



# **CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS**

COMUNIDADE EVANGÉLICA LUTERANA "SÃO PAULO"  
Recredenciado pela Portaria Ministerial nº 3.607 - D.O.U. nº 202 de 20/10/2005

**JOSE NETO LUZ CARNEIRO**

## **IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE APOIO À AUDITORIA EM REDES WINDOWS COM USO DE BASES DE DADOS RELACIONAIS.**

**ESTUDO DE CASO NO TRIBUNAL REGIONAL ELEITORAL DO TOCANTINS**

**Palmas**

**2014**



# **CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS**

COMUNIDADE EVANGÉLICA LUTERANA "SÃO PAULO"  
Recredenciado pela Portaria Ministerial nº 3.607 - D.O.U. nº 202 de 20/10/2005

**JOSÉ NETO LUZ CARNEIRO**

## **IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE APOIO À AUDITORIA EM REDES WINDOWS COM USO DE BASES DE DADOS RELACIONAIS.**

**ESTUDO DE CASO NO TRIBUNAL REGIONAL ELEITORAL DO TOCANTINS**

Trabalho de Conclusão de Curso (TCC II) elaborado e apresentado como requisito para obtenção do Título de bacharel em Sistemas de Informação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. Mestre Fernando Luiz de Oliveira.

**Palmas**

**2014**

**JOSÉ NETO LUZ CARNEIRO**

**IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE APOIO À AUDITORIA  
EM REDES WINDOWS COM USO DE BASES DE DADOS  
RELACIONAIS.**

**ESTUDO DE CASO NO TRIBUNAL REGIONAL ELEITORAL DO TOCANTINS**

Trabalho de Conclusão de Curso (TCC II) elaborado e apresentado como requisito para obtenção do Título de bacharel em Sistemas de Informação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. Mestre Fernando Luiz de Oliveira.

APROVADA EM JULHO DE 2014

**BANCA EXAMINADORA**

---

Prof. MSc. Fernando Luiz Oliveira  
Centro Universitário Luterano de Palmas

---

Prof<sup>a</sup>. M.Sc. Madianita Bogo Marioti  
Centro Universitário Luterano de Palmas

---

Prof<sup>a</sup> M.Sc. Cristina Filipakis  
Centro Universitário Luterano de Palmas

**PALMAS**

**2014**

## DEDICATÓRIA

Dedico esta monografia à minha família pelo apoio, carinho, compreensão e confiança demonstrada, que com certeza tornaram esta longa caminhada mais fácil de ser percorrida.

## **AGRADECIMENTOS**

As primícias das gratidões dedico a Deus que me deu forças, clareza de entendimento e perseverança que me fizeram não desistir da conclusão deste Curso, de cujos conhecimentos obtidos integram a respiração diária. O meu Deus é um ser vivo que sempre esteve presente em todos os momentos da minha vida, e, neste contexto, não se ausentou de forma alguma na realização deste trabalho.

Sou muito grato à minha linda família, minha esposa Eliete e os meus já não pequenos filhos Victor, Heloisa e Samuel, ao sobrinho Arich Andrade, os quais estiveram lado a lado sempre me apoiando, apesar das indignações pela extensão de tempo que se deu do início ao fim deste curso. Sempre acreditei que as palavras têm poder, e por descuidos, às vezes brincamos com elas: Lembro-me há vários anos brincava que quando meu primogênito, que estava na sua tenra idade, estivesse entrando na faculdade eu estaria saindo. Essas palavras de brincadeiras se tornaram verdade, meu filho está muito preste a ser acadêmico. Vocês, família amiga, tens sido a razão que me motivou ir à final. Amo muito vocês.

Aos meus pais, irmãos e sogros meus sinceros agradecimentos. Vocês acompanharam de perto esta longa caminhada que, após várias paradas, finalmente Deus permitiu enfincar um ponto final. Obrigado pela cobertura em oração e por palavras otimistas. Sou grato a Deus pela vida de vocês.

Agradeço também ao Professor Fernando pelo incentivo, dedicação autêntica nas minúcias corretivas e pelas sugestões de melhorias ao trabalho, e, principalmente, paciência extrema em aprovar inúmeras propostas sem conclusão e não desistir do orientando. A sua atuação como orientador, com certeza, dá segurança aos seus orientandos durante a realização dos trabalhos. Aos demais professores a nossa gratidão pelos incentivos, companheirismo e amizade.

Aos Gestores de TI do Tribunal agradeço imensamente ao irrestrito apoio com a disponibilidade dos recursos de TI e do ambiente em que foi desenvolvido o Trabalho. Com certeza os resultados poderão ser utilizados para bem servir as demandas da Instituição. As colegas e amigos de trabalho o meu muito obrigado pelos incentivos para concluir este curso antes da aposentadoria, principalmente ao amigo Neuziron pela disponibilidade e ajuda irrestrita na montagem do laboratório de testes.

Enfim, também agradeço a todos que me ajudaram de forma direta ou indireta na realização deste trabalho. O meu muito obrigado a todos.

## SUMÁRIO

1. INTRODUÇÃO.....	15
2. REVISÃO DE LITERATURA .....	17
2.1. Gerenciamento de Redes .....	17
2.1.1. Componentes de gerenciamento.....	18
2.1.1.1. Gerentes e Agentes .....	19
2.1.1.2. Base de Informações de Gerenciamento - MIB.....	20
2.1.1.2.1. Hierarquias de Gerenciamento .....	21
2.1.1.3. Objeto Gerenciado.....	24
2.2. Modelos de gerenciamento de redes.....	25
2.2.1. Diferenças entre os Modelos OSI e Internet.....	25
2.2.2. Modelo Gerenciamento Internet.....	26
2.2.3. Modelo OSI de Gerenciamento de Rede .....	30
2.2.4. Serviços do Gerenciamento OSI.....	32
2.2.5. Áreas Funcionais de Gerenciamento de Redes.....	35
2.2.5.1. Funções de Gerenciamento.....	36
2.2.5.2. Gerenciamento de configuração .....	40
2.2.5.3. Gerenciamento de Falhas.....	40
2.2.5.4. Gerenciamento de Desempenho .....	41
2.2.5.5. Gerenciamento de Contabilização .....	41
2.2.5.6. Gerenciamento de Segurança .....	42
2.3. Protocolo CMIP.....	42
2.4. Segurança de Redes .....	47
2.5. <i>Infra-estrutura da rede Windows</i> .....	50
2.6. Auditoria em Redes Windows.....	52

2.7. <i>Windows Management Instrumentation (WMI)</i> .....	54
2.7.1. Arquitetura WMI.....	58
2.7.1.1. Recursos Gerenciados.....	59
2.7.1.2. Infra-Estrutura do WMI.....	60
2.7.1.2.1. Provedores de WMI.....	60
2.7.1.2.2. CIMOM.....	62
2.7.1.2.3. Repositório CIM.....	63
2.7.1.2.3.1. <i>NameSpaces</i> .....	66
2.7.1.2.3.2. Categorias de Classes.....	67
2.7.1.2.3.3. Tipos de Classes CIM.....	68
2.7.1.2.3.4. Componentes de uma classe.....	71
2.7.1.3. Clientes WMI.....	74
2.7.1.4. Segurança em WMI.....	74
2.7.1.4.1. <i>WMI Namespace</i> .....	75
2.7.1.4.2. Segurança DCOM.....	76
2.7.1.4.3. Padrão de Segurança do Windows.....	76
2.7.2. Descrição das principais classes WMI para a solução proposta.....	76
2.7.2.1. Classe <i>Win32_NTEventLogFile</i> .....	77
2.7.2.2. Classe <i>Win32_NTLogEvent</i> .....	82
2.7.2.3. Classe <i>Win32_NTLogEventComputer</i> .....	86
2.7.2.4. Classe <i>Win32_NTLogEventLog</i> .....	87
2.7.2.5. Classe <i>Win32_NTLogEventUser</i> .....	87
3. MATERIAIS E MÉTODOS.....	89
3.1. <i>Materiais</i> .....	89
3.1.1. <i>Infraestrutura Física</i> .....	89
3.1.2. <i>Infraestrutura Lógica</i> .....	89
3.1.2.1. <i>Servidor Windows 2003 Server</i> .....	90
3.1.2.2. <i>Ambiente de virtualização Oracle VM VirtualBox</i> .....	90
3.1.2.3. <i>Estações de trabalho Virtuais com Windows XP SP3</i> .....	91
3.1.2.4. <i>Group Policy Manager Console - GPMC</i> .....	91
3.1.2.5. <i>Banco de dados relacional SQLServer 2005</i> .....	91
3.1.2.6. <i>PowerShell Script ISE 2.0</i> .....	92
3.2. <i>Metodologia</i> .....	92



4. RESULTADOS E DISCUSSÃO .....	94
4.1. Definição da Política de Auditoria para a rede do TRE-TO .....	95
4.2. Definição e configuração das diretivas de Auditoria.....	95
4.3. Segurança de acesso aos registros e filtragem para coletas dos logs .....	100
4.4. Gravação dos registros de logs em arquivos locais.....	101
4.5. Gravação dos registros de logs em banco de dados .....	102
4.6. Propriedades da classe WMI utilizada no Trabalho.....	103
4.7. Tabelas integrantes do esquema relacional do banco de dados .....	105
4.8. Cenário de testes e Resultados .....	107
4.8.1. Infraestrutura do cenário de testes.....	107
4.8.1.1. Configuração do Ambiente Virtual .....	108
4.8.1.2. Configuração do Controlador de Domínio .....	110
4.8.2. SCRIPT WMI .....	111
4.8.3. Cenário de testes .....	118
4.8.4. Primeiro teste do cenário e resultado .....	118
4.8.5. Segundo teste do cenário e resultado .....	124
4.8.6. Terceiro teste do cenário e resultado .....	127
4.8.7. Quarto teste do cenário e resultado .....	131
4.8.8. Quinto teste do cenário e resultado.....	134
5. CONSIDERAÇÕES FINAIS .....	139
6. REFERÊNCIAS BIBLIOGRÁFICAS .....	141
APÊNDICE A .....	143
APÊNDICE B.....	146

## LISTA DE FIGURAS

Figura 1: Relação Gerente-Agente, modificada de (FERNANDEZ, 2007, On-Line). ....	19
Figura 2: Comunicação entre gerente e agente (RAIMIR, 1998, p. 21).....	20
Figura 3: Hierarquia de classes de objetos gerenciados (BRISA, 1993, p.141).....	22
Figura 4: Ex. de Hierarquia de Registro–Modelo OSI (BRISA, 1993, p. 145).....	23
Figura 5: Ex. de Hierarquia de Registro – Modelo Internet (BRISA, 1993, p.164).....	24
Figura 6: Modelo de Gerenciamento Internet (EMBRATEL, 1994, p. 535). ....	28
Figura 7: Pilha de protocolos utilizada pelo SNMP (EMBRATEL, 1994, p.545).....	29
Figura 8: Entidade de aplicação de Gerenciamento de Sistemas .....	31
Figura 9: Comunicação entre entidades de aplicação de ger. de sistemas .....	32
Figura 10: Detalhamento da Camada de Aplicação (BRISA, 1993, p. 173).....	34
Figura 11: Áreas funcionais do gerenciamento de redes (RAIMIR, 1998, p. 24).....	35
Figura 12: Estados de gerenciamento (EMBRATEL, 1994, p.517).....	37
Figura 13: Gerenciamento de relatórios de falhas (EMBRATEL, 1994, p.521).....	38
Figura 14: Modelo de controle de Acesso (BRISA, 1993, p. 103).....	40
Figura 15: Processo de Segurança baseado em Riscos e Bens (HORTON, 2003, p.5). ....	48
Figura 16: Arquitetura WMI (WMI, 2008, On-Line).....	59
Figura 17: Estrutura interna do Repositório CIM (WMI, 2008, On-Line).....	64
Figura 18: Estrutura de definição de classe de recurso gerenciado.....	71
Figura 19: Exemplo de registro de log do evento de <i>logon</i> (JNLC, 2006, p. 92).....	85
Figura 20: Configuração de diretiva de auditoria utilizando GPMC.....	96
Figura 21: Ambiente de gerenciamento da ferramenta GPMC).....	97
Figura 22: Edição de GPOs via GPMC.....	98
Figura 23: Link de GPO em objetos do AD utilizando GPMC.....	99
Figura 24: Aplicação de GPO por herança.....	100
Figura 25: Critérios de Gravação dos <i>logs</i> dos eventos de auditoria.....	102

Figura 26: Propriedade “Message” – campo Descrição. ....	105
Figura 27: Esquema Relacional do Banco de dados .....	106
Figura 28: Infraestrutura do cenário de testes. ....	108
Figura 29: Ambiente Virtualizado com VirtualBox. ....	109
Figura 30: Estrutura do AD do ambiente de rede do cenário de testes. ....	111
Figura 31: Script de Coleta e Gravação de <i>Logs</i> de Auditoria .....	112
Figura 32: Script de Coleta e Gravação de <i>Logs</i> de Auditoria. Cont... ..	113
Figura 33: Script de Coleta e Gravação de <i>Logs</i> de Auditoria. Cont... ..	114
Figura 34: Script de Coleta e Gravação de <i>Logs</i> de Auditoria. Cont... ..	115
Figura 35: Script de Coleta e Gravação de <i>Logs</i> de Auditoria. Cont... ..	116
Figura 36: Script de Coleta e Gravação de <i>Logs</i> de Auditoria. Cont... ..	117
Figura 37: 1ª Consulta de Auditoria do primeiro teste. ....	120
Figura 38: 2ª Consulta de Auditoria do primeiro Teste. ....	121
Figura 39: 3ª Consulta de Auditoria do primeiro Teste. ....	122
Figura 40: 4ª Consulta de Auditoria do primeiro Teste. ....	124
Figura 41: 1ª Consulta de Auditoria do segundo Teste. ....	126
Figura 42: 2ª Consulta de Auditoria do segundo Teste. ....	127
<b>Figura 43: 1ª Consulta de Auditoria do terceiro Teste. ....</b>	<b>129</b>
<b>Figura 44: 2ª Consulta de Auditoria do terceiro Teste. ....</b>	<b>131</b>
<b>Figura 45: 1ª Consulta de Auditoria do quarto Teste. ....</b>	<b>133</b>
<b>Figura 46: 2ª Consulta de Auditoria do quarto Teste. ....</b>	<b>134</b>
<b>Figura 47: 1ª Consulta de Auditoria do quinto Teste. ....</b>	<b>136</b>
<b>Figura 48: 2ª Consulta de Auditoria do quinto Teste. ....</b>	<b>137</b>

## LISTA DE TABELAS

Tabela 1: Primitivas de definição que define o serviço de troca de informações. ....	33
Tabela 2: Mapeamento das Operações de Gerenciamento .....	36
Tabela 3: Classes de operação e associações definidas .....	42
Tabela 4: Relacionamento das classes de operação do CMIP com os serviços CMISE. ....	43
Tabela 5: PDUs do protocolo CMIP .....	43
Tabela 6: Serviços do CMIP em relação ao ACSE .....	44
Tabela 7: Serviços do CMIP em relação ao ROSE .....	44
Tabela 8: Mapeamento dos parâmetros do CMIP nos parâmetros das APDUs do ROSE. ....	45
Tabela 9: Exemplo de Script WMI – Memória .....	56
Tabela 10: Exemplo de Script WMI - Serviços.....	56
Tabela 11: Exemplo de Script WMI – Logs de Eventos .....	57
Tabela 12: Lista parcial dos provedores padrões WMI.....	61
Tabela 13: Como especificar uma NameSpace no script .....	66
Tabela 14: Recuperando NameSpaces CIM usando script WMI. ....	67
Tabela 15: Recuperando as classes definidas na NameSpaces root\cimv2. ....	68
Tabela 16: Comparação das propriedades das classes Pais e Filhas. ....	69
Tabela 17: Local de armazenamento do CIM .....	73
Tabela 18: Permissões WMI NameSpace .....	75
Tabela 19: Sintaxe da classe Win32_NTEventlogFile .....	77
Tabela 20: Descrição dos Métodos da classe Win32_NTEventlogFile .....	78
Tabela 21: Descrição das Propriedades da classe Win32_NTEventlogFile.....	79
Tabela 22: Sintaxe da classe Win32_NTLogEvent.....	82
Tabela 23: Descrição das Propriedades da classe Win32_NTLogEvent.....	83
Tabela 24: Exemplo de utilização da classe Win32_NTLogEvent .....	84
Tabela 25: Sintaxe da classe Win32_NTLogEventComputer.....	86

Tabela 26: Descrição das Propriedades da classe Win32_NTLogEventComputer.....	86
Tabela 27: Sintaxe da classe Win32_NTLogEventLog .....	87
Tabela 28: Descrição das Propriedades da classe Win32_NTLogEvenLog .....	87
Tabela 29: Sintaxe da classe Win32_NTLogEventUser .....	88
Tabela 30: Descrição das Propriedades da classe Win32_NTLogEvenUser .....	88
Tabela 31: Diretivas definidas pela Política de Auditoria.....	96
Tabela 32: Propriedades da classe Win32_NTLogEvent utilizadas no Trabalho .....	103
Tabela 33: Unidades Organizacionais do Active Directory .....	110

## LISTA DE SIGLAS

OSI	Open Systems Interconnection
ISO	Internacional Organization for Standardization
MIB	Management Information Base
CMIP	Common Management Information Protocol
SNMP	Simple Network Management Protocol
SMI	Structure of Management information
GDMO	Guidelines for the Definition of Managed Objects
SGMP	Simple Gateway Management Protocol
RMON	Remote Network Monitoring
ICMP	Internet Control Message Protocol
SMAE	System Management Application Entity
SMASE	System Management Application Service Element
ACSE	Association Control Service Element
ASE	Application Service Elements
PDU	Protocol Data Units
CMIS	Common Management Information Service
CMISE	Common Management Information Service Element
WMI	Windows Management Instrumentation
WBEM	Web-Based Enterprise Management
CIM	Modelo de Informações Comuns
CIMON	Gerente de objetos do Modelo de Informações Comuns
API	Application Programming Interface
AD	Active Directory
OU	Unidade Organizacional

## RESUMO

A crescente expansão das redes de computadores e a heterogeneidade de padrões e equipamentos tornaram complexo o controle o gerenciamento das redes. Em consequência dessa complexidade as instituições internacionais resolveram definir padrões de gerenciamento a serem seguidos pelos fabricantes de hardware e softwares de redes, resultando nos modelos de gerenciamento de rede OSI e no modelo Internet (TCP/IP). Dentre as áreas de gerenciamento definidos se encontra o gerenciamento de segurança, o qual incorpora o processo de auditoria da rede. Fazer auditoria do que acontece em uma rede em termos de acessos indevidos a equipamentos, arquivos, roubos de informações estratégicas das instituições, etc., é um fator bastante complexo e de domínio de poucos que se dedicam nessa árdua tarefa. O presente trabalho objetiva minimizar essa complexidade para realização de auditoria em redes Windows, por meio do processo de aplicação de Política de Auditoria de Segurança e dos mecanismos de transformação dos dados registrados em *logs* de eventos de auditoria em informações úteis para tomadas de decisões, utilizando, para tanto, a tecnologia de gerenciamento do Windows, disponibilizada pela Microsoft, denominada de *Windows Management Instrumentation* (WMI).

**PALAVRA CHAVE:** Gerenciamento, segurança, rede, auditoria, OSI, Internet, script, WMI.

## **1. INTRODUÇÃO**

O princípio de redes de computadores surgiu como solução simples de integrar os recursos tecnológicos, compartilhar informações e equipamentos de uma empresa, tendo o foco principal reduzir custos e aumentar a produtividade. Porém, a expansão das redes agregou novos valores e a cada dia surgem novas tecnologias de redes para atender a crescente demanda dos usuários. Neste contexto, as redes se expandiram das simples conexões locais para redes de longo alcance (WAN), formando posteriormente a rede mundial de computadores denominada internet. Para concretizar essa realidade diversos fabricantes de equipamentos de redes, com tecnologias diversas e padrões de funcionamentos diferentes surgiram e foram inseridas no mercado, causando uma enorme complexidade para manter o controle e o gerenciamento dessas redes.

Na prática quando se implanta uma rede os investimentos são voltados para montar a estrutura física e lógica da rede, não integrando no primeiro momento os investimentos com o gerenciamento. O surgimento de problemas de conexões, baixo desempenho, roubos de informações, acessos indevidos e falta de controle de tráfego da rede deixam desesperados os administradores da rede e conseqüentemente faz emergir a necessidade de gerenciamento da rede de forma que integre todas as tecnologias adotadas. Surgiram, então, esforços isolados para encontrar soluções de gerenciamento, porém, sem sucesso porque não havia padrão a ser seguido pelos fabricantes de hardwares e softwares de rede de forma que possibilitasse integração na comunicação entre os diversos componentes da rede.

Diante dessa problemática foram definidos dois padrões de gerenciamento de redes, o modelo OSI (Open Systems Interconnection) e o modelo Internet (TCP/IP). Ambos os modelos apresentam similaridades nas funções básicas de interação entre os elementos da rede para fins de obter informações e fazer o gerenciamento. Os dois



modelos adotam os componentes de gerenciamento conceitualmente denominados de gerente, agentes, base de informações de gerenciamento e o protocolo de transporte das operações e informações de gerenciamento. Apesar de terem complexidades diferentes, estes dois modelos atingem as tarefas básicas de gerência de redes: obter informações da rede e tratá-las, fazer diagnóstico e apresentar soluções (EMBRATEL, 1994, p.514-515).

O gerenciamento de redes para ser eficiente engloba diversas áreas funcionais, tais como o gerenciamento de segurança, na qual está inserido o processo de auditoria de rede, o foco do presente trabalho. Através do processo de auditoria torna possível monitorar o acesso à rede, detectar intrusos, acesso às informações sigilosas e de acesso restritos, enfim, traçar o perfil da rede e dos usuários.

A auditoria se faz com base nas informações registradas em arquivos de *logs* gerados nos diversos objetos da rede. A análise desses *logs*, além de ser uma tarefa tediosa e complexa, é impraticável quando se tem que acessar máquina a máquina para obtê-los. Dentro deste contexto, este trabalho tem por objetivo minimizar essa complexidade para realização de auditoria em redes Windows, por meio do processo de aplicação de Política de Auditoria de Segurança e dos mecanismos de transformação dos dados registrados em *logs* de eventos de auditoria em informações úteis para tomadas de decisões, cuja solução proposta contempla a construção de Script WMI para realização de coletas dos *logs* de eventos de auditoria da rede de computadores e armazená-los em uma base de dados centralizada. Desta forma, pretende-se facilitar a obtenção das informações para auditoria através de consultas e relatórios gerenciais originadas em uma única base de dados.

O presente trabalho foi desenvolvido em duas fases distintas e complementares: na primeira parte procurou-se aprofundar em pesquisas para compor o embasamento teórico que norteou e refletiu a possibilidade real de aplicação de uma sensata política de auditoria de segurança em rede de computadores. A segunda contemplou a parte prática do Trabalho, a qual fez prova dos conceitos ora desenvolvidos e por meio dos resultados demonstrou quão valiosas são as informações obtidas da aplicação de Política de Auditoria para obter o conhecimento do perfil da Rede e dos usuários para fins de gerência e controle de uso dos recursos de Tecnologia da Informação.

## **2. REVISÃO DE LITERATURA**

Esta seção objetiva apresentar o estudo teórico sobre os principais conceitos e tecnologias necessárias para o desenvolvimento do projeto proposto. Como o assunto auditoria está incluso no tema gerenciamento de redes e para melhor entendimento e ampliação do conhecimento sobre o trabalho serão abordados conceitos sobre gerenciamento de redes, modelos de gerenciamento, segurança de redes, infra-estrutura de redes windows bem como auditoria em rede windows. Por último será abordada a parte teórica da tecnologia *Windows Management Instrumentation* – WMI, a qual será fundamental para o objetivo final do projeto.

### **2.1. Gerenciamento de Redes**

A existência de equipamentos oriundos de tecnologias e fornecedores diferentes em uma rede de computadores, assim como, a diversidade de protocolos de comunicação e padrões de gerenciamentos próprios desses equipamentos poderá deixar o processo de gerenciamento da rede bastante complexo.

Por outro lado, a infra-estrutura lógica de uma rede poderá coexistir com vários domínios gerenciais e organizacionais com políticas de gerenciamento e de segurança conforme a necessidade inerente a cada um. Esta situação dificulta ainda mais a adoção de políticas de segurança e padrões de gerenciamento uniformes para a rede, bem como unir conhecimentos técnicos capaz de dominar essa diversidade.

Portanto, para ser possível obter um bom nível de gerenciamento da rede torna-se necessária a adoção de recursos tecnológicos apropriados e recursos humanos bem capacitados e dedicados à gerência da rede.

A principal função do gerenciamento de redes é manter o ambiente em pleno funcionamento com bom desempenho, com as informações e recursos compartilhados

sempre disponíveis e com segurança no controle de acesso e uso dos dados que são transportados na rede. Para tanto, as tarefas básicas de gerência em redes precisam ser bem implementadas e executadas para que seja possível descobrir, prever e reagir a problemas. As tarefas básicas de gerência são (EMBRATEL, 1994, p.514-515):

- Obter informações da rede e tratá-las;
- Fazer diagnóstico com base nas informações tratadas; e
- Encaminhar soluções para os problemas.

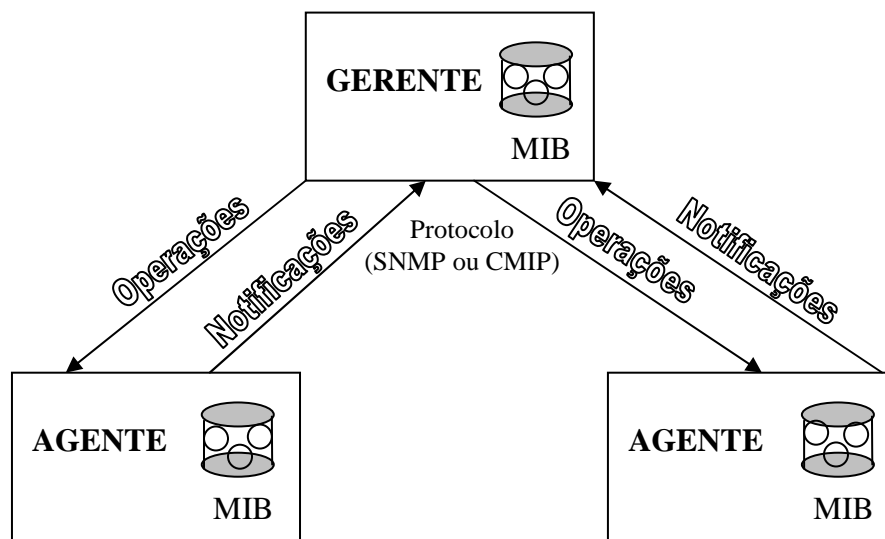
Assim, fazer a gerência de pequenas redes, contendo poucos equipamentos, não é tão complicado, mas gerenciar uma rede composta de várias sub-redes que abrangem diversos departamentos e pavimentos, ou ainda uma rede remota (WANs/MANs) com uma cobertura geográfica extensa, a qual incorpora vários pontos e equipamentos de interconexão, é uma tarefa indispensável e bastante complexa. Por isto é exigido uma dedicação e conhecimento técnico, além de boas ferramentas de gerenciamento de redes para que a mesma possa ter uma gerência aceitável e controlável. Para tanto, para melhor aproveitar os recursos e vantagens que as redes propiciam é necessário investir em ferramentas de gerenciamento para facilitar a coordenação, controle e monitoração do comportamento dos elementos da rede de forma integrada (EMBRATEL, 1994, p.508).

Diante dessas dificuldades e para resolver principalmente os problemas quanto à heterogeneidade de soluções de redes e interoperabilidade entre domínios diferentes, a qual é fundamental para prover sistemas que suportem as cinco áreas funcionais de gerenciamento: gerenciamento de falhas, de configuração, de desempenho, de segurança e de contabilização; os institutos de normalização e fabricantes de equipamentos de rede de computadores têm se juntado e definido padrões de gerenciamento de redes (EMBRATEL, 1994, p.508). Os padrões mais conhecidos para gerência de redes são: o modelo OSI (*Open Systems Interconnection*) de gerenciamento de redes, definido pela ISO (*International Organization for Standardization*) e o modelo de Gerenciamento Internet (TCP/IP).

### **2.1.1. Componentes de gerenciamento**

Tanto o modelo OSI como o modelo Internet apresentam semelhanças quanto ao gerenciamento, os quais incluem os conceitos de agentes, gerente da rede, uma base de informações de gerenciamento (MIB - *Management Information Base*) e um protocolo de

aplicação com a finalidade de transportar operações e informações de gerenciamento entre os agentes e gerentes. A figura 1, apresentada a seguir, demonstra esta relação.



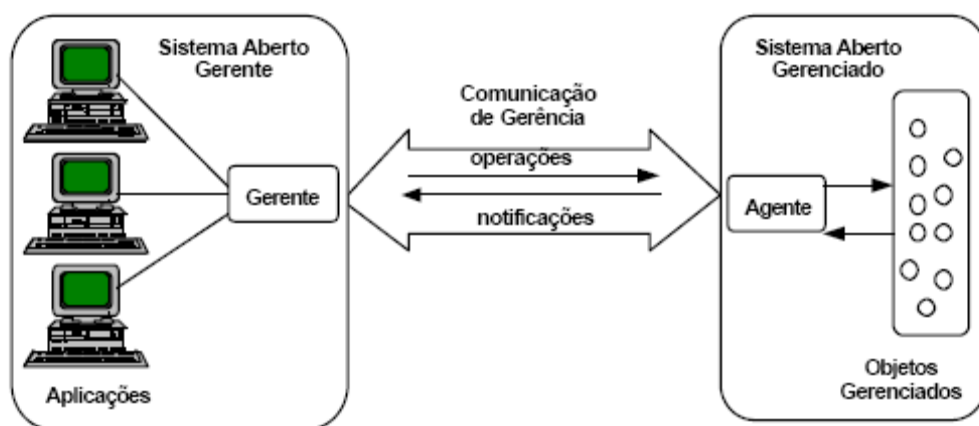
**Figura 1:** Relação Gerente-Agente, modificada de (FERNANDEZ, 2007, On-Line).

Conforme demonstra a figura 1, o gerente possui sua base de informações gerenciais que contempla dados de todos os objetos gerenciados da rede, e possui a função de enviar solicitações e operações a serem executadas pelos agentes. Os agentes, além de atender as solicitações e executar as operações enviadas pelos gerentes, possuem a função de enviar notificações sobre ocorrências na rede ou nos objetos, assim como possuem uma base de informações sobre os respectivos objetos.

#### 2.1.1.1. Gerentes e Agentes

O sistema gerente possui a responsabilidade de coordenar as atividades a serem executadas, através do envio de solicitações aos processos agentes. Os agentes possuem a responsabilidade de coletar e armazenar informações dos respectivos objetos gerenciados e notificar o sistema gerente sobre qualquer alteração ocorrida no estado dos objetos, além de ser responsável pela execução das operações sobre os objetos gerenciados e responder às solicitações dos gerentes. A comunicação entre gerente e agentes se dá conforme figura 2. A disponibilização das informações para os sistemas gerentes ocorre de duas formas (FREITAS, 2001, p. 6):

- *Request-response* ou *Polling*: quando o sistema gerente solicita a um agente o envio de alguma informação armazenada na MIB agente, caracterizando uma interação do tipo pergunta-resposta entre o gerente e agente.
- *Event-reporting* ou relato de eventos: quando o agente é o autor da ação de enviar informações ou notificações ao gerente, se posicionando o gerente na condição de ouvinte. Esses eventos ocorrem quando o agente envia informações sobre o seu estado atual ou quando envia notificações a respeito de uma ocorrência relevante, por exemplo, quando acontece algo anormal na rede.



**Figura 2:** Comunicação entre gerente e agente (RAIMIR, 1998, p. 21).

O modelo OSI de gerenciamento de redes utiliza o protocolo CMIP (*Common Management Information Protocol*) para definir as regras de comunicação entre os sistemas gerente e agente, enquanto o modelo de Gerenciamento Internet utiliza o protocolo SNMP (*Simple Network Management Protocol*).

### 2.1.1.2. Base de Informações de Gerenciamento - MIB

A base de informações de gerenciamento é o repositório dos objetos gerenciados e pode-se classificá-la como um componente fundamental em um sistema de gerenciamento. Isto porque quando há uma solicitação do sistema gerente sobre um objeto gerenciado, o agente com base nas informações armazenadas em sua MIB, responde à solicitação do gerente.

Não faz parte do conceito da MIB nenhuma definição de estrutura interna da base de informação, nem a forma como será armazenada. O que tem que ser visualizado na definição de uma MIB é o local onde estarão os objetos gerenciados, e a possibilidade dos

sistemas identificarem de forma correta os respectivos objetos. Para tanto, devem ser seguidas as regras de definição e identificação das variáveis na MIB, definidas pela Estrutura de Informação de Gerenciamento – SMI (*Structure of Management Information*). Esta estrutura determina que os nomes e tipos de variáveis da MIB devem ser definidos e referenciados de acordo com a linguagem formal padronizada pela ISO, a saber: a ASN.1 (*Abstract Syntax Notation one*). O padrão ASN.1 define regras tanto para a linguagem humana quanto para a linguagem codificada, utilizada nos protocolos de comunicação (RAIMIR, 1998, p.23).

A MIB é usada tanto no agente quanto no gerente para armazenar e trocar informações de gerenciamento, conforme figura 1. A MIB associada ao agente é denominada MIB agente, e a associada ao gerente é denominada MIB gerente. Uma MIB gerente consiste de informações de todos os componentes que ela gerencia, enquanto a MIB agente necessita conhecer somente sua informação local.

Conforme definição da SMI (*Structure Management Information*) a MIB, em termos gerais, é caracterizada tanto pela organização dos itens e pela forma de identificá-los, como também pelos tipos de operações realizadas sobre os mesmos. Portanto, a diferença existentes entre as MIBs (MIBs da ISO e da Internet) se concentram exatamente nas hierarquias utilizadas para representar os objetos que serão gerenciados. No modelo OSI são definidas as hierarquias de herança, nomeação e de registro. No entanto, em ambos os modelos de gerenciamentos referenciados a hierarquia de registros é utilizada para identificar de forma genérica os objetos, cuja hierarquia foi especificada de acordo com as normas definidas pelo padrão ASN.1 usada na atribuição de identificadores de objetos (BRISA, 1993, p. 144).

#### **2.1.1.2.1. Hierarquias de Gerenciamento**

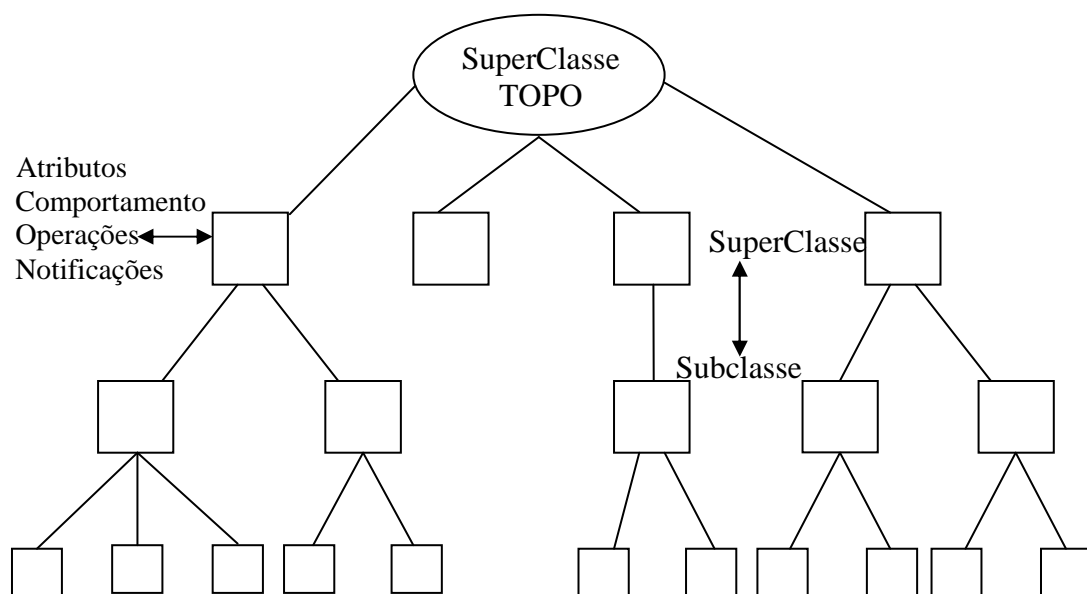
A SMI (*Structure of Management Information*), com base na abordagem de orientação a objetos, introduz os conceitos de hierarquia de herança, hierarquia de nomeação e de registros usados na caracterização e identificação dos objetos gerenciados. Além disso, é responsável pela definição do conjunto de operações que pode ser realizado sobre os objetos da MIB e o comportamento dos mesmos em relação à execução das operações (BRISA, 1993, p. 139).

Convém salientar que os três tipos de hierarquias são definidos para o modelo OSI, visto que no modelo Internet não é definida nenhuma hierarquia de nomeação para identificar instâncias de objeto, nem hierarquia de herança (classes). Neste modelo é definido somente o conceito de instância de objeto desvinculado do conceito de atributos, como também somente tipos de objetos e não classes de objetos (BRISA, 1993, p. 139). As hierarquias serão melhores descritas nos itens abaixo:

- **Hierarquia de Herança:** a hierarquia de herança é responsável por definir o conceito de classes de objetos e está relacionada às propriedades associadas aos objetos, as quais são descritas através de seus atributos, comportamento, pacotes condicionais, operações e notificações (BRISA, 1993, p. 139). Uma classe de objetos hierarquizados contém objetos que possuem propriedades similares.

Dentro do conceito de classes são definidas a superclasse e subclasses, sendo que estas herdam, de forma irrestrita, todas as propriedades da superclasse conhecida também como classe pai, conforme exemplifica a figura 3. No entanto, as peculiaridades exclusivas das subclasses são inseridas como propriedades adicionais.

Todos os objetos gerenciados pertencentes a uma classe devem possuir o mesmo comportamento, o qual pode ser definido pelos atributos. O comportamento dos objetos reflete a semântica dos atributos, as operações e notificações, como também a resposta às operações de gerenciamento, as circunstâncias em que as notificações devem ser emitidas, as dependências entre valores dos atributos particulares e os efeitos dos relacionamentos entre objetos gerenciados (BRISA, 1993, p. 141).

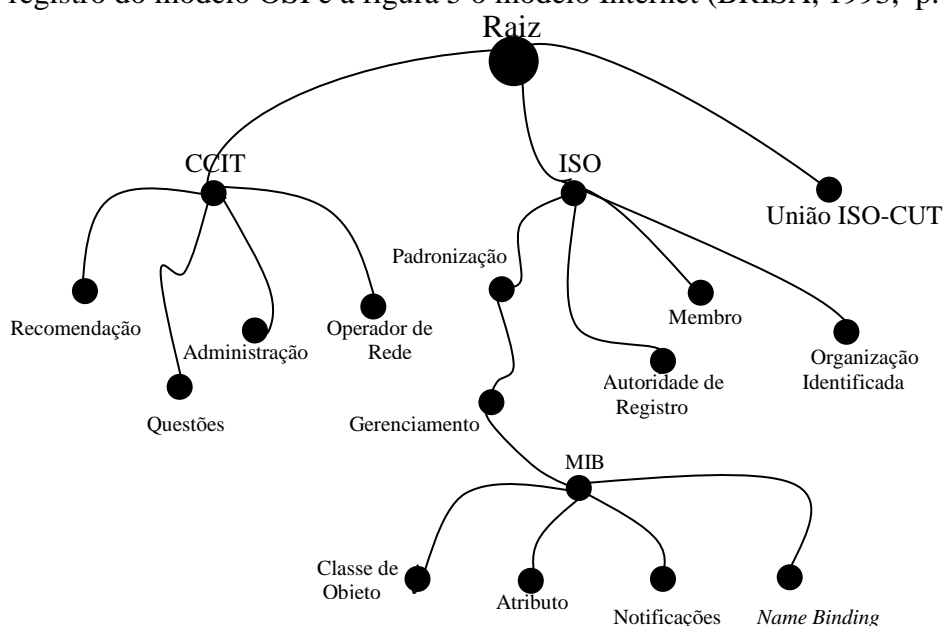


**Figura 3:** Hierarquia de classes de objetos gerenciados (BRISA, 1993, p.141).

- **Hierarquia de Nomeação:** esta hierarquia descreve os relacionamentos entre as instâncias de objetos com seus respectivos nomes. Esses relacionamentos aplicados aos objetos são do tipo “estar contido em”. Assim o objeto que contém outros objetos, podendo ser da mesma classe ou de outras, é definido como superior e o que está contido como subordinado. Contudo, um objeto gerenciado pode estar contido somente dentro de um objeto gerenciado superior.

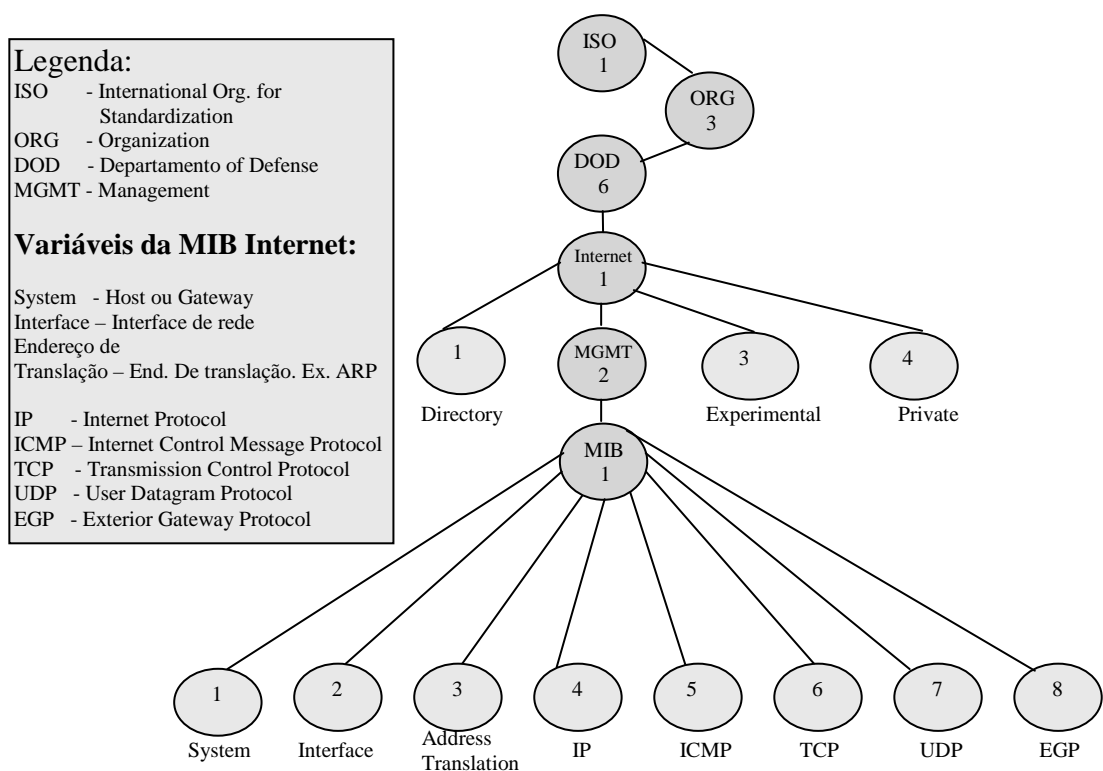
Este relacionamento descrito pela hierarquia de nomeação, também conhecido como *containment*, pode ser usado para modelar hierarquias de partes do mundo real ou hierarquias organizacionais, como: diretórios, arquivos, registros e campos. Deste modo para cada classe de objeto são definidas uma ou mais regras para identificação da classe superior e do atributo característico, cujas regras são nomeadas de “*name bindings*”. Um *name binding* é definido para uma classe de objetos, sendo que o mesmo fica disponível para ser utilizado em todas as classes derivadas da original (BRISA, 1993, p. 143).

- **Hierarquia de Registro:** esta hierarquia é utilizada para identificar de maneira universal os objetos (tanto no modelo OSI como no modelo Internet) e é especificada conforme as regras estabelecida pela notação ASN.1 (*Abstract Syntax Notation.One*) para árvore de registros usada na atribuição de identificadores a objetos, sendo que cada nó desta árvore está associada a uma autoridade de registro que determina como são atribuídos os seus números. A figura 4 exemplifica o modelo de hierarquia de registro do modelo OSI e a figura 5 o modelo Internet (BRISA, 1993, p. 144).



**Figura 4:** Ex. de Hierarquia de Registro–Modelo OSI (BRISA, 1993, p. 145).





**Figura 5:** Ex. de Hierarquia de Registro – Modelo Internet (BRISA, 1993, p.164)

Como as MIBs, tanto do modelo OSI como do modelo Internet, são modeladas através do conceito de orientação a objetos, os recursos que serão gerenciados são denominados de “objetos gerenciados”.

### 2.1.1.3. Objeto Gerenciado

Objeto gerenciado é uma representação de um recurso de rede, seja de ordem física ou lógica. Como o gerenciamento de redes trabalha com uma base de informações construída com base nos conceitos de orientação a objetos, estes constituem a base do gerenciamento de redes. Nesta proporção, caso haja algum elemento de rede não gerenciado, o mesmo não é contemplado no sistema de gerenciamento. A definição de objeto gerenciado apresenta os seguintes aspectos (BRISA, 1993, p. 18):

- a localização do objeto dentro da estrutura de rede que está sendo gerenciada;
- os atributos do objeto, os quais representam a natureza do objeto;
- as operações em que são submetidos;
- as notificações que podem emitir ao gerente; e

- as relações com outros objetos.

Convém reforçar que para definição de objeto deve-se ater aos padrões definidos pela linguagem ASN.1, a qual descreve os princípios necessários para especificação de objetos – GDMO (*Guidelines for the Definition of Managed Objects*).

As informações estáticas e dinâmicas identificam a interação entre os sistemas de gerenciamento e os objetos gerenciados. As primeiras não causam alteração no conteúdo dos objetos, enquanto que as informações dinâmicas têm a função de manter a coerência das informações de gerenciamento entre os recursos reais e seus respectivos objetos gerenciados. Recursos reais pode ser software ou hardware, ou ambos, e podem gerar eventos que quando transmitidos podem causar modificação nos objetos gerenciados, por exemplo, atualização dos atributos dos objetos. A coerência é mantida pela atualização do objeto através da ação enviada antes dessa ação ser aplicada sobre o recurso real (RAIMIR, 1998, p.19).

## **2.2. Modelos de gerenciamento de redes**

Como mencionado anteriormente, existem dois modelos a serem seguidos para implantação de gerenciamento de redes: o modelo de referência OSI e o modelo de gerenciamento TCP/IP. Como também já visto, esses dois modelos são similares quanto aos tipos de componentes adotados na arquitetura de gerenciamento, conhecidos por gerente, agente, objetos gerenciados e uma base de informações de gerenciamento – MIB. Cada modelo também possui um protocolo de aplicação exclusivo, o qual é responsável pelo transporte de operações e informações de gerenciamento entre os agentes e o gerente. Apesar da similaridade quanto aos tipos de componentes, existem diferenças substanciais quanto à forma e abrangência de comunicação entre os mesmos, inclusive quanto aos protocolos, visto que o protocolo CMIP é bem mais completo e complexo do que o protocolo SNMP. Popularmente esses dois modelos são conhecidos como “Modelo de Gerenciamento OSI” e “Modelo de Gerenciamento Internet”.

### **2.2.1. Diferenças entre os Modelos OSI e Internet**

O modelo de Gerenciamento OSI usa como base de transporte o protocolo CMIP (*Common Management Information Service Over TCP*), enquanto o modelo de Gerenciamento Internet utiliza o protocolo SNMP (*Simple Network Management*

*Protocol*). O protocolo SNMP não é orientado a conexão, o qual utiliza dos serviços prestados pelo UDP (*User Datagram Protocol*). Enquanto o protocolo CMIP é orientado a conexão e é executado sobre a pilha de protocolos OSI de gerenciamento, o que o torna mais confiável do que o modelo Internet (EMBRATEL, 1994, p. 548).

O protocolo CMIP apresenta uma melhor qualidade das informações, maior nível de confiabilidade e de segurança, porém trouxe conseqüências que impossibilitaram a sua adoção em maior escala, tais como:

- requer mais memória e processamento devido número de recursos de sistemas;
- o grande número de variáveis dificulta a compreensão sobre todas elas por parte do programador, o que a torna mais complicada a programação.

Já o protocolo SNMP foi o primeiro protocolo de gerenciamento público, como o próprio nome sugere é simples e de fácil implementação e possibilita o gerenciamento de ambientes diversos. Apesar das deficiências das primeiras versões, a simplicidade fez com que fosse adotado em um maior número de equipamentos e sistemas de gerenciamento de redes, tornando na prática um padrão de uso abrangente em gerência de redes.

No Gerenciamento Internet, o elemento agente possui a função básica de responder às operações emitidas pelo gerente, o envio de notificação ocorre somente em algumas situações de erros. Enquanto que no Gerenciamento OSI o elemento agente tanto responde às solicitações do gerente como emite notificações diversas de gerenciamento, cujas operações tendem a ser bem mais complexas do que as executadas pelo modelo Internet.

Apesar das duas arquiteturas adotarem os conceitos de orientação a objetos para descrever e especificar as informações das MIBs, no modelo Internet apenas são definidos os tipos de objetos a serem armazenados, enquanto o modelo OSI vai mais além ao especificar algumas classes de objetos a serem utilizadas nos sistemas de gerenciamento. O modelo OSI especifica um conjunto mais completo de operações de gerenciamento, além de ser possível a sincronização das operações realizadas sobre vários objetos (EMBRATEL, 1994, p. 547).

### **2.2.2. Modelo Gerenciamento Internet**

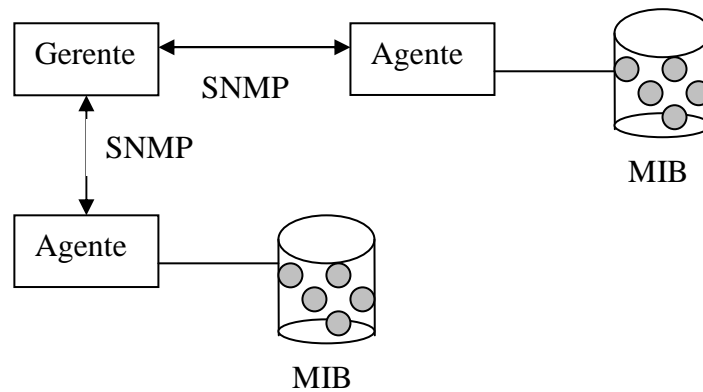
O ambiente de Gerenciamento Internet tem sido mais utilizado do que o modelo OSI e de certa forma dominado o mercado de sistema de gerenciamento de redes. O motivo de sua expansão tem sido sua simplicidade na implementação e por consumir poucos recursos de

rede e de processamento, o que tem facilitado a sua inclusão desde os mais simples equipamentos até nos mais sofisticados.

O gerenciamento Internet baseia-se no protocolo SNMP (*Simple Network Management Protocol*), que teve origem no protocolo para monitoração de *gateways* IP, o *Simple Gateway Management Protocol* – SGMP. Nessa evolução, a partir do original SGMP para o SNMP, foram inclusas as definições de uma Estrutura de Informação e Gerenciamento – SMI (*Structure of Management Information*) e de uma Base de Informação de Gerenciamento – MIB (*Management Information Base*) (EMBRATEL, 1994, p. 532).

Apesar da abrangência na utilização do Gerenciamento Internet ser atribuída à sua leveza e facilidade de implementação, estas características tornaram-se insuficiente à medida que as redes tomaram maiores dimensões e complexidades, principalmente quanto à carência de funcionalidades, eficiência e segurança. A segunda versão do protocolo SNMP, denominado SNMPv2, surgida em 1993, além de suportar as operações *getbulkrequest* e *informrequest* para transferência de grandes blocos de informação e possibilitando o gerenciamento distribuído, teve o objetivo de amenizar as deficiências da primeira versão do protocolo, no entanto, o quesito segurança somente foi implementado na versão 3 do protocolo SNMP, SNMPv3 (RFC 2571). Esta última versão não aborda uma especificação completa de um protocolo de gerenciamento de redes, mas apenas traz uma documentação com definições das características de segurança, até então carente, e um *framework* para ser utilizado com as funcionalidades já existentes nas versões anteriores (FREITAS, 2001, p. 34). Outro ponto é que foram definidas as versões 1 e 2 do RMON (*Remote Network Monitoring*) com o objetivo de diminuir o tráfego no gerenciamento e monitoração no cenário onde o sistema de gerenciamento e as redes gerenciadas são interconectadas através de redes de longa distância.

O modelo de Gerenciamento Internet apresenta um esquema centralizado, onde o gerente é configurado em uma estação de trabalho e os agentes ou *proxy agents* consistem nos outros elementos da rede. O *proxy agent* faz o papel de um procurador para os equipamentos que não implementam o protocolo SNMP. Os componentes de gerenciamento do modelo Internet são os mesmos já referenciados: Gerente e agentes, objetos gerenciados associados a uma MIB e o protocolo de gerenciamento, conforme figura 6 (EMBRATEL, 1994, p. 534).



**Figura 6:** Modelo de Gerenciamento Internet (EMBRATEL, 1994, p. 535).

No modelo Internet não é definido o conceito de classes e atributos de objetos a serem empregados pelos sistemas de gerenciamentos, conforme o modelo OSI, apenas são definidos tipos de objetos a serem armazenados na MIB (EMBRATEL, 1994, p.536), cuja definição contém cinco campos:

- nome textual com o respectivo identificador;
- definição da semântica associada ao tipo de objeto;
- a sintaxe ASN.1;
- o tipo de acesso (*Read-only*, *read-write*, *write-only* ou não acessível); e
- o *status* (obrigatório, opcional ou obsoleto).

Os tipos de objetos definidos são:

- simples: utilizando quatro tipos primitivos da ASN.1: *Integer*, *Octet String*, *Null* e *Object Identifier*;
- contexto de Aplicação: utilizando tipos especiais definidos pela SMI: *IpAddress*, *NetworkAddress*, *Counter*, *Gauge*, *TimeTicks* e *Opaque*; e
- construídos: utilizando para sua definição tipos construtores da ASN.1: *<list>* e *<table>*.

A funcionalidade do sistema de Gerenciamento Internet é alcançada através das seguintes operações (EMBRATEL, 1994, p.543-544):

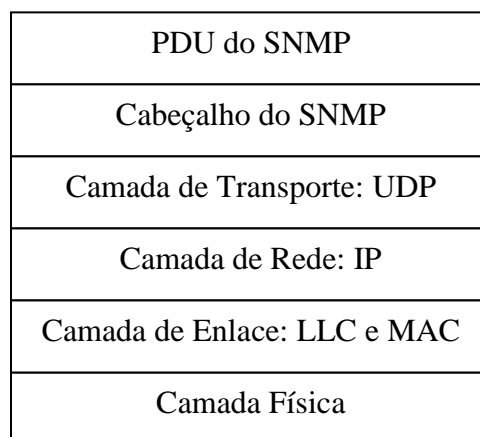
- *get-Request*: usada para recuperar uma informação de gerenciamento;
- *get-Next-Request*: utilizada quando não está definido ou não tem conhecimento das variáveis a serem recuperadas;
- *get-Response*: corresponde a uma operação de *Get-Request*, *Get-Next-Request* ou *Set-Request*;

- *set-Request*: utilizada para atribuição de valores às variáveis que contêm informações de gerenciamento;
- *trap*: utilizada para relatar ocorrências de eventos extraordinários;
- *get-Bulk-Request*: corrige uma deficiência do SNMPv1 na recuperação de grandes blocos de informação de gerenciamento;
- *inform-Request*: possibilita um gerente enviar informações de gerenciamento para outro gerente.

As mensagens do SNMP são constituídas de um cabeçalho de autenticação seguida de uma das seguintes unidades de dados de protocolo – PDU (*Protocol Data Unit*): *get-request*, *get-next-request*, *set-request*, *get-response* e *trap*. O mecanismo de *request-response* é implementado utilizando as quatro primeiras PDUs. O monitoramento da rede é feito utilizando as PDUs *get-request* e *get-next-request*, enquanto que para o controle da rede utiliza-se a PDU *set-request*, a qual permite modificar, criar e remover novas instâncias de informações de gerenciamento. A PDU *get-response* é transmitida pelo agente em resposta às PDUs *get-request*, *get-next-request*, *set-request* (EMBRATEL, 1994, p.542).

Como o protocolo SNMP não é orientado à conexão, em caso de redes locais o mesmo utiliza a pilha de protocolos abaixo, a qual está também exemplificada na figura 7:

- Camada de enlace: Protocolos LLC (*Logical Link Control*) e MAC (*Medium access Control*);
- Camada de Rede: Protocolos IP (*Internet Protocol*) e ICMP (*Internet Control Message Protocol*);
- Camada de transporte: Protocolo UDP



**Figura 7:** Pilha de protocolos utilizada pelo SNMP (EMBRATEL, 1994, p.545).

### 2.2.3. Modelo OSI de Gerenciamento de Rede

Em 1989, a ISO (*International Organization for Standardization*) apresentou a Arquitetura de Gerenciamento OSI, com o objetivo de cobrir a necessidade de uma arquitetura que fosse capaz de gerenciar os mais diversos elementos gerenciáveis existentes em uma rede, mantendo as características de integração, simplicidade, segurança e flexibilidade. A arquitetura de Gerenciamento OSI define os seguintes itens (EMBRATEL, 1994, p. 508-509):

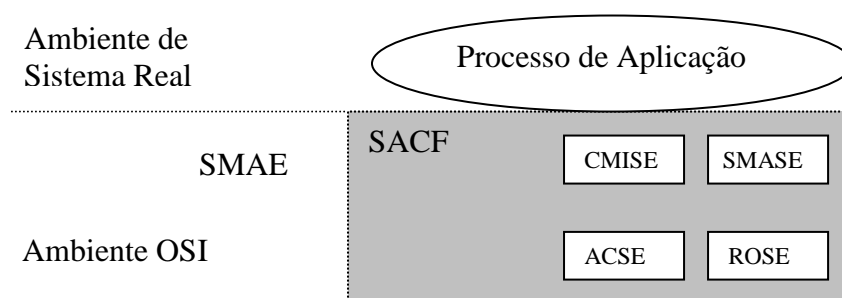
- estrutura de gerenciamento;
- os componentes de gerenciamento;
- estrutura da informação de gerenciamento;
- os serviços e o protocolo para a troca de informações de gerenciamento.

A estrutura do modelo de Gerenciamento OSI se subdivide da seguinte forma, (EMBRATEL, 1994, p. 509):

- gerenciamento de Sistemas: exige funções de apoio em todas as sete camadas da arquitetura de redes OSI.
- gerenciamento de Camada: utiliza protocolos de gerenciamento de propósito especial e restringe-se ao gerenciamento de recursos relacionados com as atividades de comunicação de uma única camada.
- operação de Camada: é uma forma de gerenciamento restrita a uma única instância de comunicação em uma camada.

Os componentes de gerenciamento do modelo OSI são os já descritos anteriormente, os quais incluem: os conceitos de gerente e agente, objetos gerenciados (entidades lógicas), a base de informação de gerenciamento – MIB e o protocolo de transporte das informações de gerenciamento. Logo, para desenvolver aplicações de gerenciamento é necessário utilizar os processos distribuídos formado pelos gerentes, responsáveis pela gerência dos objetos da rede, e pelos agentes, responsáveis por realizar as operações e pelo envio de notificações aos gerentes. A MIB é constituída pelo conjunto de objetos gerenciados e respectivos atributos, operações que podem ser enviadas pelo gerente para ser executada pelos agentes e as notificações que os agentes podem enviar ao sistema gerente.

Ao se tratar de gerenciamento de sistemas, duas ou mais entidades de aplicação podem associar-se para prover uma instância de aplicação de gerenciamento. A entidade de aplicação de gerenciamento de sistemas é denominada como SMAE (*System Management Application Entity*), a qual é composta pelo: SMASE (*System Management Application Service Element*), ACSE (*Association Control Service Element*) e por outros ASE (*Application Service Elements*), conforme mostra a figura 8 (EMBRATEL, 1994, p.510).



**Figura 8:** Entidade de aplicação de Gerenciamento de Sistemas (EMBRATEL, 1994, p.510).

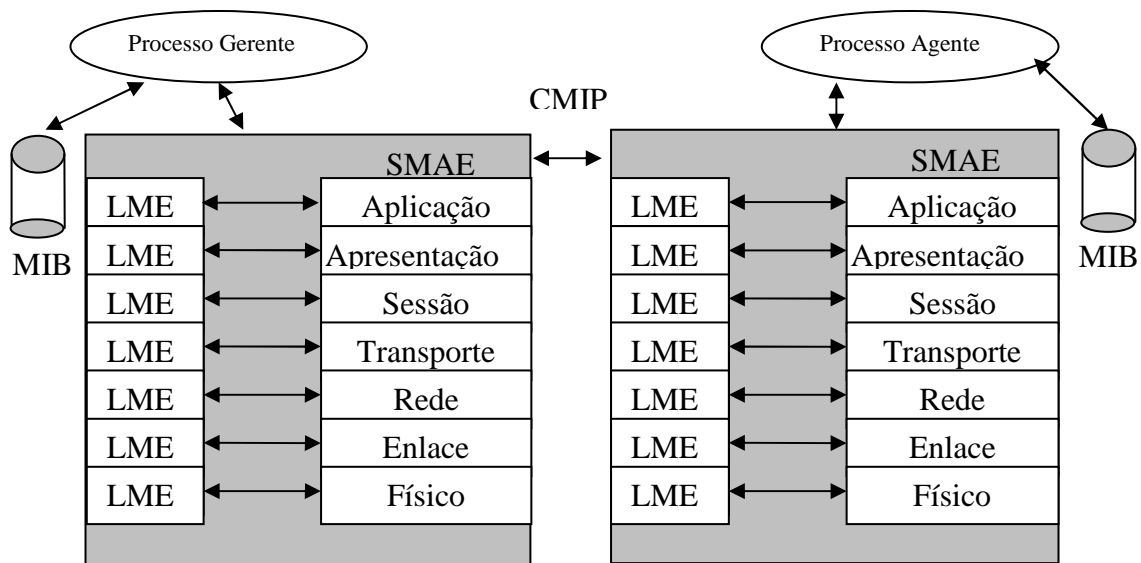
O serviço geral de Gerenciamento OSI é denominado de CMIS (*Common Management Information Service*), sendo que a aplicação que utiliza este serviço é denominada de MIS-User (*Management Information Service - User*). As interações entre os MIS-Users são realizadas através de trocas de informações de gerenciamento.

O protocolo utilizado pelo modelo OSI para transporte de informações de gerenciamento é o CMIP (*Common Management Information Protocol*), o qual implementa as primitivas oferecidas pelo CMIS. Para estabelecimento da comunicação este protocolo utiliza os elementos de serviço ACSE e ROSE (*Remote Operations Service Element*) (EMBRATEL, 1994, p. 511).

Através da MIB, o modelo de Gerenciamento OSI proporciona uma interface com cada uma das sete camadas de arquitetura de rede OSI, oferecendo as operações necessárias para executar o gerenciamento da rede em todas as camadas. Uma LME (*Layer Management Entity*) é uma entidade de gerenciamento de camada que concentra a funcionalidade da camada sob sua responsabilidade. Sendo que a integração e a função de interface com o gerente são realizadas pela SMAE, conforme demonstra a figura 9, cuja



figura apresenta ainda a comunicação entre duas entidades de aplicação de gerenciamento de sistemas.



**Figura 9:** Comunicação entre entidades de aplicação de ger. de sistemas (EMBRATEL, 1994, p.511).

#### 2.2.4. Serviços do Gerenciamento OSI

A camada de aplicação do modelo de Gerenciamento OSI fornece os serviços e o protocolo utilizado para implementação da aplicação de gerenciamento, dos quais incluem: gerenciamento de desempenho, do nível de falhas, de segurança, de configuração e de contabilidade. Na aplicação de gerenciamento de sistemas estão envolvidos os seguintes elementos de serviço de aplicação (BRISA, 1993, p. 169):

- o SMASE (*System Management Application Service Element*): este serviço define a sintaxe abstrata e semântica das informações transferidas para o gerenciamento OSI. Este elemento está relacionado com as áreas funcionais de gerenciamento.
- o CMISE (*Common Management Information Service Element*) implementa os serviços definidos pelo CMIS (Serviço geral de gerenciamento OSI) executando o protocolo CMIP. CMISE é o elemento de serviço que tem a função de prover meio para a troca de informações de gerenciamento comum. Cujas trocas são realizadas através das CMIPDU (*Common Management Information Protocol Data Units*), o qual constitui as unidades de dados do Protocolo CMIP. O elemento CMISE é tido como comum por ser utilizado nas cinco áreas funcionais de gerenciamento.

O serviço proporcionado pelo CMISE às aplicações de gerenciamento para fins de troca de informações é definido conforme Tabela 1 (BRISA, 1993, p. 161):

**Tabela 1:** Primitivas de definição que define o serviço de troca de informações.

Primitivas	Descrição
M-CREATE-request	Solicita a criação de uma instância de um objeto gerenciado
M-CREATE-response/confirmação	Confirma a criação da instância do objeto gerenciado
M-DELETE-request/indications	Solicita remoção de uma instância do objeto
M-DELETE-response/confirmation	Confirma a remoção da instância
M-GET-request/indication	Solicita a leitura de valores de atributos
M-GET-response/confirmação	Retorna o valor do atributo solicitado
M-CANCEL-GET-request/indication	Solicita cancelamento do M-GET
M-CANCEL-GET-response/confirmation	Confirma o cancelamento de uma operação M-GET
M-SET-request/indication	Solicita a atribuição de valor a um atributo
M-SET-response/confirmation	Confirma a atribuição de valor ao objeto
M-EVENT-REPORT	Emite relatório de eventos.
M-EVENT-REPORT:	Confirma a recepção de algum tipo de relatório de redes
M-ACTION-request/indication	Solicita execução de uma ação sobre objeto gerenciado
M-ACTION-response/confirmation	Confirma a execução da ação sobre o objeto.

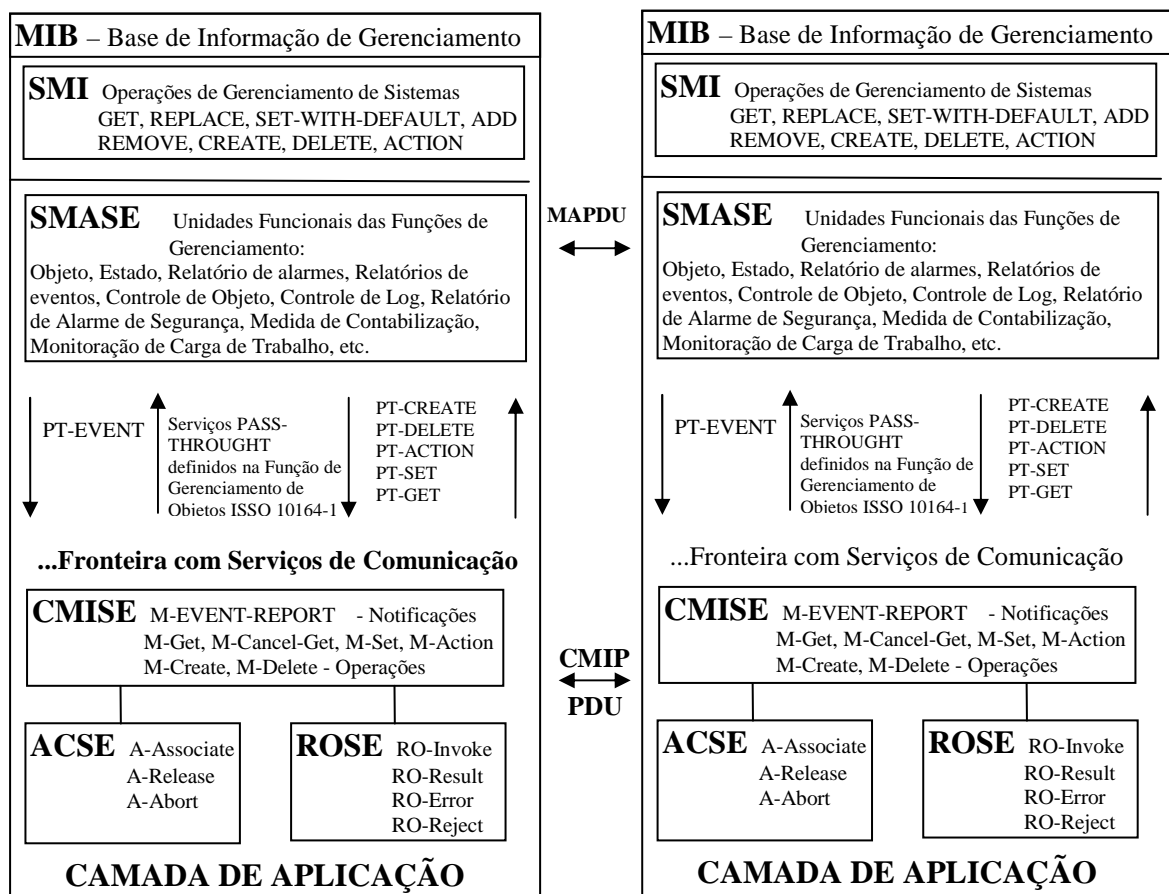
No CMISE existe a unidade funcional *Kernel* e um conjunto de unidades funcionais adicionais, como: de seleção de objetos múltiplos, de filtro, de respostas múltiplas, de serviço estendido e *Cancel Get*. Na unidade funcional *kernel* estão inclusos todos os serviços listados na tabela 1, exceto o *M-Cancel-Get* por estar incluído na unidade *Cancel Get*.

A figura 10 apresenta o detalhamento da camada de aplicação e o relacionamento entre os elementos de serviços de aplicação de gerenciamento de sistemas. Convém destacar que as operações e notificações de gerenciamento de sistemas estão dentro da

SMI (Estrutura de Informação de Gerenciamento). Essas operações de gerenciamento são definidas em dois tipos (BRISA, 1993, p. 172):

- operações orientadas a atributos, as quais são aplicadas sobre os atributos, como: *Get, Replace, Set-With-Default, Add member e Remove member*.
- operações que se aplicam sobre os objetos gerenciados, como: *Create, Delete e Action*. A semântica destas operações faz parte da definição das classes dos objetos gerenciados.

A figura 10 apresenta o relacionamento entre os conceitos apresentados anteriormente, bem como deixa mais nítido o que de fato existe em uma MIB, ou seja, o conjunto de informações de gerenciamento pertencente a um sistema que é transferido através dos protocolos de gerenciamento.



**Figura 10:** Detalhamento da Camada de Aplicação (BRISA, 1993, p. 173).

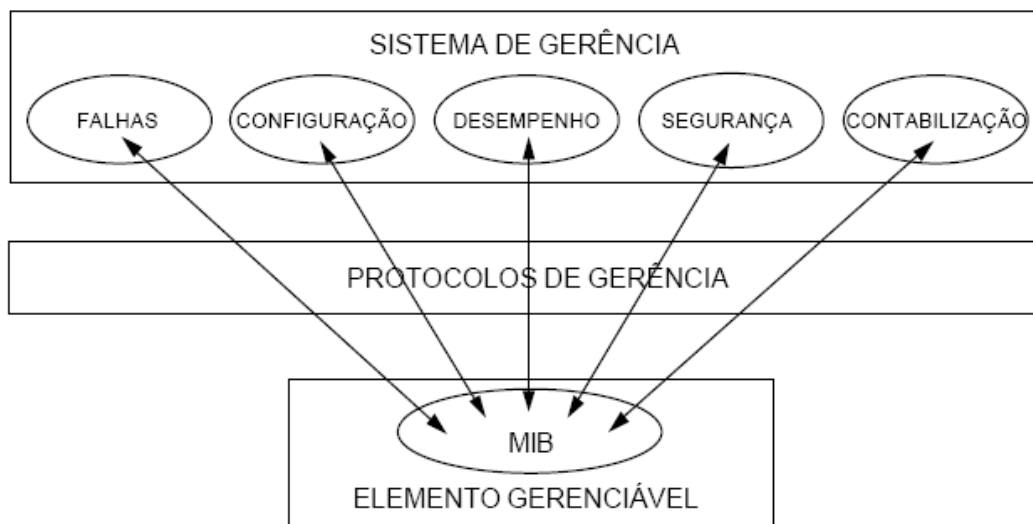
Explicitando as informações da figura 10 destaca-se que a SMI especifica a sintaxe da informação que é transferida para fins de gerenciamento; o SMASE traz as unidades funcionais das funções de gerenciamento identificadas pelas respectivas aplicações, as

quais especificam as informações de gerenciamento a serem trocadas entre os processos de gerenciamento através das MAPDUS; e a interação entre SMASE e o CMISE comprovando que o serviço de comunicação utilizado pelo SMASE é o CMISE, o qual define o serviço e procedimentos para transferência das CMIPDUS (BRISA, 1993, p. 172-175).

### 2.2.5. Áreas Funcionais de Gerenciamento de Redes

Áreas funcionais de gerência são as diferentes partes que podem ocorrer num problema de gerenciamento de redes, como: configuração da rede, falhas de componentes, o nível de desempenho apresentado no uso da rede, o nível de segurança relativo aos acessos e às informações e a contabilização da utilização da rede.

Essas atividades de gerenciamento de redes foram classificadas pela ISO (*International Organization for Standardization*) em cinco áreas funcionais de aplicação (gerenciamento de configuração, de falhas, de desempenho, de contabilização e de segurança), conforme demonstra a figura 11, as quais foram adotadas na maioria dos sistemas de gerenciamento de redes.



**Figura 11:** Áreas funcionais do gerenciamento de redes (RAIMIR, 1998, p. 24).

As informações relevantes para o funcionamento dessas áreas funcionais estão nas MIBs, as quais são constituídas pelos objetos gerenciados, seus atributos, as notificações passíveis de serem enviadas ao sistema gerente e as operações que podem ser executadas. O padrão de representação dessas informações de gerenciamento, as ferramentas de

coletas das mesmas e o controle dos objetos gerenciados constituem a estrutura de dados que foi definida com a linguagem ASN.1 (BRISA, 1993, p. 178).

### 2.2.5.1. Funções de Gerenciamento

Como as informações geradas em uma área funcional podem ser úteis como suportes para decisões em outras áreas, a ISO definiu algumas funções de gerenciamento para garantir os requisitos intrínsecos às áreas funcionais. Por exemplo, funções de gerenciamento de objeto, estado, atributos para representação de relacionamentos, relatórios de alarmes, relatórios de eventos, controle de *logs* e controle de acesso, os quais serão melhores descritos a seguir (EMBRATEL, 1994, p. 515):

- Função de Gerenciamento de Objeto – OMF (*Object Management Function*), que tem o objetivo de gerenciar a criação e a remoção de um objeto gerenciado, as alterações nos respectivos atributos e gerar relatórios dessas ações. Essa função descreve a forma de utilização do serviço no mapeamento de uma operação de gerenciamento sobre o serviço correspondente do CMISE, conforme demonstrado na Tabela 2 (EMBRATEL, 1994, p. 516);

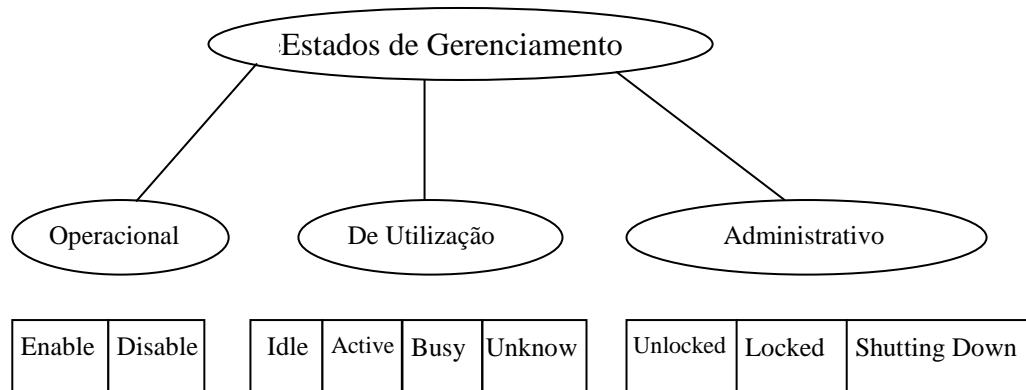
**Tabela 2:** Mapeamento das Operações de Gerenciamento

OPERAÇÃO DE GERENCIAMENTO	PASS-THROUGH	CMISE
CREATE	PT-CREATE	M-CREATE
DELETE	PT-DELETE	M-DELETE
ACTION	PT-ACTION	M-ACTION
REPLACE	PT-SET	M-SET
ADD MEMBER	PT-SET	M-SET
REMOVE MEMBER	PT-SET	M-SET
SET-WITH-DEFAULT	PT-SET	M-SET
GET	PT-GET	M-GET
NOTIFICATION	PT-EVENT	M-EVENT-REPORT

- Função de Gerenciamento de Estado – STMF (*State Management Function*), responsável por representar as condições reais e instantâneas sobre a disponibilidade e operacionalização de um recurso. Cada classe de objetos gerenciados possui o próprio conjunto de atributos de estado. Essa função deve ser padronizada visto ser comum a uma grande quantidade de recursos gerenciados. Três fatores, conforme

figura 12, afetam o estado de gerenciamento de um objeto em relação à disponibilidade dos recursos associados:

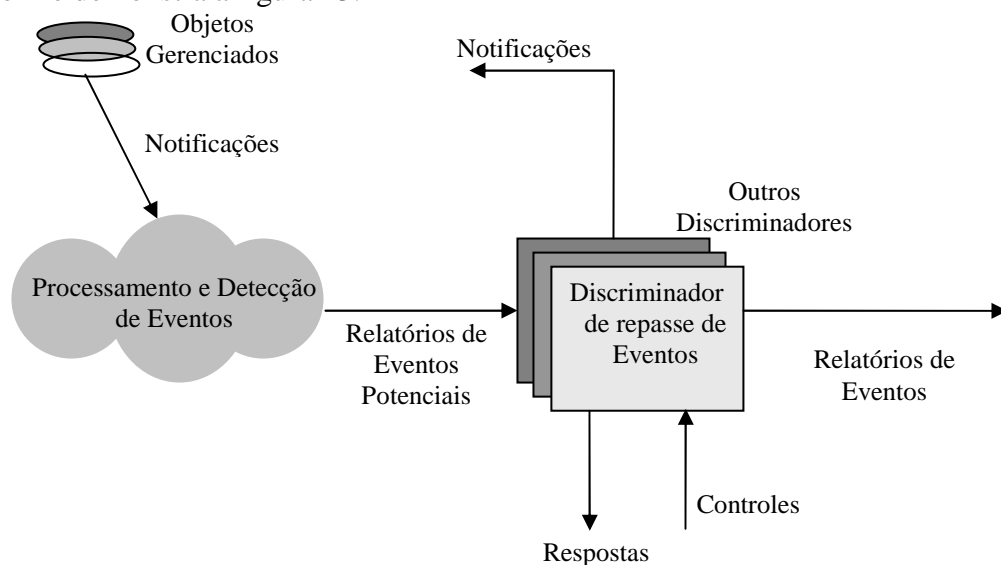
- operacionalização: ver se um dado recurso está ou não em operação;
- utilização: ver se um recurso está ou não em uso e se pode aceitar ou não outros usuários;
- administração: atribui permissão ou proibição quanto ao uso de um recurso.



**Figura 12:** Estados de gerenciamento (EMBRATEL, 1994, p.517).

- Atributos para Representação de Relacionamento – ARR (*Attributes for Representing Relationship*), que ocorre quando uma operação de um objeto gerenciado afeta outro objeto, e pode ser do tipo (EMBRATEL, 1994, p. 517-518):
  - direto: ocorre quando partes das informações de gerenciamento de um objeto identifica de forma expressa outro objeto gerenciado;
  - indireto: ocorre quando há dois ou mais relacionamentos diretos concatenados;
  - simétrico: ocorre quando os conjuntos de regras que governam as interações entre dois objetos são idênticos;
  - assimétrico: ocorre quando os papéis de dois objetos são complementares.
- Função de relatório de Alarme – ARF (*Alarm Report Function*), que permite monitorar a taxa de erros do sistema e a evolução do nível de severidade dos alarmes, com o objetivo de manter as condições operacionais dos serviços do sistema gerenciado. Nesta função são definidas cinco classes de alarme e seis níveis de severidade (EMBRATEL, 1994, p. 519):
  - Classes de Alarme;
    - alarme de comunicação: relacionado a procedimentos e processos usados na transferência de informações;

- alarme de qualidade de serviço: Indica a degradação da qualidade do serviço;
  - alarme de processamento: indica falha de processamento ou de software;
  - alarme de equipamento: indica falha de equipamentos;
  - alarme ambiental: indica as condições do ambiente em que se encontra em funcionamento o objeto gerenciado;
- Níveis de Severidades:
    - *cleared*: remoção de alarmes;
    - indeterminado: sem possibilidade de definir como foram afetadas as condições de funcionamento;
    - crítico: exigência de ação corretiva imediata;
    - maior: condições de funcionamento afetadas;
    - menor: necessário prevenir ocorrências de falhas;
    - alerta: suspeita de ocorrência de falhas.
- Função de Gerenciamento de Relatório de Evento - ERMF (*Event Report Management Function*), sendo que esta função tem como objetivo controlar o repasse de relatórios de eventos. Os relatórios de eventos potenciais são obtidos a partir das notificações geradas pelos objetos gerenciados e processadas através do componente de processamento e detecção de eventos. Estes relatórios são distribuídos aos discriminadores de repasse de eventos existentes no sistema, conforme demonstra a figura 13.



**Figura 13:** Gerenciamento de relatórios de falhas (EMBRATEL, 1994, p.521)

- Função de controle de *Log* – LCF (*Log Control Function*): o *log* nada mais é do que um repositório de registros, os quais contêm informações sobre eventos ocorridos ou por operações executadas pelos objetos gerenciados que precisam ser preservados. O controle da preservação desses registros é o objetivo da função de controle de *log* (EMBRATEL, 1994, p. 521);
- Função de registro para Auditoria de Segurança – SATF (*Security Audit Trail Function*): através desta função são gravados todos os eventos relativos à segurança em registros de *log* de auditoria de segurança, seguindo o modelo da função de registro de *log*. A partir da análise dos relatórios e alarmes de segurança poderá ter controle da manutenção da política de segurança especificada para o sistema, com a possibilidade de detectar desvios quanto à mesma, ou ainda pontos vulneráveis de segurança (EMBRATEL, 1994, p. 523).

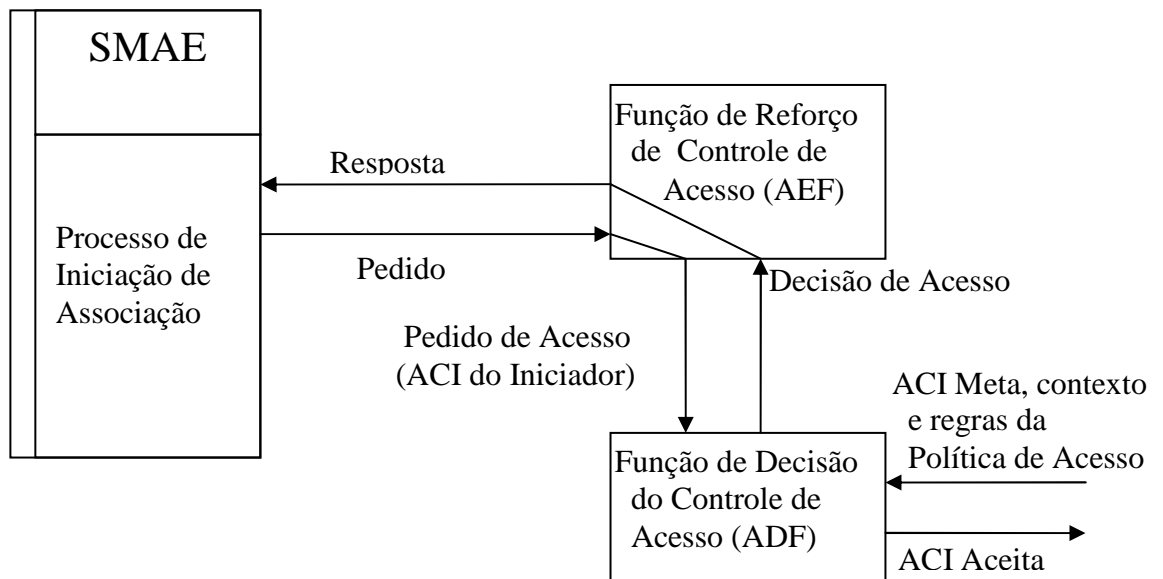
A coleção de registros de auditoria de segurança pode ser local ou remota. No caso local é necessário ter a possibilidade do sistema gerente recuperar os registros, enquanto que no caso remoto os registros são enviados ao sistema gerente à medida que os eventos de segurança ocorrem. No entanto, para que seja possível o envio dos eventos ao sistema gerente é preciso ter no sistema local um Discriminador de repasse de eventos, conforme demonstra a figura 13. Cujas função é filtrar os relatórios de eventos a serem enviados além de determinar o endereço dos sistemas destinatários dos eventos (BRISA, 1993, p. 98);

Segundo a BRISA (1993, p. 99) existem dois tipos de registros de auditoria de segurança:

- Relatório de serviço: indica se o registro está associado a um evento ligado à provisão, negação ou recuperação de um serviço;
  - Relatórios estatísticos: indica se o registro contém informações estatísticas.
- Função de controle de acesso (OAAC): a função de controle de acesso permite ao administrador prevenir-se contra acessos não autorizados aos recursos gerenciados. Através dessa função podem-se fazer avaliações dos pedidos de acessos a objetos gerenciados para que as decisões de permissão ou negação do acesso sejam feitas de acordo com a política de segurança, considerando fazer parte desta a política de controle de acesso.



Os objetos de controle de acesso, no gerenciamento OSI, são utilizados nas funções de controle de acesso da associação de aplicação e das operações de gerenciamento. Essa associação de gerenciamento é definida na estrutura da camada de aplicação, de sistemas abertos diferentes, com o objetivo de troca de informações necessárias à interoperação entre tais aplicações, conforme a figura 14 (BRISA, 1993, p. 102).



**Figura 14:** Modelo de controle de Acesso (BRISA, 1993, p. 103).

### 2.2.5.2. Gerenciamento de configuração

Para o bom gerenciamento de uma rede é extremamente importante manter a documentação atualizada com os registros técnicos relativos à instalação, inicialização, modificação e demais registros de parâmetros de configurações dos agentes e gerentes. Inclui-se ainda nesse gerenciamento de configuração o projeto da rede com localização e especificação de todos os pontos e elementos pertencentes à mesma.

### 2.2.5.3. Gerenciamento de Falhas

O gerenciamento de falhas tem como principal objetivo auxiliar o administrador da rede quanto ao número, tipos, horas das ocorrências e as localizações dos erros apresentados na rede, com o objetivo de que o retorno da rede ao estado operacional ocorra o mais breve possível. Logo, quando ocorre uma falha deve haver uma concentração das atividades para

localizar, isolar a falha do restante da rede, recuperar os componentes e serviços em falha, identificar a causa do problema e registrar a solução para fins de uso futuro (BRISA, 1993, p. 18).

No gerenciamento de falhas, o administrador da rede, com base nas informações estatísticas, poderá propor projetos que contemplem soluções de tolerância à falhas, com o uso de elementos redundantes, sobressalentes, rotas de comunicação alternativas e etc (FREITAS, 2001, p.9).

#### **2.2.5.4. Gerenciamento de Desempenho**

Quando ocorre uma queda no desempenho da rede, os usuários reclamam quase que incansavelmente à medida que vê as suas atividades serem prejudicadas. Portanto, os componentes de uma rede precisam ser monitorados de forma contínua e os dados estatísticos sobre o desempenho da rede devem ser registrados para subsidiar o planejamento e o controle de qualidade dos serviços disponíveis na rede. A partir destes dados, o administrador passa a possuir elementos que sustentarão ações de pró-atividades no sentido de antecipar-se ao surgimento de problemas.

Para gerenciar o desempenho da rede devem ser seguidos os passos: coletar dados do uso corrente dos dispositivos da rede, analisar os dados relevantes, estabelecerem limiares e simulação da rede. Quanto aos serviços, é importante ter registros do tempo de resposta, taxa de rejeição e disponibilidade (FREITAS, 2001, p.9).

#### **2.2.5.5. Gerenciamento de Contabilização**

A função de contabilização é medir a utilização dos recursos da rede para fins de quantificação dos custos, tarifação, cotas de utilização, etc. Este gerenciamento, além de prover um melhor acompanhamento do aumento de uso dos recursos da rede, poderá apresentar indícios de necessidade de novas aquisições de elementos para a mesma. A gerência de contabilização facilita o trabalho do administrador no sentido de possibilitar a identificação de usuários que abusam dos privilégios de acesso, uso ineficiente e até mesmo no planejamento de crescimento da rede com base no aumento da demanda (EMBRATEL, 1994, p.515).

### 2.2.5.6. Gerenciamento de Segurança

O controle de acesso à rede e às informações deve garantir que somente pessoas ou aplicações autorizadas tenham permissão aos recursos da rede. O objetivo é proteger o maior bem para as instituições atualmente: a informação. Dentre as funções do gerenciamento de segurança, destacam-se (BRISA, 1993, p. 87):

- a coleta, armazenamento e examine dos registros de auditoria e *logs* de segurança, bem como ativação e desativação de diretivas de auditoria;
- o registro das tentativas de ataques de forma efetiva;
- a definição das normas, políticas e procedimentos de segurança para a empresa, de modo que proteja a rede contra os acessos indevidos e sirva de subsídios para projetos de implementação de mecanismos de segurança, como: *firewall*, técnicas de criptografias, armazenamento seguros das informações.

### 2.3. Protocolo CMIP

Como mencionado anteriormente, a ISO (*International Organization for Standardization*) estabeleceu como solução para gerenciamento de redes o Protocolo CMIP (*Common Management Information Protocol*), o qual fornece um formato inteligível comum para a transferência de informação de gerenciamento entre entidades pares. Neste caso, um dos pares atua como gerente e outro como agente durante a comunicação de gerenciamento para troca de informações sobre os objetos gerenciados e armazenados nas respectivas MIBs (MIB gerente e MIB Agente) (BRISA, 1993, p.321).

O protocolo CMIP utiliza classes de operação que são relacionadas com os serviços do CMISE (Elemento de Serviço de Informação de Gerenciamento Comum), conforme Tabela 4, cujos serviços podem ser confirmados ou não-confirmados. Esses serviços são mapeados em operações aplicadas sobre objetos ou recursos gerenciados da rede. A tabela 3 apresenta as classes de operação e associações definidas em termos gerais (BRISA, 1993, p. 181).

**Tabela 3:** Classes de operação e associações definidas

Classes	Descrição
Classe de Operação 1	Síncrona relatando sucesso ou falha
Classe de Operação 2	Assíncrona relatando sucesso ou falha

Classe de Operação 3	Assíncrona relatando somente falha (erro)
Classe de Operação 4	Assíncrona relatando somente sucesso
Classe de Operação 5	Assíncrona resultado não relatado.
Classe de Associação 1	Somente a entidade de aplicação iniciadora da associação pode invocar operações
Classe de Associação 2	Somente a entidade de aplicação respondedora da associação pode invocar operações
Classe de Associação 3	Ambas as entidades de aplicação iniciadora e respondedora da associação podem invocar operações

Dentre as classes de operações e associações apresentadas, o Protocolo CMIP utiliza a classe de associação 3 e as classes de operação 1, 2 e 5.

**Tabela 4:** Relacionamento das classes de operação do CMIP com os serviços CMISE.

Serviço	Tipo	Classe de Operação
M-Event Report	Confirmado/não confirmado	2 ou 1/5
M-Get	Confirmado	2 ou 1
M-Cancel-Get	Confirmado	2 ou 1
M-Set	Confirmado/não confirmado	2 ou 1/5
M-Action	Confirmado/não confirmado	2 ou 1/5
M-Create	Confirmado	2 ou 1
M-Delete	Confirmado	2 ou 1

O protocolo CMIP, de forma semelhante ao protocolo SNMP, engloba vários tipos de PDUs (*Protocol Data Units*), as quais são apresentados na Tabela 5 (EMBRATEL, 1994, p. 512).

**Tabela 5:** PDUs do protocolo CMIP

PDUs	Descrição
M-Action	Execução de alguma ação sobre um objeto gerenciado
M-Create	Criação de uma instância de um objeto gerenciado
M-Delete	Remoção de uma instância de um objeto gerenciado
M-Get	Leitura de atributos dos objetos gerenciados
M-Set	Modificação de atributos de objetos gerenciados

MEvent-Report	Notificação de um evento associado a um objeto gerenciado
---------------	---

Para estabelecer a comunicação, o CMIP faz uso dos elementos de serviço de Controle de Associação – ACSE e do ROSE (*Remote Operations Service Element*). Por isto, convém destacar os serviços assumidos pelo CMIP em relação ao ACSE e ao ROSE e suas respectivas APDUS (*Application Protocol Data Units*), conforme demonstram as tabelas 6 e 7 (BRISA, 1993, p.182).

**Tabela 6:** Serviços do CMIP em relação ao ACSE

APDUs	Serviços	Primitivas de Serviço
AARQ	A-Associate	A-Associate-Request e indication
AARE		A-Associate-Response e confirmation
RLRQ	A-Release	A-Release-request e indication
RLRE	A-Release	A-Release-response e Confirmation
ABRT	A-Abort	A-Abort-request e indication

**Tabela 7:** Serviços do CMIP em relação ao ROSE

APDUs	Serviços	Primitivas de Serviço
ROIV	RO-Invoke	RO-Invoke-request e indication
RORS	RO-Result	RO-Result-request e indication
ROER	RO-Error	RO-Error-request e indication
RORJ	RO-Reject	RO-Reject-U-request e indication

Nas comunicações entre aplicação gerente e agente, onde a aplicação gerente envia operações a serem executadas pelo agente sobre os objetos gerenciados e os agentes enviam notificações de gerenciamento ao gerente, para transporte dessas operações e notificações, são utilizadas as PDUs do ROSE. As operações remotas são identificadas dentro do contexto de aplicação da seguinte forma (BRISA, 1993, p.185):

- *REMOTEoperation {joint-isso-ccitt remote operation(4) apdu(1)}*
- *ROIV apdu(1), RORS apdu(2), ROER apdu (3) e RORJ apdu (4)*

Cada PDU do ROSE possui seus próprios parâmetros que são mapeados com os parâmetros do CMIP, conforme Tabela 8 (BRISA, 1993, p.186).

**Tabela 8:** Mapeamento dos parâmetros do CMIP nos parâmetros das APDUs do ROSE

APDUs do ROSE	Parâmetros das APDUs	Parâmetros do CMIP
ROIV (RO-INVOKE)	Invoked-Id	Invoked-Id
	Linked-Id	Linked-Id
	Operation Value	Operation Value + Mode
	Argument	Base Object Class Base Object Instance Manage Object Class Manage Object Instance Attribute List Access Control Action Type Action Information Current Time Action Reply Errors Linked Identifier Scope Filter Synchronization
RORS (RO-RESULT)	Invoked – Id	Invoked-Id
	Operation Value	Operation Value
	Result	Managed Object Class Managed Object Instance Current Time Attribute List
ROER (RO-ERROR)	Invoked-Id	Invoked-Id
	Error Value	Error Value
	Error Parameter	AccessDenied-Error-Local Value(LV) 2 ClassInstanceConflict-ERROR-LV 19 ComplexityLimitation-ERROR-LV 20 DuplicateManagedObjectIntance- ERROR – LV 11 getListError ERROR – LV 7 invalidArgumentValue-ERROR –LV 15 invalidAtributeValue-ERROR-LV 6 invalididFilter-ERROR-LV 4 invalidScope-ERROR-LV 16 invalidObjectInstance-ERROR-LV 17 missingAttributeValue-ERROR-LV 18 noSuchAction-ERROR-LV 9 noSuchArgument--ERROR-LV 14 noSuchAttribute--ERROR-LV 5

		noSuchEventType-ERROR-LV 13 noSuchObjectClass-ERROR-LV 0 noSuchObjectInstance-ERROR-LV 1 noSuchReferenceObject-ERROR-LV 12 processingFailure-ERROR-LV 10 setListError-ERROR-LV 8 syncNotSupported-ERROR-LV 3
RORJ(RO-REJECT)	Invoked-Id	Invoked-Id
	Problem	General Problem Invoke Problem Return Result Problem Return Result Problem

Convém explicitar que os parâmetros do CMIP correspondentes ao parâmetro argumento (*Argument*) da APDU ROIV referem-se a todas as operações e notificações que podem ocorrer. Já os parâmetros CMIP mapeados com o *Error Parameter* (RO-ERROR) dizem respeito às notificações ou com quaisquer das operações que estão sendo realizadas em uma determinada associação (BRISA, 1993, p.187).

Dentro da abrangência do protocolo CMIP, ainda existem alguns recursos relevantes que facilitam o gerenciamento e merecem destaque, como os disponibilizados pelo Serviço de Informação de Gerenciamento Comum (CMIS) (EMBRATEL, 1994, p.531):

- *scoping*: permite realizar uma única operação sobre um grupo de instância de objetos selecionado;
- filtro: permite definir um conjunto de testes a ser aplicado a um grupo de instâncias de objetos previamente selecionado através do *Scoping*, proporcionando formar um grupo menor a partir deste, sobre o qual as operações de gerenciamento devem ser aplicadas;
- sincronização: sincroniza várias operações de gerenciamento realizadas sobre instâncias de objetos selecionados por meio dos recursos de *Scoping* e de Filtro;

Analisando a quantidade de recursos providos pelo protocolo CMIP, é notável a diferença em relação ao Protocolo SNMP, do Gerenciamento Internet. Este é o protocolo concebido para o padrão de sistemas de gerenciamento distribuídos de grandes e complexas redes corporativas e redes públicas de telecomunicações.

Como o trabalho proposto aborda a questão da segurança e que para promover o gerenciamento de redes é necessário prover segurança aos dados armazenados nos

equipamentos da rede e às informações que trafegam na mesma, convém abordar conceitos relativos à segurança em redes de computadores.

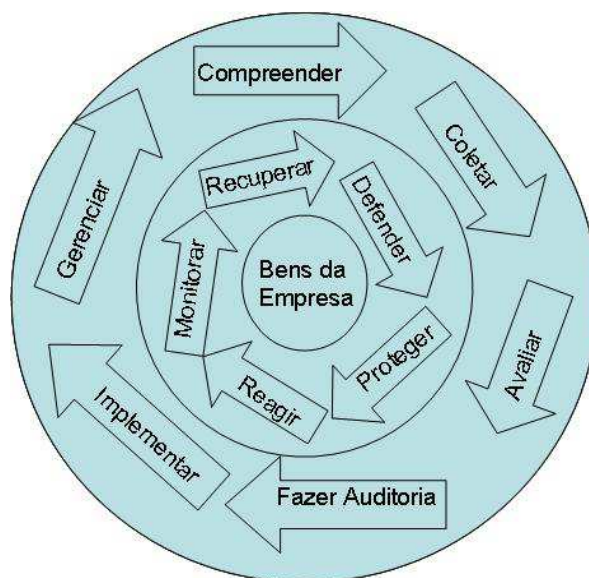
## **2.4. Segurança de Redes**

Segundo Wadlow (2000, p.4), “segurança é a direção que se pode viajar, mas nunca chegar de fato ao destino”, ou seja, é o caminho que tem começo, mas não tem fim. “Segurança é um processo que se pode aplicar seguidamente à rede e à empresa que a mantém e, assim, melhorar a segurança dos sistemas. Se não iniciar ou interromper a aplicação do processo, a segurança será cada vez pior, à medida que surgirem novas ameaças e técnicas” (Wadlow, 2000, p.4). O processo de segurança que deve ser aplicado continuamente, se resume em analisar o problema, sintetizar uma solução para o problema com base na análise realizada e avaliar a solução empregada procurando entender e aprender em quais aspectos não correspondeu às expectativas. Nesse contexto o gerente de segurança deve ter habilidades e disponibilidade para aprender o máximo que puder sobre os tipos de ameaças encontradas, saber planejar soluções com base no que aprendeu antes de aplicá-las e em seguida implementá-las sendo fiel ao que foi projetado.

Esse trabalho deve ser constante porque os sistemas e informações a serem protegidos mudam ou sofrem alterações frequentemente, e, ao mesmo tempo surgem novas vulnerabilidades, novos riscos e tipos de ameaças à segurança. As novas ameaças são originadas da dedicação dos criminosos e terroristas que se dedicam em adquirir conhecimentos sobre as novas tecnologias e como violarem os sistemas (HORTON, 2003, p.4). Portanto, métodos de segurança devem sempre ser inovados e readequados para manter um bom nível de segurança. É um árduo trabalho, porém, extremamente necessário para uma empresa ou instituição.

Para promover uma efetiva proteção da rede de dados de uma instituição, devem ser seguidas algumas diretrizes que são fundamentais e que fazem parte do ciclo de vida da segurança da informação, conforme apresentado na figura 15. Estas diretrizes abrangem a proteção dos bens da rede (servidores, computadores e etc) com base no gerenciamento dos riscos, conhecimento sobre as ameaças e vulnerabilidades da rede (HORTON, 2003, p.5).





**Figura 15:** Processo de Segurança baseado em Riscos e Bens (HORTON, 2003, p.5).

A seguir será explanada cada ação a ser executada dentro do ciclo vida para que seja mantida a segurança das informações.

- compreensão e conhecimento dos bens da instituição, dos produtos e serviços, da estrutura organizacional e seus objetivos;
- coleta de informações sobre a infra-estrutura computacional e da rede, o que existe de proteção e o que precisa ser protegido;
- avaliação das estratégias para a rede e da arquitetura computacional, definindo a sua inserção no papel e objetivos da instituição, considerando os mecanismos de proteção existentes que estão sendo aplicados e o que está proposto;
- auditoria dos recursos de rede para obter conhecimento da segurança atual e os pontos críticos que devem ser protegidos para fechar as vulnerabilidades existentes;
- implementação de ações corretivas;
- gerenciamento dos controles e mecanismos de proteção aplicados;
- implementar mecanismos de proteção para os bens da instituição;
- fazer monitoração com base no processo de auditoria e registrar alertas e dados do sistema, avaliando essas informações nos eventos de segurança. A maior parte de tempo deve ser dedicada nesta fase de monitoração;
- reação quanto ao preparo de recursos para iniciar a defesa e a recuperação em tempo hábil, quando houver incidentes; e
- reavaliação das necessidades de segurança.

A execução dessas funções de forma cíclica propicia um ambiente com bom nível de segurança para uma instituição, protegendo o que existe de mais valioso atualmente, a informação. Vale ressaltar que prejuízos incalculáveis são imputados a empresas e instituições que não dão valor e, conseqüentemente, não investem o necessário no processo de segurança da informação. Os prejuízos são calculados com base nos valores tangíveis e intangíveis, por exemplo, a perda de confiança dos clientes.

A segurança de uma rede, considerando ser o meio de acesso às valiosas informações de uma empresa ou instituição, deve se ater a três objetivos principais: confidencialidade, integridade e disponibilidade. Dentro do objetivo de confidencialidade os dados devem estar disponíveis somente para quem precisa ter acesso a eles, mantendo o controle de acesso e evitando acessos indevidos. O fator integridade é bastante crítico e devem ter medidas severas de proteção para garantir que os dados não sejam alterados indevidamente. Contudo, não adianta garantir a confidencialidade e proteger os dados contra alterações indevidas, se não incluir garantias de que os dados estejam disponíveis quando necessários. A falta de disponibilidade poderá causar grandes prejuízos à empresa ou instituição, principalmente danos financeiros (HORTON, 2003, p.7). Portanto, a quebra de uma ou mais dessas três propriedades constitui ameaça à segurança da informação de qualquer instituição. Existem diversas formas e ferramentas que podem ser aplicadas no processo de segurança de uma rede, como (HORTON, 2003, p.12):

- *firewalls* de rede e de estação;
- IDS (Sistema de detecção de intrusão) de rede e de estação;
- servidor *proxy*;
- VPN de hardware/software;
- implementações de VLANs dos *Switchs*;
- softwares de gerenciamento e verificação de vulnerabilidades;
- softwares de alerta e monitoração de *logs*;
- softwares de Antivírus de servidor e estação;
- servidor central de autenticação;
- servidor de controle de diretivas e acesso;
- máquina virtual;
- criptografia de arquivos e de e-mails; e
- auditoria/avaliação de segurança da rede.

Como apresentado na relação acima, dentre os meios de prover segurança em uma rede, inclui o processo de auditorias. Com a aplicação desse processo em uma rede ocorre a gravação, em arquivos de *logs*, dos registros das ações dos usuários sobre os objetos que estão sendo auditados. Através da análise desses registros é possível traçar o comportamento de uso da rede, dos usuários e dos serviços, o que facilita a administração da rede, até mesmo, pelo simples fato de detecção de desvios de padrões conhecidos e aceitáveis para a rede em análise (BRISA, 1993, p.97).

Na rede Windows, o processo de auditoria é possível através de aplicação de diretivas de segurança de auditoria, as quais fazem parte da estrutura lógica e administrativa da rede construídas a partir da criação do serviço de diretório do Windows, denominado de *Active Directory*. Para melhor entendimento dos serviços providos pelo *Active Directory* faz-se necessário apresentar a infra-estrutura da rede Windows.

## **2.5. Infra-estrutura da rede Windows**

A rede do Windows normalmente é composta por um ou mais servidores e estações de trabalho. O servidor funciona com o sistema operacional Windows 2000 ou 2003 Server e possui a função de controlador do domínio da rede. Ao assumir essa função ocorre a construção da estrutura dos Serviços de Diretórios (*Active Directory*), o qual é a base para a implementação da infra-estrutura da rede.

O *Active Directory* funciona como uma base de dados da rede onde são cadastrados todos os objetos da mesma, como: usuários e grupos de usuários, computadores, impressoras, unidades organizacionais, esquemas e etc. Essa base contempla todas as informações necessárias para funcionamento da rede baseada no *Windows 2003 Server* (BATTISTI, 2006. Parte 2, p4).

Segundo Battisti (2006. Parte 1, p. 3), o *Active Directory* possui duas estruturas: uma física e outra lógica. A estrutura lógica é formada pelos elementos: domínios, árvores, florestas, relações de confiança, objetos do *Active Directory*, unidades organizacionais e esquemas, enquanto que a estrutura física é responsável por definir o local de armazenamento das informações do *Active Directory* e como será feita a sincronização das informações entre os controladores de domínio *máster* e membros de um domínio.

Na definição da estrutura lógica constituem-se os limites de segurança e da gerência administrativa da rede. No topo da hierarquia administrativa temos o conceito de

floresta, a qual pode ser formada por uma ou mais árvores de domínios. Dentro de uma floresta a relação de confiança entre as árvores existentes e os respectivos domínios se dá de forma automática. No entanto, se a rede possuir mais de uma floresta, a relação de confiança entre as mesmas não se dá de forma automática, estabelecendo nesse ponto o limite de segurança da rede.

Uma árvore pode ser formada por um ou mais domínios, sendo que o domínio de uma rede representa o limite de gerenciamento do conjunto de objetos do *Active Directory* que faz parte desse domínio. Em um domínio poderá existir um ou vários controladores do domínio, os quais compartilham e gerenciam a mesma base de dados do *Active Directory*. A idéia de construir uma rede com vários domínios ou subdomínios proporciona uma independência administrativa entre alguns setores de uma empresa ou instituição (DOMÍNIOS, 2004, *On-Line*).

Apesar da independência administrativa estabelecida pela presença de domínios e/ou subdomínios existentes em uma rede, o processo automático da relação de confiança por pertencerem a uma mesma árvore ou floresta propicia a formação do catálogo global da rede através dos vários conjuntos dos objetos dos serviços de diretórios (BATTISTI, 2006. Parte 4, p. 4).

O conceito de esquema do *Active Directory* é definido pela constituição de classes de objetos e respectivos atributos, contendo as informações essenciais para os objetos do serviço de diretório. A classe contém todos os atributos capazes de definirem as informações que deve conter no objeto, contudo, um atributo pode pertencer a várias classes, uma vez que são definidos de forma separada da classe. Para definir os atributos de uma classe, é realizada uma associação entre a classe e os respectivos atributos (DIRECTORY, 2005, *On-Line*).

Na constituição de um domínio para a rede Windows, o primeiro controlador, denominado de máster, deverá agregar também a função de servidor de DNS (*Domain Name System*), o qual possui a função de resolver nomes dos objetos do *Active Directory*, mais precisamente dos computadores e servidores, fazendo uma associação entre os nomes dos objetos e os respectivos números dos endereços IPs (*Internet Protocol*). Considerando que os equipamentos se comunicam através dos endereços, essa associação facilita o trabalho dos usuários por ser muito mais fácil trabalhar com nomes do que com números.

Dentro da estrutura do *Active Directory* fazem presente também as diretivas de segurança, que por sua vez incorpora as diretivas de auditorias que são as ferramentas utilizadas para aplicação de auditoria em uma Rede Windows.

## 2.6. Auditoria em Redes Windows

No Sistema Windows, os processos de auditorias de segurança são realizados através de diretivas de grupo de auditoria. Essas diretivas, através dos recursos dos serviços de diretório, podem ser aplicadas em todos os computadores e/ou usuários de um domínio, ou em apenas alguns ou ainda em modo local (AUDITORIA, 2004, *On-Line*).

Para possibilitar a ocorrência de auditoria de segurança é necessário que os *logs* de auditoria sejam gravados com os eventos relativos à segurança. Esta gravação pode ocorrer tanto localmente em cada máquina auditada, como também de forma remota, dependendo do tipo de auditoria definida. Portanto, os eventos de auditoria utilizados para detecção de mau uso dos recursos da rede, atentados contra a segurança das informações ou de problemas que prejudiquem a concretização da política de segurança, como alteração indevida das configurações de auditorias, precisam ser controlados pelos usuários do grupo de gerenciamento de segurança (BRISA, 1993, p.98).

As opções de auditoria, por padrão em qualquer sistema, possuem *status* de não configuradas ou não definidas, exatamente porque exige definição do que precisa ser auditado de acordo com a política de auditoria da empresa ou instituição, o que é bastante peculiar a cada interesse. Para realizar uma boa análise de auditoria de segurança, dois passos são necessários: primeiramente é preciso definir e aplicar a política de auditoria e, posteriormente, é necessário ter o conhecimento sobre a estrutura dos arquivos de *logs* gerados para auditoria.

As diretivas de auditoria da rede Windows, quando aplicadas, geram arquivos de *logs* dos eventos de várias áreas de segurança, como (AUDITORIA, 2004, *On-Line*):

- acesso a objetos;
- acesso ao serviço de diretório;
- alteração de diretiva;
- controle de processo;
- eventos de *logon*;
- eventos de sistema;

- gerenciamento de conta;
- uso de privilégios; e
- eventos de *logon* de conta.

É notório que ativar todas essas diretivas de auditoria de segurança tornam impraticável a gerência e a análise dos arquivos de *logs*, visto que cada diretiva destas gera eventos e arquivos de *logs* com estruturas diferentes, o que obrigaria ter o conhecimento sobre todos eles, além de uma boa ferramenta de administração e filtragem desses *logs* e grande espaço em disco. Todas essas diretivas de auditoria ativadas geram milhares de *logs* por dia para cada máquina, o que poderia prejudicar a performance do equipamento e consequentemente da rede. Portanto, a recomendação para se ter bons resultados oriundos da aplicação de auditoria em uma rede é a definição sensata da política de auditoria de segurança a ser adotada.

Uma boa política de segurança, segundo Wadlon(2000, p.12), precisa atender a alguns propósitos, adequando-os a uma boa política de auditoria, tais como:

- descrever o por quê, como e o que será auditado;
- definir prioridades sobre o que precisa ser auditado, conforme a necessidade e o impacto da auditoria;
- definir métodos de retenção dos registros de eventos;
- definir meios de gerenciamento e acompanhamento sistemático dos registros de eventos;
- impedir que o departamento de segurança tenha um desempenho fútil e sem objetivo perante a administração da instituição;
- permitir um acordo explícito e o convencimento das várias partes da instituição sobre o valor da auditoria.

O maior desafio para o gerenciamento de uma rede Windows através de uma política de auditoria se concentra na forma como coletar as informações dos *logs* que são gravados localmente em cada máquina. Isto porque se conectar, mesmo que de forma remota, em cada estação para fazer análise dos *logs* é uma tarefa impraticável, cujos objetivos dificilmente serão alcançados. Portanto, como solução desse impasse projeta-se a busca de métodos capazes de fazer coletas dessas informações de forma automática e armazená-la em uma base de dados. Contudo, antes de descrever os métodos e os recursos que serão utilizados, convêm reforçar a base teórica sobre as normas, tecnologias e recursos necessários para esta finalidade.

## **2.7. Windows Management Instrumentation (WMI)**

WMI é uma tecnologia da Microsoft que permite a manipulação de informações úteis para gerenciamento e controle de redes corporativas Windows. WMI surgiu em 1998 como um componente no *Service Pack 4* do Windows NT 4.0, mas, a partir do Windows 2000, passou a fazer parte dos componentes nativos dos Sistemas Operacionais Windows. Para as versões anteriores existe um pacote para *download* no próprio site da *Microsoft* (WMI, 2008, *On-Line*).

Essa instrumentação surgiu com base na iniciativa da WBEM (*Web-Based Enterprise Management*), sendo esta criada pela indústria responsável por buscar o desenvolvimento de tecnologias padrões para o acesso às informações de gerenciamento de ambientes corporativos. Como a Microsoft estava entre as empresas que propuseram a criação desse padrão, a tecnologia WMI é conhecida como a implementação da Microsoft para o padrão WBEM (WMI, 2008, *On-Line*).

WMI pode ser considerada a tecnologia de gerenciamento primária para os Sistemas Operacionais da Microsoft. Essa tecnologia permite aos administradores de rede e de sistemas uma forma consistente para acessar informações de gerência de sistemas e foi projetado desde o início para trabalhar através de redes. Utilizando as informações de gerência acessadas por meio do WMI é possível fazer consultas, monitoração, configuração de computadores *desktop* e servidores, assim como do próprio Sistema Operacional, além de aplicações de rede e até mesmo dispositivos físicos. Portanto, entender o funcionamento dos recursos do WMI é tão importante e útil quanto entender o funcionamento dos serviços do *Active Directory*.

Utilizando a biblioteca de *Scripting WMI*, administradores de sistemas podem criar *scripts* para trabalhar com WMI e criarem diversos sistemas de gerenciamento e *scripts* de monitoramento, o que facilita sobremaneira o árduo trabalho de gerência de redes, principalmente pela possibilidade de realizar acessos tanto a máquinas locais como remotas.

A tecnologia WMI permite que o gerenciamento de redes Windows através do fornecimento de acesso aos objetos gerenciáveis seja feito de forma simples, utilizando os conceitos de orientação a objetos, ou seja, classes, métodos, propriedades e relações entre os recursos gerenciáveis. Para cada recurso há uma classe WMI correspondente, de forma

que exista uma descrição, mesmo que sucinta, das propriedades do recurso gerenciável e as ações que o WMI pode realizar para gerenciar o recurso (WMI, 2008, *On-Line*).

Um recurso ou objeto gerenciável através da tecnologia WMI pode ser um computador, software, serviços, conta de usuários e grupos, pastas, recursos compartilhados, eventos registrados em *logs*, arquivos de sistemas, segurança, componentes de redes, enfim, uma diversidade de recursos subordinados à gerência através do WMI. A grande vantagem da utilização do WMI se destaca pelo fato de não ser necessário o uso de diversas ferramentas para fazer a gerência dessa diversidade de recursos, dependendo apenas que o administrador saiba como escrever *scripts* que faz uso da biblioteca de *Scripting* WMI.

Utilizando o Windows *Script Host* (WSH), *VBScript* ou *Microsoft JScript* é possível escrever *scripts* WMI para automação do gerenciamento de redes e sistemas. WMI possui várias características e aplicações, as quais se destacam para o desenvolvimento do trabalho proposto (WMI, 2008, *On-Line*):

- *scripts* para administrar *logs* de eventos, segurança da rede e de sistemas, sistemas de arquivos, configurações, impressoras, processos e muitos outros componentes do sistema operacional;
- *scripts* para administrar serviços de redes, como: DNS, DHCP e outros; e
- *scripts* para monitoração em tempo real.

Uma das principais vantagens em utilizar a tecnologia WMI é a possibilidade de aplicar os *scripts* para fazer a gerência ou configurações em computadores tanto locais como remotos. Desta forma, torna-se possível obter *logs* de segurança dos computadores de uma rede, instalar software remotamente, obter a lista de usuários que estão logados em cada máquina da rede em tempo real, obter características dos equipamentos da rede e softwares instalados, enfim, recursos gerenciais que se tornaram possíveis graças ao surgimento do WMI. Antes do WMI, essas gerências eram realizadas através de ferramentas de terceiros e dificilmente tinha o acesso remoto, sendo que as ferramentas próprias do Windows eram bastante limitadas.

Executando o simples *script* WMI da tabela 9 no *prompt* de comando utilizando o *CScript* é possível obter o total da memória física em qualquer computador da rede, basta atribuir o nome do computador à variável *strComputer*. Para executar o *script* deve ser utilizada a sintaxe: “*cscript* [nome do *script*]” no *prompt* de comando de uma estação ou



servidor Windows, sendo que o *script* abaixo deverá ser salvo com a extensão ‘.vbs’ (WMI, 2008, *On-Line*).

**Tabela 9:** Exemplo de Script WMI – Memória

Linha	Sintaxe
1.	strComputer = "."
2.	
3.	Set objSWbemServices = GetObject("winmgmts:\\\" & strComputer)
4.	Set colSWbemObjectSet = _
5.	objSWbemServices.InstancesOf("win32_LogicalMemoryConfiguration")
6.	
7.	For Each objSWbemObject In colSWbemObjectSet
8.	Wscript.Echo "Total Physical Memory (kb): " & _
9.	objSWbemObject.TotalPhysicalMemory
10.	
11.	Next

Analisando o *script* da Tabela 9 comprova-se que para cada recurso gerenciável existe uma classe WMI correspondente. Na linha 5 é instanciada a classe “*Win32\_LogicalMemoryConfiguration*” a qual contém as propriedades tanto da memória física como virtual. Para mostrar a memória virtual basta substituir a propriedade do objeto para *TotalVirtualMemory* na linha 9 do *script* exemplo. Ao executar o *script* WMI exemplificado na Tabela 10 é possível obter todos os serviços de um determinado computador ordenados pelo nome, *status* e o modo de inicialização dos mesmos.

**Tabela 10:** Exemplo de Script WMI - Serviços

Linha	Sintaxe
1.	strComputer = "."
2.	
3.	Set objSWbemServices = GetObject("winmgmts:\\\" & strComputer)
4.	Set colSWbemObjectSet =
5.	objSWbemServices.InstancesOf("Win32_Service")
6.	For Each objSWbemObject In colSWbemObjectSet
7.	Wscript.Echo "Display Name: " & objSWbemObject.DisplayName &
8.	vbCrLf & _
9.	" State: " & objSWbemObject.State & vbCrLf & _
10.	" Start Mode: " & objSWbemObject.StartMode
11.	Next

Observa-se que no exemplo da Tabela 10, linha 4, está sendo instanciada a classe de objetos de serviços do Windows “*Win32\_Service*”, e que as propriedades são exclusivas para cada tipo de objeto. As classes WMI são na prática representatividades

virtuais da realidade, enquanto que as propriedades dos objetos são as mesmas dos objetos reais. Outra análise importante sobre WMI é a existência de semelhanças entre os *scripts*. Por exemplo, os *scripts* da Tabela 9 e Tabela 10 basicamente diferem somente na classe a ser instanciada e nas propriedades dos objetos. Logo, escrever *scripts* WMI não é uma tarefa muito complicada, uma vez que envolve apenas três passos (WMI, 2008, *On-Line*):

- a conexão com o serviço WMI;
- recuperação de informações sobre a classe WMI; e
- realizar alguma tarefa com as informações.

Como exemplo da semelhança entre os *scripts* WMI a Tabela 11 apresenta um *script* que recupera e mostra os *logs* de eventos do Windows.

**Tabela 11:** Exemplo de Script WMI – *Logs* de Eventos

Linha	Sintaxe
1.	strComputer = "."
2.	Set objSWbemServices = GetObject("winmgmts:\\." & strComputer)
3.	Set colSWbemObjectSet =objSWbemServices.InstancesOf("Win32_ NTLogEvent")
4.	For Each objSWbemObject In colSWbemObjectSet
5.	Wscript.Echo "Log File: " & objSWbemObject.LogFile & vbCrLf & _
6.	"Record Number: " & objSWbemObject.RecordNumber & vbCrLf & _
7.	"Type: " & objSWbemObject.Type & vbCrLf & _
8.	"Time Generated: " & objSWbemObject.TimeGenerated & vbCrLf & _
9.	"Source: " & objSWbemObject.SourceName & vbCrLf & _
10.	"Category: " & objSWbemObject.Category & vbCrLf & _
11.	"Category <i>String</i> : " & objSWbemObject.CategoryString & vbCrLf & _
12.	"Event: " & objSWbemObject.EventCode & vbCrLf & _
13.	"User: " & objSWbemObject.User & vbCrLf & _
14.	"Computer: " & objSWbemObject.ComputerName & vbCrLf & _
15.	"Message: " & objSWbemObject.Message & vbCrLf
16.	Next

Analisando o *script* da Tabela 11, as linhas 2, 3 e 4 mostram os três passos a serem seguidos para escrita de *scripts* WMI, assim como apresenta a estrutura quase que permanente do *script*. Comparando com os *scripts* dos exemplos anteriores, nota-se que houve alterações apenas com relação às classes instanciadas e as propriedades dos objetos. Essas características facilitam o entendimento e o trabalho de escrita de *scripts* WMI para as diversas finalidades que se propõe através das informações obtidas dos objetos gerenciáveis.

### 2.7.1. Arquitetura WMI

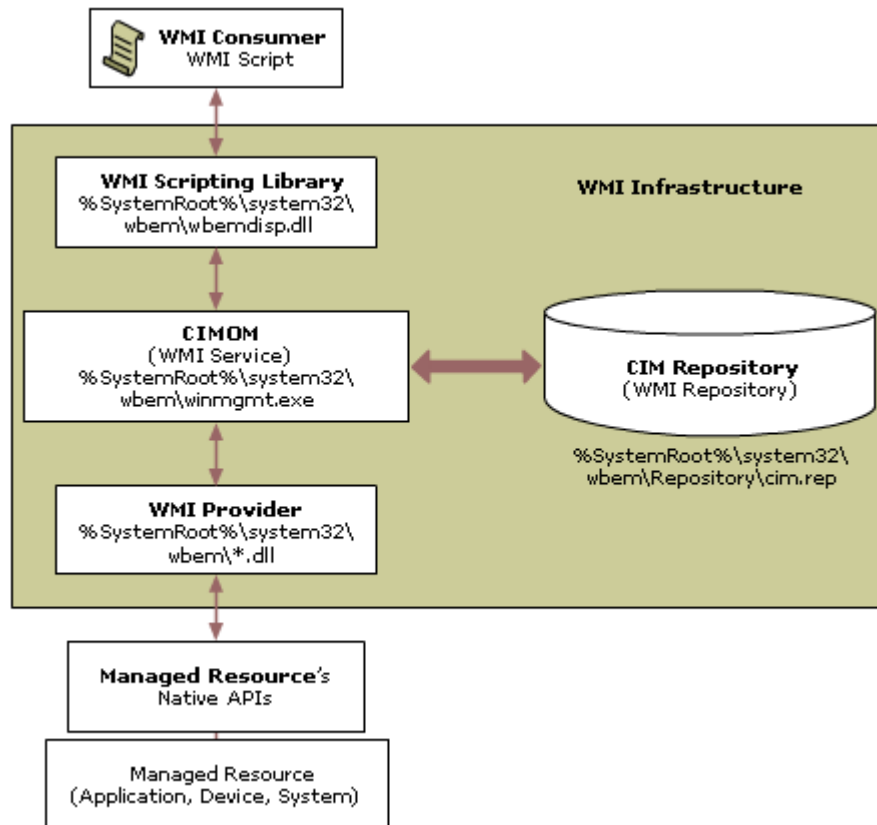
Apesar da facilidade para se construir *scripts* WMI que possibilitam obter informações gerenciais a partir de um *script* padrão, torna-se necessário conhecer quais elementos, classes, métodos e propriedades que podem ser utilizados no *script* padrão para alcançar o objetivo esperado. Vale ressaltar que também é importante conhecer como WMI trabalha e como e onde são armazenadas as informações do WMI. Para um bom trabalho baseado em *script* WMI é indispensável que o administrador saiba responder três questões básicas, como (WMI, 2008, *On-Line*):

- quais classes estão disponíveis;
- quais são os nomes dessas classes; e
- quais propriedades e métodos podem ser usados para cada classe.

O Modelo de Informações Comuns (CIM) fornece uma cobertura detalhada da arquitetura de WMI, o que traz uma compreensão mais abrangente para construção de ferramentas gerenciais baseadas na tecnologia WMI. A arquitetura WMI se desenvolve em três camadas principais a seguir, as quais são mostradas também na figura 16 (WMI, 2008, *On-Line*).

- clientes;
- infra-estrutura WMI; e
- recursos gerenciados.

Além dessas três camadas principais, será explorado também na seqüência a forma como WMI trabalha com o aspecto segurança, visto ser bastante salutar o administrador de redes ou de sistemas entender todo esse conjunto de informações para então construir *scripts* úteis e eficientes para gerência de redes.



**Figura 16:** Arquitetura WMI (WMI, 2008, On-Line).

A estrutura, a composição e o funcionamento de cada camada apresentada na figura 16, serão expostos nos tópicos seguintes.

### 2.7.1.1. Recursos Gerenciados

Os recursos gerenciados formam a camada base da arquitetura WMI. Tais recursos podem ser componentes físicos ou lógicos, como: sistemas computacionais, discos, dispositivos periféricos, *logs* de eventos, arquivos e pastas, arquivos de sistemas, componentes de redes, subsistemas do Sistema Operacional Windows, impressoras, processos, configurações de registros, segurança, serviços, compartilhamentos, usuários e grupos, serviço de diretório do Windows (*Active Directory*), instalador do Windows, modelo de driver Windows (WDM) e também a base de informações de gerenciamento (MIB) (WMI, 2008, *On-Line*).

O recurso gerenciado comunica com o WMI através de um provedor. Como já descrito, a prática de construção de *scripts* para interagir com recursos gerenciados WMI faz uso do termo “instância”, o qual se refere a uma representação de uma implementação atual de algum objeto que pode ser gerenciado pelo WMI, cuja representação é realizada

através da classe instanciada. Por exemplo, o *script* da Tabela 9, que tem como resultado de sua execução uma representação real do total de memória física do computador, é realizada através da instância da Classe “*Win32\_LogicalMemoryConfiguration*”.

### 2.7.1.2. Infra-Estrutura do WMI

A infra-estrutura é a camada intermediária do modelo arquitetural que consiste em 03 (três) componentes primários (WMI, 2008, *On-Line*):

- CIMOM: Gerente de objetos do Modelo de Informações Comum, conhecido também como *WMI Service*;
- CIM: Modelo de Informações Comum, conhecido também como repositório WMI; e
- *WMI providers*: Provedores WMI.

Esses três componentes formam a infra-estrutura sobre a qual as configurações e dados de gerenciamento são definidos, expostos, acessados e recuperados.

#### 2.7.1.2.1. Provedores de WMI

Os provedores de WMI possuem a função de interligar o CIMON aos recursos gerenciados. Esses provedores solicitam informações e as enviam para os recursos gerenciados WMI em nome da aplicação cliente (WMI, 2008, *On-Line*). Por exemplo, o *script* que recupera registro dos *logs* de eventos do Windows utiliza o provedor embutido *Event Log* e o que recupera o tamanho da memória utiliza o provedor *Win32*.

Outro fator importante reside na forma de comunicação que os provedores WMI fazem com seus respectivos recursos gerenciados, cuja comunicação é realizada através das APIs (*Application Programming Interfaces*). Já a comunicação com o CIMOM (Gerente de objetos do Modelo de Informações Comum) se concretiza com o uso de interfaces de programação WMI (WMI, 2008, *On-Line*). Por exemplo, o provedor *Event Log* chama a API *Win32 Event Log* para acessar os *logs* de eventos. Veja que o administrador responsável por criar os *scripts* não interage diretamente com as APIs, o próprio provedor WMI estabelece o uso das mesmas de forma transparente para o administrador, até porque diversas APIs, tipo a *Win32*, não permitem ser chamadas através de *scripts*, o que forçaria o uso de linguagens de programação diferente e a perda das características de simplicidade de se trabalhar com WMI, como: construir *scripts* em poucos minutos, sem necessidade de uso de ferramentas diferentes do Notepad; além de a

construção de *scripts* não requerer que seja incluso cabeçalho de arquivos, nem fazer uso de compilador.

Outra vantagem refere-se ao fato de que uma pessoa que necessita construir *scripts* WMI não tem a necessidade de conhecer as APIs dos recursos gerenciados nem as diferenças que existem. WMI possui a responsabilidade de traduzir todos os comandos WMI inseridos nos *scripts* em comando entendíveis pelas APIs.

Para ficar mais claro, os provedores WMI são implementados como bibliotecas de *links* dinâmicos residentes na própria estrutura de instalação do sistema operacional Windows (2000, XP, 2003 e posteriores), especificadamente no diretório “*systemroot\System32\Wbem*”. Por se tratar de recursos nativos nos sistemas operacionais Windows, os desenvolvedores podem utilizá-los para incluir recursos de gerenciamento em seus produtos (WMI, 2008, *On-Line*).

Como existe uma diversidade de provedores padrões embutidos nos sistemas operacionais Windows, a Tabela 12 traz uma amostra desses provedores.

Tabela 12: Lista parcial dos provedores padrões WMI

Provedor	DLL	NameSpace	Descrição
<i>Active Directory</i>	Dsprov.dll	Root\directory\ldap	Mapas dos objetos do <i>Active Directory</i> para WMI.
<i>Event Log</i>	Ntevt.dll	Root\cimv2	Administração dos <i>Logs</i> de Eventos do Windows, como: leitura, backup, limpeza, cópias, exclusão, renomeação, compactação e descompactação dos arquivos de <i>logs</i> de eventos e alteração nas configurações dos <i>logs</i> de eventos.
<i>Performance Counter</i>	Wbemperf.dll	Root\cimv2	Prover acessos a dados de apresentação.
<i>Registry</i>	Stdprov.dll	Root\default	Leitura, gravação, monitoração, criação e exclusão de chaves e valores dos registros.
SNMP	Snmpincl.dll	Root\snmp	Prover acesso aos dados da MIB SNMP.
<i>Security Provider</i>	Cimwin32.dll	Root\cimv2	Recupera ou altera configurações de segurança que controla propriedades, auditoria, direitos de acessos a arquivos, diretórios e compartilhamentos.

WDM	Wmiprov.dll	Root\wmi	Prover acesso às informações sobre <i>drivers</i> dos dispositivos WDM.
Win32	Cimwin32.dll	Root\cimv2	Prover informações sobre o computador, discos, dispositivos periféricos, arquivos e pastas, compartilhamentos, arquivos de sistemas, componentes de redes, sistemas operacionais, segurança, serviços, usuários e grupos etc.
<i>Windows installer</i>	Msiprov.dll	Root\cimv2	Prover acessos às informações sobre softwares instalados.

### 2.7.1.2.2. CIMOM

O CIMOM lida com interações entre os clientes e os provedores. O termo CIMOM surgiu da iniciativa do WBEM (*Web-Based Enterprise Management*) e a especificação do CIM (*Comom Information Model*) é mantida pelo DMTF (*Distributed Management Task Force*) (WMI, 2008, *On-Line*).

Todas as informações e solicitações passam pelo CIMOM. Os *scripts* são direcionados para o CIMOM, entretanto, este não lida diretamente com as informações. O CIMOM recebe as solicitações através dos *scripts* e localiza um provedor que possa atender as solicitações. Quando o provedor processa as requisições, os resultados são enviados para o CIMOM, que possui a responsabilidade de entregá-las para o usuário solicitante (WMI, 2008, *On-Line*).

Os serviços WMI (*winmgmt.exe*) fornecem regras para o CIMOM sobre o Windows XP e são semelhantes à maioria dos serviços do Sistema Operacional, podendo ser parados e inicializados utilizando os seguintes comandos “*Net stop winmgmt*” e “*Net start winmgmt*”.

O CIMOM provê uma interface comum para os clientes acessarem o WMI e ainda fornece os seguintes serviços para a infra-estrutura do WMI (WMI, 2008, *On-Line*):

- *Provider Registration*: registro de provedores WMI e informações de capacidade com o CIMOM. As informações são armazenadas no CIM pelo CIMOM;
- *Request routing*: o CIMOM utiliza das informações dos provedores para direcionar as requisições dos clientes para o provedor apropriado;

- *Remote Access*: os clientes acessam de forma remota os sistemas WMI habilitados conectando ao CIMOM. Uma vez estabelecida a conexão, com poucas exceções, qualquer operação que pode ser feita localmente pode também, facilmente, ser realizada no computador remoto;
- *Security*: o CIMOM faz o controle dos acessos aos recursos gerenciados WMI validando o cadastro de acesso de cada usuário antes de conceder permissão para conexão WMI ao computador local ou remoto;
- *Query processing*: permite ao cliente fazer consulta de qualquer recurso gerenciado WMI usando a linguagem de consulta WMI (WQL). Através dessa linguagem pode ser feita consulta de apenas um tipo (ID) específico de eventos que por acaso tenha ocorrido em um determinado tempo; e
- *Event Processing*: permite ao cliente subscrever eventos que representam as alterações em um recurso gerenciado WMI. Assim, o cliente pode submeter *scripts* que disparam alertas quando, por exemplo, o espaço de uma unidade lógica do disco está abaixo do tamanho aceitável. O CIMOM gerencia os recursos em um intervalo especificado e gera eventos de notificações quando a subscrição é satisfeita.

Com base nesses serviços, quando é feita solicitações para o CIMOM recuperar dados, inscrever-se para eventos, ou alguma outra tarefa relacionada com gerenciamento, o CIMOM recupera o provedor e informações da classe necessárias para os serviços solicitados do repositório CIM.

### **2.7.1.2.3. Repositório CIM**

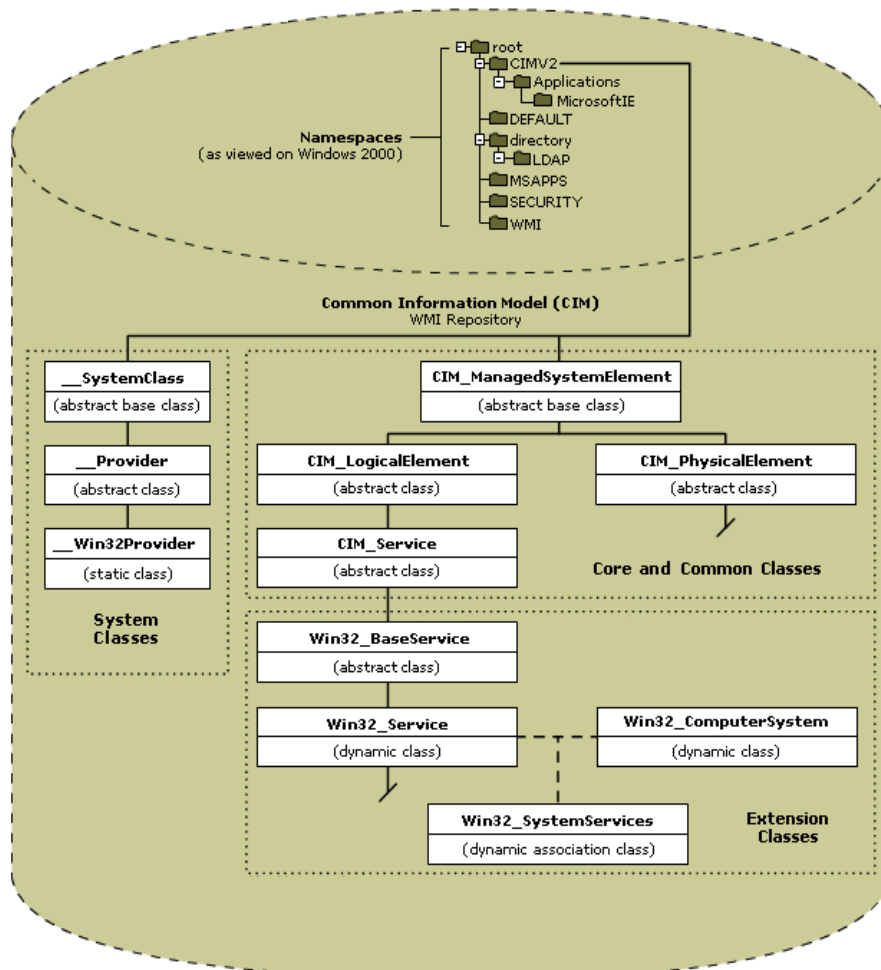
WMI baseia-se na idéia de que configuração e informações sobre a gestão de diferentes fontes poderão ser uniformemente representadas com um esquema. O repositório CIM, também conhecido como repositório de objetos ou de classes detém o esquema que modela o ambiente gerenciado e define várias partes dos dados expostos através do WMI. Tal esquema é baseado nas Informações Comuns do Modelo Padrão (DTMF) (WMI, 2008, *On-Line*).

O esquema CIM é semelhante a um projeto arquitetônico da estrutura física de uma casa, o CIM modela o hardware, o sistema operacional, e o software que constitui um computador. O CIM é um modelo de dado para WMI. Portanto, conhecer e saber como interpretar o projeto WMI para gerenciamento, denominado de repositório CIM, é de



fundamental importância para quem for escrever *scripts* WMI. Conhecer o repositório CIM implica em saber navegar na estrutura CIM e poder determinar o computador e os recursos de softwares expostos através do WMI. Saber como interpretar o projeto de gerenciamento WMI, ou o projeto de um recurso gerenciado, ajuda a ter o conhecimento sobre quais tarefas podem ser executadas sobre o recurso gerenciado.

A figura 17 mostra uma visão conceitual da estrutura interna e da organização do repositório CIM. Como ilustrado na figura 17, o CIM usa classes para criar o modelo de dados. O CIM contém muito mais classes do que as mostradas no diagrama. Isto é importante para entender que o repositório CIM é o armazém de classes que define o ambiente de gerenciamento WMI e todos os recursos controláveis expostos através do WMI, (WMI, 2008, *On-Line*).



**Figura 17:** Estrutura interna do Repositório CIM (WMI, 2008, On-Line)

Detalhando a estrutura apresentada na figura17, destacam-se três conceitos do Modelo de Informações Comuns que são relevantes para aprender como navegar no

repositório CIM de forma eficiente e saber interpretar o esquema WMI, (WMI, 2008, *On-Line*):

1. O repositório CIM está dividido em vários *NameSpaces*;
2. Cada *NameSpace* pode conter um ou mais grupo das seguintes classes:
  - classes de sistemas;
  - classes comuns e de núcleo;
  - extensão de classes.
3. Há três tipos de classes primárias: abstrata, estática, e dinâmica. Um quarto tipo de classe, conhecida como classe de associação, é também suportado.
  - uma classe abstrata é o modelo usado para derivar novas classes abstratas ou não abstratas, e não pode ser usada para recuperar instâncias de recursos gerenciados;
  - uma classe estática define os dados que são armazenados fisicamente no repositório CIM, dos quais os mais comuns é a configuração WMI e os dados operacionais;
  - classes dinâmicas é classe que modela os recursos gerenciados WMI que são recuperados dinamicamente de um provedor;
  - uma classe de associação pode ser abstrata, estática, ou dinâmica. Utilizada para descrever um relacionamento entre duas classes ou recursos gerenciados.

De forma análoga ao esquema do *Active Directory*, o CIM é construído com base nos conceitos de classes, sendo que uma classe é um modelo de um recurso gerenciado. A diferença reside no fato de que as classes do *Active Directory* representam objetos estáticos enquanto as classes CIM representam recursos dinâmicos. Para melhor entendimento dos recursos dinâmicos, as instâncias de recursos não são armazenadas no repositório CIM, porém, são dinamicamente recuperadas através de um provedor com base nas solicitações do cliente. O dinamismo também reside no fato de o *status* operacional da maioria dos recursos gerenciados mudarem com frequência, então, torna-se necessário que as informações recuperadas sejam as mais atuais possíveis. Como a execução de *scripts* WMI apresenta boa performance, não se resume em um complicador o fato de buscar informações sempre atuais, mesmo que o volume seja grande, (WMI, 2008, *On-Line*).

Para melhor entendimento e clareza dos conceitos apresentados na figura 17, faz-se necessário uma abordagem mais abrangente sobre as *NameSpaces* e sobre os tipos de classes do repositório CIM.

### 2.7.1.2.3.1. NameSpaces

*NameSpaces* são mecanismos de partições empregado pelo CIM para controlar o escopo e ter visibilidade de definições de classes dos recursos gerenciados. A grosso modo, *NameSpaces* é semelhante a pastas de um *drive* de armazenamento de dados. Cada *NameSpaces* no CIM contém um grupo lógico de classes relacionadas que representa uma tecnologia específica ou uma área de gerenciamento. Assim como as pastas, os *NameSpaces* contribuem para que o usuário identifique de forma única cada item, por exemplo, uma *NameSpace* de nome *MicrosoftActiveDirectory* provavelmente deve conter classes WMI usadas para gerenciar o *Active Directory*. De forma similar às pastas que não permite dois arquivos com mesmo nome dentro da mesma pasta, também não é permitido classes de mesmo nome dentro da mesma *NameSpace*, podendo, é claro, existir classes de mesmo nome em *NameSpaces* distintas (WMI, 2008, *On-Line*).

Uma diferença importante entre pastas e *NameSpaces* diz respeito à profundidade de níveis de subpastas. Nas pastas a profundidade é livre, podendo ter vários níveis de profundidade. Já nas *NameSpaces* dificilmente se encontrará mais do que três níveis de profundidades.

É importante saber que a maiorias das classes que modelam recursos gerenciados Windows estão armazenadas na *NameSpace* *root\cimv2*, tais como: *logs* de ventos, analisador de desempenho, instalador do Windows, e todos os provedores Win32.

O *script* da Tabela 13 mostra como especificar uma *NameSpace* para conexão, bem como exemplifica o uso da variável *strComputer*. Assim como no Windows, que ao executar um arquivo sem informar o caminho completo, o sistema procura nos caminhos especificados na variável %PATH%, quando não se especifica a *NameSpace* no *script*, assume-se que trata-se da *NameSpace* padrão configurada no Registro. A *NameSpace* padrão é configurada no seguinte endereço: HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\WBEM\Scripting\Default *NameSpace*. No entanto, faz parte das boas práticas para construções de *scripts* a definição da *NameSpace* que será utilizada no *script* para realização das tarefas desejadas diretamente no *script* (WMI, 2008, *On-Line*).

**Tabela 13:** Como especificar uma *NameSpace* no *script*

Linha	Sintaxe
1.	<code>strComputer = "."</code>
2.	<code>Set objSwbemServices = GetObject("winmgmts:\\\" &amp; strComputer &amp; \"\root\cimv2")</code>

A parte inicial de um *script* WMI deve constar sempre o nome do computador sobre o qual será realizada alguma atividade através do *script* WMI. Quando não está especificado o nome do computador (ou a variável *strComputer* tem como valor o ponto (.)), o *script* será executado no computador local. No entanto, percebe-se a facilidade para executar o *script* em um computador remoto, pois será necessário apenas alterar o valor da variável *strComputer*. Esta mesma facilidade estende-se à atribuição do nome da *NameSpace* e da classe, visto que somente será necessário alterá-las em um corpo básico do *script* WMI.

WMI é uma tecnologia bastante versátil e sem muitas complicações, podendo utilizar os próprios *scripts* para aprender sobre a própria tecnologia. Por exemplo, pode-se utilizar o mesmo padrão de *scripts* já demonstrados para recuperar informações das *NameSpaces* do CIM, alterando apenas o nome da classe destino. Isto porque todas as informações sobre as *NameSpaces* são armazenadas no repositório CIM como instâncias estática da classe *\_\_NameSpace*, logo, percebe-se que as instâncias de classes estáticas são recuperadas diretamente do CIM, sem uso de provedor WMI. O *script* da Tabela 14 recupera e exibe todas as *NameSpaces* que estão abaixo da *namespace* principal (WMI, 2008, *On-Line*).

**Tabela 14:** Recuperando NameSpaces CIM usando script WMI.

Linha	Sintaxe
1.	<code>strComputer = "."</code>
2.	
3.	<code>Set objSWbemServices = GetObject("winmgmts:\\\" &amp; strComputer &amp; "\root")</code>
4.	<code>Set colNameSpaces = objSwbemServices.InstancesOf("__NAMESPACE")</code>
5.	
6.	<code>For Each objNameSpace In colNameSpaces</code>
7.	<code>    Wscript.Echo objNameSpace.Name</code>
	<code>Next</code>

### 2.7.1.2.3.2. Categorias de Classes

As classes de sistemas suportam a configuração e operações internas do WMI como, por exemplo, as configurações de *NameSpaces*, segurança de *NameSpaces*, registro de provedores, e eventos de subscrições e notificações. As classes de sistemas são identificadas pelas duas sublinhas (*underscores*) que prefaciam cada nome de classe de sistema, conforme demonstrado na figura 17. Por exemplo, as classes *\_\_SystemClass*, *\_\_Provider*, e *\_\_Win32Provider* e *\_\_NameSpace* são exemplos de classes de sistema,

cujas classes dificilmente farão parte da construção de *script* WMI, a não ser com o objetivo de monitoração (WMI, 2008, *On-Line*).

Classes comuns e de núcleo servem para duas funções: primeiro, elas representam classes abstratas das quais sistemas e desenvolvedores de softwares podem derivar e criar classes de extensões específicas. Segundo, elas definem recursos comuns para áreas de gerenciamento particular.

As extensões de classes são tecnologias de classes específicas criadas pelo sistema e desenvolvedores de softwares aplicativos. As classes *Win32\_BaseService*, *Win32\_Service*, *Win32\_SystemServices* e *Win32\_ComputerSystem*, mostradas na figura 17, são extensões de classes. A maioria das classes de extensão Microsoft da *NameSpace root\cimv2* podem ser identificada através do prefixo *Win32\_*, (WMI, 2008, *On-Line*).

Assim como é possível listar todas as *NameSpaces* através de *scripts* WMI, também é possível listar todas as classes armazenadas dentro de uma *NameSpace*. O *script* da Tabela 15 exemplifica como listar todas as classes definidas na *NameSpace root\cimv2*. Observa que esse *script* faz uso do método *SubclasseOf*, o qual tem como retorno todas as classes filhas da classe pai especificada.

**Tabela 15:** Recuperando as classes definidas na NameSpaces root\cimv2.

Linha	Sintaxe
1.	strComputer = "."
2.	
3.	Set objSWbemServices = GetObject("winmgmts:\\." & strComputer & "\root\cimv2")
4.	Set colClasses = objSwbemServices.SubClassesOf( )
5.	
6.	For Each objClass In colClasses
7.	Wscript.Echo objClass.Path.Path
	Next

### 2.7.1.2.3.3. Tipos de Classes CIM

De forma semelhante ao *Active Directory*, as classes CIM são organizadas de forma hierárquica, classes filhas herdam das classes topo ou pai propriedades, métodos, e qualificadores. O padrão DTMF mantém a base comum de classes, das quais sistemas e aplicações de desenvolvimento derivam e criam sistemas ou aplicações específicas de extensões de classes. Por exemplo, obedecendo a hierarquia de herança, a classe *Win32\_Service* tem como classes superiores na ordem ascendente as seguintes classes: *Win32\_BaseService*, *CIM\_Service*, *CIM\_LogicalElement*, *CIM\_ManagedSystemElement*

das quais herda propriedades, métodos e qualificadores, conforme mostra a figura 17 (WMI, 2008, *On-Line*).

A tabela 16 traz uma comparação das propriedades que não são de sistemas, encontradas em cada uma dessas classes, facilitando a percepção da herança aplicada às classes filhas, bem como as particularidades de cada uma em relação à sua classe pai.

**Tabela 16:** Comparação das propriedades das classes Pais e Filhas.

<b>CIM_ManagedSystemElement and CIM_LogicalElement</b>	<b>CIM_Service</b>	<b>Win32_BaseService</b>	<b>Win32_Service</b>
Description	Description	Description	Description
InstallDate	InstallDate	InstallDate	InstallDate
Name	Name	Name	Name
Status	Status	Status	Status
	CreationClass	CreationClass	CreationClass
	Name	Name	Name
	Description	Description	Description
	Name	Name	Name
	Started	Started	Started
	StartMode	StartMode	StartMode
	Status	Status	Status
	SystemCreation	SystemCreation	SystemCreation
	ClassName	ClassName	ClassName
	SystemName	SystemName	SystemName
		AcceptPause	AcceptPause
		AcceptStop	AcceptStop
		DesktopInteract	DesktopInteract
		DisplayName	DisplayName
		ErrorControl	ErrorControl
		ExitCode	ExitCode
		PathName	PathName
		ServiceSpecific	ServiceSpecific
		ExitCode	ExitCode
		ServiceType	ServiceType
		StartName	StartName
		State	State
		TagID	TagID
			Checkpoint
			ProcessID
			WaitHint

Como visto anteriormente, os tipos de classes CIM se resumem em classes abstratas, estáticas e Classes dinâmicas.

- As classes abstratas CIM são utilizadas como modelo para definir novas classes e servem como base para outras classes abstratas, estáticas e dinâmicas. Este tipo de classe pode ser identificado examinando o qualificador abstrato. Uma classe abstrata deve definir um qualificador abstrato e configurar o qualificador abstrato com valor “true”. O uso mais comum de tipo de classe abstrata é para definir classes comuns e de núcleos. Classes abstratas são raramente usadas em *scripts* WMI, porque não se podem recuperar instâncias de classes abstratas (WMI, 2008, *On-Line*).
- As classes estáticas são responsáveis por definir os dados que são fisicamente armazenados no repositório CIM. As instâncias de classes estáticas são armazenadas e recuperadas diretamente do repositório CIM, pois não utilizam provedores. Estas classes são identificadas examinando o qualificador da classe. Entretanto, diferente dos tipos abstrata e dinâmica que são identificadas pela presença de um qualificador específico, classes estáticas são identificadas pela ausência dos qualificadores abstratos e dinâmicos.
- As classes dinâmicas são responsáveis por modelar (projeto) os recursos gerenciados WMI, os quais são dinamicamente recuperados dos provedores e são identificadas examinando o qualificador dinâmico. Uma classe dinâmica deve definir um qualificador dinâmico e configurar o qualificador dinâmico com valor igual a “true”. O tipo de classe dinâmica é tipicamente utilizado para definir extensões de classes. Classes dinâmicas é o tipo mais comum de classes usadas em *script* WMI.
- Um quarto tipo de classe, conhecida como classe de associação, são classes que descrevem um relacionamento entre duas classes ou entre recursos gerenciados. Classe de Associação pode ser uma classe abstrata, estática ou dinâmica. A classe *Win32\_SystemServices*, mostrada na figura 17, é um exemplo de uma classe de associação dinâmica que descreve o relacionamento entre um computador e o serviços que estão sendo executados nesse computador. Este tipo de classe pode ser identificado examinando o qualificador “associação”. Uma classe de associação abstrata, estática ou dinâmica deve definir o qualificador “associação” e configurá-lo com o valor igual “true” (WMI, 2008, *On-Line*).

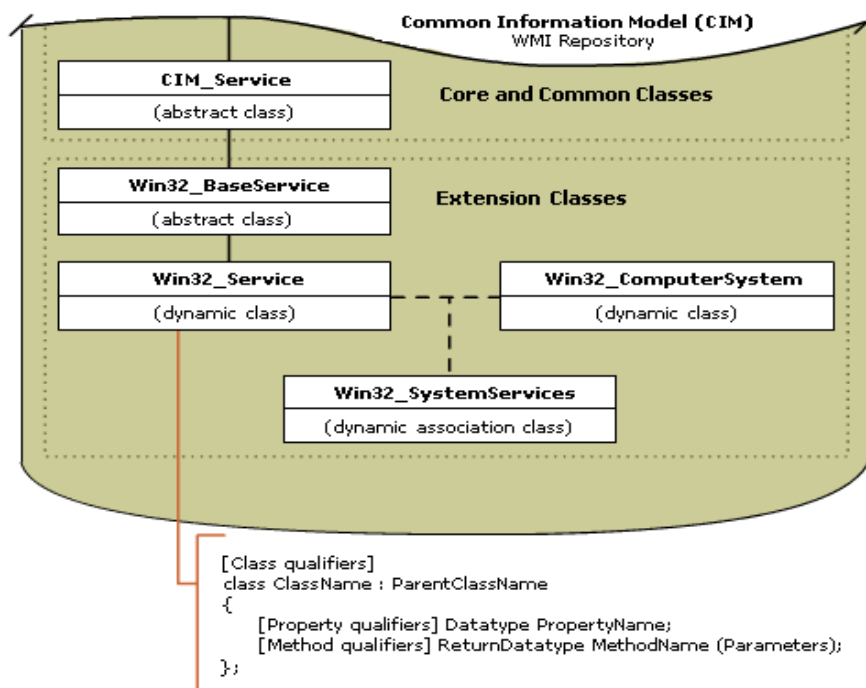
Como CIM é baseado nos princípios da orientação a objetos, as classes CIM incluem propriedades e métodos. As propriedades descrevem as configurações e o *status*

dos recursos gerenciados WMI, enquanto os métodos são funções executáveis que agem sobre os recursos gerenciados associados com as classes correspondentes.

#### 2.7.1.2.3.4. Componentes de uma classe

Cada hardware e recursos de software que é gerenciável através do WMI são definidos por uma classe. Uma classe é um projeto (ou modelo) para um discreto recurso gerenciado WMI, e todas as instâncias do recurso usam o projeto. Por exemplo, todos os serviços (os quais são instâncias da classe *Win32\_Service*) têm as mesmas propriedades. Isto não significa que terão os mesmos valores para as propriedades. Por exemplo, “*Alerter*” tem um nome “*Alerter*”, enquanto o serviço WMI tem o nome *Winmgmt*. Entretanto, todos os serviços têm uma propriedade “Nome”.

Cada definição de classe de recurso gerenciado consiste de propriedades, métodos, e qualificadores. Esta definição é aplicada em todas as instâncias de uma classe e determina o que pode e não pode ser feito para uma instância. Por exemplo, com serviços pode-se especificar ações que deveriam ser tomadas no caso do serviço falhar (reiniciar o serviço, reiniciar o computador, executar um programa, ou não tomar nenhuma ação). A figura 18 demonstra a estrutura de uma definição de classe de um recurso gerenciado (WMI, 2008, *On-Line*).



**Figura 18:** Estrutura de definição de classe de recurso gerenciado (WMI, 2008, *On-Line*).



Propriedades são como nomes (substantivo) que descreve um recurso gerenciado. Classes usam propriedades para descrever coisas tais como a identidade, configuração e o status do recurso gerenciado. Serviços, por exemplo, tem um nome, nome de exibição, descrição, tipo de inicialização e *status*. A classe *Win32\_service* tem as mesmas propriedades. Cada propriedade tem um nome, um tipo e um opcional qualificador (atributo) da propriedade.

Métodos são como verbos que executam uma ação sobre um recurso gerenciado. Por exemplo, para a classe *Win32\_Service* deve existir métodos iniciar, parar, pausar e reiniciar serviços. Cada método tem um nome, um tipo de retorno, parâmetros opcionais, e opcionais qualificadores (atributos) de métodos. Assim como propriedades, é possível usar o nome do método em combinação com *SWbemObject* para chamar o método.

Os qualificadores (atributos) são como adjetivos que provêm informações adicionais sobre a classe, propriedade, ou método para as quais são aplicadas. Os qualificadores definem as características operacionais da propriedade que está sendo atualizada ou do método que está sendo chamando. Há três tipos de qualificadores, os quais provêm três tipos de informações (WMI, 2008, *On-Line*):

- Qualificadores de classes: provêm informações operacionais sobre a classe:
  - qualificadores abstratos, dinâmicos, e de associação indica o tipo da classe;
  - o qualificador do provedor indica o provedor que serve a classe. Por exemplo, o qualificador do provedor para a classe *Win32\_Service* indica que a classe utiliza o provedor *CIMWin32 (cimwin32.dll)*; e
  - o qualificador *EnumPrivileges* informa o privilégio especial requerido para usar uma classe.
- Qualificadores de propriedades provêm informações sobre cada propriedade:
  - o qualificador *CIMType* diz o tipo de dados da propriedade. WMI suporta um número de tipos de dados, incluindo *Strings*, inteiros, data e hora, *arrays*(vetores), e valores *Booleanos (boolean)*;
  - o qualificador “*Read*” indica se a propriedade é legível;
  - o qualificador “*Write*” indica se pode modificar o valor da propriedade; e
  - o qualificador *Key* indica que a propriedade é a chave da classe (chave primária) e é usada para identificar a instância única do recurso gerenciado na coleção de recursos idênticos.

- Qualificadores de Métodos provêm informações sobre cada método. Por exemplo:
  - o qualificador “*implemented*” indica que o método tem uma implementação suprida por um provedor. Isto é importante porque classes WMI frequentemente incluem métodos que são válidos, mas, não realizam nenhuma tarefa porque não foram implementados ainda, estão vazios;
  - o qualificador *ValueMap* define a configuração de valores permitido para o parâmetro de um método ou tipo de retorno; e
  - o qualificador *Privileges* informa privilégios especiais requeridos para chamar o método.

Propriedades e métodos são importantes porque as versões do WMI diferem entre sistemas operacionais; a classe *Win32\_ComputerSystem* no Windows XP tem várias novas propriedades e métodos não suportados na classe *Win32\_ComputerSystem* do Windows 2000. Portanto, é interessante ter conhecimento sobre esses detalhes porque, diferente das interfaces de serviço do *Active Directory* (ADSI), as propriedades e métodos WMI devem estar disponíveis no computador destino em ordem para ser possível um *script* realizar alguma atividade.

Complementando, ainda, é importante saber o local de armazenamento do CIM, visto que o local varia de acordo com a versão do Sistema Operacional Windows. Para facilitar a localização, a Tabela 17 mostra o diretório onde fica armazenado o CIM nos diferentes Sistemas Operacionais Windows (WMI, 2008, *On-Line*).

**Tabela 17:** Local de armazenamento do CIM

Sistema Operacional – Versão	Diretório
Windows ME, 98 e 95 OSR 2.5	% windir%\System\Wbem\Repository\cim.rep
Windows 2000 e NT 4.0 SP4	Systemroot\System32\Wbem\Repository\cim.rep
Windows XP	Systemroot\System32\Wbem\Repository\FS

O Modelo de Informações Comuns (CIM) consiste basicamente de quatro arquivos, a saber:

- *index.btr*: árvore binária de índice de arquivos;
- *index.map*: arquivo de controle de transações;
- *objects.data*: repositório CIM onde são armazenadas definições dos recursos gerenciados; e
- *objects.map*: arquivo de controle de transações.

### 2.7.1.3. Clientes WMI

Conforme demonstrado na arquitetura do WMI, figura 16, os clientes formam a última camada na infra-estrutura do WMI. Quanto à identificação o cliente poder ser (WMI, 2008, *On-Line*):

- um *script*;
- uma aplicação de gerenciamento;
- uma aplicação baseada na Web;
- alguma ferramenta administrativa que acessa e controla informações de gerenciamento disponíveis através da infra-estrutura do WMI.

Os *scripts* das Tabelas 09, 10 e 11, os quais recuperam o tamanho da memória física, os serviços do Windows e os *logs* de eventos do Windows respectivamente, são exemplos de *script* cliente WMI.

### 2.7.1.4. Segurança em WMI

WMI é uma tecnologia extremamente poderosa e versátil para auxiliar os trabalhos de administração de sistemas. A aplicação dos recursos providos pelo WMI torna-se bastante simples com a escrita de *scripts* usando nada mais que o *Notepad*, cujos *scripts* podem ser facilmente executados tanto em computadores locais como remotos. O resultado da execução desses *scripts* pode ser útil para a administração de sistemas ou de redes, como também podem causar prejuízos irreversíveis dependendo da má intenção do construtor do *script*. A ciência é sempre livre e isenta de maldades, o homem utilizador dessa ciência é que pode desvirtuar a sua aplicação para o lado ruim (WMI, 2008, *On-Line*).

Pensando na segurança na aplicação dessa tecnologia, para evitar que *hackers* façam mau uso dos *scripts* para causar prejuízos a terceiros foram impostas algumas restrições na execução dos *scripts*, sem abrir mão da facilidade de construção dos mesmos, ou seja, construir *scripts* continua fácil, mas, a execução somente é permitida se atendido alguns critérios de segurança, os quais evitam ações de *hackers* como também execução de *scrips* por pessoas indevidas. Por exemplo, a execução de *scripts* somente é permitida por um administrador que tenha privilégios específicos.

A segurança de WMI é uma extensão do subsistema de segurança construído dentro do sistema operacional Windows e inclui os seguintes itens (WMI, 2008, *On-Line*):

- nível de segurança baseado no *WMI NameSpace*;

- *Distributed Component Object Model (DCOM) security*;
- segurança padrão do sistema operacional Windows.

#### 2.7.1.4.1. WMI NameSpace

As permissões WMI são aplicadas a nível de *NameSpace* e por padrão aplicadas em todas classes que faz parte da *NameSpace*. As permissões são aplicadas somente para a *NameSpace* principal (*root*) e replicada, por herança, para todas as *NameSpaces* filhas.

O nível segurança aplicado através do WMI *NameSpace* faz com que sejam validadas as permissões da conta do usuário com as permissões aplicadas e armazenadas no repositório CIM. Por padrão, para quem faz parte do grupo de administradores é dado o privilégio completo no WMI e no repositório CIM, tanto nos computadores locais como nos computadores remotos, enquanto os demais usuários são concedidos privilégios somente no computador local (WMI, 2008, *On-Line*).

A lista de permissões WMI disponíveis está na tabela 18, as quais são configuradas através da aba segurança do controle WMI acessado através do MMC, *systemroot\system32\Wmimgmt.msc* (WMI, 2008, *On-Line*).

**Tabela 18:** Permissões WMI NameSpace

Permissão	Descrição	Administrador	Todos
Executar Métodos	O usuário chama o método de um específico <i>NameSpace</i> , entretanto, a execução somente se realiza após o provedor checar se o usuário tem privilégio de executar a tarefa a ser realizada.	•	•
Escrita completa ( <i>Full Write</i> )	Permissão para o usuário criar ou modificar uma <i>NameSpace</i> , uma classe de sistema ou uma instância.	•	
Escrita Parcial	Permissão para criar ou modificar classe estática ou alguma instância de classes que não seja de sistema.	•	
Modificar Provedor	Permissão para o usuário escrever classes e instâncias para provedores WMI.	•	•
Habilitar conta	Conceder permissão de leitura para um WMI <i>NameSpace</i> . Isto permite que o usuário execute <i>script</i> que recupere dados, porém, somente em computador local.	•	•
Habilitar remoto	Permite um usuário acessar um WMI <i>NameSpace</i> a partir de um computador remoto. Este direito é concedido somente para os administradores.	•	
Leitura de	Permite o usuário ler, porém não modificar, a	•	

segurança	descrição de segurança para um WMI <i>NameSpace</i> .		
Editar Segurança	Permite um usuário modificar a descrição de segurança para um WMI <i>NameSpace</i>	•	

#### 2.7.1.4.2. Segurança DCOM

DCOM pode ser definida como a arquitetura subjacente a interação da biblioteca de *script* WMI com o serviço WMI. Esta arquitetura provê um mecanismo de personificação capaz de permitir especificar de qual usuário (ou perfil do usuário) o serviço WMI irá utilizar os privilégios para executar uma tarefa, agindo como se fosse o próprio usuário. Essa personificação permite o serviço WMI agir em nome de um usuário usando suas credenciais. É possível também atribuir o privilégio de delegação, ou seja, dar o direito do serviço WMI permitir o uso das credenciais do usuário também em outros serviços ou em um terceiro computador (WMI, 2008, *On-Line*).

#### 2.7.1.4.3. Padrão de Segurança do Windows

De forma adicional à segurança em WMI, as permissões para realização de tarefas através dos *scripts* estão subordinadas aos critérios de segurança do próprio sistema operacional Windows. Por exemplo, se o usuário executar um *script* para adicionar algum arquivo em uma determinada pasta e caso este mesmo usuário não possuir permissão Windows de escrita nessa pasta, a execução do *script* não terá sucesso. É isto que obrigará o administrador primeiramente a definir as permissões no Windows para, assim, seja possível realizar da tarefa definida no *script* WMI (WMI, 2008, *On-Line*).

### 2.7.2. Descrição das principais classes WMI para a solução proposta

Foi apresentado nos tópicos anteriores, dentro da estrutura do repositório CIM, as categorias e tipos de classes WMI, bem como os componentes que formam uma classe. Foi apresentado, ainda, o conceito e a estrutura das *namespaces* e como as classes estão organizadas nas mesmas. No entanto, dentre as diversidades e quantidade de classes existentes não faz sentido para o escopo do presente trabalho o enorme esforço de enumerar e detalhar todas as classes WMI, até porque deixaria o trabalho bastante extenso.

Portanto, serão apenas listadas e detalhadas as principais classes correlacionadas com o trabalho.

Para o escopo de auditoria em redes Windows, WMI disponibiliza o provedor “*Event Log*”, o qual fornece os seguintes serviços (MSDN, 2008, *On-Line*):

- acesso aos dados do serviço de *logs* de eventos; e
- notificações de ventos.

O provedor de *logs* de eventos (*Event Log*) usa a classe *Win32\_NTEventLogFile* para mapear dados de *logs* de eventos para objetos WMI, e usa a classe *Win32\_NTLogEvent* para representar os eventos. O provedor “*Event Log*” implementa a interface padrão *IWbemProviderInit*, comum a todos os provedores, disponibilizando os seguintes métodos que fazem parte da interface *IWbemServices*: *CreateInstanceEnumAsync*, *ExecMethodAsync*, *GetObjectAsync* e *PutInstanceAsync* (MSDN, 2008, *On-Line*).

Dentro da estrutura do repositório do WMI (CIM) existem os arquivos que formatam os objetos gerenciáveis, denominados de arquivos MOFs. Esses arquivos contêm a definição de classes que descreve as capacidades providas pelos provedores. Por exemplo, o arquivo “*Ntevt.mof* “ contêm informações do provedor “*Event Log*”, as associações e classes registradas. Todas as classes suportadas pelo provedor “*Event Log*” estão armazenadas na *namespace root\cimv2*, as quais serão listadas e detalhadas a seguir por serem importantes para o escopo do presente trabalho. Vale ressaltar que estas seções foram baseadas no tutorial oferecido pela Microsoft e disponibilizadas em (MSDN, 2008, *On-Line*).

### 2.7.2.1. Classe *Win32\_NTEventLogFile*

A classe WMI *Win32\_NTEventlogFile* representa um arquivo ou diretório lógico dos eventos do sistema operacional. A sintaxe demonstrada na Tabela 19 é o código simplificado do MOF – Formato do objeto gerenciado, e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

**Tabela 19:** Sintaxe da classe *Win32\_NTEventlogFile*

Sintaxe da classe <i>Win32_NTEventlogFile</i>
<pre>class win32_NTEventlogFile : CIM_DataFile {     uint32 AccessMask;     boolean Archive;</pre>

---

```

    String Caption;
    boolean Compressed;
    String CompressionMethod;
    String CreationClassName;
    datetime CreationDate;
    String CSCreationClassName;
    String CSName;
    String Description;
    String Drive;
    String EightDotThreeFileName;
    boolean Encrypted;
    String EncryptionMethod;
    String Extension;
    String FileName;
    uint64 FileSize;
    String FileType;
    String FSCreationClassName;
    String FSName;
    boolean Hidden;
    datetime InstallDate;
    uint64 InUseCount;
    datetime LastAccessed;
    datetime LastModified;
    String LogfileName;
    String. Manufacturer;
    uint32 MaxFileSize;
    String Name;
    uint32 NumberOfRecords;
    uint32 OverwriteOutDated;
    String OverwritePolicy;
    String Path;
    boolean Readable;
    String Sources[];
    String Status;
    boolean System;
    String Version;
    boolean writeable;
};

```

---

A Tabela 20 traz a lista e a descrição dos métodos definidos pela classe *Win32\_NTEventlogFile* (MSDN, 2008, *On-Line*):

**Tabela 20:** Descrição dos Métodos da classe *Win32\_NTEventlogFile*

Métodos	Descrição
<i>TakeOwnership</i>	Método da classe que obtém propriedade do arquivo lógico especificado na propriedade " <i>Name</i> ".
<i>changeSecurityPermissions</i>	Método que altera a permissão de segurança do arquivo lógico especificado na propriedade " <i>Name</i> ".

<i>Copy</i>	Método que copia o arquivo ou diretório lógico especificado na propriedade “Name” para o local especificado pelo parâmetro “input”.
<i>Rename</i>	Método que renomeia o arquivo lógico (ou diretório) especificado na propriedade “Name”.
<i>Delete</i>	Método que deleta o arquivo lógico (ou diretório) especificado na propriedade “Name”.
<i>Compress</i>	Método que compacta o arquivo lógico (ou diretório) especificado na propriedade “Name”.
<i>Uncompress</i>	Método que descompacta o arquivo lógico (ou diretório) especificado na propriedade “Name”.
<i>TakeOwnershipEx</i>	Método para obter a propriedade do arquivo lógico especificado na propriedade “Name”.
<i>ChangeSecurityPermissionsEx</i>	Método que altera a permissão de segurança do arquivo lógico especificado na propriedade “Name”.
<i>CopyEx</i>	Método para copia o arquivo lógico (ou diretório) especificado na propriedade “Name” para o local especificado pelo parâmetro “FileName”.
<i>deleteEx</i>	Método que deleta o arquivo lógico (ou diretório) especificado na propriedade “Name”.
<i>compressEx</i>	Método que usa compactação NTFS para compactar o o arquivo lógico (ou diretório) especificado na propriedade “Name” .
<i>uncompressEx</i>	Método que descompacta o arquivo lógico (ou diretório) especificado na propriedade “Name” .
<i>GetEffectivePermission</i>	Método que determina se o visitante tem permissões agregadas especificadas não somente pelo argumento “Permission” do arquivo do objeto, mas sobre o arquivo ou diretório residentes no compartilhamento.
<i>ClearEventLog</i>	Limpa o log do evento especificado.
<i>backupEventLog</i>	Salva o log do evento especificado em um arquivo de backup.

A Tabela 21 traz a lista e a descrição das propriedades definidas pela classe *Win32\_NTEventLogFile* (MSDN, 2008, *On-Line*):

**Tabela 21:** Descrição das Propriedades da classe *Win32\_NTEventLogFile*

Propriedades	Qualificadores	Descrição
<i>AccessMask</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read-only</i>	Representa os direitos de acesso exigidos para acessar ou executar uma operação específica sobre o arquivo de log de evento.
<i>Archive</i>	Tipo de dados: <i>boolean</i> Tipo de acesso: <i>Read-only</i>	Se verdadeiro, o arquivo que contém os eventos do Windows deve ser arquivado.
<i>Caption</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Resumo da descrição do objeto.
<i>Compressed</i>	Tipo de dados: <i>boolean</i>	Se verdadeiro, o arquivo que contém os



	Tipo de acesso: <i>Read-only</i>	eventos do Windows é compactado.
<i>CompressionMethod</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Algoritmo ou ferramenta utilizada para compactar o arquivo lógico que contém os eventos Windows.
<i>CreationClassName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i> Qualifiers: Key, Maxlen(256)	Nome da primeira classe concreta que aparece em um corrente de herança usada na criação de uma instância. Quando usada com a outra propriedade “key” da classe, esta propriedade permite todas as instâncias dessa classe e subclasses serem unicamente identificadas.
<i>creationDate</i>	Tipo de dados: <i>datetime</i> Tipo de acesso: <i>Read-only</i>	Data em que o arquivo que contém os eventos Windows foi criado.
<i>CSCreationClassName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Classe do sistema do computador.
<i>CSName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do sistema do computador
<i>Description</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Descrição do objeto
<i>Drive</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Letra do <i>drive</i> do arquivo que contém os eventos Windows. Por exemplo: “C:”.
<i>EightDotThreeFileName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do arquivo DOS compatível para o arquivo que contém os eventos Windows. Por exemplo: “C:\progra~1”
<i>Encrypted</i>	Tipo de dados: <i>boolean</i> Tipo de acesso: <i>Read-only</i>	Arquivo que contém os eventos Windows está criptografado.
<i>encryptionMethod</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Algoritmo ou ferramenta usada para criptografar o arquivo lógico.
<i>Extension</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Extensão do arquivo (sem o ponto) para o arquivo que contém os eventos do Windows. Por exemplo: “txt”, “mof”.
<i>fileName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do arquivo (sem extensão) definido para o arquivo que contém os eventos do Windows. Por exemplo: “autoexec”.
<i>fileSize</i>	Tipo de dados: <i>uint64</i> Tipo de acesso: <i>Read-only</i>	Tamanho (em <i>bytes</i> ) do arquivo que contém eventos do Windows.
<i>fileType</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Tipo de arquivo (indicado pela propriedade <i>Extension</i> ).
<i>FSCreationClassName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Classe do sistema de arquivo.
<i>FSName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do sistema de arquivo.
<i>Hidden</i>	Tipo de dados: <i>boolean</i> Tipo de acesso: <i>Read-only</i>	Se verdadeiro, o arquivo que contém os eventos do Windows está oculto.
<i>InstallDate</i>	Tipo de dados: <i>datetime</i> Tipo de acesso: <i>Read-only</i>	Objeto está instalado. Esta propriedade não necessita de um valor para indicar que o objeto está instalado.
<i>InUseCount</i>	Tipo de dados: <i>uint64</i>	Número de arquivos abertos que estão

	Tipo de acesso: <i>Read-only</i>	atualmente ativos em relação ao arquivo que contém os eventos Windows.
<i>LastAccessed</i>	Tipo de dados: <i>datetime</i> Tipo de acesso: <i>Read-only</i>	Data e hora que o arquivo que contém eventos Windows foi acessado pela última vez.
<i>LastModified</i>	Tipo de dados: <i>datetime</i> Tipo de acesso: <i>Read-only</i>	Data e hora que o arquivo que contém eventos Windows foi modificado pela última vez.
<i>LogfileName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do arquivo que contém eventos Windows. O padrão para nomes dos arquivos de log inclui: aplicação, sistema e segurança.
<i>Manufacturer</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Elaborador do recurso da versão, se está presente.
<i>MaxFileSize</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read-only</i>	Tamanho máximo (em <i>bytes</i> ) permitido para o arquivo que contém eventos Windows.
<i>Name</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Nome herdado que serve como uma chave para uma instância do arquivo lógico que contém eventos Windows dentro de um sistema de arquivos. Deve ser informado o caminho completo dos nomes. Por exemplo: “ <i>c:\windows\system\win.ini</i> ”
<i>NumberOfRecords</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read-only</i>	Número de registros no arquivo que contém eventos windows. Esse valor é determinado ao chamar a função <i>Windows GetNumberOfEventLogRecords</i>
<i>OverwriteOutDated</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read/Write</i> <i>Qualifiers: Units (Days)</i>	Número de dias após os quais um evento pode ter sido sobrescrito.
<i>OverWritePolicy</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	<i>Policy</i> atual que sobrescreve o serviço de <i>log</i> de eventos empregados para esse arquivo de <i>log</i> . O dado pode ser nunca sobrescrito, ou sobrescrito quando necessário ou quando fora de data(época). Quando o dado está fora de data depende do valor de <i>OverwriteOutDated</i> .
<i>Path</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	O caminho do arquivo que contém o evento do Windows. Por exemplo: “ <i>\windows\system\</i> ”.
<i>Readable</i>	Tipo de dados: <i>boolean</i> Tipo de acesso: <i>Read-only</i>	Se verdadeiro, o arquivo que contém eventos Windows pode ser lido.
<i>Sources</i>	Tipo de dados: <i>String array</i> Tipo de acesso: <i>Read-only</i>	Lista de aplicações que registram <i>logs</i> dentro do arquivo de <i>logs</i> .
<i>Status</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Status atual do objeto. Os possíveis valores são: <i>OK, error, degraded, unknown, pred fail, starting, stopping, service, stressed, nonrecover, no contact, lost conn.</i>
<i>System</i>	Tipo de dados: <i>boolean</i>	Se verdadeiro, o arquivo que contém

	Tipo de acesso: <i>Read-only</i>	eventos Windows é um arquivo de sistema.
<i>Writeable</i>	Tipo de dados: <i>boolean</i> Tipo de acesso: <i>Read-only</i>	Se verdadeiro, o arquivo que contém os eventos Windows pode ser escrito (alterável).

### 2.7.2.2. Classe *Win32\_NTLogEvent*

A classe WMI *Win32\_NTLogEvent* pertencente ao provedor “*Event Log*” (*Ntevt.dll*) é utilizada para traduzir as instâncias do *log* de evento do Windows. Para receber *logs* de eventos de segurança a aplicação deve ter o privilégio definido por *SeSecurityPrivilege*, caso não tenha será retornado para aplicação a mensagem de “*Access Denied*”.

A sintaxe demonstrada na Tabela 22 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

**Tabela 22:** Sintaxe da classe *Win32\_NTLogEvent*

Sintaxe da classe *Win32\_NTLogEvent*

```
class Win32_NTLogEvent
{
    uint16 Category;
    String CategoryString;
    String ComputerName;
    uint8 Data[];
    uint16 EventCode;
    uint32 EventIdentifier;
    uint8 EventType;
    String InsertionStrings[];
    String Logfile;
    String Message;
    uint32 RecordNumber;
    String SourceName;
    datetime TimeGenerated;
    datetime TimeWritten;
    String Type;
    String User;
};
```

A classe *Win32\_NTLogEvent* não possui métodos definidos. A Tabela 23 traz a lista e a descrição das propriedades definidas pela classe *Win32\_NTLogEvent* (MSDN, 2008, *On-Line*):

**Tabela 23:** Descrição das Propriedades da classe Win32\_NTLogEvent

Propriedades	Qualificadores	Descrição
<i>Category</i>	Tipo de dados: <i>uint16</i> Tipo de acesso: <i>Read-only</i>	Subcategoria para os eventos. Essa subcategoria está em local específico.
<i>CategoryString</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Tradução de subcategoria. A tradução está em local específico.
<i>ComputerName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do computador que gerou o evento.
<i>Data</i>	Tipo de dados: <i>uint8 array</i> Tipo de acesso: <i>Read-only</i>	Lista de dados binários que acompanham o relatório do evento do Windows.
<i>EventCode</i>	Tipo de dados: <i>uint16</i> Tipo de acesso: <i>Read-only</i>	Valor 16-bits inferiores da propriedade <i>EventIdentifier</i> . Esse valor está presente corresponder com o valor exibido no visualizador de eventos do Windows.
<i>EventIdentifier</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read-only</i>	Identificador de eventos. Esse é específico para a origem que gerou a entrada do <i>log</i> de evento e é usado, juntamente com <i>SourceName</i> , para identificar de forma única o tipo de evento do Windows.
<i>EventType</i>	Tipo de dados: <i>uint8</i> Tipo de acesso: <i>Read-only</i>	Os Sistemas Windows server 2003, Windows 2000 e XP possui os seguintes tipos de eventos: Valores: 1 → <i>Error</i> - 2 → <i>Warning</i> 3 → <i>Information</i> ; 4 → <i>Security Audit Success</i> ; 5 → <i>Security Audit Failure</i> .
<i>InsertionStrings</i>	Tipo de dados: <i>String array</i> Tipo de acesso: <i>Read-only</i>	Lista das <i>Strings</i> de inserção que acompanham o relatório dos eventos Windows.
<i>Logfile</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Nome do arquivo de <i>log</i> de eventos Windows. Juntamente com <i>RecordNumber</i> , é utilizado para identificar de forma única uma instância dessa classe.
<i>Message</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Como aparece a mensagem do evento no <i>log</i> de eventos do Windows. Esta é uma mensagem padrão com zero ou mais <i>Strings</i> de inserção fornecida pela origem do evento do Windows. As <i>Strings</i> de inserção são inseridas dentro da mensagem padrão em um formato predefinido.
<i>RecordNumber</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Identifica o evento dentro do arquivo de <i>log</i> de eventos Windows. Esse identificador é específico para o arquivo de <i>log</i> e é usado juntamente com o nome do arquivo de <i>log</i> para identificar de forma única a instância dessa classe.

<i>SourceName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome da origem (aplicação, serviço, driver ou subsistema) que gerou a entrada do <i>log</i> . Este é usado juntamente com o <i>EventIdentifier</i> para identificar de forma única o tipo de vento do Windows.
<i>TimeGenerated</i>	Tipo de dados: <i>datetime</i> Tipo de acesso: <i>Read-only</i>	Data e hora que o evento foi gerado.
<i>TimeWritten</i>	Tipo de dados: <i>datetime</i> Tipo de acesso: <i>Read-only</i>	Data e hora que o evento foi gravado no arquivo de <i>log</i> .
<i>Type</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Tipo do evento. Este é uma <i>String</i> enumerada. É preferencial usar a propriedade <i>EventType</i> do que a propriedade <i>Type</i> . Os tipos podem ter os seguintes valores: <i>1</i> → <i>Error</i> ; <i>2</i> → <i>Warning</i> ; <i>4</i> → <i>Information</i> ; <i>8</i> → <i>Security Audit Sucess</i> ; <i>16</i> → <i>Security Audit Failure</i> .
<i>User</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do usuário logado quando o evento ocorreu. Se o nome do usuário não pode ser determinado, será atribuído o valor <i>NULL</i> .

A Tabela 24 mostra o exemplo de utilização da classe *Win32\_NTLogEvent* para consultar registros de *logs* de ventos de segurança (MSDN, 2008, *On-Line*).

**Tabela 24:** Exemplo de utilização da classe *Win32\_NTLogEvent*

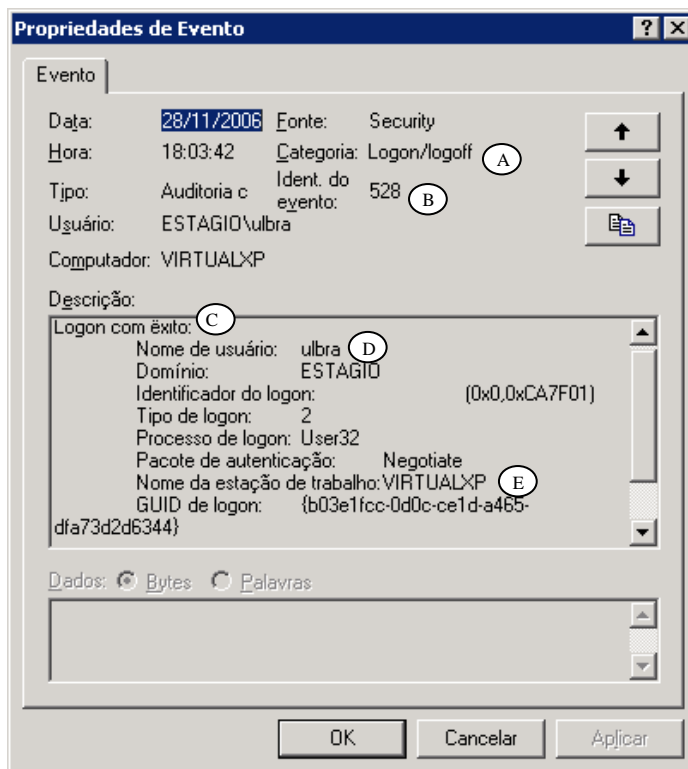
Sintaxe da classe *Win32\_NTLogEvent*

```

strComputer = "."
Set objWMIService = GetObject("winmgmts:" _
    & "{impersonationLevel=impersonate}!\\" _
    & strComputer & "\root\cimv2")
Set colLoggedEvents = objWMIService.ExecQuery _
    ("select * from win32_NTLogEvent " _
    & "where logfile = 'security'")
For Each objEvent in colLoggedEvents
    wscript.Echo "Category: " & objEvent.Category & VBNewLine _
    & "Computer Name: " & objEvent.ComputerName & VBNewLine _
    & "Event Code: " & objEvent.EventCode & VBNewLine _
    & "Message: " & objEvent.Message & VBNewLine _
    & "Record Number: " & objEvent.RecordNumber & VBNewLine _
    & "Source Name: " & objEvent.SourceName & VBNewLine _
    & "Time Written: " & objEvent.TimeWritten & VBNewLine _
    & "Event Type: " & objEvent.Type & VBNewLine _
    & "User: " & objEvent.User
Next

```

Percebe-se, através do *script* do exemplo anterior, a facilidade para obter os registros em computadores remotos com alteração do valor da variável *strComputer*, bem como o uso da linguagem de consulta (SQL) para filtrar os dados desejados. Vale ressaltar que as propriedades desta classe serão as mais utilizadas para obtenção das informações registradas nos *logs* dos eventos relacionados à auditoria. A figura 19 apresenta os tipos de registros de eventos relacionados à auditoria.



**Figura 19:** Exemplo de registro de log do evento de *logon* (JNLC, 2006, p. 92).

Analisando a figura 19 torna-se notável a relação das propriedades listadas na figura com as propriedades da classe WMI *Win32\_NTLogEvent*. Desse tipo de registro no arquivo de *logs* de eventos, por exemplo, seria importante para a auditoria o armazenamento das seguintes informações: a categoria do evento em A, o código do evento em B, o tipo do evento em C, o nome do usuário que efetuou *logon* em D e a estação de trabalho acessada pelo usuário em E.

Convém salientar que os procedimentos de configuração das diretivas de segurança para ativar a geração de registros de *logs* de eventos de auditoria, bem como a descrição dos códigos de eventos importantes para auditoria estão documentados no trabalho de

conclusão de estágio intitulado “Documentação sobre Diretivas de Auditoria de Segurança em Domínio Windows 2003 Server” (JNLC, 2006).

### 2.7.2.3. Classe Win32\_NTLogEventComputer

A classe WMI *Win32\_NTLogEventComputer* faz parte do provedor “Event Log” (*Ntevt.dll*) e é definida como uma classe de associação WMI por fazer o relacionamento entre um computador e o um evento.

A sintaxe demonstrada na Tabela 25 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

**Tabela 25:** Sintaxe da classe Win32\_NTLogEventComputer

Sintaxe da classe *Win32\_NTLogEventComputer*

```
class win32_NTLogEventComputer
{
    win32_ComputerSystem ref Computer;
    win32_NTLogEvent ref Record;
};
```

A classe *Win32\_NTLogEventComputer* não possui métodos definidos. A Tabela 26 traz a lista e a descrição das propriedades definidas pela classe *Win32\_NTLogEventComputer*:

**Tabela 26:** Descrição das Propriedades da classe Win32\_NTLogEventComputer

Propriedades	Qualificadores	Descrição
<i>Computer</i>	Tipo de dados: <i>Win32_ComputerSystem</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Faz referência à instância sobre a qual o evento ocorreu.
<i>Record</i>	Tipo de dados: <i>Win32_NTLogEvent</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Faz referência à instância que representa o evento.

#### 2.7.2.4. Classe Win32\_NTLogEventLog

A classe WMI *Win32\_NTLogEventLog* faz parte do provedor “Event Log” (*Ntevt.dll*) e é definida como uma classe de associação WMI por fazer o relacionamento entre um evento do Windows com um arquivo de log de ventos do Windows.

A sintaxe demonstrada na Tabela 27 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

**Tabela 27:** Sintaxe da classe Win32\_NTLogEventLog

Sintaxe da classe *Win32\_NTLogEventLog*

```
class Win32_NTLogEventLog
{
    Win32_NTEventLogFile ref Log;
    Win32_NTLogEvent ref Record;
};
```

A classe *Win32\_NTLogEventLog* também não possui métodos definidos. A Tabela 28 traz a lista e a descrição das propriedades definidas pela classe *Win32\_NTLogEventLog*:

**Tabela 28:** Descrição das Propriedades da classe Win32\_NTLogEventLog

Propriedades	Qualificadores	Descrição
<i>Log</i>	Tipo de dados: <i>Win32_NTEventLogFile</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Faz referência à instância que representa o arquivo de <i>log</i> de evento.
<i>Record</i>	Tipo de dados: <i>Win32_NTLogEvent</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Faz referência à instância que representa o evento que está sendo registrado.

#### 2.7.2.5. Classe Win32\_NTLogEventUser

A classe de associação WMI *Win32\_NTLogEventUser* faz parte do provedor “Event Log” (*Ntevt.dll*) e é responsável por fazer o relacionamento entre o evento Windows e o usuário correntemente logado no sistema quando da ocorrência do evento.



A sintaxe demonstrada na Tabela 29 é o código simplificado do MOF – Formato do objeto gerenciado (*Ntevt.mof*), e inclui todas as propriedades herdadas (MSDN, 2008, *On-Line*):

**Tabela 29:** Sintaxe da classe Win32\_NTLogEventUser

Sintaxe da classe *Win32\_NTLogEventUser*

```
class Win32_NTLogEventUser
{
    Win32_NTLogEvent ref Record;
    Win32_UserAccount ref User;
};
```

A classe *Win32\_NTLogEventUser* também não possui métodos definidos. No entanto, a Tabela 30 mostra a lista e a descrição das propriedades definidas pela classe *Win32\_NTLogEventUser*:

**Tabela 30:** Descrição das Propriedades da classe Win32\_NTLogEventUser

Propriedades	Qualificadores	Descrição
<i>Record</i>	Tipo de dados: <i>Win32_NTLogEvent</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Faz referência à instância que representa o evento que está sendo associado com o usuário.
<i>User</i>	Tipo de dados: <i>Win32_UserAccount</i> Tipo de acesso: <i>Read-only</i> <i>Qualifiers: Key</i>	Faz referência à instância que representa o usuário associado com o evento.

Diversas outras classes WMI importantes para gerenciamento do ambiente Windows podem ser localizadas no próprio site da Microsoft destinados à comunidade técnica como também às equipes de desenvolvimento de aplicações, conhecidas como Microsoft Technet (WMI, 2008) e Microsoft MSDN respectivamente (MSN, 2008).

### **3. MATERIAIS E MÉTODOS**

Nesta seção serão descritas as ferramentas e a metodologia utilizada no presente trabalho, desde a fase conceitual até os procedimentos adotados para aplica-los na execução do trabalho, o qual tem como objetivo viabilizar auditoria em rede Windows, em específico na Rede do TRE-TO.

#### **3.1. Materiais**

Para o desenvolvimento deste trabalho foi disponibilizada, no ambiente de rede do TRE-TO, as seguintes infraestruturas:

##### **3.1.1. Infraestrutura Física**

Para hospedagem dos sistemas e ferramentas úteis para desenvolvimento do trabalho foi disponibilizado pelo Tribunal um computador tipo Desktop, marca ITAUTEC modelo Infoway SM3330, com Processador AMD Phenom II X2 550, 3.10 Ghz, 4Gb de memória RAM e HD de 500GB.

##### **3.1.2. Infraestrutura Lógica**

A infraestrutura lógica utilizada foi composta por: Sistema Operacional Windows XP SP3 como hospedeiro do ambiente de virtualização Oracle VM VirtualBox. O ambiente virtual formado por 07 máquinas virtuais (VM): 01 VM Windows 2003 Server SP2 com Sistema de Banco de Dados SQL Server 2005, a ferramenta de gerenciamento de políticas do Active Directory: GPMC (Group Policy Manager Console) e a ferramenta PowerShell

Script ISE 2.0 e Microsoft .NET Framework v. 3.5; e 05 VMs Windows XP SP3 como estações de trabalho simulando ambiente de trabalho dos usuários da rede.

### **3.1.2.1. Servidor Windows 2003 Server**

Para o ambiente de testes foi utilizado o Windows 2003 Server Standard Edition Service Pack 2 com os seguintes serviços: *Active Directory* e Serviço DNS (*Domain Name System*) ativos.

O serviço de diretório do Windows 2003 Server é construído ao configurar o Windows 2003 Server para exercer a função de Controlador de domínio da rede. Este serviço é indispensável para realização deste trabalho, uma vez que é o elemento principal para o planejamento e implementação da infraestrutura de rede no Windows 2003 Server. Neste ambiente foi criado os objetos da rede, como: usuários, grupos de usuários e computadores. Para o presente trabalho foi configurado o Domínio “**tcc.neto.com**”, sendo, portanto, o nome do host do servidor como: **server.tcc.com.br**.

O serviço de DNS foi instalado no servidor Windows 2003 Server com o objetivo de fazer a resolução de nomes para a rede utilizada no trabalho. Ao inserir uma estação de trabalho ao Domínio da Rede é adicionado em uma tabela o nome do host e o respectivo endereço IP. Dessa forma, ao ser consultado pelos usuários determinados objetos da rede como: computadores, sites, serviços, etc primeiramente ocorre a consulta ao servidor de DNS local, como consulta de 1º Nível. Este não sendo capaz de resolver os nomes solicitados, o mesmo reencaminha a solicitação aos servidores de DNS de níveis superiores (JNLIC, 2006).

### **3.1.2.2. Ambiente de virtualização Oracle VM VirtualBox**

Para construção do ambiente virtual foi utilizada a ferramenta de virtualização de sistemas operacionais Oracle VM Virtual Box versão 4.3.12. Apesar da ferramenta ser comercializada, não houve custo porque há disponibilidade do proprietário para uso com fins acadêmicos (licença na modalidade PUEL - *Use and Evaluation License*).

O Virtual Box foi originalmente desenvolvido pela empresa alemã Innotek. Posteriormente foi vendida para a empresa SUN e esta em 2010 vendeu para a Oracle Corporation, quando recebeu o nome de “Oracle VM Virtual Box” (MIRANDA, 2010).

O Virtual Box é uma ferramenta portátil e pode ter uma variedade de sistemas operacionais como seu hospedeiro, como: Microsoft Windows, Mac OS, Linux, solareis, etc.

O ponto forte decisivo para a escolha dessa ferramenta, além da familiaridade com o uso da mesma, foi o fácil gerenciamento das máquinas virtuais, o que inclui: criação, configuração, redimensionamento, duplicação, *export/import* e inicialização, pausa e parada de máquinas virtuais. Convém ressaltar que ao realizar a escolha dessa ferramenta, apesar de não ter sido realizado estudo comparativo em face de outras ferramentas similares disponíveis no mercado, houve pesquisas em trabalho dessa natureza realizado por outros acadêmicos, como o trabalho desenvolvido pelo acadêmico Vitor de Deus Miranda (MIRANDA, 2010).

### **3.1.2.3. Estações de trabalho Virtuais com Windows XP SP3**

No ambiente de virtualização foram criadas cinco máquinas virtuais com o Windows XP Service Pack 3 para realização dos testes de aplicação de diretivas de auditoria e coleta de *logs* de eventos.

### **3.1.2.4. Group Policy Manager Console - GPMC**

Esta ferramenta foi utilizada para criação, aplicação e gerenciamento das políticas de auditorias na rede. Esta ferramenta tem a função de integrar e facilitar o gerenciamento das políticas de grupos do *Active Directory*, além das vantagens de possibilitar a realização de backup e restauração, cópia e importação de GPOs (Group Policy Objects) e permitir usar filtros WMI. (JNLC, 2006).

### **3.1.2.5. Banco de dados relacional SQLServer 2005**

Para gravação dos *logs* de ventos de segurança em base de dados centralizada foi adotado o Sistema de banco de dados SQL Server 2005. O sistema de banco de dados Sql Server 2005 é de propriedade da Microsoft, porém, não foi necessário aquisição de licença por se tratar de uso facultado pelo fabricante para trabalho acadêmico (*open license for academic*).

### **3.1.2.6. PowerShell Script ISE 2.0**

Inicialmente foi prevista a utilização da linguagem de aplicação universal VbScript, nativa do Windows, para construção dos scripts propostos no trabalho. No entanto, diante de algumas dificuldades encontradas no decorrer do desenvolvimento do trabalho utilizando VbScript, o que poderia inviabilizar os resultados esperados, foi analisada a ferramenta PowerShell Script ISE 2.5 e, posteriormente, decidiu-se migrar os Scripts para o padrão dessa ferramenta por apresentar maior facilidade de uso e maior disponibilidade de recursos para construção dos Scripts. Por exemplo, a construção de script para conexão com banco de dados é mais facilitado e de melhor compreensão. Houve também a necessidade de extrair informações indispensáveis para auditoria de parte da descrição (*substring*) dos *logs* de eventos de segurança, o que foi melhor resolvido com o uso da ferramenta PowerShell Script.

Como a substituição da ferramenta de Script não impactaria e não mudaria o objeto do trabalho, assim como não alteraria os resultados esperados, não foi vislumbrado nenhum problema quanto à mudança da ferramenta inicialmente prevista, até porque a ferramenta de construção de scripts é apenas o meio facilitador para execução do trabalho.

A ferramenta PowerShell ISE 2.5 é um *shell* de comando baseado em tarefas e em linguagem de script especialmente projetada para administração do Sistema Operacional Windows. A ferramenta foi construída sobre a plataforma do .NET Framework 3.5, o que explica a necessidade de instalação prévia desse Framework disponibilizado pela Microsoft como plataforma de desenvolvimento de aplicativos.

## **3.2. Metodologia**

A primeira fase deste trabalho teve como objetivo elucidar os principais conceitos relacionados ao tema por meio de pesquisas bibliográficas em livros técnicos, artigos técnicos da Microsoft, dissertações e artigos relacionados à gerência e segurança de redes de computadores. Dentre os conteúdos abordados destaca-se:

- **Gerenciamento de Redes:** foram contemplados os principais conceitos relativos à gerência de redes, incluindo os conceitos de Gerente e Agentes, bem como a hierarquia de gerenciamento;
- **Modelos de gerenciamento de Redes:** foram abordados conteúdos dos dois modelos: OSI e INTERNET, sendo mais focado no modelo OSI em razão de contemplar

conteúdo específico para o trabalho proposto. Ainda sobre os modelos de gerenciamento foram evidenciados os protocolos utilizados, bem como as áreas funcionais de gerenciamento de rede, incluindo nestas a área de segurança a qual incorpora o processo de Auditoria;

- **Segurança de redes:** relativo à segurança além dos conceitos aplicados ao assunto, foi evidenciado o tripé dos objetivos a serem alcançados com processo de segurança de redes, quais sejam: confidencialidade, integridade e disponibilidade.
- **Infraestrutura de rede Windows:** principais conceitos sobre a estrutura do controlador de domínio, incluindo os serviços de diretório, suas classes de objetos gerenciáveis e respectivos atributos e o serviço de resoluções de nomes (DNS).
- **Auditoria em Redes Windows:** aplicação de diretivas de auditoria de segurança, configuração e geração de *logs* de eventos de segurança, bem como definição das boas práticas para construção de uma sensata política de auditoria para rede Windows;
- **WMI – Windows Management Instrumentation:** principais conceitos e possibilidades de gerência tanto de rede quanto de sistemas por meio dos recursos dessa ferramenta. Foi abordada ainda a arquitetura do WMI composta pelos recursos gerenciados, provedores, repositórios, categorias de classes e os critérios de segurança para uso dos recursos gerenciais dessa tecnologia, principalmente quanto às permissões para execução de scripts.

O conteúdo teórico desenvolvido na primeira fase deste trabalho deu sustentação e garantia da possibilidade de realizar gerência de Rede Windows por meio da área funcional de segurança, em destaque neste trabalho o processo de auditoria em rede Windows. É válido ressaltar que o processo de auditoria se dá por meio de rastreamento de *logs* de eventos de segurança gerados em consequências de adoção de boas práticas de ativações e configurações de Diretivas de auditoria.

A segunda parte deste trabalho consistiu no desenvolvimento da aplicação. Para isto, foram necessários:

- Definir a política de segurança;
- Implantar a política;
- Analisar os dados disponíveis pelas ferramentas e montar a base de dados;
- Construir os Scripts;
- Definir e Aplicar os testes.

## 4. RESULTADOS E DISCUSSÃO

O presente trabalho foi desenvolvido com o objetivo de minimizar a complexidade para realização de auditoria em redes Windows, por meio do processo de aplicação de Política de Auditoria de Segurança e dos mecanismos de transformação dos dados registrados em *logs* de eventos de auditoria em informações úteis para tomadas de decisões. Cujas soluções propostas contemplam a construção de Script WMI para realização de coletas dos *logs* de eventos de auditoria da rede de computadores e armazená-los em uma base de dados centralizada, bem como a construção de consultas SQL para obter as informações para auditoria. Para atingir o objetivo do trabalho, além da base teórica já exposta e antes da apresentação do cenário de testes faz-se necessário a apresentação dos seguintes tópicos, os quais foram os procedimentos base e indispensáveis para o entendimento e construção da solução proposta.

- Definição da política de auditoria para a rede do TRE-TO;
- Definição e configuração das diretivas de auditoria;
- Segurança de acesso aos registros e filtragem para coletas dos logs;
- Gravação dos registros de logs em arquivos locais;
- Gravação dos registros de logs em banco de dados;
- Propriedades da classe WMI utilizada no Trabalho;
- Tabelas integrantes do esquema relacional do banco de dados; e
- Cenário de testes.

#### **4.1. Definição da Política de Auditoria para a rede do TRE-TO**

Apesar das vantagens da implementação de auditoria na rede, ativar diretivas de auditoria sem uma sólida política de auditoria pode ocasionar que os resultados sejam ineficazes em virtude da falta da legalidade dos atos da auditoria pelos gestores da instituição. Portanto, previamente à execução da auditoria em ambiente real de produção devem-se definir com critérios, vantagens e objetivos claros a política de auditoria de segurança da rede de computadores da instituição. Os passos definidos como boas práticas para construção de uma política de auditoria foram colocadas na seção “Auditoria em Redes Windows” item 2.6 do conteúdo teórico, os quais estão enumerados a seguir:

- Descrição do por que, como e o que será auditado;
- Definição das prioridades sobre o que precisa ser auditado, conforme a necessidade e o impacto da auditoria;
- Definição dos métodos de retenção dos registros de eventos;
- Definição dos meios de gerenciamento e acompanhamento sistemático dos registros de eventos;

Com base nos critérios exposto, no **APENDICE A**, consta a Minuta da Política de Auditoria de Segurança para a Rede Windows do TRE-TO. Cujá minuta, já avalizada pela coordenação de infraestrutura tecnológica do Tribunal será submetida ao comitê de segurança da Instituição com o objetivo de alcançar o aval da alta gestão e posteriormente fazer parte integrante da política de segurança da informação do Tribunal.

Após a definição da Política de Auditoria de Segurança tornou-se viável e perceptível a definição dos cenários de testes necessários para comprovação da aplicação da política de auditoria.

A seguir será exposto o procedimento utilizado para definição e configuração das diretivas de segurança úteis para auditoria.

#### **4.2. Definição e configuração das diretivas de Auditoria**

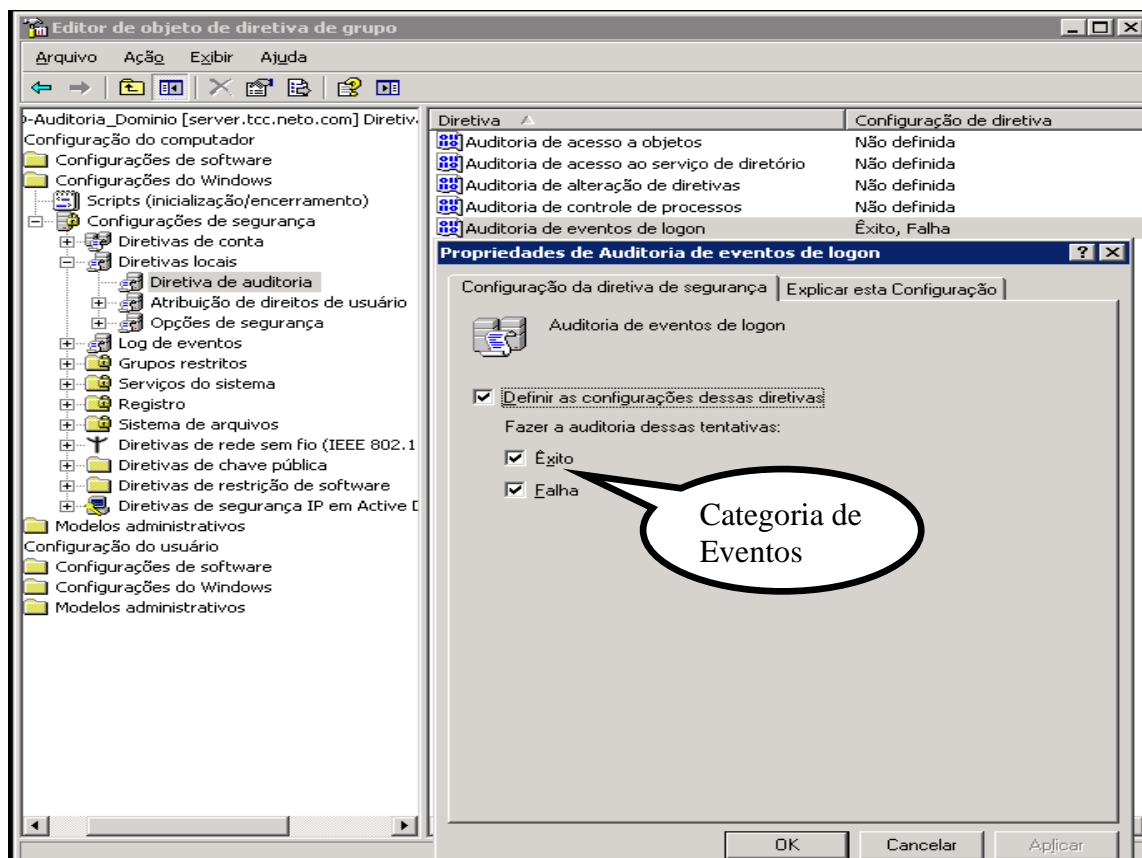
Para possibilitar o registro das ações definidas na Política de Auditoria de Segurança, constante do **Apêndice A** foi definidas as diretivas de auditoria constantes da Tabela 31.



**Tabela 31:** Diretivas definidas pela Política de Auditoria

DIRETIVA	AÇÕES REGISTRADAS	LIMITE DE APLICAÇÃO
Auditoria de eventos de <i>logon</i>	Êxito e Falha	Domínio da Rede
Auditoria de alteração de diretivas	Exito	Servidor DC
Auditoria de acesso a objetos	Êxito e Falha	Domínio da Rede ou em OUs específicas
Auditoria de Gerenciamento de conta	Êxito	Servidor DC
Auditoria de eventos de Sistema	Êxito	Servidor DC

A configuração das diretivas de auditorias foi realizada por meio da ferramenta *Group Policy Management (GPMC)* instalada no Servidor Controlador do Domínio (DC). A configuração consistiu em definir para cada diretiva a categoria do evento para registro em arquivos de log: “**Eventos de Êxito**” ou “**Eventos de falha**”, conforme demonstrado na **figura 20**.



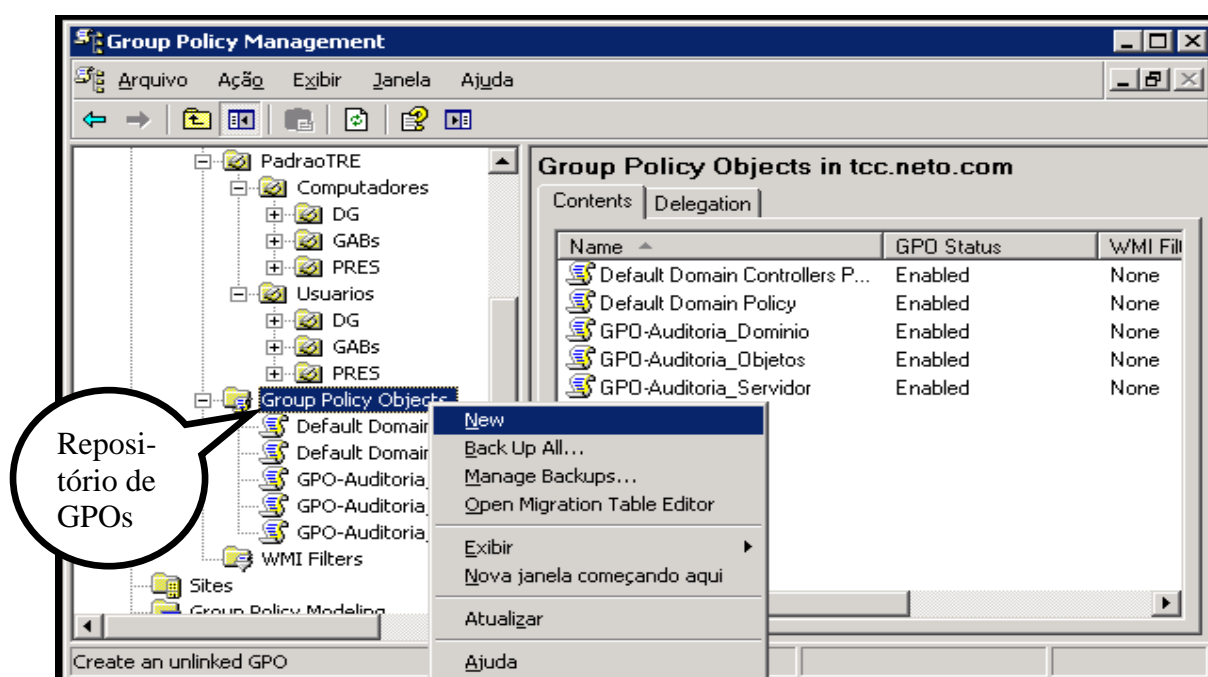
**Figura 20:** Configuração de diretiva de auditoria utilizando GPMC.

Os **eventos de êxito** são diferenciados pelo **ícone de uma chave** e indicam que o sistema operacional concluiu a ação ou operação com êxito, enquanto que os **Eventos de Falha** são diferenciados pelo **ícone de um cadeado** e indicam que houve uma tentativa de ação ou operação mal sucedida.

Para melhor organização da aplicação das diretivas foram criadas as seguintes GPOs (*Group Policy Objects*) para configuração e aplicação de diretivas específicas para auditoria:

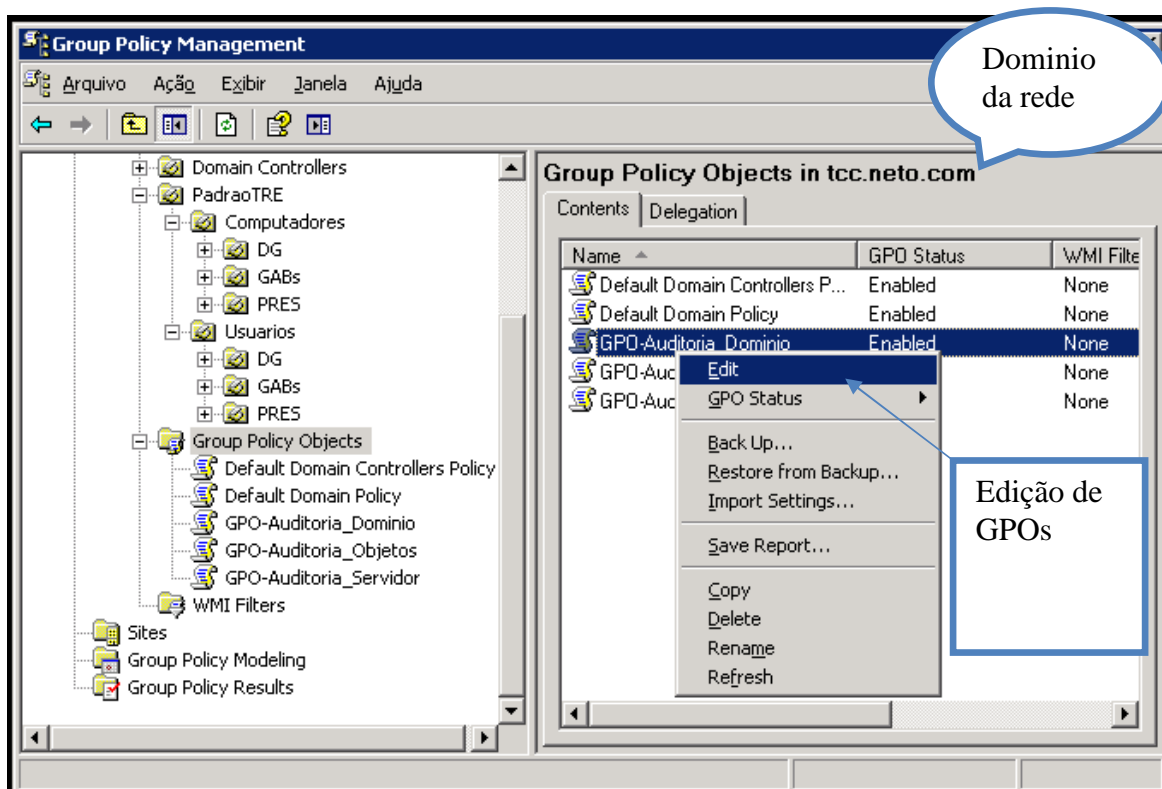
- ✓ GPO - Auditoria\_Dominio: para aplicação de diretiva por herança em todo o domínio. Nesta política deverá configurar as diretivas a serem aplicadas no Servidor e em todas as Estações.
- ✓ GPO - Auditoria\_Servidor: Para aplicação de diretiva de auditoria exclusivamente no servidor DC.
- ✓ GPO - Auditoria\_Objeto: para aplicação de diretiva específicas para setores (Unidades Organizacionais do *Active Directory*) onde há objetos a serem auditados.
- ✓ As novas GPOs devem ser criadas no repositório de GPOs do *Active Directory* denominado de “*Group Policy Objects*”.

A figura 21 mostra o ambiente de gerenciamento de políticas da ferramenta GPMC utilizado para criação das GPOs, bem como o repositório das GPOs criadas.



**Figura 21:** Ambiente de gerenciamento da ferramenta GPMC).

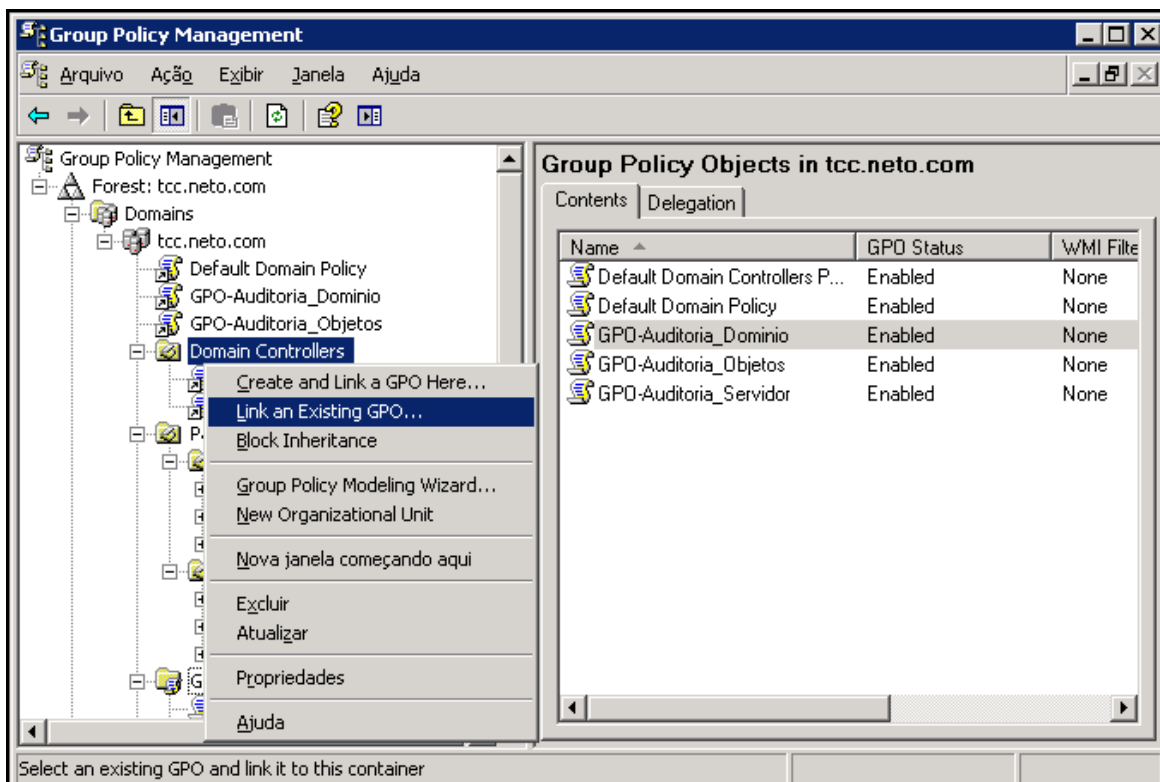
A configuração das diretivas integrantes da Política de Auditoria foi realizada editando as GPOs criadas para essa finalidade no ambiente de gerenciamento de GPOs da ferramenta GPMC, conforme Figura 22. Para tanto, foi selecionada a “Diretiva de Auditoria” e escolhida na sequencia a diretiva para configuração, conforme demonstra a figura 22. O processo de configuração foi o mesmo para todas as diretivas contempladas na Política de Auditoria.



**Figura 22:** Edição de GPOs via GPMC.

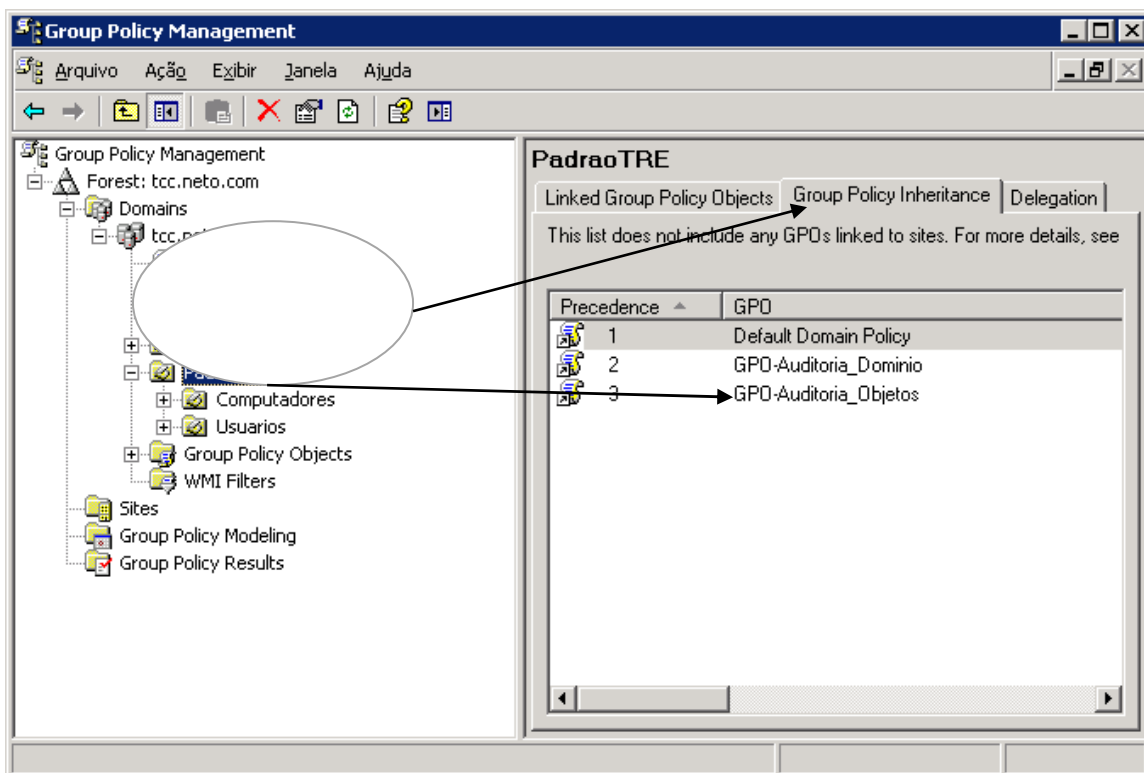
Após a criação e configuração das diretivas de auditoria o passo seguinte foi o estabelecimento do LINK das unidades organizacionais do *Active Directory* com as respectivas GPOs de auditoria.

A opção de link com GPOs já existentes fica disponível ao clicar com o botão direito do mouse sobre o objeto a ser aplicado a GPO, no caso demonstrado na **Figura 23** o link está sendo realizado no Controlador de Domínio da rede.



**Figura 23:** Link de GPO em objetos do AD utilizando GPMC.

Ao aplicar a GPO no Domínio, TCC.NETO.COM, por herança, conforme demonstrado na figura 24, a aplicação aconteceu também em todos os objetos pertencentes ao Domínio, e, conseqüentemente, quando ocorreu ações previstas nas Diretivas configuradas *logs* de eventos começaram a ser gerados onde houve ação auditada, seja nas estações ou no próprio Controlador de Domínio. Lembrando que os conceitos de aplicação de políticas, incluindo bloqueio e herança de aplicação, foram contemplados no conteúdo teórico desenvolvido durante o estágio (JNLC, 2006).



**Figura 24:** Aplicação de GPO por herança.

Uma vez que exposto o procedimento adotado para definição e configuração das diretivas de auditoria para fins de geração dos *logs* de eventos de segurança, o próximo tópico será abordado o procedimento adotado para acesso seguro e filtragem para coleta dos registros de *logs* dos eventos de auditoria.

#### **4.3. Segurança de acesso aos registros e filtragem para coletas dos logs**

A Classe *WMI Win32\_NTLogEvent* possui a estrutura de geração de *logs* de eventos de várias categorias, como: Aplicação, sistema, *Active Directory* e segurança. Para cada categoria existe um arquivo de extensão “.evt” para gravação local dos registros de log. No caso dos *logs* de eventos de segurança o arquivo “SecEvent.evt” é o responsável por receber localmente os *logs* gerados.

O acesso aos registros de *logs* de eventos de segurança é restrito aos usuários com perfil de administrador. Portanto, os scripts de coletas dos *logs* de ventos de auditoria devem ser executados por usuários que tenha o privilégio adequado para acessar as estações de trabalho e o controlador do Domínio da Rede.

A gravação dos *logs* de eventos de segurança no respectivo arquivo local se dá de forma cronológica à execução das ações provocadoras dos eventos. Para o presente trabalho adotou-se a metodologia de filtragem dos registros de *logs* com base no último registro de *log* de evento coletado de cada computador da rede e registrado no Banco de Dados. Assim, a coleta seguinte sempre partirá do número do registro maior do que o último registro de *log* de eventos em banco de dados para o computador.

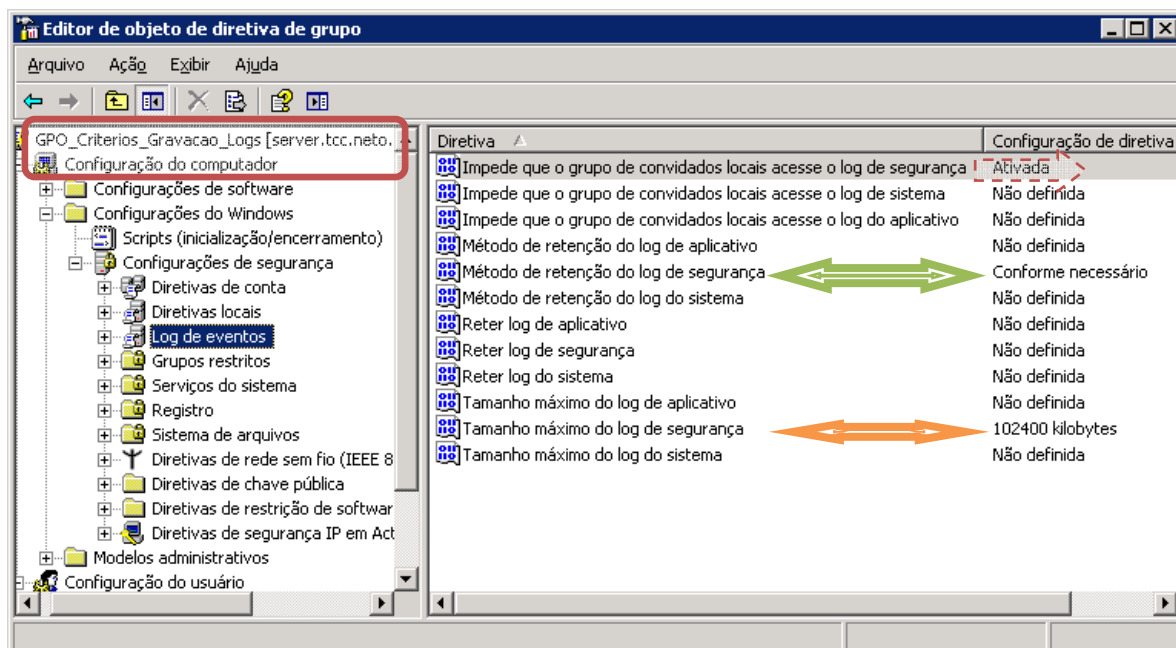
#### **4.4. Gravação dos registros de logs em arquivos locais**

Para execução das propostas do presente trabalho adotou-se a metodologia de gravação dos registros de *logs* de eventos na modalidade “CIRCULAR”. Nesta modalidade não ocorre o impedimento de *logon*, caso o espaço em disco reservado para gravação dos *logs* de eventos esteja esgotado, visto que nessa modalidade ocorre automaticamente a exclusão dos registros de *logs* mais antigos para abrir espaço para os novos registros. No entanto, se há geração de *logs* de evento de forma indiscriminada a coleta dos *logs* de auditoria poderá ser prejudicada pela sobrescrita prematura dos registros.

Usufruindo do passo-a-passo disponível no trabalho de estágio (JNLC, 2006), a configuração da metodologia de gravação dos registros de *logs* para o presente trabalho se deu da seguinte forma:

- ✓ Criação de uma GPO com o nome de: “GPO\_Critérios\_gravacao\_logs” utilizando a ferramenta GMPC;
- ✓ Ao editar a GPO “Critérios\_gravacao\_logs” foi configurado o tamanho máximo que pode chegar o arquivo de *logs* de segurança, a forma de retenção dos *logs* e a negação da permissão de acesso ao grupo convidados:
  - Tamanho máximo do arquivo de log de segurança: 100 Mb
  - Forma de retenção dos *logs*: Circular (Conforme necessário).
  - Ativado o impedimento do grupo de convidado local acessar o log de segurança.

A figura 25 demonstra como ficou configurada a gravação dos *logs* dos eventos de auditoria.



**Figura 25: Critérios de Gravação dos logs dos eventos de auditoria.**

Para coleta dos logs de eventos de auditoria foram utilizados Scripts desenvolvidos no ambiente e recursos da ferramenta PowerShell Script 2.5 ISE. Por meio dos scripts foi possível acessar aos computadores da rede, objetos do *Active Directory*, e fazer a coleta dos eventos de auditoria, assim como estabelecer conexão com o Banco de dados para fins de gravação dos logs coletados em tabela específica no Banco.

#### **4.5. Gravação dos registros de logs em banco de dados**

Para gravação dos logs coletados em banco de dados SQL Server 2005 foi construída a tabela 'Eventos\_Auditoria', contemplando todos os campos (colunas) necessários para gravação dos logs de eventos de auditoria das diretivas suficientes para atender a Política de auditoria, visto que os logs de evento de auditoria pertencem a uma única classe WMI – *Win32\_NTLogEvent*. No entanto, houve a necessidade de adicionar outras tabelas no esquema relacional do banco de dados para fins de cruzamento de informações úteis nas consultas de auditoria, de forma a contemplar as propriedades utilizadas de cada classe no esquema do banco, cujas tabelas estão descritas no item 4.7 “Tabelas integrantes do esquema relacional do banco de dados”.

#### 4.6. Propriedades da classe WMI utilizada no Trabalho

A classe WMI *Win32\_NTLogEvent* possui as propriedades que absorve o registro de logs de todos os eventos de auditoria. No entanto, para alcançar o objetivo da Política de auditoria apenas as propriedades constantes da Tabela 32 foram necessárias.

**Tabela 32:** Propriedades da classe *Win32\_NTLogEvent* utilizadas no Trabalho

Propriedades	Qualificadores	Descrição
<i>Category</i>	Tipo de dados: <i>uint16</i> Tipo de acesso: <i>Read-only</i>	Subcategoria para os eventos. Essa subcategoria está em local específico.
<i>CategoryString</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Tradução de subcategoria. A tradução está em local específico.
<i>ComputerName</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Nome do computador que gerou o evento.
<i>EventCode</i>	Tipo de dados: <i>uint16</i> Tipo de acesso: <i>Read-only</i>	Valor 16-bits inferiores da propriedade <i>EventIdentifier</i> . Esse valor está presente corresponder com o valor exibido no visualizador de eventos do Windows.
<i>EventIdentifier</i>	Tipo de dados: <i>uint32</i> Tipo de acesso: <i>Read-only</i>	Identificador de eventos. Esse é específico para a origem que gerou a entrada do log de evento e é usado, juntamente com <i>SourceName</i> , para identificar de forma única o tipo de evento do Windows.
<i>EventType</i>	Tipo de dados: <i>uint8</i> Tipo de acesso: <i>Read-only</i>	Os Sistemas Windows server 2003, Windows 2000 e XP possui os seguintes tipos de eventos: Valores: 1 → Error - 2 → Warning 3 → Information; 4 → Security Audit Success; 5 → Security Audit Failure.
<i>Logfile</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i> Qualifiers: <i>Key</i>	Nome do arquivo de log de eventos Windows. Juntamente com <i>RecordNumber</i> , é utilizado para identificar de forma única uma instância dessa classe.
<i>Message</i>	Tipo de dados: <i>String</i> Tipo de acesso: <i>Read-only</i>	Como aparece a mensagem do evento no log de eventos do Windows. Esta é uma mensagem padrão com zero ou mais Strings de inserção fornecida pela origem do evento do Windows. As Strings de inserção são inseridas dentro da mensagem padrão em um formato predefinido.



<i>RecordNumber</i>	<i>Tipo de dados: uint32 Tipo de acesso: Read-only Qualifiers: Key</i>	<i>Identifica o evento dentro do arquivo de log de eventos Windows. Esse identificador é específico para o arquivo de log e é usado juntamente com o nome do arquivo de log para identificar de forma única a instância dessa classe.</i>
<i>SourceName</i>	<i>Tipo de dados: String Tipo de acesso: Read-only</i>	<i>Nome da origem (aplicação, serviço, driver ou subsistema) que gerou a entrada do log. Este é usado juntamente com o <i>EventIdentifier</i> para identificar de forma única o tipo de evento do Windows.</i>
<i>TimeGenerated</i>	<i>Tipo de dados: datetime Tipo de acesso: Read-only</i>	<i>Data e hora que o evento foi gerado.</i>
<i>User</i>	<i>Tipo de dados: String Tipo de acesso: Read-only</i>	<i>Nome do usuário logado quando o evento ocorreu. Se o nome do usuário não pode ser determinado, será atribuído o valor <i>NULL</i>.</i>

Além dessas propriedades da classe *Win32\_NTLogEvent* foi necessário extrair outras informações “*substrings*” da propriedade “*message*” úteis para o processo de auditoria, por exemplo:

- tipo de *logon*;
- Domínio;
- Endereço de rede de origem (IP); e
- Porta de origem.

Como a propriedade “*message*” traz a descrição do evento em forma de *String* foi bastante complicado extrair as *substrings* acima por meio de Script para compor as informações de auditoria. A **figura 26** demonstra e as informações da propriedade “*message*” registradas em um evento, no campo “*Descrição*”.

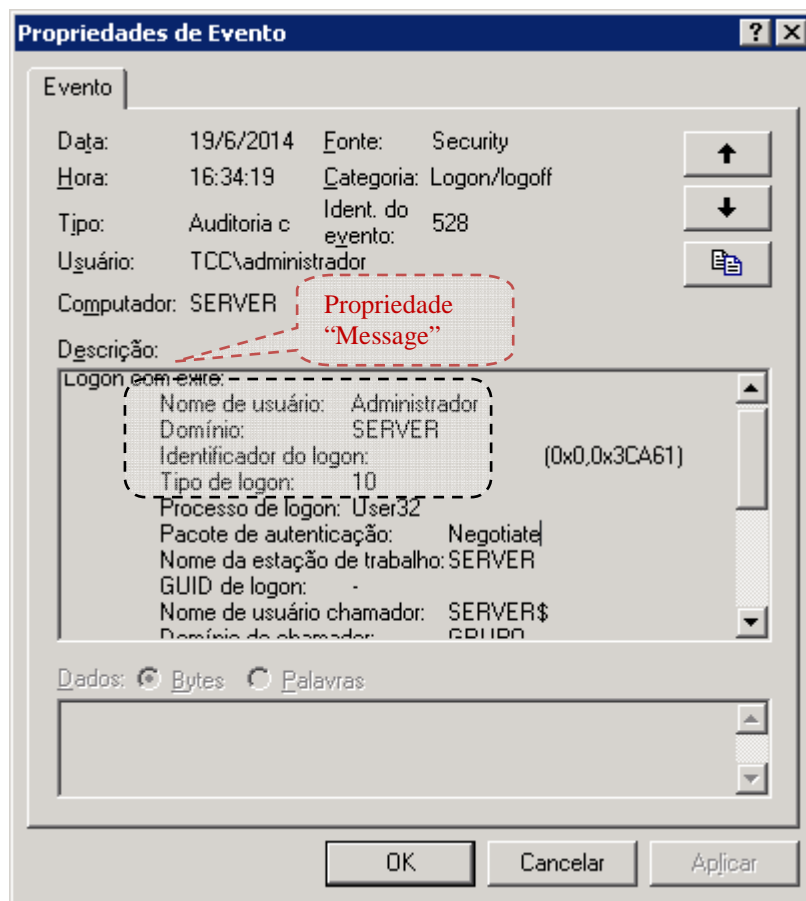


Figura 26: Propriedade “Message” – campo Descrição.

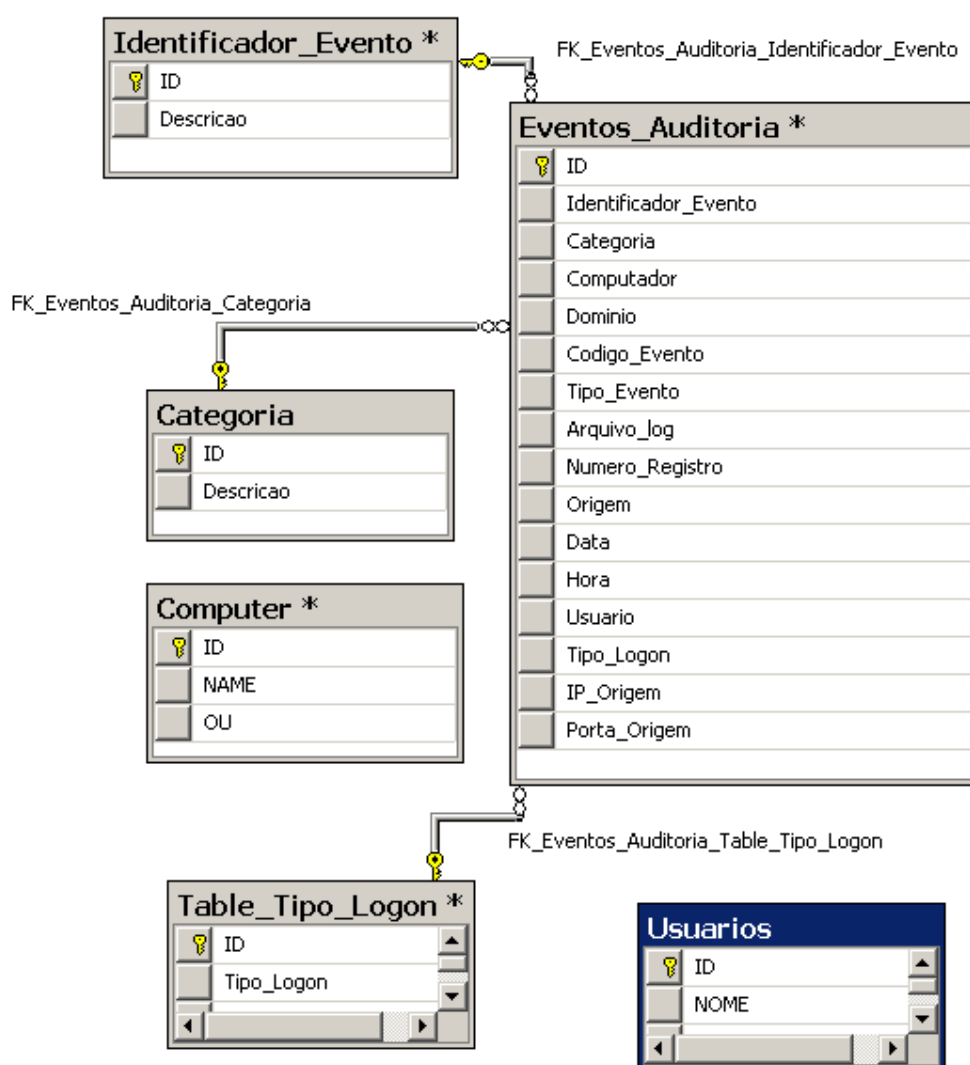
#### 4.7. Tabelas integrantes do esquema relacional do banco de dados

O esquema relacional do banco de dados no qual será armazenados os dados para realização das auditorias constantes da Política de Auditoria (APÊNDICE A) foi construído com as seguintes tabelas:

- **Eventos\_Auditoria:** para gravação dos registros de *logs* de eventos de auditoria coletados via Script dos computadores da Rede. Esta tabela é composta pelas seguintes colunas: as seguintes colunas: ID, Identificador\_Evento, Categoria, Computador, Domínio, Código\_Evento, Tipo\_Evento, Arquivo\_log, Numero\_Registro, Origem, DataHora, usuário, Tipo\_Logon, IP\_Origem, Porta\_Origem;
- **Identificador\_Evento:** para constar a relação dos identificadores dos eventos que serão armazenados em banco de dados e as respectivas descrições. Esta tabela é composta por: ID e Descrição;
- **Categoria:** esta tabela guardará a relação de códigos de categorias de evento e as respectivas descrições. Esta tabela é composta por: ID e Descrição;

- **Table\_Tipo\_Logon:** para constar a relação dos tipos de *Logons* e respectiva descrição. Esta tabela é composta por: *Tipo\_Logon* e *Descrição*;
- **Computer:** Relação dos computadores do Domínio e respectivas Unidades Organizacionais (OUs);
- **Usuarios:** Relação dos usuários do Domínio e respectivas Unidades Organizacionais (OUs);

O esquema relacional do banco foi criado conforme demonstrado na figura 27.



**Figura 27: Esquema Relacional do Banco de dados**

Uma vez definidos e documentados os procedimentos base para a solução proposta, tornou-se possível a definição do cenário de testes para finalmente comprovar a viabilidade

prática da implantação da Política de Auditoria de Segurança na rede Windows do Tribunal. A seguir serão expostos o cenário de testes utilizados no presente trabalho e os resultados obtidos.

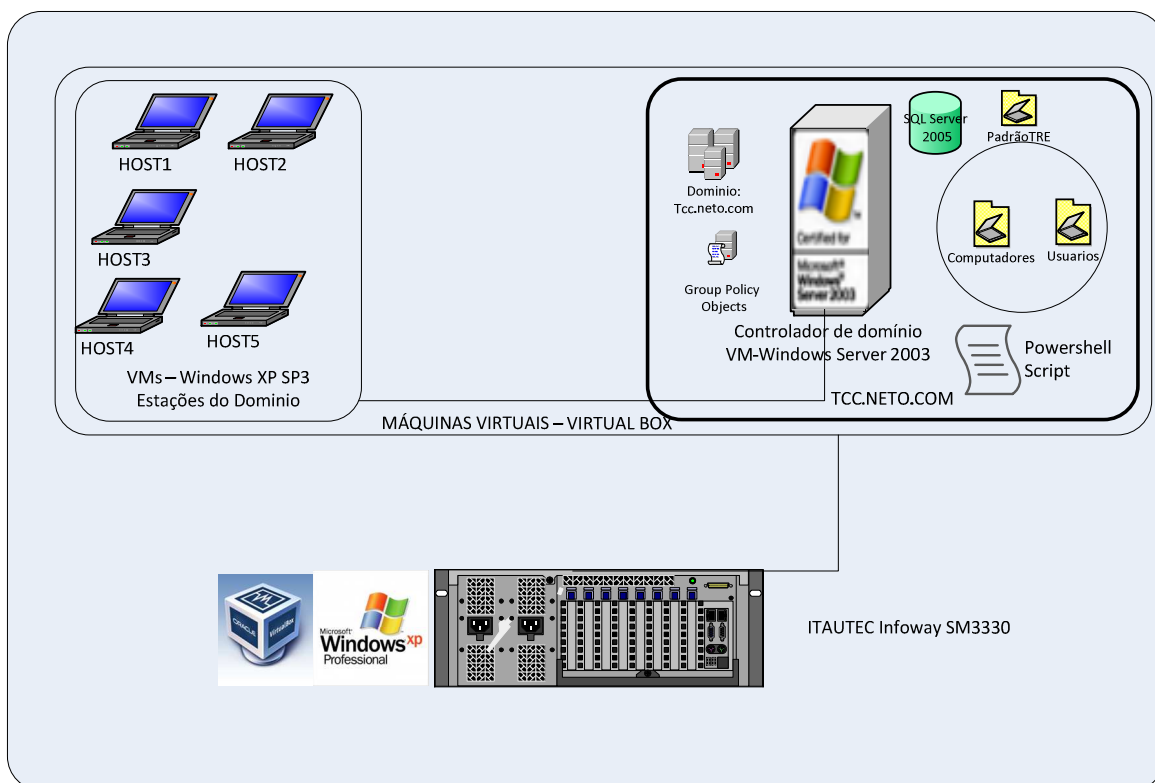
#### **4.8. Cenário de testes e Resultados**

Para realização deste trabalho foi construída infraestrutura exclusiva no ambiente de Rede do TRE-TO, com o objetivo de evitar o uso do ambiente de produção do Tribunal para implementação da proposta e evitar impactos desnecessários nas atividades demandantes da rede principal. Tal medida vai ao encontro das boas práticas que devem ser adotadas para implementação de novas tecnologias em ambiente de produção. As quais norteiam para a existência de ambiente similar para realização prévia dos testes e para colocar em funcionamento, em fase experimental, as inovações tecnológicas, bem como implementações de novas rotinas de trabalho que provoquem impacto no desempenho da rede, tanto na comunicação como em processamento.

##### **4.8.1. Infraestrutura do cenário de testes**

A infraestrutura para os testes foi construída em um ambiente virtual similar ao existente em ambiente de rede Windows em produção do Tribunal, conforme descrito a seguir:

- Todos os testes do cenário foram realizados utilizando a infraestrutura construída da seguinte forma: 01 máquina física hospedeira das virtuais com Windows XP Service Pack 3 e 06 máquinas virtuais criadas de forma similar ao ambiente de produção da rede Windows do Tribunal, sendo 01 VM com Windows 2003 Server SP2 como Controlador do Domínio “TCC.NETO.COM”, com Sistema de Banco de Dados SQL Server 2005, a ferramenta de gerenciamento de políticas do *Active Directory*: GPMC (*Group Policy Manager Console*), a ferramenta PowerShell Script ISE 2.0 e Microsoft .NET Framework v. 3.5; e 05 VMs Windows XP SP3 como estações de trabalho simulando ambiente de trabalho dos usuários da rede, conforme demonstrado na figura 28.



**Figura 28: Infraestrutura do cenário de testes.**

No ambiente virtual foram realizados os testes com objetivo de avaliar os conceitos defendidos nos estudos teóricos quanto à gerência de rede por meio de aplicação de diretivas de auditoria sobre os recursos e sobre as ações dos usuários da rede. Para realização dos testes foram criados cenários de forma a atender cada item definido na Política de Auditoria de Segurança da Instituição. A seção seguinte serão abordados os procedimentos de configuração do ambiente virtual utilizado.

#### **4.8.1.1. Configuração do Ambiente Virtual**

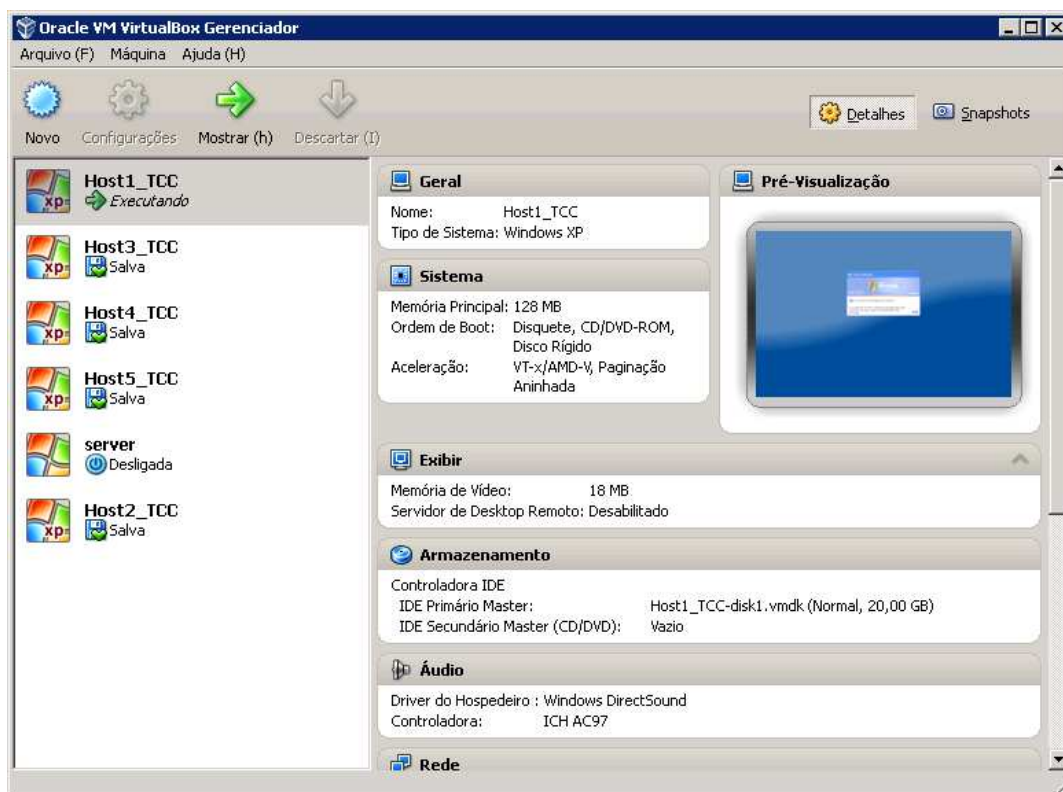
A configuração do ambiente virtual utilizado no cenário de testes ocorreu da seguinte forma:

- **Configuração da máquina hospedeira do ambiente virtual:** nessa máquina foi instalado o Windows XP SP3 apenas como base para o ambiente de virtualização. A configuração de rede dessa máquina foi feita de modo a pertencer à rede do Tribunal para possibilitar acesso à internet.
- **Ambiente de Virtualização:** para virtualização do ambiente de rede exclusivo para os testes (Controlador de Domínio e as estações de trabalho) foi utilizado o Sistema

Oracle VM Virtual Box, versão 4.3.12 livre para utilização acadêmica. As máquinas virtuais foram criadas com a seguinte configuração, conforme Figura 28:

- Configurações das Máquinas virtuais com Windows XP SP3 utilizadas como Estação de Trabalho:
  - Nomes das VMs: Host1\_TCC, Host2\_TCC, Host3\_TCC, Host4\_TCC e Host5\_TCC.
  - Nome dos hosts: host1, host2, host3, host4 e host5.
  - HD de 20 Gb;
  - Memória RAM de 128Mb e memória de Vídeo de 18Mb.
- Configuração da máquina Virtual com Windows 2003 Server:
  - Nome da VM: server;
  - Nome do host: Server;
  - HD de 40 Gb;
  - Memória RAM de 1,5G e memória de Vídeo de 18Mb.

A Figura 29, a seguir, apresenta o ambiente virtualizado.



**Figura 29: Ambiente Virtualizado com VirtualBox.**

#### 4.8.1.2. Configuração do Controlador de Domínio

O Controlador de Domínio da Rede Windows foi construído em uma Máquina Virtual com Windows 2003 Server e foi configurado da seguinte forma:

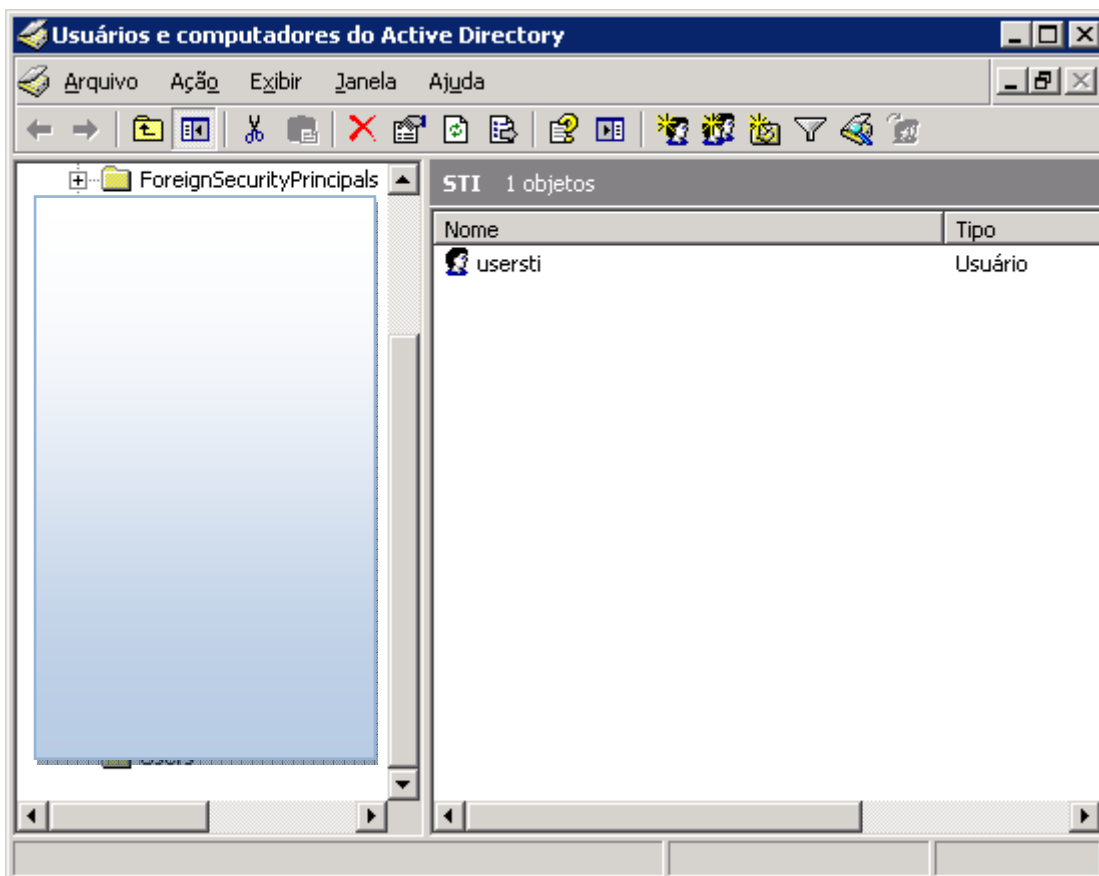
- A estrutura do AD da rede Windows foi configurada contemplando de forma resumida parte da estrutura do Tribunal, a qual é organizada em Unidades Organizacionais – OUs para facilitar a gerência de segurança implementada por meio de *Group Policy Objects*, conforme consta na Tabela 33.

**Tabela 33: Unidades Organizacionais do Active Directory**

PadrãoTRE											
Computadores					Usuários						
PRES	GABs	DG				PRES	GABs	DG			
		STI	SGP	SJI	SADOR			STI	SGP	SJI	SADOR

- Descrição das unidades organizacionais:
  - OU PadrãoTRE: OU geral que contempla todas as OUs de computadores e Usuários
  - OU Computadores: Dentro dessa OU foram criadas novas OUs seguindo o organograma do Tribunal, em cada OU setorial estão inseridos os respectivos computadores.
  - OU Usuários: Dentro dessa OU foram criadas novas OUs seguindo o organograma do Tribunal, em cada OU setorial estão inseridos os respectivos usuários.
- O controlador de Domínio está incorporando ainda a função de servidor de banco de dados, tendo instalado como gerenciador do Banco o sistema SQLServer 2005.
- Por último no controlador de domínio foi instalada e configurada a ferramenta de construção de Script PowerShell ISE 2.5. Cujas execuções dos scripts também ocorreram a partir do Servidor.

A figura 30 demonstra como ficou a estrutura do AD do ambiente de testes.



**Figura 30: Estrutura do AD do ambiente de rede do cenário de testes.**

No tópico seguinte será demonstrado e explicado o Script WMI utilizado para Coleta de informações dos registros de *logs* de eventos de auditoria da rede.

#### **4.8.2. SCRIPT WMI**

Assim como foi construída infraestrutura única para execução de todos os testes do cenário, também foi construído um único Script de Coleta de informações dos registros de *logs* de eventos de auditoria da rede, visto que os *logs* de eventos de auditoria pertencem a Categoria de segurança (*Security*) e todos os registros são feitos utilizando as propriedades da Classe WMI *Win32\_NTLogEvent*.

A seguir será mostrado e explicado por meio das Figuras 31 a 36 o SCRIPT que foi construído para o fim específico de fazer as coletas dos Registros de *Logs* de Eventos de Auditoria e a respectiva gravação no Banco de Dados *SqlServer 2.5*. O Script completo consta no **Apendice B**.



```

5
6 $ArrComputers = "host1" # , "host2", "host3", "host4", "host5"
7
8 foreach ($Computer in $ArrComputers)
9 {
10
11 $name = Get-WmiObject -Class Win32_NTLogEvent -Namespace "root\cimv2" -Computer $Computer `
12 | Where-Object {$_.Category -Match "2" -and $_.EventCode -eq "517" -or $_.EventCode -eq "528"
13 -or $_.EventCode -eq "529" -or $_.EventCode -eq "530" -or $_.EventCode -eq "531" -or $_.EventCode -eq "532"
14 -or $_.EventCode -eq "533" -or $_.EventCode -eq "534" -or $_.EventCode -eq "540" -or $_.EventCode -eq "550"
15 -or $_.EventCode -eq "552" -or $_.EventCode -eq "560" -or $_.EventCode -eq "563" -or $_.EventCode -eq "564"
16 -or $_.EventCode -eq "611" -or $_.EventCode -eq "612" -or $_.EventCode -eq "620" -or $_.EventCode -eq "624"
17 -or $_.EventCode -eq "627" -or $_.EventCode -eq "628" -or $_.EventCode -eq "630" -or $_.EventCode -eq "632"
18 -or $_.EventCode -eq "633" -or $_.EventCode -eq "636" -or $_.EventCode -eq "637"}
19
20 function Do-FilesInsertRowByRow ([Data.SqlClient.SqlConnection] $OpenSqlConnection) {
21
22 write-host "EventCode" + $name.RecordNumber
23
24 $sqlCommand = New-Object System.Data.SqlClient.SqlCommand
25 $sqlCommand.Connection = $sqlConnection
26
27 $sqlCommand.CommandText = "SET NOCOUNT ON; " +
28 "INSERT INTO dbo.Eventos_Auditoria (Identificador_Evento, Categoria, Computador, Dominio, Codigo_Evento,
29 Tipo_Evento, Arquivo_log, Numero_Registro, Origem, Data, Hora, Usuario, Tipo_Logon, IP_Origem, Porta_Origem) " +
30 "VALUES (@Identificador_Evento, @Categoria, @Computador, @Dominio, @Codigo_Evento, @Tipo_Evento, @Arquivo_log,
31 @Numero_Registro, @Origem, @Data, @Hora, @Usuario, @Tipo_Logon, @IP_Origem, @Porta_Origem); " +
32 "SELECT SCOPE_IDENTITY() as [InsertedID]; "
33
34 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Identificador_Evento", [Data.SqlDbType]::bigint)))
35 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Categoria", [Data.SqlDbType]::bigint)))
36 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Computador", [Data.SqlDbType]::NVarChar, 50)))
37 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Dominio", [Data.SqlDbType]::NVarChar, 50)))
38 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Codigo_Evento", [Data.SqlDbType]::bigint)))
39 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Tipo_Evento", [Data.SqlDbType]::NVarChar, 250)))
40 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Arquivo_log", [Data.SqlDbType]::NVarChar, 500)))
41 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Numero_Registro", [Data.SqlDbType]::bigint)))
42 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Origem", [Data.SqlDbType]::NVarChar, 50)))
43 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Data", [Data.SqlDbType]::NVarChar, 50)))
44 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Hora", [Data.SqlDbType]::NVarChar, 50)))
45 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Usuario", [Data.SqlDbType]::NVarChar, 50)))
46 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Tipo_Logon", [Data.SqlDbType]::NVarChar, 50)))
47 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@IP_Origem", [Data.SqlDbType]::NVarChar, 50)))
48 $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Porta_Origem", [Data.SqlDbType]::NVarChar, 50)))
49
50

```

**Figura 31: Script de Coleta e Gravação de Logs de Auditoria**

### Entendendo parte do Script da Figura 31:

Linha 6: A variável *\$arrComputers* recebe a coleção (*Array*) dos computadores (*hosts*) da rede, de onde serão coletadas as informações de logs de eventos.

Linha 8: Foi declarado um loop tipo *ForEach* e este laço é repetido para cada item da coleção de computadores *\$ArrComputers* por meio da variável *\$Computer*.

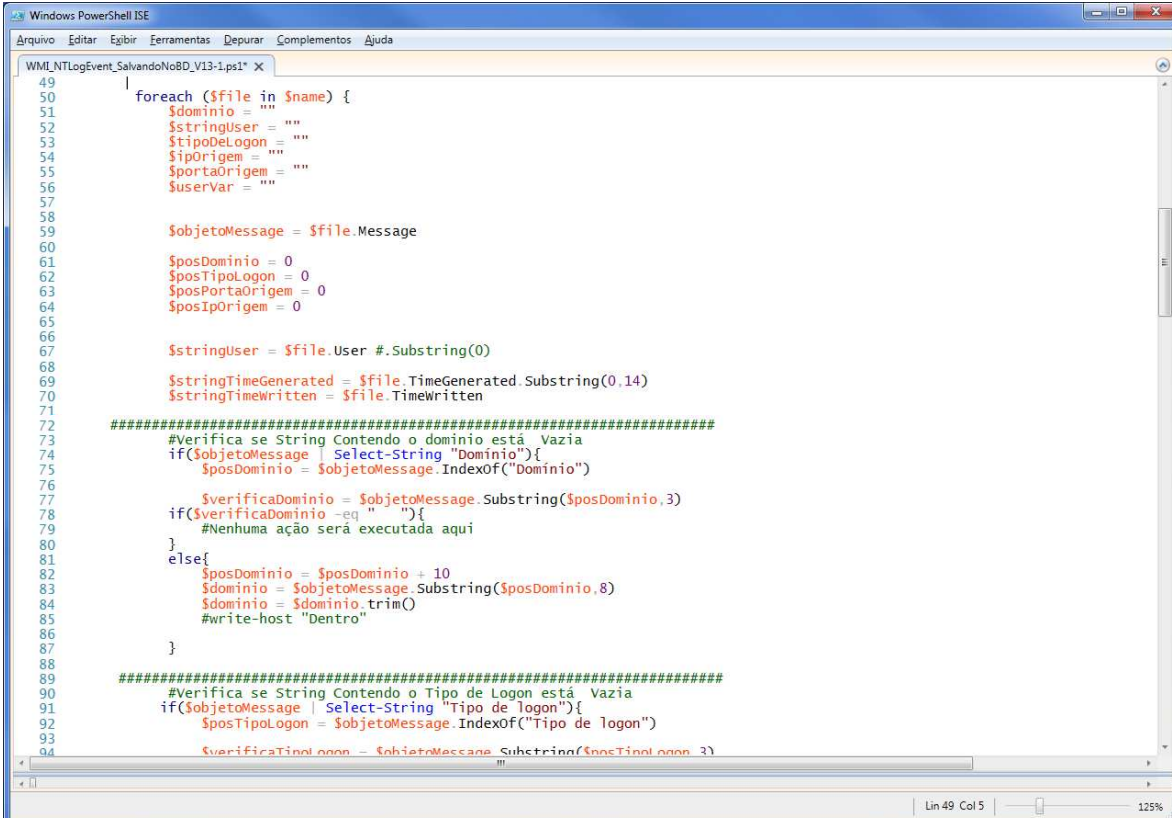
Linha 11: A variável *\$name*, do tipo *Array*, recebe objetos WMI provenientes da Classe *Win32\_NTLogEvent*, pertencentes a cada computador da coleção e que sejam da categoria 2 (*security*) e que tenha os *EventCode* declarados na clausula *where*. Esses *EventCodes* são os objetos cujas propriedades serão armazenadas em banco de dados centralizado, visto que são os objetos gerados pelas ações dos usuários previstas na política de auditoria.

Linha 20: Ocorre a declaração da função que será chamada posteriormente para fazer a inserção linha a linha no banco de dados;

Linha 24: A variável *sqlCommand* recebe uma nova instância da classe *System.Data.SqlClient.SqlCommand*, a qual contém os comandos/propriedades do banco;

Linha 27: Acontece a inicialização de nova instância da classe `SqlCommand` com o texto (string) da query “Insert into...”, contendo as propriedades da tabela do banco e a definição os valores que serão atribuídos às propriedades.

Linhas 34 a 48: Acontece a declaração dos parâmetros das propriedades do banco, respeitando os mesmos tipos declarados quando da criação do banco.



```
49 |
50 |     foreach ($file in $name) {
51 |         $dominio = ""
52 |         $stringUser = ""
53 |         $tipoDeLogon = ""
54 |         $ipOrigem = ""
55 |         $portaOrigem = ""
56 |         $userVar = ""
57 |
58 |
59 |         $objetoMessage = $file.Message
60 |
61 |         $posDominio = 0
62 |         $posTipoLogon = 0
63 |         $posPortaOrigem = 0
64 |         $posIpOrigem = 0
65 |
66 |
67 |         $stringUser = $file.User #.Substring(0)
68 |
69 |         $stringTimeGenerated = $file.TimeGenerated.Substring(0,14)
70 |         $stringTimeWritten = $file.TimeWritten
71 |
72 |         #####
73 |         #Verifica se String Contendo o dominio está vazia
74 |         if($objetoMessage | Select-String "Dominio"){
75 |             $posDominio = $objetoMessage.IndexOf("Dominio")
76 |
77 |             $verificaDominio = $objetoMessage.Substring($posDominio,3)
78 |             if($verificaDominio -eq " "){
79 |                 #Nenhuma ação será executada aqui
80 |             }
81 |             else{
82 |                 $posDominio = $posDominio + 10
83 |                 $dominio = $objetoMessage.Substring($posDominio,8)
84 |                 $dominio = $dominio.Trim()
85 |                 #write-host "Dentro"
86 |             }
87 |
88 |
89 |         #####
90 |         #Verifica se String Contendo o Tipo de Logon está vazia
91 |         if($objetoMessage | Select-String "Tipo de logon"){
92 |             $posTipoLogon = $objetoMessage.IndexOf("Tipo de logon")
93 |
94 |             $verificaTipoLogon = $objetoMessage.Substring($posTipoLogon,3)
```

Figura 32: Script de Coleta e Gravação de Logs de Auditoria. Cont...

### Entendendo parte do Script da Figura 32:

Linha 50: A variável `$name` na linha 11 recebeu a coleção de objetos WMI provenientes da Classe `Win32_NTLogEvent`, pertencentes a cada computador da coleção `$ArrComputer`. Na linha 50 para cada objeto da coleção `$name` irá executar as ações que estão dentro do laço “`foreach`”. A variável `$file` representará cada objeto do Array `$name` dentro do loop.

Linhas 51 a 56: São declaradas as variáveis que receberão *substrings* a serem extraídas da propriedade “Message”, *tipo string*, de cada objeto, a qual corresponde à descrição de cada evento. Essas variáveis irão receber informações valiosas para auditoria, como: Domínio, Usuário, Tipo de Logon, IP de origem e Porta de Origem.

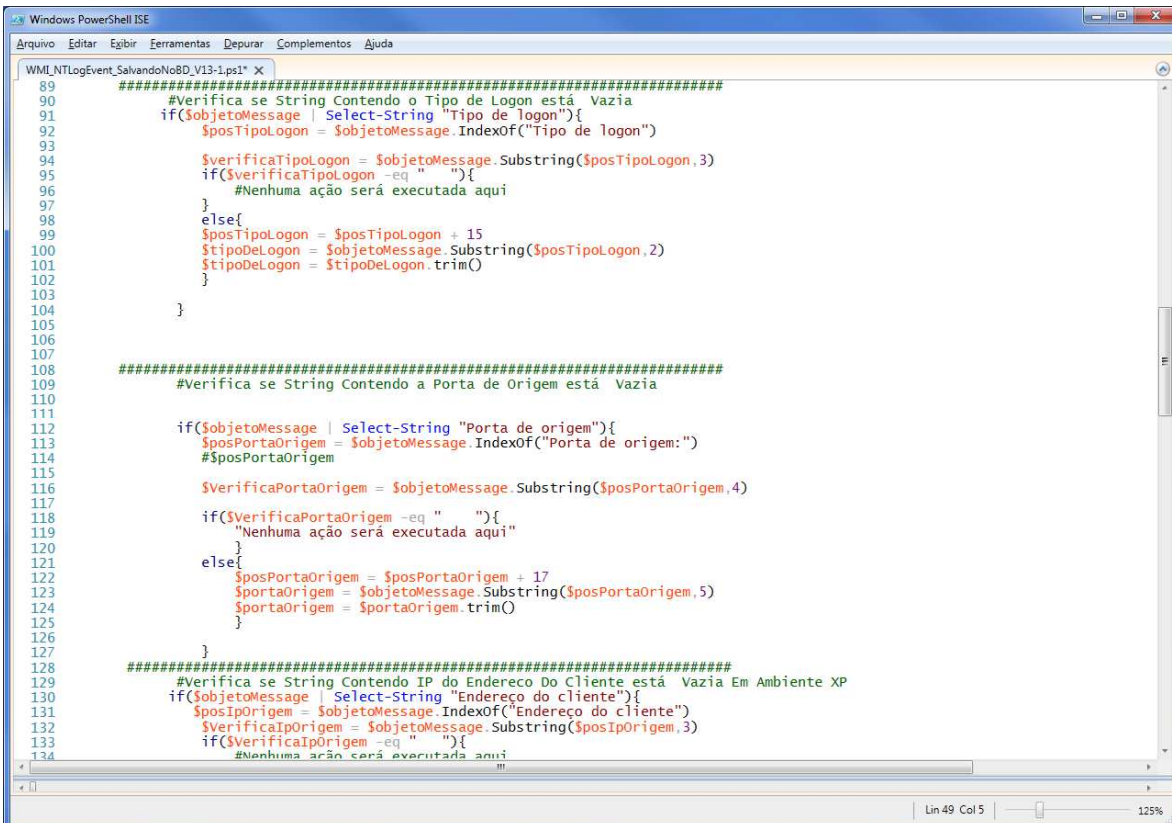
Linha 59: Instanciada a variável que receberá a *string* completa da propriedade “Message”

Linhas 61 a 64: São declaradas variáveis estáticas zeradas para evitar receber lixo em tempo de execução antes de atribuição de valor às mesmas. Essa é uma particularidade do PowerShell, quando o mesmo tem execução interrompida ele continua guardando as variáveis em cache e quando retorna a execução poderá ter lixo. Por isso, é recomendado zerar as variáveis estáticas.

Linha 67: Cada objeto (eventcode) há a propriedade user = usuário que praticou a ação geradora do evento. A variável \$stringuser irá receber essa propriedade.

Linhas 69 e 70: As variáveis \$stringTimeGenerated e \$stringTimeWritten irão receber a data e hora de geração e gravação em arquivo de log de evento, respectivamente.

Linhas 73 a 87: Ocorre o tratamento da *string* da propriedade *Message* para extrair a *substring* com a informação de domínio da rede.



```
Windows PowerShell ISE
Arquivo Editar Exibir Ferramentas Depurar Complementos Ajuda
WML_NTLogEvent_SalvandoNoRD_V13-1.ps1 X
89 #####
90 #Verifica se String Contendo o Tipo de Logon está Vazia
91 if($objetoMessage | Select-String "Tipo de logon"){
92     $posTipoLogon = $objetoMessage.IndexOf("Tipo de logon")
93
94     $verificaTipoLogon = $objetoMessage.Substring($posTipoLogon,3)
95     if($verificaTipoLogon -eq " "){
96         #Nenhuma ação será executada aqui
97     }
98     else{
99         $posTipoLogon = $posTipoLogon + 15
100         $tipoDeLogon = $objetoMessage.Substring($posTipoLogon,2)
101         $tipoDeLogon = $tipoDeLogon.trim()
102     }
103 }
104
105
106
107
108 #####
109 #Verifica se String Contendo a Porta de Origem está Vazia
110
111
112 if($objetoMessage | Select-String "Porta de origem"){
113     $posPortaOrigem = $objetoMessage.IndexOf("Porta de origem:")
114     # $posPortaOrigem
115
116     $verificaPortaOrigem = $objetoMessage.Substring($posPortaOrigem,4)
117
118     if($verificaPortaOrigem -eq " "){
119         "Nenhuma ação será executada aqui"
120     }
121     else{
122         $posPortaOrigem = $posPortaOrigem + 17
123         $portaOrigem = $objetoMessage.Substring($posPortaOrigem,5)
124         $portaOrigem = $portaOrigem.trim()
125     }
126 }
127
128 #####
129 #Verifica se String Contendo IP do Endereço do Cliente está Vazia Em Ambiente XP
130 if($objetoMessage | Select-String "Endereço do cliente"){
131     $posIpOrigem = $objetoMessage.IndexOf("Endereço do cliente")
132     $verificaIpOrigem = $objetoMessage.Substring($posIpOrigem,3)
133     if($verificaIpOrigem -eq " "){
134         #Nenhuma ação será executada aqui
135     }
136 }
```

Figura 33: Script de Coleta e Gravação de *Logs* de Auditoria. Cont...

### Entendendo parte do Script da Figura 32 e 33:

Linhas 90 a 127: Acontece o tratamento da *String* “Message” para extrair as informações de “Tipo de Logon” e “Porta de Origem”.

```
128 #####
129 #Verifica se String Contendo IP do Endereco Do Cliente está vazia Em Ambiente XP
130 if($objetoMessage | Select-String "Endereço do cliente"){
131     $posIpOrigem = $objetoMessage.IndexOf("Endereço do cliente")
132     $verificaIpOrigem = $objetoMessage.Substring($posIpOrigem,3)
133     if($verificaIpOrigem -eq " "){
134         #Nenhuma ação será executada aqui
135     }
136     else{
137         $posIpOrigem = $posIpOrigem + 20
138         $ipOrigem = $objetoMessage.Substring($posIpOrigem, 15)
139         $ipOrigem = $ipOrigem.trim()
140     }
141 }
142 }
143 }
144 #####
145 #Verifica se String Contendo IP do Endereco Do Cliente está vazia Em Ambiente Server
146 if($objetoMessage | Select-String "Endereço de rede de origem"){
147     $posIpOrigem = $objetoMessage.IndexOf("Endereço de rede de origem:")
148     $verificaIpOrigem = $objetoMessage.Substring($posIpOrigem,4)
149     if($verificaIpOrigem -eq " "){
150         "Nenhuma ação será executada aqui"
151     }
152     else{
153         $posIpOrigem = $posIpOrigem + 28
154         $ipOrigem = $objetoMessage.Substring($posIpOrigem, 14)
155         $ipOrigem = $ipOrigem.trim()
156     }
157 }
158 }
159 #####
160 #Verifica se Propriedade com nome do usuario está vazia. Em caso afirmativo
161 # Coleta usuário do Campo Message
162 #"Valor Atual de User"+ $stringUser
163 #####
164
165 $dataString = $file.TimeGenerated
166
167 $ano = $dataString.Substring(0,4)
168 $mes = $dataString.Substring(4,2)
169 $dia = $dataString.Substring(6,2)
170 $hora = $dataString.Substring(8,2)
171 $min = $dataString.Substring(10,2)
172 $sec = $dataString.Substring(12,2)
173
```

**Figura 34: Script de Coleta e Gravação de Logs de Auditoria. Cont...**

### Entendendo parte do Script da Figura 34:

Nas linhas 128 a 158 acontece o tratamento da *String* da propriedade *Message* para obter o IP de origem do evento coletado do Windows XP e Windows 2003 server, respectivamente. Foi precisa fazer tratamento separados porque não houve padronização de *substring* para o mesmo tipo de informação extraída do WinXP e Win2003 Server.

O processo de extração das substrings teve que seguir o seguinte procedimento:

- Primeiro verificou se a *substring* “Dominio” existe dentro da *String*(linha 74);
- em seguida foi guardado na variável *\$posDominio* a posição da palavra *Dominio*;
- Na linha 98 e 99 foi verificado se as três posições após a palavra *Dominio* estão vazias;



- Caso não estejam vazias, nas linhas 103 a 105 foi atribuído à variável domínio o nome do domínio e excluída os espaços após o nome por meio da função trim()).

Esses procedimentos foram repetidos para extração das demais substrings úteis para a auditoria;

```

165
166     $dataString = $file.TimeGenerated
167
168     $ano = $dataString.Substring(0,4)
169     $mes = $dataString.Substring(4,2)
170     $dia = $dataString.Substring(6,2)
171     $hora = $dataString.Substring(8,2)
172     $min = $dataString.Substring(10,2)
173     $seg = $dataString.Substring(12,2)
174
175     $dataTratada = $ano + "/" + $mes + "/" + $dia + " " + $hora + ":" + $min + ":" + $seg
176
177     $sqlCommand.Parameters[0].Value = $file.EventIdentifier
178     $sqlCommand.Parameters[1].Value = $file.Category
179     $sqlCommand.Parameters[2].Value = $file.ComputerName
180     $sqlCommand.Parameters[3].Value = $dominio
181     $sqlCommand.Parameters[4].Value = $file.EventCode
182     $sqlCommand.Parameters[5].Value = $file.EventType
183     $sqlCommand.Parameters[6].Value = $file.LogFile
184     $sqlCommand.Parameters[7].Value = $file.RecordNumber
185     $sqlCommand.Parameters[8].Value = $file.SourceName
186     $sqlCommand.Parameters[9].Value = $dataTratada
187     $sqlCommand.Parameters[10].Value = $stringTimeWritten
188     $sqlCommand.Parameters[11].Value = $stringUser
189     $sqlCommand.Parameters[12].Value = $tipoDeLogon
190     $sqlCommand.Parameters[13].Value = $ipOrigem
191     $sqlCommand.Parameters[14].Value = $portaOrigem
192
193     # Run the query and get the scope ID back into $InsertedID
194
195     $InsertedID = $sqlCommand.ExecuteScalar()
196     # Write to the console.
197     "Inserido novo registro com ID $InsertedID para o LOG " + $file.EventIdentifier
198 }
199
200 }
201 }
202
203 # Abre conexão com o banco
204 $DBServer = "SERVER\TCC"
205 $DBName = "tcc"
206 $sqlConnection = New-Object System.Data.SqlClient.SqlConnection
207 $sqlConnection.ConnectionString = "Server=$DBServer;Database=$DBName;Integrated Security=True;"
208 $sqlConnection.Open()
209
210 # Quit if the SQL connection didn't open properly

```

**Figura 35: Script de Coleta e Gravação de Logs de Auditoria. Cont...**

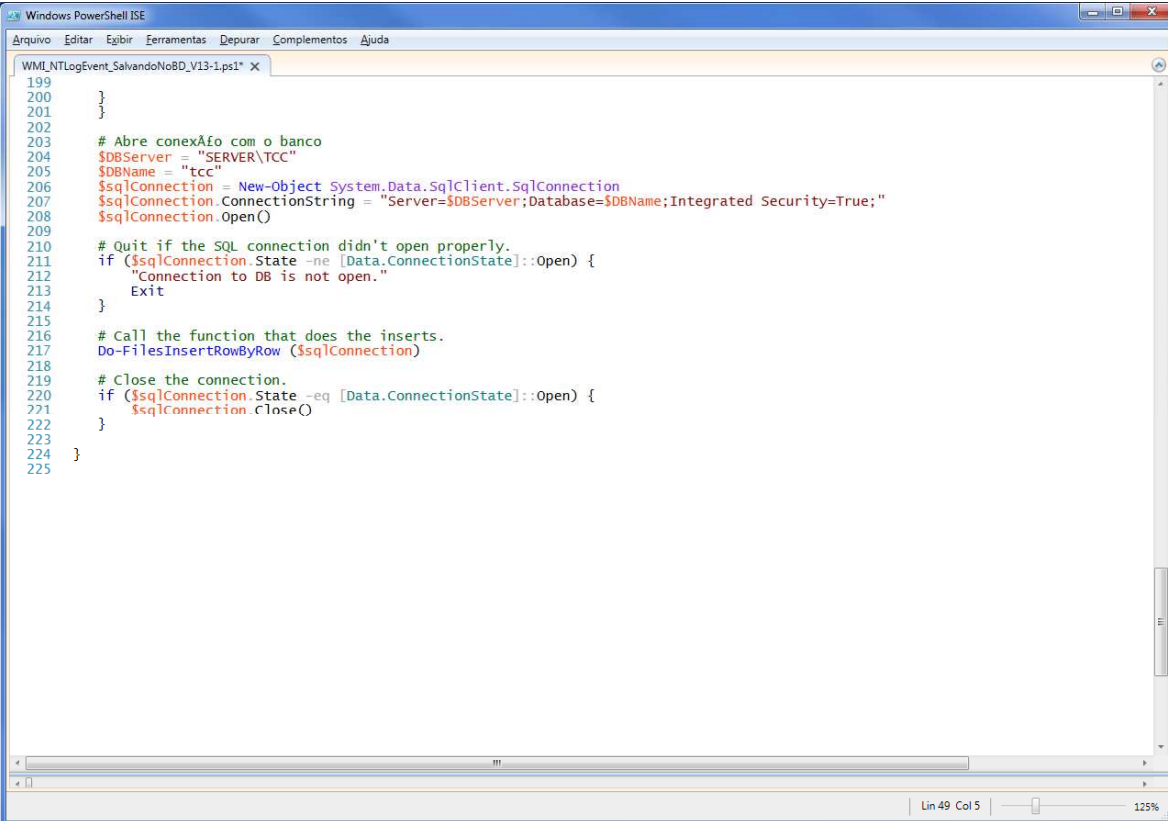
### Entendendo parte do Script da Figura 35:

Linhas 166 a 175: Acontece o tratamento da *String DateTime* para inserção da data e hora de geração do evento no banco de dados. De toda a *String* são utilizados somente as 14 primeiras posições, conforme declarado na Linha 69.

Linhas 177 a 191: Nessas linhas os valores das propriedades de cada *EventCode*, cujos parâmetros já foram declarados, estão sendo atribuídos ao banco de dados, sendo que cada coluna da tabela *Eventos\_Auditoria* irá receber o valor correspondente.

Linhas 195: Acontece a atribuição da função *ExecuteScalar()*, a qual é a responsável pela inserção dos registros na tabela do banco, obedecendo as atribuições dos

valores de cada coluna conforme ordem definida nas linhas 177 a 191, as quais corresponde às colunas da Tabela *Eventos\_Auditoria*



```
199
200     }
201 }
202
203 # Abre conexão com o banco
204 $DBServer = "SERVER\TCC"
205 $DBName = "tcc"
206 $sqlConnection = New-Object System.Data.SqlClient.SqlConnection
207 $sqlConnection.ConnectionString = "Server=$DBServer;Database=$DBName;Integrated Security=True;"
208 $sqlConnection.Open()
209
210 # Quit if the SQL connection didn't open properly.
211 if ($sqlConnection.State -ne [Data.ConnectionState]::Open) {
212     "Connection to DB is not open."
213     Exit
214 }
215
216 # Call the function that does the inserts.
217 Do-FilesInsertRowByRow ($sqlConnection)
218
219 # Close the connection.
220 if ($sqlConnection.State -eq [Data.ConnectionState]::Open) {
221     $sqlConnection.Close()
222 }
223
224 }
225 }
```

**Figura 36: Script de Coleta e Gravação de Logs de Auditoria. Cont...**

### **Entendendo parte do Script da Figura 36:**

Linhas 203 a 208: Acontece o processo de conexão com o banco. Primeiro as \$DBServer, \$DBName recebem informações do banco para conexão. Após a variável \$sqlConnection recebe uma nova instancia da classe System.Data.SqlClient.SqlConnection. Na linha 207 a propriedade ConnectionString recebe as informações para conexão ao banco: Servidor do banco e nome da instância. Na linha 208 é invocada a função Open() para proceder a abertura do banco para inserção dos dados.

Linhas 2011 a 222: Foi construído um condicional para verificar previamente se o banco está aberto para então proceder com inserção dos dados no banco e após as inserções a função Close() é chamada para fechar o banco.

Uma vez populado o banco de dados com os registros de *logs* de eventos das ações regulamentadas na Política de Auditoria de Segurança foi possível construir o cenário de testes.

#### **4.8.3. Cenário de testes**

Esta Seção abordará os artefatos que foram construídos no cenário de testes para comprovar a viabilidade de aplicação da Política de Auditoria de Segurança do Tribunal na Rede Windows.

A política de Auditoria de Segurança constante do **APENDICE A** enumera 06 itens de informações que devem ser coletadas da rede e armazenadas de forma centralizado em Banco de Dados para fins de possibilitar consultas para obter informações úteis para uma auditoria sensata e eficaz.

Para cada item passível de auditoria definido na Política de Auditoria de Segurança do TRE-TO foi realizado testes específicos com a finalidade exclusiva de comprovar sua aplicação na rede.

#### **4.8.4. Primeiro teste do cenário e resultado**

Criado para testar se os registros de *logs* dos acessos (*logon*) bem sucedidos nos equipamentos da rede, incluindo o servidor, provenientes da aplicação da diretiva de eventos de *logon*, foram coletados e gravados em banco de dados. Tais registros visa cumprir ao determinado no item 1 da Política de Auditoria de Segurança adotada pelo Tribunal, descrito a seguir:

##### **Item 1: “Registro dos Acessos (*logon*) com êxito nos controladores de domínio e nas estações de trabalho da rede Windows do Tribunal”**

Para atingir o fim esperado pela Política de Auditoria foram utilizados dentre os eventos coletados somente os eventos de códigos: 528, 540, 550, 552. A documentação sobre os códigos dos eventos foi abordada no Trabalho de Estágio (JNLC, 2006). Resumidamente segue a descrição dos códigos dos eventos:

- 528: Um usuário fez *logon* com sucesso em um computador.
- 540: Um usuário fez *logon* com sucesso na rede.

- 550: Mensagem de notificação que pode indicar um possível ataque de negação de serviço.
- 552: Um usuário fez *logon* com sucesso a um computador que usa credenciais explícitas enquanto já havia feito *logon* como um usuário diferente.

Para obter as informações úteis para auditoria referentes ao determinado no item 1 da Política de Auditoria de Segurança adotada pelo Tribunal foram criadas as seguintes consultas:

1. Quais usuários acessaram a estação X em um determinado período a ser definido no ato da consulta localmente e pela rede;
  - Informações necessárias para construção dessa consulta:
    - Host auditado: Host1
    - Banco de dados: TCC
    - Tabela: Eventos\_Auditoria
    - Código de eventos: 528
    - Tipo de *logons*: todos
    - Período: 01/06/2014 a 26/06/2014.
  - Resultado Esperado para essa consulta:
    - O identificador de cada evento;
    - A relação dos usuários que acessaram o computador no período definido;
    - A descrição dos tipos de *logons* utilizados;
    - IP de origem, caso tenha;

A Figura 37 contempla o resultado da 1ª Consulta SQL do 1º Teste referente ao Item 1 da Política de Auditoria adotada pelo Tribunal.



The screenshot shows the Microsoft SQL Server Management Studio interface. The query window contains the following SQL code:

```

select Computador, Usuario, Identificador_Evento, Table_Tipo_Logon.Tipo_Logon, IP_Origem, Data
from Eventos_Auditoria, Table_Tipo_Logon
where Computador = 'HOST1' AND Codigo_Evento = '528' AND Data between '06/01/2014' AND '06/27/2014'
AND
Table_Tipo_Logon.ID = Eventos_Auditoria.Tipo_Logon /*MM/DD/YYYY*/

```

The Results pane displays the following data:

	Computador	Usuario	Identificador_Evento	Tipo_Logon	IP_Origem	Data
23	HOST1	TCC\UserDG	528	Interactive		2014-06-25 23:14:39.000
24	HOST1	TCC\UserDG	528	CachedInteractive		2014-06-25 23:14:39.000
25	HOST1	TCC\usergabs	528	RemoteInteractive		2014-06-25 22:50:33.000
26	HOST1	TCC\usersji	528	RemoteInteractive		2014-06-25 22:45:15.000
27	HOST1	TCC\usersji	528	RemoteInteractive		2014-06-25 22:39:22.000
28	HOST1	TCC\Administrador	528	Interactive		2014-06-25 22:11:58.000
29	HOST1	TCC\Administrador	528	CachedInteractive		2014-06-25 22:11:58.000
30	HOST1	TCC\userpres	528	Interactive		2014-06-25 22:05:06.000
31	HOST1	TCC\userpres	528	CachedInteractive		2014-06-25 22:05:06.000
32	HOST1	TCC\userpres	528	Interactive		2014-06-25 21:35:56.000
33	HOST1	TCC\userpres	528	CachedInteractive		2014-06-25 21:35:56.000
34	HOST1	AUTORIDADE NT\NETWORK SERVICE	528	Service		2014-06-26 22:39:26.000

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (60) tcc 00:00:00 66 rows

**Figura 37: 1ª Consulta de Auditoria do primeiro teste.**

A Consulta da Figura 37 teve como objetivo mostrar os usuários que tiveram acesso ao Host1 no período de 01/06 a 26/06/2014. Foram utilizadas na consulta SQL 02 Tabelas do esquema do banco de dados “TCC”: *Eventos\_Auditoria* e *Table\_Tipo\_Logon*. Esses eventos foram gerados pela ativação da diretiva de *logon* e registrados nos eventos de código 528. A informação é importante para auditoria porque além de saber quais usuários e respectivos departamentos, há a informação de como foi o acesso por meio do “*Tipo\_logon*”.

2. Quais estações foram acessadas pelo usuário Y em um determinado período a ser definido no ato da consulta;
  - Informações necessárias para construção dessa consulta:
    - Usuário auditado: estagiario
    - Banco de dados: TCC
    - Tabela: *Eventos\_Auditoria*
    - Código de eventos: 528
    - Tipo de *logons*: todos
    - Período: 01/06/2014 a 26/06/2014.

- Resultado Esperado da 2ª consulta:
  - O identificador de cada evento;
  - A relação das Estações acessadas pelo usuário “estagiário”;
  - Data e hora dos acessos;
  - Os tipos de *logons* utilizados e a descrição;

A Figura 38 contempla o resultado da 2ª Consulta SQL do 1º Teste referente ao Item 1 da Política de Auditoria adotada pelo Tribunal.

The screenshot displays the Microsoft SQL Server Management Studio interface. The query editor contains the following SQL query:

```

select usuario, Computador, Identificador_Evento, Table_Tipo_Logon.Tipo_Logon, IP_Origem, Numero_Registro
from Eventos_Auditoria, Table_Tipo_Logon
where usuario = 'TCC\estagiario'
AND Codigo_Evento = '528'
AND Data between '06/01/2014'
AND '06/27/2014'
AND Table_Tipo_Logon.ID = Eventos_Auditoria.Tipo_Logon
  
```

The Results pane shows the following data:

	usuario	Computador	Identificador_Evento	Tipo_Logon	IP_Origem	Numero_Registro
1	TCC\estagiario	HOST3	528	Interactive		202
2	TCC\estagiario	HOST3	528	CachedInteractive		200
3	TCC\estagiario	HOST3	528	Interactive		5
4	TCC\estagiario	HOST5	528	Interactive		5

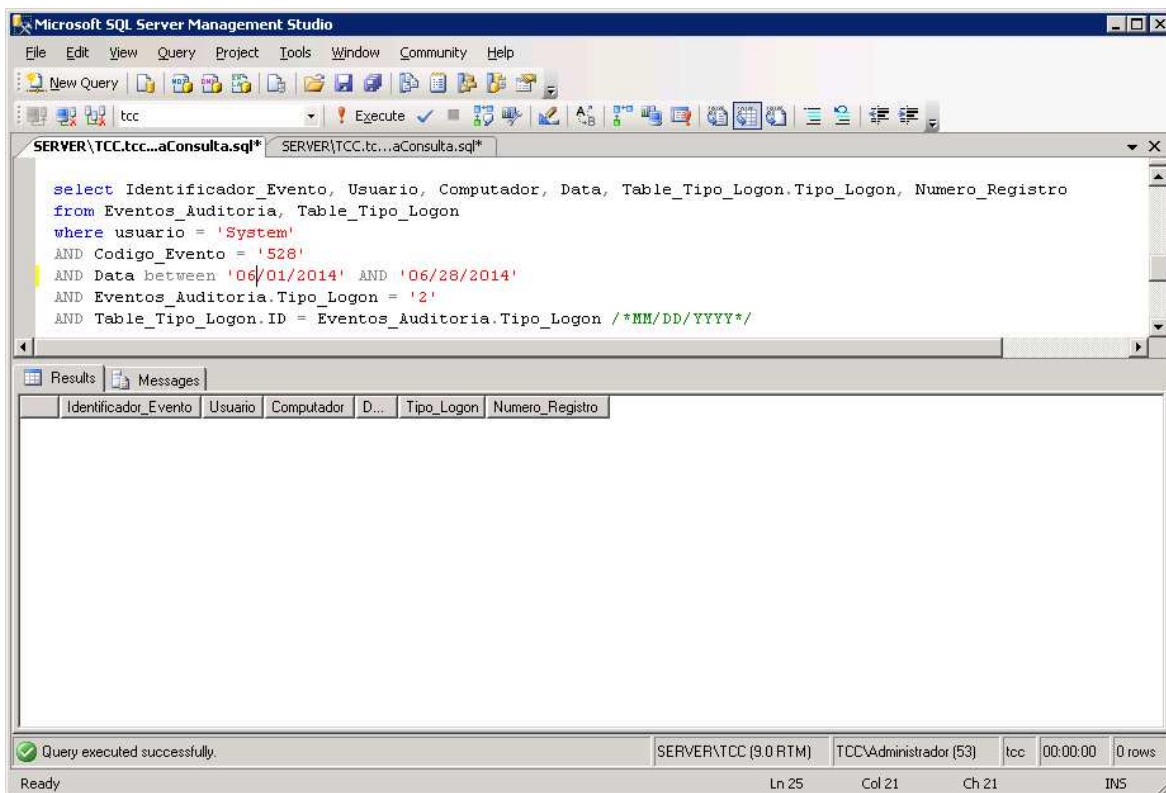
The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (52) tcc 00:00:00 4 rows

**Figura 38: 2ª Consulta de Auditoria do primeiro Teste.**

A Consulta da Figura 38 teve como objetivo rastrear acessos do usuário “estagiário” na rede. Essa informação é fundamental para auditoria quando já há conhecimento de ações indevidas de usuários da Rede. Pelo horário, tipo de *logon* e computadores acessados é possível certificar o perfil do usuário.

3. Quantos acessos “bem sucedidos” ocorreram na rede com usuário “System” sendo o tipo de *logon* igual a 2 (usuário realizou *logon* nesse computador). O que indica que obtiveram a senha do usuário System e realizaram *logon* com o mesmo.
- Informações necessárias para construção dessa consulta:
    - Usuário auditado: System
    - Banco de dados: TCC
    - Tabela: Eventos\_Auditoria
    - Código de eventos: 528
    - Tipo de *logons*: 2
    - Período: 01/06/2014 a 26/06/2014.
  - Resultado Esperado da 3ª consulta:
    - O identificador de cada evento;
    - A relação das Estações acessadas pelo usuário “System”;
    - Data e hora dos acessos;
    - Os tipos de *logons* utilizados e a descrição;

A Figura 39 contempla o resultado da 3ª Consulta SQL do 1º Teste referente ao Item 1 da Política de Auditoria adotada pelo Tribunal.



**Figura 39: 3ª Consulta de Auditoria do primeiro Teste.**

A Consulta da Figura 39 é muito importante para auditoria porque uma vez detectado acesso com Tipo de *logon*=2 para o usuário “System” significa que o usuário foi utilizado indevidamente. Neste caso ações imediatas devem ser tomadas pelo administrador de redes para evitar prejuízos à segurança da instituição. Na consulta não foi encontrado nenhum registro de *logon* com usuário “system”.

4. Quantos acessos através da rede foram realizados nos computadores, utilizando a conta de quais usuários?

- Informações necessárias para construção dessa consulta:
  - Usuário auditado: Todos
  - Banco de dados: TCC
  - Tabela: Eventos\_Auditoria
  - Código de eventos: 528, 540
  - Tipo de *logons*: 3, 10 e 11
  - Período: 01/06/2014 a 26/06/2014.
  
- Resultado Esperado da 4ª consulta:
  - O identificador de cada evento;
  - Nome dos usuários que realizaram acessos pela rede;
  - Nome dos computadores que sofreram acessos pela rede;
  - Data e horário dos acessos.
  - IP\_Origem;
  - Porta\_Origem;

A Figura 40 contempla o resultado da 4ª Consulta SQL do 1º Teste referente ao Item 1 da Política de Auditoria adotada pelo Tribunal.

Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Community Help

SERVER\tcc\tcc...aConsulta.sql\* SERVER\tcc\tcc...aConsulta.sql\* SERVER\tcc\tcc...aConsulta.sql\*

```

select Numero_Registro, Computador, usuario, Data, Identificador_Evento,
       Table_Tipo_Logon.Tipo_Logon, IP_Origem, Porta_Origem
from Eventos_Auditoria, Table_Tipo_Logon
where usuario NOT IN ('AUTORIDADE NT\SYSTEM', 'AUTORIDADE NT\LOGON ANÔNIMO', 'AUTORIDADE NT\NETWORK SERVICE',
                    'AUTORIDADE NT\SERVIÇO DE REDE', 'AUTORIDADE NT\SERVIÇO LOCAL', 'AUTORIDADE NT\LOCAL SERVICE')
AND Codigo_Evento IN ('528', '540') AND Data between '06/01/2014' AND '06/27/2014' AND Table_Tipo_Logon.ID =

```

	Numero_Registro	Computador	usuario	Data	Identificador_Evento	Tipo_Logon	IP_Origem	Porta_Origem
5...	4286	SERVER	TCC\Administrador	2014-06-19 22:50:33.000	528	RemotelInteractive	10.27.67.203	59518
5...	4242	SERVER	TCC\HOST1\$	2014-06-19 22:43:03.000	540	RemotelInteractive	192.168.56.101	
5...	4201	SERVER	TCC\Administrador	2014-06-19 22:31:52.000	528	RemotelInteractive	10.27.67.203	59518
5...	4170	SERVER	TCC\HOST1\$	2014-06-19 22:28:03.000	540	RemotelInteractive	192.168.56.101	
5...	4110	SERVER	TCC\HOST1\$	2014-06-19 22:13:04.000	540	RemotelInteractive	192.168.56.101	
5...	4053	SERVER	TCC\HOST1\$	2014-06-19 21:58:05.000	540	RemotelInteractive	192.168.56.101	
5...	4044	SERVER	TCC\HOST1\$	2014-06-19 21:56:04.000	540	RemotelInteractive	192.168.56.101	
5...	4038	SERVER	TCC\HOST1\$	2014-06-19 21:55:45.000	540	RemotelInteractive	192.168.56.101	
5...	4037	SERVER	TCC\HOST1\$	2014-06-19 21:55:45.000	540	RemotelInteractive	192.168.56.101	
5...	4036	SERVER	TCC\HOST1\$	2014-06-19 21:55:45.000	540	RemotelInteractive	192.168.56.101	
5...	4035	SERVER	TCC\HOST1\$	2014-06-19 21:55:45.000	540	RemotelInteractive	192.168.56.101	
5...	4000	SERVER	TCC\Administrador	2014-06-19 21:51:16.000	540	RemotelInteractive	10.27.64.98	
5...	3993	SERVER	TCC\Administrador	2014-06-19 21:51:16.000	528	RemotelInteractive	127.0.0.1	0

Query executed successfully. SERVER\tcc (9.0 RTM) TCC\Administrador (54) tcc 00:00:00 5240 rows

Ready Ln 1 Col 1 IN5

**Figura 40: 4ª Consulta de Auditoria do primeiro Teste.**

A Consulta da Figura 40 mostra os computadores que tiveram acessos através da rede, bem como os usuários que efetivaram esses acessos. Dessas informações é possível analisar com base no horário do acesso e no computador que tentou fazer o acesso mensurar o risco de intrusão na rede.

#### **4.8.5. Segundo teste do cenário e resultado**

Criado para testar se os registros de *logs* dos acessos (*logon*) “SEM ÊXITO” nos equipamentos da rede, incluindo o servidor, provenientes da aplicação da diretiva de eventos de *logon*, foram coletados e gravados em banco de dados. Tais registros visa cumprir ao determinado no item 2 da Política de Auditoria de Segurança adotada pelo Tribunal, descrito a seguir:

Item 2: “Registro das Tentativas de acesso “sem êxito” nos servidores e em todas as estações de trabalho do Tribunal”.

Para atingir o fim esperado pela Política de Auditoria foram utilizados dentre os eventos coletados somente os eventos de códigos: 529, 530, 531, 532, 533 e 534. Suscintamente segue o significado desses códigos de eventos de auditoria;

- 529: Falha de *logon*. Foi feita uma tentativa de *logon* com nome de usuário desconhecido ou senha inválida.
- 530: Falha de *logon*. Foi feita uma tentativa de *logon* por uma conta de usuário que tentou fazer *logon* fora do horário permitido.
- 531: Falha de *logon*. Foi feita uma tentativa de *logon* usando-se uma conta desabilitada.
- 532: Falha de *logon*. Foi feita uma tentativa de *logon* usando-se uma conta expirada.
- 533: Falha de *logon*. Foi feita uma tentativa de *logon* por um usuário que não tem permissão para fazer *logon* nesse computador.
- 534: Falha de *logon*. O usuário tentou fazer *logon* com um tipo que não é permitido.

Para obter as informações úteis para auditoria referentes ao determinado no item 2 da Política de Auditoria de Segurança adotada pelo Tribunal foram criadas as seguintes consultas:

1. Quais tentativas de acesso “sem êxito” ocorreram localmente ou na rede utilizando nome de usuário desconhecido ou senha inválida em um determinado período a ser definido no ato da consulta?
  - Informações necessárias para construção dessa consulta:
    - Usuário auditado: Todos
    - Banco de dados: TCC
    - Tabela: Eventos\_Auditoria
    - Código de eventos: 529
    - Tipo de *logons*: 3, 10 e 11
    - Período: 01/06/2014 a 26/06/2014.
  - Resultado Esperado da 1ª consulta:
    - O identificador de cada evento;
    - Nome dos usuários que realizaram tentativas de acessos;
    - Nome dos computadores que sofreram tentativas de acessos;
    - Data e horário das tentativas dos acessos.

- IP\_Origem;
- Porta\_Origem;

A Figura 41 contempla o resultado da 1ª Consulta SQL do 2º Teste referente ao Item 2 da Política de Auditoria adotada pelo Tribunal.

The screenshot displays the Microsoft SQL Server Management Studio interface. The query window contains the following SQL code:

```
select Numero_Registro, Computador, Usuario, Data, Identificador_Evento, Tipo_Logon, IP_Origem, Porta_Origem
from Eventos_Auditoria
where Codigo_Evento = '529'
AND Tipo_Logon IN ('3', '10', '11')
AND Data between '06/01/2014' AND '06/26/2014'
```

The Results pane shows the following data:

	Numero_Registro	Computador	Usuario	Data	Identificador_Evento	Tipo_Logon	IP_Origem	Porta_Origem
9	181	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 22:44:39.000	529	10		
10	180	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 22:44:22.000	529	10		
11	179	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 22:44:14.000	529	10		
12	164	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 22:39:10.000	529	10		
13	22	HOST3	AUTORIDADE NT\SYSTEM	2014-06-25 23:18:21.000	529	11		
14	20	HOST3	AUTORIDADE NT\SYSTEM	2014-06-25 23:18:15.000	529	11		
15	18	HOST3	AUTORIDADE NT\SYSTEM	2014-06-25 23:18:10.000	529	11		
16	35386	SERVER	AUTORIDADE NT\SYSTEM	2014-06-24 04:33:44.000	529	3	10.27.67.203	0
17	35385	SERVER	AUTORIDADE NT\SYSTEM	2014-06-24 04:33:44.000	529	3	10.27.67.203	0
18	35384	SERVER	AUTORIDADE NT\SYSTEM	2014-06-24 04:33:41.000	529	3	10.27.67.203	0
19	35383	SERVER	AUTORIDADE NT\SYSTEM	2014-06-24 04:33:41.000	529	3	10.27.67.203	0

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (57) tcc 00:00:00 19 rows. Ready Ln 1 Col 1 IN5

**Figura 41: 1ª Consulta de Auditoria do segundo Teste.**

2. Quais tentativas de acesso “**sem êxito**” ocorreram localmente ou na rede utilizando conta do domínio expirada ou desabilitada ou utilizando conta ativa, porém, sem permissão para fazer *logon* no computador, em um determinado período a ser definido no ato da consulta?

- Informações necessárias para construção dessa consulta:
  - Usuário auditado: todo
  - Banco de dados: TCC
  - Tabela: Eventos\_Auditoria
  - Código de eventos: 531, 532, 533



- Tipo de *logons*: 2, 3, 10 e 11
- Período: 01/06/2014 a 26/06/2014.
- Resultado Esperado da 2ª consulta:
  - O identificador de cada evento;
  - Descrição do Evento;
  - Nome dos usuários que realizaram tentativas de acessos;
  - Nome dos computadores que sofreram tentativas de acessos;
  - Data e horário dos acessos.
  - IP\_Origem;
  - Porta\_Origem;

A Figura 42 contempla o resultado da 2ª Consulta SQL do 2º Teste referente ao Item 2 da Política de Auditoria adotada pelo Tribunal.

The screenshot displays the Microsoft SQL Server Management Studio interface. The query editor contains the following SQL query:

```

select Numero_Registro, Computador, Usuario, Data, Identificador_Evento, Tipo_Logon, IP_Origem, Porta_Origem
from Eventos_Auditoria
where Codigo_Evento IN ('531','532','533')
AND Tipo_Logon IN ('2','3','10','11')
AND Data between '06/01/2014' AND '06/26/2014'

```

The Results pane shows the following data:

	Numero_Registro	Computador	Usuario	Data	Identificador_Evento	Tipo_Logon	IP_Origem	Porta_Origem
1	358	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 23:45:17.000	531	10		
2	357	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 23:45:07.000	531	10		
3	356	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 23:44:39.000	533	10		
4	313	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 23:35:23.000	533	10		
5	295	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 23:23:50.000	533	10		
6	17	HOST2	AUTORIDADE NT\SYSTEM	2014-06-25 23:22:36.000	533	10		
7	48484	SERVER	AUTORIDADE NT\SYSTEM	2014-06-25 23:52:24.000	531	10	10.27.67.203	64300
8	48479	SERVER	AUTORIDADE NT\SYSTEM	2014-06-25 23:51:58.000	533	10	10.27.67.203	64300
9	48467	SERVER	AUTORIDADE NT\SYSTEM	2014-06-25 23:51:43.000	533	10	10.27.67.203	64300

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (59) tcc 00:00:00 9 rows. Ready Ln 29 Col 25 Ch 22 INS

**Figura 42: 2ª Consulta de Auditoria do segundo Teste.**

#### **4.8.6. Terceiro teste do cenário e resultado**

Criado para testar certificar que os registros de *logs* de eventos de auditoria provenientes da aplicação da diretiva de Alteração de Diretiva foram devidamente coletados e gravados



em bancos de dados. Tais registros gravados em banco de dados visam cumprir ao determinado no Item 3 da Política de Auditoria de Segurança adotada pelo Tribunal, descrito a seguir:

**Item 3: “Registro das Alterações das diretivas de auditoria, com registro da hora e do usuário que efetuou a alteração.”**

Essa informação é importante para auditoria por que as alterações indevidas ou não gerenciadas das diretivas podem comprometer a segurança e o funcionamento do servidor DC da rede.

Para atingir o fim esperado pela Política de Auditoria foram utilizados dentre os eventos coletados somente os eventos de códigos: 517, 611, 612, 620, cuja documentação completa sobre os códigos dos eventos foram abordados no Trabalho de Estágio (JNLC, 2006). Suscintamente segue o significado desses códigos de eventos de auditoria:

- 517: O Arquivo de log de segurança foi limpo;
- 611: A relação de confiança com outro domínio foi removida;
- 612: Uma diretiva de auditoria foi alterada;
- 620: A relação de confiança com outro domínio foi modificada;

Para obter as informações úteis para auditoria referentes ao determinado no item 3 da Política de Auditoria de Segurança adotada pelo Tribunal foram criadas as seguintes consultas:

1. Quando houve limpeza do arquivo de log de segurança e qual usuário realizou a limpeza?
  - Informações necessárias para construção dessa consulta:
    - Hosts auditados: Todos
    - Banco de dados: TCC
    - Tabela: Eventos\_Auditoria
    - Código de eventos: 517
    - Período: 01/06/2014 a 26/06/2014.
  - Resultado Esperado para essa consulta:

- O identificador de cada evento;
- Identificação do usuário que fez a limpeza no arquivo de log;
- Unidade Organizacional que pertence o usuário;
- Nomes dos computadores que tiveram *logs* limpos;
- Unidade Organizacional que pertence o computador;
- Data e hora do evento.

A Figura 43 contempla o resultado da 1ª Consulta SQL do 3º Teste referente ao Item 3 da Política de Auditoria adotada pelo Tribunal.

The screenshot displays the Microsoft SQL Server Management Studio interface. The query window shows the following SQL query:

```
select Numero_Registro, Computador, Usuario, Data, Identificador_Evento
from Eventos_Auditoria
where Codigo_Evento = '517'
AND Data between '04/01/2014' AND '06/26/2014'
```

The Results pane shows the following data:

	Numero_Registro	Computador	Usuario	Data	Identificador_Evento
1	1	HOST1	AUTORIDADE NT\SYSTEM	2014-06-25 21:34:27.000	517
2	1	HOST2	AUTORIDADE NT\SYSTEM	2014-06-25 21:31:33.000	517
3	1	HOST3	AUTORIDADE NT\SYSTEM	2014-06-25 21:51:07.000	517
4	1	HOST5	AUTORIDADE NT\SYSTEM	2014-06-25 21:58:28.000	517

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (52) tcc 00:00:00 4 rows

**Figura 43: 1ª Consulta de Auditoria do terceiro Teste.**

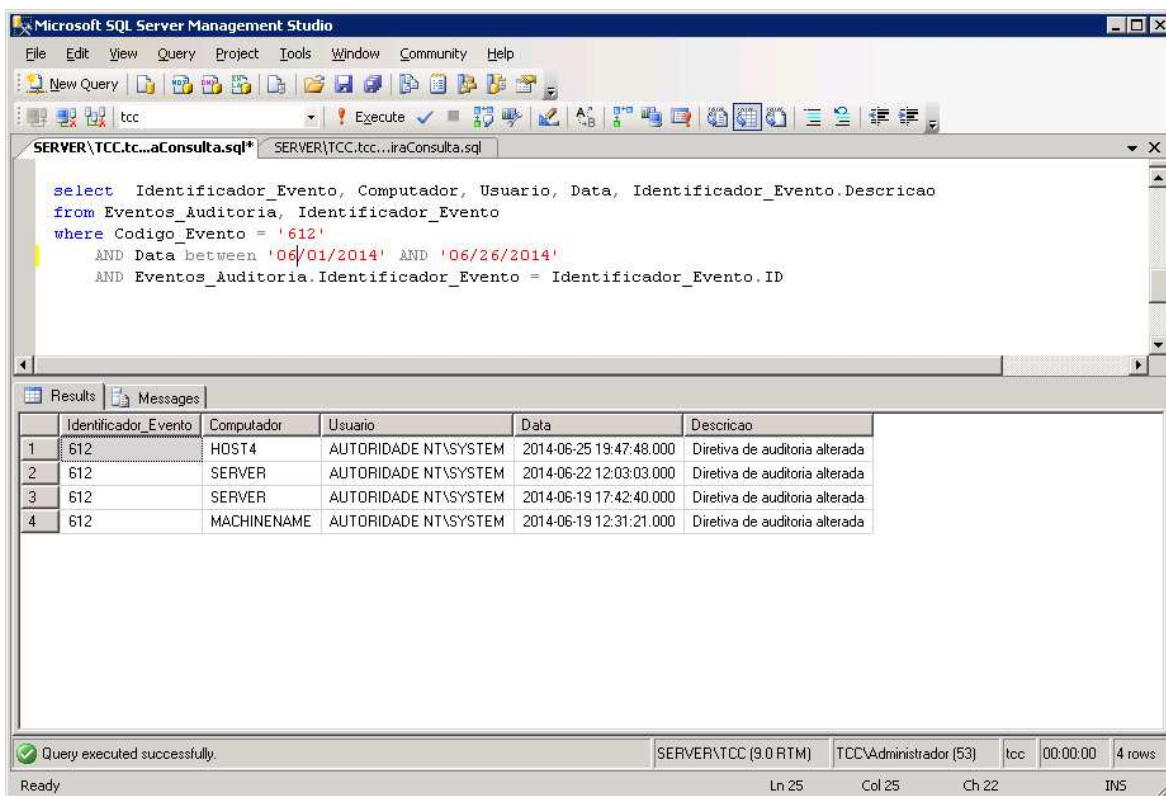
A primeira ação quando algum usuário pratica alguma ação indevida na rede é tentar apagar o rastro de suas ações eliminando os arquivos de registro de *logs*. Nesse contexto essa consulta é muito importante para auditoria porque mesmo não encontrando os *logs* dos eventos fica o registro de quem limpou o arquivo de *logs*, informação esta valiosa para o início de uma investigação mais precisa. No evento mostrado o usuário foi o “System”, porém, na *String* do corpo da descrição do evento

há a informação do usuário real. Precisa somente tratar a *String* na coleta dos registros para extrair a informação do usuário da mesma forma como foi feito com: Tipo de *logon*, Dominio e IP de Origem.

2. Quando ocorreram alterações de diretivas de auditoria e qual usuário realizou as alterações?

- Informações necessárias para construção dessa consulta:
  - Hosts auditados: Todos
  - Banco de dados: TCC
  - Tabela: Eventos\_Auditoria
  - Código de eventos: 612
  - Período: 01/06/2014 a 26/06/2014.
  
- Resultado Esperado para essa consulta:
  - O identificador de cada evento;
  - Descrição do evento;
  - Identificação do usuário que fez alteração das diretivas;
  - Data e hora do evento.

A Figura 44 contempla o resultado da 2ª Consulta SQL do 3º Teste referente ao Item 3 da Política de Auditoria adotada pelo Tribunal.



**Figura 44: 2ª Consulta de Auditoria do terceiro Teste.**

#### **4.8.7. Quarto teste do cenário e resultado**

Criado especificamente para certificar que os registros de *logs* de eventos de auditoria provenientes da aplicação da diretiva de Acessos a Objetos foram devidamente coletados e gravados em bancos de dados. Foi configurado para auditoria somente a pasta “ParecerJuridico” da estação de trabalho localizada na Presidência (host1). Tais registros gravados em banco de dados visam cumprir ao determinado no Item 4 da Política de Auditoria de Segurança adotada pelo Tribunal, descrito a seguir:

**Item 4: “Registro das Alterações e exclusão de pastas e arquivos definidos como críticos e/ou sigilosos pelos Gestores do Tribunal.”**

O referido teste visa detectar alterações e exclusões de pastas e arquivos definidos como críticos e/ou sigilosos pelos Gestores do Tribunal. O procedimento para configuração de auditoria de acesso a objetos consta da documentação do Estágio (JNLC, 2006).

Para realização deste teste foram coletados os *logs* de eventos de segurança dos hosts onde estavam armazenados os objetos auditados, muito embora, a política esteja aplicada em toda a rede.

Para atingir o fim esperado pela Política de Auditoria foram utilizados dentre os eventos coletados somente os eventos de códigos: 560, 563, 564. Suscintamente segue o significado desses códigos de eventos de auditoria:

- 560: Foi concedido acesso a um objeto já existente;
- 563: Foi feita uma tentativa de abertura de um objeto com a intenção de excluí-lo;
- 564: Um objeto protegido foi excluído;

Para obter as informações úteis para auditoria referentes ao determinado no item 4 da Política de Auditoria de Segurança adotada pelo Tribunal foram criadas as seguintes consultas:

1. Quais usuários acessaram objetos auditados em um determinado período a ser definido no ato da consulta?
  - Informações necessárias para construção dessa consulta:
    - Hosts auditados: host1
    - Banco de dados: TCC
    - Tabela: Eventos\_Auditoria
    - Código de eventos: 560
    - Período: 01/06/2014 a 26/06/2014.
  - Resultado Esperado para essa consulta:
    - O identificador de cada evento;
    - Identificação dos usuários que acessaram os objetos auditados;
    - Unidade Organizacional que pertence o usuário;
    - Data e hora do evento.

A Figura 45 contempla o resultado da 1ª Consulta SQL do 4º Teste referente ao Item 4 da Política de Auditoria adotada pelo Tribunal.

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window contains the following SQL query:

```

select Identificador_Evento, Identificador_Evento.Descricao, Computador, Usuario, Data
from Eventos_Auditoria, Identificador_Evento
where Codigo_Evento = '560'
AND Eventos_Auditoria.Identificador_Evento = Identificador_Evento.ID
AND Data between '06/01/2014' AND '06/26/2014'

```

The Results pane displays the following data:

	Identificador_Evento	Descricao	Computador	Usuario	Data
3	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:55.000
4	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:46.000
5	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:46.000
6	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:42.000
7	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:42.000
8	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:31.000
9	560	Concedido acesso a um objeto já existente	HOST1	TCC\aluno2	2014-06-25 23:42:26.000
10	560	Concedido acesso a um objeto já existente	HOST1	TCC\usergabs	2014-06-25 22:52:31.000
11	560	Concedido acesso a um objeto já existente	HOST1	TCC\usersji	2014-06-25 22:47:06.000
12	560	Concedido acesso a um objeto já existente	HOST1	TCC\usersji	2014-06-25 22:47:06.000
13	560	Concedido acesso a um objeto já existente	HOST1	TCC\usersji	2014-06-25 22:47:02.000
14	560	Concedido acesso a um objeto já existente	HOST1	TCC\usersji	2014-06-25 22:47:02.000
15	560	Concedido acesso a um objeto já existente	HOST1	TCC\usersji	2014-06-25 22:46:51.000

**Figura 45: 1ª Consulta de Auditoria do quarto Teste.**

2. Quais usuários praticaram ação de exclusão de objetos auditados em um determinado período a ser definido no ato da consulta?

- Informações necessárias para construção dessa consulta:
  - Hosts auditados: host1
  - Banco de dados: TCC
  - Tabela: Eventos\_Auditoria
  - Código de eventos: 564
  - Período: 01/06/2014 a 26/06/2014.
  
- Resultado Esperado para essa consulta:
  - O identificador de cada evento;
  - Descrição do evento;
  - Objeto acessado;
  - Identificação dos usuários que acessaram os objetos auditados;
  - Unidade Organizacional que pertence o usuário;
  - Data e hora do evento.

Obs.: Para identificar o objeto acessado é necessário verificar o resultado da consulta de acessos aos objetos e comparar o usuário e a data de acesso com a data da exclusão, visto que na descrição do evento 564 não há informação do nome do objeto excluído.

A Figura 46 contempla o resultado da 2ª Consulta SQL do 4º Teste referente ao Item 4 da Política de Auditoria adotada pelo Tribunal.:

The screenshot displays the Microsoft SQL Server Management Studio interface. The query window shows the following SQL code:

```
select Numero_registro, Identificador_Evento.ID, Identificador_Evento.Descricao, Computador, Usuario, Data
from Eventos_Auditoria, Identificador_Evento
where Computador = 'HOST1'
AND Identificador_Evento.ID = '564'
AND Usuario NOT IN ('AUTORIDADE NT\SYSTEM', 'AUTORIDADE NT\NETWORK SERVICE')
AND Data between '04/01/2014' AND '06/26/2014'
order by Data desc
```

The Results pane shows a table with the following data:

	Numero_registro	ID	Descricao	Computador	Usuario	Data
1	399	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:59:41.000
2	391	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:57:40.000
3	387	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:57:37.000
4	385	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:57:37.000
5	383	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:57:37.000
6	381	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:57:37.000
7	379	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:57:37.000
8	376	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:55:39.000
9	373	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:53:37.000
10	370	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:51:36.000
11	367	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:49:35.000
12	363	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:47:34.000
13	360	564	Um objeto protegido foi excluído	HOST1	TCC\Administrador	2014-06-25 23:45:33.000

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (53) tcc 00:00:00 141 rows. Ready Ln 1 Col 1 INS.

**Figura 46: 2ª Consulta de Auditoria do quarto Teste.**

#### **4.8.8. Quinto teste do cenário e resultado**

O Quinto teste foi criado para certificar que os registros de logs de eventos de auditoria provenientes da aplicação da diretiva de Gerenciamento de Contas foram devidamente coletados e gravados em bancos de dados. Tais registros gravados em banco de dados visam cumprir ao determinado nos Itens 5 e 6 da Política de Auditoria de Segurança adotada pelo Tribunal, descritos a seguir:

**Item 5:** “Criação, alteração e exclusão de usuários do domínio.”

**Item 6:** “Concessão de privilégio de administrador de domínio para usuários da rede.”

O referido teste visa obter informações sobre os registros de ações de criação, alteração e exclusão de contas de usuários do Domínio, bem como concessão de privilégio de administrador. Os registros foram coletados dos logs de eventos de segurança do Servidor Controlador de Domínio da rede, uma vez que as contas de usuários são criadas e gerenciadas no controlador do Domínio da rede.

Para atingir o fim esperado pela Política de Auditoria foram utilizados dentre os eventos coletados somente os eventos de códigos: 624, 627, 630 639, 640 e 685. Suscintamente segue o significado desses códigos de eventos de auditoria;

- 624: Uma conta de usuário foi criada;
- 627: Uma senha de usuário foi alterada;
- 628: Uma senha de usuário foi redefinida;
- 630: Uma conta de usuário foi excluída;
- 632: Um membro foi adicionado a um grupo global;
- 633: Um membro foi removido de um grupo global;
- 636: Um membro foi adicionado a um grupo local;
- 637: Um membro foi removido de um grupo local;

Para obter as informações úteis para auditoria referentes ao determinado nos Itens 5 e 6 da Política de Auditoria de Segurança adotada pelo Tribunal foram criadas as seguintes consultas:

1. Quais contas de usuários do domínio foram criadas, modificadas ou excluídas em um determinado período a ser definido no ato da consulta e qual usuário realizou a criação, alteração ou exclusão da conta?
  - Informações necessárias para construção dessa consulta:
    - Hosts auditados: Todos
    - Banco de dados: TCC
    - Tabela: Eventos\_Auditoria
    - Código de eventos: 624, 627, 628 e 630
    - Período: 01/06/2014 a 26/06/2014.
  - Resultado Esperado para essa consulta:
    - O identificador de cada evento;



- Descrição do Evento;
- Identificação do usuário que praticou a ação;
- Data e hora do evento.

A Figura 47 contempla o resultado da 1ª Consulta SQL do 5º Teste referente ao Item 5 da Política de Auditoria adotada pelo Tribunal.

The screenshot displays the Microsoft SQL Server Management Studio interface. The query editor shows the following SQL query:

```

select Numero_registro, Identificador_Evento.ID, Identificador_Evento.Descricao, Computador, Usuario, Data
from Eventos_Auditoria, Identificador_Evento
where Identificador_Evento.ID IN ('624','627','628','630')
AND Data between '04/01/2014' AND '06/26/2014'
AND Eventos_Auditoria.Identificador_Evento = Identificador_Evento.ID
order by Data desc

```

The Results pane shows the following data:

	Numero_registro	ID	Descricao	Computador	Usuario	Data
3	48095	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 23:24:13.000
4	48095	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 23:24:13.000
5	48024	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 23:17:40.000
6	48024	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 23:17:40.000
7	46955	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 21:42:42.000
8	46955	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 21:42:42.000
9	46944	630	Conta de usuário excluída	SERVER	TCC\Administrador	2014-06-25 21:42:29.000
10	46944	630	Conta de usuário excluída	SERVER	TCC\Administrador	2014-06-25 21:42:29.000
11	46943	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 21:42:28.000
12	46943	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 21:42:28.000
13	45678	628	Uma senha de usuário foi redefinida	SERVER	TCC\Administrador	2014-06-25 18:41:49.000

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (52) tcc 00:00:00 64 rows. Ready Ln 1 Col 1 IN5

**Figura 47: 1ª Consulta de Auditoria do quinto Teste.**

2. Quais contas de usuários foram adicionadas ou removidas de grupo de perfil de “Administrador ou similar” em um determinado período a ser definido no ato da consulta e qual usuário realizou a adição ou remoção da conta do grupo?

- Informações necessárias para construção dessa consulta:
  - Hosts auditados: Todos
  - Banco de dados: TCC
  - Tabela: Eventos\_Auditoria
  - Código de eventos: 632, 633, 636 e 637
  - Período: 01/06/2014 a 26/06/2014.
- Resultado Esperado para essa consulta:
  - O identificador de cada evento;

- Descrição do Evento;
- Identificação do usuário que praticou a ação;
- Unidade Organizacional do usuário;
- Data e hora do evento.

A Figura 48 contempla o resultado da 2ª Consulta SQL do 5º Teste referente ao Item 6 da Política de Auditoria adotada pelo Tribunal.

The screenshot displays the Microsoft SQL Server Management Studio interface. The query window shows the following SQL query:

```

select Identificador_Evento.ID, Identificador_Evento.Descricao, Usuario, Usuarios.OU, Data
from Eventos_Auditoria, Identificador_Evento, Usuarios
where Codigo_Evento IN ('632', '633', '636', '637')
AND Data between '04/01/2014' AND '06/26/2014'
AND Eventos_Auditoria.Identificador_Evento = Identificador_Evento.ID
AND Eventos_Auditoria.Usuario = Usuarios.NOME

```

The Results pane shows the following data:

ID	Descricao	Usuario	OU	Data
637	Usuário removido de um grupo local	TCC\Administrador	Users	2014-06-25 23:49:07.000
636	Usuário adicionado a grupo local	TCC\Administrador	Users	2014-06-25 23:47:40.000
632	Usuário adicionado a grupo global	TCC\Administrador	Users	2014-06-25 23:41:01.000
636	Usuário adicionado a grupo local	TCC\Administrador	Users	2014-06-25 23:30:00.000
636	Usuário adicionado a grupo local	TCC\Administrador	Users	2014-06-25 23:25:41.000
636	Usuário adicionado a grupo local	TCC\Administrador	Users	2014-06-25 23:19:05.000
632	Usuário adicionado a grupo global	TCC\Administrador	Users	2014-06-25 21:55:08.000
636	Usuário adicionado a grupo local	TCC\Administrador	Users	2014-06-25 21:47:18.000
632	Usuário adicionado a grupo global	TCC\Administrador	Users	2014-06-25 21:45:20.000
636	Usuário adicionado a grupo local	TCC\Administrador	Users	2014-06-22 17:53:16.000

The status bar at the bottom indicates: Query executed successfully. SERVER\TCC (9.0 RTM) TCC\Administrador (53) tcc 00:00:00 32 rows

**Figura 48: 2ª Consulta de Auditoria do quinto Teste.**

Percebe-se que não há diferenças significantes na elaboração das consultas, como também uma vez construído o script de coleta não há muito que alterar para atender novas demandas, a parte mais complicada no processo de coleta das informações foi o tratamento de strings devido a falta de padronização de algumas informações úteis para auditoria. Portanto, dessa complexidade ficaram faltando algumas informações em alguns cenários, por exemplo, os nomes das contas de usuários que foram excluídas, criadas ou modificadas não foram possíveis extrair tais informações da string da propriedade “Message” do objeto coletado.

Diante do foi apresentado tanto na parte teórica, materiais e metodologia, bem como por meio dos procedimentos adotados no capítulo resultado e discussão para

desenvolvimento da solução proposta foi possível comprovar a viabilidade de aplicação de política de auditoria em uma rede Windows, bem como obtenção de informações indispensáveis para auditorias para fins de tomada de decisões tanto dos gestores de uma instituição assim como para melhorar a gerência, desempenho e a segurança da rede. Além dessas vantagens os resultados obtidos por meio da auditoria podem contribuir para o dimensionamento e aplicação de segurança no perímetro da rede, como *firewall*, técnicas de criptografias, ferramentas de detecção de intrusão, etc..

No capítulo seguinte serão apresentadas as considerações finais sobre o desenvolvimento do trabalho.

## **5. CONSIDERAÇÕES FINAIS**

A primeira fase deste trabalho teve como principal objetivo a apresentação de um embasamento teórico que dê sustentação e possibilidade aos administradores de redes criarem soluções de gerenciamento, principalmente, relacionadas à realização de auditoria eficiente em redes windows. O processo de auditoria em redes windows é um recurso importante para o gerenciamento da rede e muito pouco utilizado, principalmente, devido a falta de conhecimento sobre o real significado das propriedades exibidas nos *logs* registrados e pela dificuldade em obter informações úteis para auditoria acessando cada equipamento da rede de forma individualizada. Mapear o perfil do uso da rede e dos usuários é indispensável para manter a rede sob controle gerencial.

Foi apresentada a utilização dos recursos já disponíveis na plataforma Windows, instalados juntamente com o sistema operacional ou baixados do próprio site da Microsoft, que tornam viável a construção de soluções aplicáveis ao gerenciamento de redes, em especial, para o processo de auditoria.

A viabilidade de construção de uma solução de apoio à auditoria em redes Windows com o uso de base de dados relacional para centralização das informações da rede, que sejam úteis para auditoria, é perfeitamente possível e indispensável para uma boa gerência de segurança da rede, visto que a auditoria faz parte intrínseca ao ciclo da gerência de rede no quesito segurança, principalmente por ajudar a proteger e a encontrar *gaps* de segurança ao bem mais valioso da instituição: a Informação.

Este trabalho servirá de base para trabalhos futuros de desenvolvimento de ferramentas mais completas e com interfaces gráficas mais amigáveis para tornar mais popular a implementação de Políticas de Auditoria de Segurança em prol da gerência de segurança aplicável às redes de computadores.

Apesar da solução proposta ter sido desenvolvida em ambiente de rede com Windows 2003 Server, cujo sistema operacional de rede já em desuso devido as novas versões de Windows Server disponíveis no mercado, poucas mudanças serão necessárias para aplicação da solução em novos ambientes com versões posteriores aos 2003. As modificações basicamente se resume às adequações da codificação dos eventos de logs de auditoria e quanto à utilização de alguns novos métodos da classe *Win\_NTLogEvent* disponíveis nas novas versões. O Windows 2003 Server foi utilizado em razão da estrutura do trabalho ter sido concebida com essa versão quando ainda estava em pleno uso, tanto o estágio como o desenvolvimento da primeira parte do Trabalho (conteúdo teórico). Porém, como houve lapso temporal significativo entre o desenvolvimento da primeira parte e a parte prática do trabalho, neste intervalo de tempo surgiram novas versões dos Windows Server.

Conclui-se que pelo trabalho prático apresentado é possível aplicação de uma comedia Política de Auditoria de Segurança em uma rede Windows utilizando dos recursos do WMI para obter informações importantes da rede para tomadas de decisões eficientes no combate ao mau uso dos recursos e informações disponibilizadas na rede.

## 6. REFERÊNCIAS BIBLIOGRÁFICAS

- (BATTISTI, 2006) Battisti, Júlio Cesar Fabris. Tutorial de *Active Directory*. 2006. Disponível em <http://www.juliobattisti.com.br/artigos/default.asp>. Acesso em 05/04/2007.
- (DOMÍNIOS, 2004) \_\_\_\_\_. Configurando a infra-estrutura de domínio em um ambiente *Windows Server 2003*. Atualizado em 06/04/2004. Disponível em <http://www.microsoft.com/brasil/security/guidance/prodtech/win2003/secmod118.msp> Acesso em 07/04/2007.
- (DIRECTORY, 2005) \_\_\_\_\_. *Active Directory*. Atualizado em 21/01/2005. Disponível em <https://www.microsoft.com/technet/prodtechnol/windowsserver2003/pt-br/library/ServerHelp/a9d684f0-90b1-4c67-8dca-7ebf803a003d.msp?mfr=true>. Acesso em 06/04/2007.
- (WADLOW, 2000) Wadlow, Thomas A. *Segurança de Redes: Projeto e Gerenciamento de Redes Seguras*. Rio de Janeiro: Campus, 2000. 268p.
- (HORTON, 2003) Horton, Mike; Mugge, Clinton. *Hack Notes: Segurança de Redes - referência rápida*. Rio de Janeiro: Campus, 2003. 250p.
- (BRISA, 1993) BRISA. Sociedade Brasileira Para Interconexão de Sistemas Aberto. *Gerenciamento de Redes*. São Paulo: Makron Books, 1993. 364p.
- (AUDITORIA, 2004) \_\_\_\_\_. *Diretiva de Auditoria*. Atualizado em 17/05/2004. Disponível em <https://www.microsoft.com/brasil/security/guidance/topics/ameaca/secmod50.msp#EVE>. Acesso em 10/04/2007.
- (EMBRATEL, 1994) Carvalho, Tereza Cristina Melo de Brito, Organizado e Coordenado por. *Arquitetura de Redes de Computadores – OSI e*

TCP/IP - BRISA. São Paulo: Makron Books; Rio de Janeiro:EMBRATEL; Brasília, DF: SGA, 1994.

- (RAIMIR, 1998) Filho, Raimir Holand. SAGRES: Um Sistema Baseado em Conhecimento para Apoio à Gerência de Falhas em Redes de Computadores. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal do Ceará, 1998.
- (FREITAS, 2001) Freitas, Gerado Magela Lopes. Uma Estratégia para Implementação de Gerenciamento de Redes – Estudo de Caso do TCU. Dissertação de Mestrado - Universidade Federal de Santa Catarina, 2001.
- (FERNANDEZ, 2007) Fernandez, Marcial Porto. Gerenciamento de Redes. Universidade Estadual do Ceará, 2007. Disponível em <http://www.larces.uece.br/~marcial/>. Acesso em 15/05/2007.
- (WMI, 2008) Microsoft TechNet. Windows Management Instrumentation. Disponível em [http://www.microsoft.com/technet/scriptcenter/guide/sas\\_wmi\\_overview.aspx?mfr=true](http://www.microsoft.com/technet/scriptcenter/guide/sas_wmi_overview.aspx?mfr=true). Acesso em 18/05/2008.
- (MSDN, 2008) Microsoft Developer Network. Windows Management Instrumentation. Atualizado em 19/06/2008. Disponível em [http://msdn.microsoft.com/en-us/library/aa390413\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa390413(VS.85).aspx). Acesso em 20/06/2008.
- (JNLC, 2006) Carneiro, José Neto Luz. Documentação sobre Diretivas de Auditoria de Segurança em Domínio Windows 2003 Server. Trabalho de Conclusão de Estágio do Curso de Sistema de Informação do CEULP/ULBRA. Disponível em <http://www.ulbra-to.br/ensino/monografias/VerMonografia.aspx?im=210>. Acesso em 23/06/2008.
- (MIRANDA, 2010) Miranda, Vitor de Deus. Análise comparativa entre as ferramentas VirtualBox, VMWare Workstation e QEMU em um ambiente Linux. Monografia de Pós-Graduação “Lato Sensu” – Departamento da Ciência da Computação-MG, 2010.

## **APÊNDICE A**

### **POLÍTICA DE AUDITORIA DE SEGURANÇA DA REDE WINDOWS DO TRE-TO**

#### **1. OBJETIVO**

Estabelecer critérios para realização de auditoria de segurança na Rede Windows do Tribunal com objetivo de minimizar o mau uso dos recursos de tecnologia disponíveis, apropriação indevida de informações jurídicas sigilosas e garantir meios de rastreabilidade de acessos indevidos a equipamentos e informações.

#### **2. JUSTIFICATIVA**

A rede do TRE-TO é composta por aproximadamente 500 estações de trabalho e 500 usuários, e, apesar de possuir mecanismos de segurança provido por Firewalls, regras de negócio específicas, sistemas de segurança e instalações seguras nas estações dos cartórios eleitorais, ainda há a necessidade de adicionarmos uma política de auditoria segurança principalmente quanto ao uso do maior acervo de bens de TI, que são as estações de trabalhos de uso direto dos usuários da rede Windows e o acesso a informações classificadas sigilosas ou informações de acessos controlados.

#### **3. DESCRIÇÃO DA POLÍTICA DE AUDITORIA**

As boas práticas para uma política de auditoria sustentável e gerenciável nos prudencia a limitar a política em ações de registros bem definidas e que não seja simplesmente ações geradoras de *logs* e, conseqüentemente, causadora da baixa performance dos computadores e da rede.



No Cenário da rede Windows do TRE-TO a política de auditoria será implementada por meio de ativação de Diretivas de Auditoria de Segurança capazes de provocar o registro em arquivos de *logs*, prioritariamente, das seguintes ações executadas na Rede, incluindo estações e servidores de domínio.

1. Acessos (*logon*) com êxito nos controladores de domínio e nas estações de trabalho da rede Windows do Tribunal.
  - **Objetivo: Manter o registro dos acessos sucesso para rastreamento de uso dos recursos de TI;**
2. Tentativas de acesso “sem êxito” nos servidores e em todas as estações de trabalho do Tribunal.
  - **Objetivo: Possibilitar detecção de ações de intrusões na rede e tentativas de quebra de senhas.**
3. Alterações das diretivas de auditoria, com registro da hora e do usuário que efetuou a alteração;
  - **Objetivo: Manter o controle das alterações de diretivas de auditoria para evitar prejuízos à auditoria da rede causados por desativação indevida das diretivas.**
4. Alterações e exclusão de pastas e arquivos definidos como críticos e/ou sigilosos pelos Gestores do Tribunal;
  - **Objetivo: manter o controle sobre os objetos gerenciados para possibilitar responsabilizar ações indevidas sobre os objetos.**
5. Criação, alteração e exclusão de usuários do domínio;
  - **Objetivo: Manter o registro do autor da ação, bem como de quem sofreu a ação para atender demandas específicas de auditoria.**
6. Concessão de privilégio de administrador de domínio para usuários da rede.
  - **Objetivo: Detecção de apropriação indevida de privilégios administrativos sobre os recursos da rede.**

#### **4. CRITÉRIO DE GERENCIAMENTO SOBRE OS REGISTROS**

- Todas as ações mencionadas deverão ser registradas em arquivos de *logs* e armazenadas em banco de dados por um período mínimo de 06 meses.
- Os registros de *logs* locais nos servidores e nas estações de trabalho serão armazenados localmente até o momento da coleta dos *logs*.
- Deverá ser adotado o processo de gravação circular dentro do espaço em disco pré-reservado para evitar que os *logs* de segurança impossibilite o acesso dos usuários aos respectivos computadores.
- Uma vez coletados os registros de *logs* que espelhem as ações auditadas e armazenados em bancos de dados, consultas poderão ser feitas para extrair as informações úteis para auditoria e tomada de decisões pela gerência de rede, bem como pela Administração do Tribunal quando necessário.
- As informações coletadas e armazenadas deverão ser mantidas com acessos restritos e não poderão ser publicadas ou utilizadas indevidamente para denegrir imagem dos usuários da rede, ficando passível de ação administrativa disciplinar o responsável por tais atos.

Palmas/TO, 18 de Junho de 2014.

**COORDENADORIA DE SUPORTE E INFRAESTRUTURA**

## APÊNDICE B

### SCRIPT de coleta de logs de eventos e gravação em banco de dados

```
#V12
#Script Para coletar os dados da classe Win32_NTLogEvent de todos dos computadores da rede
$ArrComputers = "host1" # , "host2", "host3", "host4", "host5"
foreach ($Computer in $ArrComputers)
{
$name = Get-WmiObject -Class Win32_NTLogEvent -Namespace "root\cimv2" -Computer $Computer `
| Where-Object {$_.Category -Match "2" -and $_.EventCode -eq "682" -or $_.EventCode -eq "528" -or $_.EventCode -eq "540" `
-or $_.EventCode -eq "550" -or $_.EventCode -eq "552" -or $_.EventCode -eq "529" -or $_.EventCode -eq "530" -or $_.EventCode -eq "531" `
-or $_.EventCode -eq "532" -or $_.EventCode -eq "533" -or $_.EventCode -eq "534" -or $_.EventCode -eq "517" -or $_.EventCode -eq "608" `
-or $_.EventCode -eq "609" -or $_.EventCode -eq "611" -or $_.EventCode -eq "612" -or $_.EventCode -eq "620" -or $_.EventCode -eq "621" `
-or $_.EventCode -eq "622" -or $_.EventCode -eq "609" -or $_.EventCode -eq "560" -or $_.EventCode -eq "563" -or $_.EventCode -eq "564"}

function Do-FilesInsertRowByRow ([Data.SqlClient.SqlConnection] $OpenSqlConnection) {
write-host "EventCode" + $name.RecordNumber

$sqlCommand = New-Object System.Data.SqlClient.SqlCommand
$sqlCommand.Connection = $OpenSqlConnection

$sqlCommand.CommandText = "SET NOCOUNT ON; " +
"INSERT INTO dbo.Eventos_Auditoria (Identificador_Evento, Categoria, Computador, Dominio,
Codigo_Evento, Tipo_Evento, Arquivo_log, Numero_Registro, Origem, Data, Hora, Usuario,
```

```

        Tipo_Logon, IP_Origen, Porta_Origen) " +
"VALUES (@Identificador_Evento, @Categoria, @Computador, @Dominio, @Codigo_Evento,
        @Tipo_Evento, @Archivo_log, @Numero_Registro, @Origen, @Data, @Hora, @Usuario,
        @Tipo_Logon, @IP_Origen, @Porta_Origen);" +
"SELECT SCOPE_IDENTITY() as [InsertedID]; "

```

```

        $sqlCommand.Parameters.Add((New-Object
Data.SqlClient.SqlParameter("@Identificador_Evento", [Data.SqlDbType]::bigint)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Categoria", [Data.SqlDbType]::bigint)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Computador", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Dominio", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Codigo_Evento", [Data.SqlDbType]::bigint)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Tipo_Evento", [Data.SqlDbType]::NVarChar, 250)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Archivo_log", [Data.SqlDbType]::NVarChar, 500)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Numero_Registro", [Data.SqlDbType]::bigint)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Origen", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Data", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Hora", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Usuario", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Tipo_Logon", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@IP_Origen", [Data.SqlDbType]::NVarChar, 50)))
        $sqlCommand.Parameters.Add((New-Object Data.SqlClient.SqlParameter("@Porta_Origen", [Data.SqlDbType]::NVarChar, 50)))

```

```

        foreach ($file in $name) {
            $dominio = ""
            $stringUser = ""
            $tipoDeLogon = ""
            $ipOrigen = ""
            $portaOrigen = ""
            $serverVar = ""

            $objetoMessage = $file.Message

            $posDominio = 0
            $posTipoLogon = 0
            $posPortaOrigen = 0
            $posIpOrigen = 0

            $stringUser = $file.User #.Substring(0)
            $stringTimeGenerated = $file.TimeGenerated.Substring(0,14)
            $stringTimeWritten = $file.TimeWritten

```

#####

```

#Verifica se String Contendo o dominio está Vazia
if($objetoMessage | Select-String "Domínio"){
    $posDominio = $objetoMessage.IndexOf("Domínio")

    $verificaDominio = $objetoMessage.Substring($posDominio,3)
if($verificaDominio -eq " "){
    #Nenhuma ação será executada aqui
}
else{
    $posDominio = $posDominio + 10
    $dominio = $objetoMessage.Substring($posDominio,8)
    $dominio = $dominio.trim()
    #write-host "Dentro"
}
}

#####
#Verifica se String Contendo o Tipo de Logon está Vazia
if($objetoMessage | Select-String "Tipo de logon"){
    $postipoLogon = $objetoMessage.IndexOf("Tipo de logon")

    $verificaTipoLogon = $objetoMessage.Substring($postipoLogon,3)
if($verificaTipoLogon -eq " "){
    #Nenhuma ação será executada aqui
}
else{
    $postipoLogon = $postipoLogon + 15
    $tipoDeLogon = $objetoMessage.Substring($postipoLogon,2)
    $tipoDeLogon = $tipoDeLogon.trim()
}
}

}

#####
#Verifica se String Contendo a Porta de Origem está Vazia

if($objetoMessage | Select-String "Porta de origem"){
    $posPortaOrigem = $objetoMessage.IndexOf("Porta de origem:")
    # $posPortaOrigem

    $verificaPortaOrigem = $objetoMessage.Substring($posPortaOrigem,4)

if($verificaPortaOrigem -eq " "){
    "Nenhuma ação será executada aqui"
}
}

```

```

else{
    $posPortaOrigem = $posPortaOrigem + 17
    $portaOrigem = $objetoMessage.Substring($posPortaOrigem,5)
    $portaOrigem = $portaOrigem.trim()
}
}
#####
#Verifica se String Contendo IP do Endereco Do Cliente está Vazia Em Ambiente XP
if($objetoMessage | Select-String "Endereço do cliente"){
    $posIpOrigem = $objetoMessage.IndexOf("Endereço do cliente")
    $verificaIpOrigem = $objetoMessage.Substring($posIpOrigem,3)
    if($verificaIpOrigem -eq " "){
        #Nenhuma ação será executada aqui
    }
    else{
        $posIpOrigem = $posIpOrigem + 20
        $ipOrigem = $objetoMessage.Substring($posIpOrigem, 15)
        $ipOrigem = $ipOrigem.trim()
    }
}
}
#####
#Verifica se String Contendo IP do Endereco Do Cliente está Vazia Em Ambiente Server
if($objetoMessage | Select-String "Endereço de rede de origem"){
    $posIpOrigem = $objetoMessage.IndexOf("Endereço de rede de origem:")
    $verificaIpOrigem = $objetoMessage.Substring($posIpOrigem,4)

    if($verificaIpOrigem -eq " "){
        "Nenhuma ação será executada aqui"
    }
    else{
        $posIpOrigem = $posIpOrigem + 28
        $ipOrigem = $objetoMessage.Substring($posIpOrigem, 14)
        $ipOrigem = $ipOrigem.trim()
    }
}
}
#####
#Verifica se Propriedade com nome do usuario está vazia. Em caso afirmativo
# Coleta usuario do Campo Message
#"Valor Atual de User"+ $stringUser
#####

```

```

$dataString = $file.TimeGenerated

$ano = $dataString.Substring(0,4)
$mes = $dataString.Substring(4,2)
$dia = $dataString.Substring(6,2)
$hora = $dataString.Substring(8,2)
$min = $dataString.Substring(10,2)
$seg = $dataString.Substring(12,2)

$dataTratada = $ano + "/" + $mes + "/" + $dia + " " + $hora + ":" + $min + ":" + $seg

        $sqlCommand.Parameters[0].Value = $file.EventIdentifier
$sqlCommand.Parameters[1].Value = $file.Category
$sqlCommand.Parameters[2].Value = $file.ComputerName
$sqlCommand.Parameters[3].Value = $dominio
$sqlCommand.Parameters[4].Value = $file.EventCode
$sqlCommand.Parameters[5].Value = $file.EventType
$sqlCommand.Parameters[6].Value = $file.LogFile
$sqlCommand.Parameters[7].Value = $file.RecordNumber
$sqlCommand.Parameters[8].Value = $file.SourceName
$sqlCommand.Parameters[9].Value = $dataTratada
$sqlCommand.Parameters[10].Value = $stringTimeWritten
$sqlCommand.Parameters[11].Value = $stringUser
$sqlCommand.Parameters[12].Value = $tipoDeLogon
$sqlCommand.Parameters[13].Value = $ipOrigem
$sqlCommand.Parameters[14].Value = $portaOrigem

        # Run the query and get the scope ID back into $InsertedID

# $sqlCommand.Parameters
        $InsertedID = $sqlCommand.ExecuteScalar()
        # write to the console.
        "Inserido novo registro com ID $InsertedID para o LOG " + $file.EventIdentifier
    }
}

# Abre conexão com o banco
$DBServer = "SERVER\TCC"
$DBName = "tcc"
$sqlConnection = New-Object System.Data.SqlClient.SqlConnection
$sqlConnection.ConnectionString = "Server=$DBServer;Database=$DBName;Integrated Security=True;"

```

```
$sqlConnection.Open()
# Quit if the SQL connection didn't open properly.
if ($sqlConnection.State -ne [Data.ConnectionState]::Open) {
    "Connection to DB is not open."
    Exit
}
# Call the function that does the inserts.
Do-FilesInsertRowByRow ($sqlConnection)
# Close the connection.
if ($sqlConnection.State -eq [Data.ConnectionState]::Open) {
    $sqlConnection.Close()
}
}
```