog Systems. Springer, Berlin, Heidelberg, pp. 35^51, 1989.

[51] KOBASA, A. Using Modeling and User-Adapted Interaction 4(2), Editorial Special Issue on User Modeling Shell Systems, iiv, 1995.

[52] MCTEAR, M. (ed.): Artificial Intelligence Review 7(3). Special issue on user modeling. Microsoft: 2000, Product and Technology Catalog. www.microsoft.com/products 1993.

[53] FININ, TIM, DAVID DRAGER, GUMS - A General User Modeling System, Proceedings of the 1986 Canadian Society for Computational Studies of Intelligence (CSCSI-86), Maio de 1986.

[54] KOBASA, ALFRED. "Generic User Modeling Systems", Kluwer Academic Publishers. Printed in the Netherlands, 2001.

[55] VAN MELLE, W.: 1982, "System Aids in Constructing Consultation Programs: EMYCIN". UMI Research Press, Ann Arbor, MI, 1996.

- [56] SHORTLIFFE, E. H. "Computer-Based Medical Consultations: MYCIN". North-Holland, New York, 1976
- [57] POHL, W. "LABOUR: Machine learning for user modeling". In: M. J. Smith, G. Salvendy and R. J. Koubek (eds.), *Design of Computing Systems: Social and Ergonomic Considerations (Proceedings of the Seventh International Conference on Human-Computer Interaction)*. Elsevier, Amsterdam. pp. 27^30, 1997.
- [58] BRAJNIK, G., TASSO, C. "A shell for developing non-monotonic user modeling systems". *International Journal of Human-Computer Studies* 40, 31^62. 1994.
- [59] KOBSA, A., POHL, W., The BGP-MS user modeling system. *User Modeling and User-Adapted Interaction* 4(2), 59^106, 1995.
- [60] BRAJNIK, G., TASSO, C. A shell for developing non-monotonic user modeling systems. *International Journal of Human-Computer Studies* 40, 31-62. 1994.
- [61] ORWANT, J. "Heterogenous learning in the Doppelganger user modeling system". *User Modeling and User-Adapted Interaction* 4(2), 107-130, 1995.
- [62] PAIVA, A., SELF, J.: TAGUS: A user and learner modeling workbench. *User Modeling and User-Adapted Interaction* 4(3), 197-226, 1995.
- [63] KAY, J. The um Toolkit for reusable, long term user models. *User modeling and User-Adapted Interaction* 4(3), 149^196, 1995.
- [64] KLEIBER, U. ERKLA: rung in interaktiven Systemen und Unterstu« tzungsmo« glichkeiten durch das System BGP-MS. WIS Memo 6, WG Knowledge-based Information Systems, Department of Information Science, University of Konstanz, Germany. Knowledge Craft: 1988, Knowledge Craft, 3.2 Edn. Carnegie Group, Inc., Pittsburgh, PA, 1994.

- [65] POHL, W. LABOUR: Machine learning for user modeling. In: M. J. Smith, G. Salvendy and R. J. Koubek (eds.), Design of Computing Systems: Social and Ergonomic Considerations (Proceedings of the Seventh International Conference on Human-Computer Interaction). Elsevier, Amsterdam. pp. 27-30, 1997.
- [66] TAYLOR, J. A., CARLETTA, J., MELLISH, C. Requirements for belief models in co-operative dialogue. User Modeling and User-Adapted Interaction 6(1), 23-68, 1999.
- [67] COSTA, N.S, COURY D.V., SEGATTO. E., SOFIANE, L. Um *Software* de Segurança Aplicado em uma Rede de Computadores de um Sistema Elétrico de Potência. SNPTEE – Seminário Nacional de Produção e Transmissão de Energia Elétrica. 14 a 17 de outubro, Rio de Janeiro, 2007.
- [68] KURUMIN; Kernel Debian; em:< [http:// http://www.guiadohardware.net/kurumin/](http://www.guiadohardware.net/kurumin/)>. Acesso em 12 de abril 2005.
- [69] BRANDÃO, ANTONIO; VIRGOS – Segurança Computacional em Sistemas Linux. ICMC, 23 de Set.2005.
- [70] ETHEREAL – Analisador de Protocolo de Rede. Disponível em:<<http://www.ethereal.com/>>. Acesso em 3 de dez.2003.
- [71] GFILANGUARD – Monitoramento do Log, Rede e Monitoramento do Servidor e Scaneamento de Segurança. Disponível em:<[http:// http://www.gfi.com/languard/](http://www.gfi.com/languard/)>. Acesso em 15 de fev.2004.
- [72] SIMON HAYKIN – “Redes Neurais: Princípios e Práticas”, Editora Bookman, segunda edição, 2007, 900 p.
- [73] ZSOLTL. KOVACS, “Redes Neurais Artificiais: Fundamentos e Aplicações”, editora Livraria da Física, Quarta edição, 2006, 174 p.

[74] NETO, LUIS G. P., NICOLETTI, MARIA DO CARMO, “Introdução às Redes Neurais Construtivas”, Editora Edusfscar, primeira edição, 2005, 192 p.

[75] CARVALHO, ANDRÉ C. PONCE DE LEON F., BRAGA, ANTONIO DE PADUA, LUDERMIR, TERESA BERNARDA, “Redes Neurais Artificiais: Teoria e Aplicações”, Editora LTC, Primeira Edição, 2000, 262 p.

[76] REZENDE, S. O., Sistemas Inteligentes Fundamentos e Aplicações, editora Manole, 2002.

Apêndice A – Conceitos Básicos sobre UML

1 UML

Para o desenvolvimento de sistemas de software de grande e médio porte, são utilizados métodos de análise e projeto que visam modelar sistemas para que toda uma equipe de projeto possa ter uma compreensão única do projeto. Para estas situações, foi criada a UML (Linguagem de Modelagem Unificada). A UML [12] é a união de um conjunto de métodos de análise e projeto orientado a objeto, tendo sido padronizada pela OMG (*Object Management Group*), um consórcio aberto de empresas fundado em 1989 justamente para atuar como facilitadora do desenvolvimento de uma arquitetura padrão para objetos. A UML é uma linguagem de modelagem, não um método. A Linguagem de Modelagem é a notação que o método usa para descrever o projeto. Os processos são passos que devem ser seguidos para se construir o projeto. A UML define uma notação e um metamodelo. As notações são todos os elementos de representação gráfica vistos no modelo, enfatizada como sintaxe de modelo de linguagem.

A UML é constituída de diagramas para que o projeto fique bem detalhado sob diversos ângulos de percepção. Semelhante a um projeto residencial em que são realizados diversos projetos para que haja destaque sobre um serviço A ou B. Como exemplo, são os projetos elétricos, hidráulicos, sanitários, estruturais e os de arquitetura. Embora a essência de todos eles seja a mesma, cada um mostra uma visão distinta e específica do projeto. Este conjunto de visões compõe todo o projeto. Os diagramas são os meios utilizados para a visualização desses blocos de construção. Um diagrama é concretizado como uma representação gráfica de um conjunto de elementos, geralmente representados como um gráfico conectado de vértices (itens) e arcos (relacionamentos). Com os diagramas, o sistema em desenvolvimento pode ser visualizado sob diferentes perspectivas. A UML define um número de diagramas que permitem dirigir o foco para aspectos diferentes de um sistema (uma coleção de subsistemas organizados para a

realização de um objetivo e descritos por um conjunto de modelos, possivelmente sob diferentes pontos de vista) de maneira independente.

Neste contexto encontram-se as classes, que são os blocos de construções mais importantes de qualquer sistema orientado a objetos. Convém explicar que uma classe é uma descrição de um conjunto de objetos que compartilham os mesmos atributos, operações, relacionamentos e semântica. As classes são utilizadas para capturar o vocabulário do sistema que está sendo desenvolvido. Cada classe deve ter um nome que a diferencie das outras classes. As classes possuem atributos. Atributo é uma propriedade nomeada de uma classe que descreve um intervalo de valores que as instâncias da propriedade podem apresentar. Um atributo é, portanto, uma abstração de tipo de dados ou estados que os objetos da classe podem abranger. As classes também possuem operações. Uma operação é a implementação de um serviço que pode ser solicitado por algum objeto da classe para modificar o seu comportamento. Uma classe pode ter qualquer número de operações ou até não ter operação alguma.

A UML possibilita trabalhar com a modelagem estrutural básica, modelagem estrutural avançada, comportamento básico de modelagem, modelagem comportamental avançada e modelagem da arquitetura.

Todas estas opções de trabalho possibilitam ao analista uma simplificação da realidade para entender melhor o sistema em desenvolvimento. Por meio da UML, constroem-se modelos (abstração semanticamente fechada de um sistema, representa uma simplificação autoconsistente e completa da realidade, criada com a finalidade de permitir uma melhor compreensão do sistema) a partir de blocos de construção básicos, como classes, interfaces, colaborações, componentes, nós, dependências, generalizações e associações.

Existem diagramas estruturais (diagramas de classe, objetos, componentes e de implantação) e diagramas comportamentais (diagramas de caso de uso, sequência, colaboração, transição de estados e de atividade). Serão destacados os utilizados por este trabalho.

a) Diagramas de Casos de Uso

Um caso de uso descreve o que um sistema (ou um subsistema) faz no contexto geral; no entanto, não especifica como ele é realizado.

Neste contexto de caso de uso se encaixam os cenários, que são uma sequência de ações que ocorrem para ilustrar um comportamento. Conclui-se que os cenários são basicamente uma instância de um caso de uso.

Uma importante observação a ser feita acerca da relação caso de uso e cenários é a de que um caso de uso possui poder de se expandir para N cenários. Na utilização de cada caso de uso, serão encontrados cenários primários (os que essencialmente poderão ocorrer), em conjunto com os cenários secundários, que definem as sequências alternativas ou secundárias para o cenário.

Um caso de uso realiza a captura de um determinado comportamento pretendido do sistema, sem a necessidade de especificar como esse comportamento será implementado.

Casos de uso podem ser agrupados e organizados em pacotes, da mesma forma que ocorre com as classes. Isto inclui especificação de relacionamentos, como generalização, inclusão e extensão, existentes entre eles. Os casos de uso são classificadores, podendo ter atributos e operações que poderão ser representadas da mesma forma que nas classes.

Os casos de uso fazem parte de um diagrama denominado Diagrama de casos de uso. Os diagramas de casos de uso são um dos cinco diagramas disponíveis na UML.

Os diagramas de caso de uso são importantes para visualizar, especificar e documentar o comportamento de um elemento. Os diagramas de casos de uso possuem:

- a) Casos de uso
- b) Atores
- c) Relacionamentos de dependência, generalização e associação.

Os diagramas de casos de uso são utilizados para fazer a modelagem da visão estática do sistema.

A notação usada pelo Diagrama de “Caso de Uso” e ator é mostrada na figura 76.

Figura 76 - Representação gráfica do ator e do caso de uso

Um ator representa qualquer entidade que interage com o sistema. Pode ser uma pessoa, outro sistema ou um subsistema. Abaixo algumas dessas características:

- a) O ator não é parte do sistema. Representa os papéis que o usuário do sistema pode desempenhar.
- b) O ator pode interagir ativamente com o sistema.
- c) O ator pode ser um receptor passivo de informação.
- d) O ator pode representar um ser humano, uma máquina ou outro sistema.

O “Caso de Uso” é uma sequência de ações que o sistema executa e produz um resultado de valor para o ator. Algumas de suas características são descritas abaixo:

- a) Um “Caso de Uso” modela o diálogo entre atores e o sistema.

- b) Um “Caso de Uso” é iniciado por um ator para invocar uma certa funcionalidade do sistema.
- c) Um “Caso de Uso” é um fluxo de eventos completo e consistente.

O conjunto de todos os “Casos de Uso” representa todas as situações possíveis de utilização do sistema. Como exemplo, temos o caso de uso na figura 77.

Figura 77 - Casos de uso de um Sistema Escolar para professores via Web

No exemplo mostrado acima, o ator “Professor” possui quatro casos de uso, “Fazer Login”, “Gerar Renda”, “Escolher Curso” e “Escolher Disciplinas”. Os casos de uso na maioria das vezes obedecem a uma determinada sequência de ocorrência.

As variações das situações nos casos de uso recebem o nome de cenários. Cenário é uma instância de um “Caso de Uso”. O “Caso de Uso” deve ser descrito através de vários cenários. Devem ser construídos tantos cenários quantos forem necessários para se entender completamente todo o sistema. O Caso de Uso pode ser considerado como teste informal, para validação dos requisitos do sistema.

b) Tipos de cenários

Cenários “Primários” são os cenários nos quais o fluxo segue normalmente. Não há quebra no fluxo por alguma espécie de erro. Os cenários “Secundários” são os casos que compõem exceção. O fluxo normal de operação é interrompido.

c) Diagramas de Classe

Os diagramas de classe mostram a estrutura a ser projetada para a construção de modelos que servirão de base para implementação da modelagem. As classes se associam de diversas formas: agregação, associação ou uso. Elas também podem se relacionar com cardinalidade.

Existem modelos básicos de classes que permitem que a mesma possa ser estruturada o nível de análise. Para estes modelos utilizam-se estereótipos, que são classes representativas de uma ou um conjunto de classes, sem ter como preocupação a identificação final das classes, identificação esta que é definida ao projeto.

Utilizam-se normalmente três tipos de estereótipos que representam respectivamente Interface, Controle e Armazenamento. Existem outros tipos de estereótipos; no entanto, utilizaremos somente estes. Eles são visualizados na figura 78.

Figura 78 – Estereótipos: Interface, Controle e Armazenamento

d) Diagrama de Interação

Os diagramas de interação são compostos pelo diagrama de sequência e diagrama de colaboração. Os diagramas de interação são utilizados para fazer a modelagem sobre os aspectos dinâmicos do sistema. No entanto, também são utilizados

para a construção de sistemas executáveis por meio de engenharia de produção e reversas, sendo suas principais funções as de visualizar, especificar, construir e documentar a dinâmica do sistema sobre o tópico básico de um cenário ou do sistema como um todo.

Em um diagrama de interação é mostrada uma interação formada por um conjunto de objetos e seus relacionamentos, incluindo as mensagens que poderão ser trocadas entre eles. Desta forma, um diagrama de sequência é um diagrama de interação que dá ênfase à ordenação temporal de mensagens. Disposto na forma gráfica, o diagrama de sequência é uma espécie de tabela que mostra objetos distribuídos no eixo X, e mensagens em ordem crescente no tempo, no eixo Y. O diagrama de colaboração é um diagrama de interação que dá ênfase à organização estrutural dos objetos que enviam e recebem mensagens.

e) Diagrama de Sequência

Um diagrama de sequência dá importância à ordenação temporal das mensagens. Os diagramas de sequência têm duas características que os diferenciam dos diagramas de colaboração. Primeiro, existe linha de vida no objeto, isto é, a linha tracejada vertical representa a existência de um objeto em um período de tempo. Segundo, existe o foco de controle que é um retângulo alto e estreito, que mostra o período durante o qual um objeto está desempenhando uma ação, diretamente ou por meio de um procedimento subordinado. Podemos observá-lo melhor no diagrama de sequência do caso de uso, como mostrado na figura 79.

Figura 79 - Visualização de um diagrama de sequência

Na figura 79, observa-se um diagrama em que o atendente realiza uma verificação dos espetáculos de um Teatro. No retorno da consulta são mostrados horários e demais informações que serão necessárias para que o cliente, mediante o atendente, possa escolher o espetáculo.

Em outras palavras, pode-se dizer que o *Foco de Controle*, utilizado neste tipo de diagrama, representa o tempo relativo que o fluxo de controle está focalizado num dado Objeto.

f) Diagrama de Colaboração

O diagrama de colaboração dá ênfase à organização dos objetos que participam de uma interação. Os diagramas de colaboração possuem suas características diferenciais. Primeiro existe o caminho, para realizar a vinculação de um objeto a outro. Segundo, existe o número de seqüência, para indicar a ordem das mensagens em relação ao tempo. Uma observação mais prática pode ser feita ao se visualizar o diagrama de colaboração, como no exemplo mostrado na figura 80.

Figura 80 - Exemplo de um diagrama de colaboração

g) Diagrama de Transição de Estados

Os diagramas de estados são empregados para fazer a modelagem de aspectos dinâmicos do sistema. Um objeto reativo tem um claro tempo de vida, cujo comportamento atual é afetado pelo seu passado. Eles são utilizados para visualizar, especificar, construir e documentar a dinâmica de uma sociedade de objetos, ou poderão ser utilizados para fazer a modelagem do fluxo de controle de um estado para outro. O diferencial deste diagrama em relação aos outros tipos de diagramas é o seu conteúdo particular.

Apresenta-se abaixo na figura 81, um exemplo de um diagrama de Estado.

Figura 81 - Exemplo de um Diagrama de Transição de Estado

Nos diagramas de Estado, os verbos que indicam a ação ficam no gerúndio, para indicar a realização do ato mediante o caso de uso em estudo.

Apêndice B – Segurança (Testes de Invasão)

Objetivamos, inicialmente, munir os administradores de sistema com informações que podem ajudá-los a melhorar o nível de segurança de sua rede. Tratou-se dos principais pontos que devem ser cuidadosamente verificados durante a realização de um teste de invasão. Os itens destacados nesse texto foram baseados em um artigo da revista *Internet Security*, cujo autor é Mark Cusick.

1 Introdução

Pode-se definir o objetivo de um teste de invasão como sendo o de verificar a resistência do sistema em relação aos métodos atuais de ataque. Tal método pode ser simplesmente um tipo de engenharia social, onde um invasor pode realizar uma ligação telefônica para a instituição alvo e dialogar com funcionários e solicitar informações de identificação do usuário, bem como senhas de acesso a determinados serviços. No entanto, as ações de invasão podem ser mais complexas, como por exemplo, a utilização de técnicas de *Buffer overflow* para possuir acesso de administrador, também chamado de acesso de *root*.

Os testes de invasão são necessários, pois diariamente são descobertos novos problemas ou *bugs* de segurança nos mais variados e destacados sistemas. Desta forma é de grande importância que a equipe que vai realizar os testes de invasão, utilize técnicas reais, pois se assim não ocorrer, todo o processo de teste poderá se tornar totalmente inválido.

Após a realização dos testes de invasão, há duas possibilidades de resultados:

- A equipe de testes obtém êxito, de modo que o sistema de segurança está aparentemente adequado.
- A equipe de testes não obtém êxito, de modo que o sistema de segurança está com falhas.

- Dependendo do sistema que foi testado, uma boa ou má notícia com respeito aos resultados dos testes servirão como realimentação de serviços a ser melhorados.

2 Testando o Sistema de Detecção de Intruso (SDI)

O sistema de detecção de Intruso (SDI) permite uma notificação ao administrador da rede de comunicação quando há ocorrência de tentativas de intrusão, isto de acordo com a verificação de certos padrões de ataque que podem ser configurados dependendo da ferramenta de detecção de intruso que se está utilizando.

Um sistema de detecção de intruso (SDI) por analogia pode ser comparado a um sistema de alarme contra roubo instalado em um carro. Este alarme contra ladrões de carros pode detectar tentativas de invasão e realizar alguma ação baseada nessa detecção (travar a direção, disparar um alarme, travar os freios, desligar todo o sistema elétrico, travar as portas etc.).

Entretanto, em razão da grande variedade de vulnerabilidade que um SDI deve se preocupar, nos dias atuais realizar a detecção de uma intrusão em uma rede de comunicação ou rede de computadores não é uma tarefa simples. Porém, além dessas limitações, as atuais ferramentas de SDI possuem problemas, tais como:

- Uma alta taxa de falsos positivos, que ocorre quando a ferramenta SDI classifica uma ação como uma possível intrusão, quando na verdade trata-se de uma ação legítima.
- Falsos negativos ocorrem quando uma intrusão real acontece, mas a ferramenta permite que o intruso passe como se fosse o originado de uma ação legítima.

Além desses principais problemas, pode ocorrer também um erro de *Subversion*, que acontece quando o intruso modifica a operação da ferramenta de SDI para forçar a ocorrência de falsos negativos (por exemplo, um ataque de *Syn Flood*). Pois com isso um simples acesso a um determinado serviço da página pode gerar uma detecção da ocorrência desse tipo de ataque. Desta forma, a razão do teste de invasão é de muita importância; com ele (teste) pode-se demonstrar efetivamente se a ferramenta de SDI está operando conforme o esperado e ajudá-lo no refinamento das regras de forma que possa reduzir a taxa de alarmes falsos. A seguir, a realização de análise de dados advinda de um SDI.

3 Tempo de coleta e análise dos dados em SDI

Dentre as características dos métodos de análise de eventos e os tipos de dados examinados, deve-se considerar particularmente importante o tempo de coleta e análise dos dados.

Na abordagem *batch*, também conhecida como orientada a intervalo, os mecanismos de obtenção de dados guardam os registros em arquivos, que serão periodicamente analisados pelo SDI em busca de sinais de ataques ou mau uso. Desse modo, o fluxo de dados dos pontos de monitoramento não é continuamente analisado.

Esse método é adequado para situações onde a ameaça de ataque é pequena e o interesse maior é saber quem são os intrusos e não tomar medidas imediatas. O modo *batch* requer menos carga de processamento, sendo ideal para instituições que dispõem de recursos limitados. Os ataques geralmente acontecem nos mesmos alvos, em diversas etapas. Nesses casos, o método orientado a intervalo pode detectar as assinaturas de ataque sem maiores problemas.

As desvantagens básicas são que os incidentes na segurança raramente são detectados em tempo hábil para que as devidas providências sejam executadas, maximizando os danos e pode haver a necessidade de grande espaço em disco para armazenar os arquivos gerados, principalmente em ambientes de rede.

Os métodos em tempo real garantem que a análise dos dados coletados será feita continuamente, de forma que o SDI possa agir com a rapidez necessária para frustrar uma tentativa de intrusão qualquer. Dependendo da velocidade da análise, o ataque pode ser detectado rápido o bastante para que ele seja interrompido. Mesmo não sendo possível deter o intruso imediatamente, com uma análise sensível e ágil, os administradores do sistema podem mensurar melhor com os danos causados. É possível, também, coletar as informações necessárias prontamente para que o invasor seja logo penalizado.

No entanto, nesse método, há um consumo de memória e demanda de processamento bem maior que no método *batch*. A configuração de sistemas de tempo real é crítica, pois qualquer assinatura mal caracterizada pode gerar uma enorme quantidade de falsos alarmes em um curto espaço de tempo.

4 Resposta aos incidentes

Depois da aquisição dos dados e de sua posterior análise, os sistemas de detecção de intrusos devem reagir às tentativas de invasão ou mau uso. Tal reação pode ser feita de diversas formas, enquadradas em duas categorias básicas: as reações ativas e passivas.

Quando o SDI possui reação ativa, as respostas são dadas automaticamente sempre que determinados tipos de intrusão são detectados. As mais comuns são:

- a) Coletar informação adicional

Quando uma tentativa de ataque é detectada, uma boa reação é analisar as evidências com um enfoque mais apurado. Isto pode ser feito aumentando-se a sensibilidade dos sensores de rede ou *host* para que mais informações possam ser capturadas e estudadas ou criar armadilhas virtuais (*Honeypots*). Com mais dados sobre a tentativa de ataque, é possível obter mais parâmetros para deter o intruso, ou mesmo mais provas para investigá-lo e puni-lo sob os limites legais.

b) Alterar as configurações

Em um ambiente de rede, uma solução bastante direta para deter um atacante é simplesmente cortar sua conexão. Isto não é tarefa trivial para um SDI, mas pode ser feito, interagindo-se com o *firewall* e roteadores para tomar algumas providências, entre as quais: bloquear os pacotes de endereço IP de onde o ataque é iniciado; bloquear as portas, protocolos ou os serviços usados pelos atacantes ou, em caso extremo, fechar todas as conexões que usam determinadas interfaces de rede.

c) Executar ações diretas contra o intruso

A ação mais agressiva que se pode tomar em caso de intrusão é atacar o invasor. Assim, o *hacker* sofrerá com as estratégias que ele próprio usou. Uma abordagem menos hostil é tentar obter informações do computador ou *site* do intruso, para posterior tomada de medidas legais.

O problema de reagir dessa forma tão enérgica é a probabilidade de infringir alguma lei. Assim como é ilegal alguém invadir os sistemas de uma instituição, também o é responder de forma semelhante. Outro problema é que os ataques geralmente partem de endereços de rede falsos, fazendo com que aumente o risco de que *sites* ou usuários inocentes sejam prejudicados.

As respostas passivas delegam ao pessoal de segurança a tarefa de tomar as providências necessárias em relação ao atacante. A reação mais comum é disparar alarmes.

Quando um ataque é detectado, notificações podem ser geradas pelo SDI para avisar aos usuários. A forma mais comum de alarme é um aviso na tela; entretanto, em grandes organizações, é comum a notificação de alertas remotos. A informação provida nas mensagens pode variar bastante, indo de uma simples notificação da intrusão até mensagens detalhadas sobre o grau de severidade do ataque, especificando seu tipo, o endereço IP de onde partiu, o alvo e as técnicas utilizadas.

5 Testando o Plano de Respostas a Incidentes

Um plano de resposta a incidentes é um fator tão importante e crítico, que no caso de o administrador de segurança da rede de computadores não possuir um, o teste de invasão e especialmente o trabalho de se preparar e tornar a rede um pouco mais segura não servirá para muita coisa. Um plano de resposta a incidentes deverá conter e abranger basicamente as seguintes questões:

- Qual o objetivo do plano de resposta para cada tipo de incidente?
- Quais são as ações legais existentes?
- Serão realizadas ações legais (jurídicas) no caso de um incidente?
- Qual tipo de publicidade (a respeito ao ataque) é permitido?
- Quem é o responsável em conduzir uma resposta ao incidente?
- Quem fará parte do grupo de resposta a incidente?
- Qual nível de autoridade é requerido para o grupo de resposta a incidente?

Vale ressaltar que a condução de uma resposta a um incidente deverá estar diretamente relacionada ao tipo de negócio da instituição. Por exemplo, os bancos como instituições financeiras, tomam algumas ações junto à federação nacional de bancos.

6 Riscos envolvidos em um teste de Invasão

Um aspecto que deve ser levado em consideração, referente ao teste de invasão, é que ele pode gerar uma falsa sensação de segurança. Pois a idéia de que a rede suportou com êxito um teste de invasão e por isso está completamente segura não é válida. Isto ocorre em razão da rede ter vulnerabilidades que a equipe de teste não tenha encontrado ou talvez não existam no momento do teste de invasão, mas podem vir a existir após alguma mudança na configuração de uma rede.

Por isso a importância da recomendação para que a equipe que realizará os testes de invasão apresente um relatório detalhado mostrando os tipos de ataques realizados e os resultados que foram alcançados. Deve ser considerada a condição de realização de testes de invasão em vários momentos do sistema.

Um fator relevante que também deve ser considerado refere-se aos possíveis danos causados durante um teste de invasão, sendo possível que o sistema possa ser afetado durante o teste, podendo ocasionar problemas em arquivos importantes e que podem ser perdidos. Isto tudo exige uma definição clara e objetiva dos parâmetros que identificam os pontos onde o teste possui validade. Em sua realização, para que seja considerado um bom teste de invasão, é bom definir que ele não pode ficar preso ou focalizado a um único aspecto do sistema. Para isso é recomendado que seja indicado um departamento que assuma as responsabilidades no caso de ocorrer algum dano ao sistema.

7 Ameaças da Internet

Para a realização dos testes de invasão, algumas instituições contratam para a equipe *Crackers* (ou *Hackers* do mal) para realizarem os testes de invasão. Isto pode ser muito perigoso, pois as pessoas possuem princípios éticos diferentes, de modo que este é um risco que deverá ser levado em consideração quando for realizado um teste de invasão. Para evitar situações desagradáveis, é importante formalizar um contrato com cláusulas que estabeleçam quais as informações referentes aos testes de invasão não podem ser divulgadas. A seguir, o que um contrato de invasão deverá conter:

- Após a realização dos testes de invasão, deverá ser criado um relatório detalhado das vulnerabilidades encontradas, bem como ser dado suporte na correção desses pontos.
- A equipe responsável pelo teste não deve, em momento algum, receber informações sobre o sistema ou sobre parceiros comerciais da instituição. Isto é necessário para proteger o sistema com relação à validade do teste, de forma que intrusos verdadeiros não possuam acesso a estas informações. Isto tudo deve ser executado enquanto o administrador de segurança não tiver testado a ferramenta SDI ou montado o plano de resposta de incidentes.
- Todos os testes de invasão deverão ser conduzidos aplicando-se ferramentas previamente definidas.
- Enquanto os itens acima não forem atendidos, a equipe de teste de invasão não deve ter acesso a sua rede ou sistema.
- Informações públicas, como a estrutura da empresa ou a lista de telefones internos, deverão ser enviadas para a equipe de teste de invasão.

- A equipe de teste de invasão, em nenhum momento, deverá violar a privacidade e os direitos individuais, pois se trata de um teste para avaliar o sistema, e não as informações privadas.
- Após os testes, todos os dados coletados, como arquivos, senhas e qualquer outro tipo de informação obtida, devem ser devolvidos à instituição em que foi realizado o teste sem que cópias sejam retidas pela organização que realizou os testes.
- Todos os testes deverão ser realizados somente para fins instrutivos e não destrutivos. O objetivo é descobrir vulnerabilidades.
- Se algumas etapas dos testes causarem algum dano ao sistema, deverá ser realizado em períodos de baixa ou sem atividades.
- O relatório das atividades de teste de invasão deverá apresentar todos os passos executados, detalhando onde houve acesso e quando não houve. Sendo importante verificar que o relatório deve conter recomendações detalhadas para a correção de qualquer vulnerabilidade que foi encontrada.

8 Fases para realização de um teste de Invasão

a) Coleta de dados e Planejamento

Nesta fase inicial, a equipe de testes de invasão vai procurar aprender tudo o que puder sobre o alvo a ser testado, sendo que inicialmente não terá a preocupação com as vulnerabilidades do sistema. Para isto, a equipe deverá obter informações sobre a estrutura da diretoria, número de telefones/ramais, lista dos parceiros, vendedores e de qualquer outra informação útil. A relação de telefones possui muitas informações para o invasor. A topologia da rede é outra informação que deverá ser verificada, pois conforme essa topologia o invasor poderá determinar quais são os pontos vulneráveis.

Faz-se necessária, também, a construção de um detalhado plano de ataque. Esta etapa é denominada de aproximação indireta, uma espécie de estratégia militar bem planejada em cada ação.

b) Pesquisa do Sistema

Esta etapa é também conhecida como aproximação indireta, fase em que a instituição que será testada é estudada, sendo recolhidas todas as informações possíveis. Isto inclui os IP's da instituição.

c) Teste do Sistema

Denominada também de aproximação direta, apresenta os seguintes passos:

- A identificação do caminho de acesso à instituição alvo, o que pode ser feito por ferramentas como *traceroute*. O objetivo é determinar o caminho e as ACLS implementadas nos roteadores e *Firewalls* (para isso pode-se utilizar a ferramenta *Firewalk*), que identifica quais os serviços permitidos através da ACL.
- A próxima etapa é verificar que tipo de informação pode ser recuperado do servidor de DNS. Um exemplo é a recuperação do registro HINFO. Se esta informação estiver disponível haverá condições do tipo de Sistema Operacional da Instituição alvo.
- Agora é o momento de determinar o endereço do *Firewall* para a realização de testes, para em seguida verificar as máquinas que estão mal configuradas. Elas são local ideal para tentar um acesso não autorizado.
- Tendo o mapa das máquinas da instituição alvo, o passo seguinte é determinar quais os *hosts* que estão ativos e conectados na Internet.
- Caso alguma máquina esteja ativa e conectada à Internet, é possível realizar um Port Scan. Um Port Scan possui o objetivo de determinar

quais as portas (TCP/UDP) estão ativas. Existem várias ferramentas que podem fazer o *Port scan*: *nmap*, *strobe*, *tcp_scan*, *mdp_scan*, *netcat* e *queso*.

- Descobertas as portas ativas dos *hosts*, são utilizadas para se obter mais informações dos *hosts* (exemplo *banner*) como, por exemplo, informações sobre os usuários de cada sistema. Conecta-se a cada uma das portas TCP/UDP e analisam-se as respostas, para identificar as versões e vulnerabilidades de serviços.

Colhidas as informações anteriores, é chegado o momento de associar as informações do sistema com as vulnerabilidades conhecidas. Por exemplo:

- Versão do Sistema Operacional
- Versões dos serviços
- Arquitetura do Sistema

E a partir dessas informações, montar o mapa.

O invasor até o momento não realizou ação alguma de intrusão no sistema alvo, mas o futuro invasor já possui informações suficientes da instituição. Como por exemplo, quem é quem na instituição bem como as empresas que prestam serviços a ela. O invasor pode falar em nome de funcionários da Instituição ou de empresas contratadas.

As técnicas de engenharia social podem ser bem eficazes quando o invasor pode ligar para as pessoas se identificando, por exemplo, como funcionário de uma companhia telefônica, de uma distribuição de equipamentos de hardware e software que podem ser úteis para atingir a instituição alvo. Um exemplo disso é a solicitação da identificação do usuário e senha para a realização de uma atualização automática de determinado hardware ou software, ou até mesmo um teste de funcionalidade de um

equipamento. Isso tudo pode ser feito por um invasor muito educado e cortês, lembrando que o uso de uma delicada e graciosa voz feminina pode aumentar a probabilidade de obtenção de informações.