UM MODELO PARA VERIFICAÇÃO, HOMOLOGAÇÃO E CERTIFICAÇÃO DE ADERÊNCIA À NORMA NACIONAL DE SEGURANÇA DE INFORMAÇÃO - NBR-ISO/IEC-17799

Luís Rodrigo de Oliveira Gonçalves

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Luís Felipe Magalhães de Moraes, Ph.D.

Prof. Luís Alfredo Vidal de Carvalho, D.Sc.

Prof. Bruno Richard Schulze, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
FEVEREIRO DE 2005

GONÇALVES, LUIS RODRIGO DE OLIVEIRA

Um modelo para verificação, homologação e certificação de aderência à Norma Nacional de Segurança de Informação- ISO/IEC-17799[Rio de Janeiro] 2005

XII, 133 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2005)

Tese – Universidade Federal do Rio de Janeiro, COPPE

1. Normas de Segurança
2. Segurança de Redes
3. Certificação de Segurança

I. COPPE/UFRJ II. Título (série)

DEDICATÓRIA

Dedico este trabalho à minha família, em especial aos meus pais, Sônia Maria e Luiz Carlos, que desde o início desta jornada na terra sempre estiveram ao meu lado, guiando-me, apoiando-me e sempre que necessário corrigindo os meus passos. Vocês são meus heróis e estarão em meu coração hoje e sempre.

À minha amada esposa, Simone Sattler, pelo seu apoio, incentivo e compreensão por tantas noites em que passei em claro, me empenhando para a confecção deste trabalho. Te amo muito e obrigado por existir.

Aos meus amados avos Jair e Norma que, por desejo de Papai do Céu, já estão no andar de cima e mesmo tão longe dos nossos olhos jamais deixarão de estar presentes em nossos pensamentos. Muito obrigado por tudo que fizeram por mim. Obrigado por terem sido tão presentes na minha vida, sem vocês e seus carinhos meu caminho não teria tantas alegrias e vitórias. Sei que um dia poderei dizer pessoalmente quanto os amo e o quanto são importantes.

AGRADECIMENTOS

Agradeço a Deus e a todos que contribuíram para este trabalho, em especial ao Professor *Luís Felipe de Moraes*; à grande amiga *Michelini Christop Lima de Freitas* pelas infinitas vezes em que me ajudou durante a breve estadia no Laboratório Ravel; à minha amada irmã *Sandra Maris* pela ajuda na revisão deste texto; à amiga *Ana Elisa Moreira Tavares* pela força na elaboração do *design* do *front-end* do Tamanduá-Mirim e de seu logo; à amiga *Daniela C. C. Nery* por ter realizado a revisão final do texto; ao professor e amigo *Marcelo Lanza* pelo papo agradável e pelas dicas trocadas nas idas e vindas para Petrópolis.

Gostaria de agradecer a todos os amigos do *Laboratório Ravel-COPPE/UFRJ*, aos professores *Inês de Castro Dutra*, *Vítor Santos Costa* e *Felipe Maia Galvão França*. A *Cláudia Helena Prata* - Secretária Executiva, *Josefina Solange Silva Santos* - Secretária Acadêmica, *Maria Lúcia Ramos de Paula* - Assistente Administrativo, *Sônia Regina Galliano* - Responsável pelo Centro de Documentação e demais membros do Programa de Engenharia e Sistemas e Computação da Universidade do Rio de Janeiro.

E as entidades de fomento *CAPES* - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, *CNPq* - Conselho Nacional de Desenvolvimento Científico e Tecnológico e a *FAPERJ* - Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro, pelo apoio prestado às Univesidades e Laboratórios de Pesquisa Brasileiros, especialmente à Universidade Federal do Rio de Janeiro e ao Laboratório Ravel.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM MODELO PARA VERIFICAÇÃO, HOMOLOGAÇÃO E CERTIFICAÇÃO DE ADERÊNCIA À NORMA NACIONAL DE SEGURANÇA DE INFORMAÇÃO - NBR-ISO/IEC-17799

Luís Rodrigo de Oliveira Gonçalves
Fevereiro/2005

Orientador : Luís Felipe Magalhães de Moraes
Programa: Engenharia de Sistemas e Computação

Um dos problemas envolvendo a adoção e a implementação de políticas ou normas de segurança para sistemas de informação consiste em verificar se as regras efetivamente implementadas estão em conformidade com aquelas definidas nos respectivos padrões adotados. Este trabalho descreve uma proposta de modelo e a implementação de um protótipo, baseado nessa descrição, para auxiliar na verificação do nível de aderência de sistemas de computação em rede à Norma Nacional de Segurança da Informação *NBR-ISO/IEC 17799*. Diferentemente da maioria das propostas conhecidas neste contexto, tais como, por exemplo, o sistema denominado *COBRA - Consultive, Objective and Bi-functional Risk Analisis* [13], que se baseiam no uso de questionários eletrônicos, a presente proposta visa permitir a análise automática do ambiente a ser verificado. Conforme será mostrado, além de permitir a análise automática da aderência à norma *NBR ISO/IEC 17799*, o modelo proposto pode ser utilizado em qualquer tipo de rede, independentemente do seu tamanho (em termos do número de dispositivos) e da arquitetura utilizada. Além disso, é feita a implementação de um protótipo operacional baseado na proposta descrita, de modo a ilustrar o uso da ferramenta e dos principais conceitos e aplicações abordados neste trabalho. Este protótipo utiliza a própria rede de computadores para realizar a coleta das evidências da aderência à norma em questão. Sob este aspecto, tal aplicativo apresenta uma inovação em relação as outras soluções conhecidas, com a vantagem adicional que seus coletores podem ser facilmente adaptados para serem utilizados em outros ambientes de *software* e *hardware*.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

UM MODELO PARA VERIFICAÇÃO, HOMOLOGAÇÃO E CERTIFICAÇÃO DE ADERÊNCIA À NORMA NACIONAL DE SEGURANÇA DE INFORMAÇÃO - NBR-ISO/IEC-17799

Luís Rodrigo de Oliveira Gonçalves

February/2005

Advisor: Luís Felipe Magalhães de Moraes

Department: Computing Systems Engineering

One of thee problems involving adoption and implementation of policies or norms of security for information systems consist of verifying if the rules effectively implemented are in compliance with those defining in the respective adopted standards. This work describes a model proposal and the implementation of an archetype, based on this description, to assist in the verification of the level of conformity of computer systems in network to the National Norm of Security of Information NBR-ISO/17799. Differently of the majority of the proposals known in this context, such as, for example, the called system COBRA - Consultive, Objective and Bi-functional Risk Analysis [13], that if they base on the use of electronic questionnaires, present the proposal aims at to allow the automatic analysis of the environment to be verified. As it will be shown, besides allowing the automatic analysis of the conformity to Norm NBR ISO/IEC 17799, the considered model can be used in any type of network, independently of its size (in terms of the device number) and of the used architecture. Moreover, the implementation of an operational archetype based in the proposal described is made, in order to illustrate the boarded use of the tool and the main concepts and applications in this work. This archetype uses the proper computer network to carry through the collection of the evidences of the tack to the norm in question. Under this aspect, such applicatory one presents an innovation in relation the other known solutions, with the additional advantage that its collectors can easily be adapted to be used in other environments of software and the hardware.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

DESDE os primórdios da humanidade sempre foi possível perceber uma constante preocupação com as informações e com o conhecimento que geram. No Egito antigo por exemplo, somente as castas dominantes tinham acesso à escrita e à leitura, visto a complexidade e o seu valor agregado.

A medida em que a sociedade evoluía, a preocupação com a segurança das informações aumentava. Um exemplo clássico desta preocupação são as máquinas de cifrar, utilizadas durante o período da primeira grande guerra. A função destas máquinas era alterar o conteúdo das mensagens antes de seu envio, e ao capturar uma mensagem cifrada, o inimigo obtinha apenas um texto cifrado e não a mensagem original. Isso permitiu que segredos e estratégias fossem trocadas de forma segura entre aliados. Nos dias de hoje, a criptografia e a esteganografia continuam sendo largamente utilizadas, principalmente no comércio eletrônico.

O surgimento dos computadores e de sua interconexão através de redes mundialmente distribuídas permitiu uma maior capacidade de processamento e de distribuição das informações. Mas por outro lado, permitiu que várias pessoas acessassem informações confidenciais e sigilosas, o que causou a criação de mecanismos que evitassem o acesso e a alteração indevida dessas informações. Isto resultou, em propostas e publicações de várias normas de segurança em todo o mundo, principalmente nos Estados Unidos.

Dentre as normas que foram publicadas pode-se destacar: O *Orange Book*, ou

DoD 5200.28-STD, que foi a norma publicada pelo Departamento de Defesa dos Estados Unidos em 15 de agosto de 1983 [17], e a publicação *An Introduction to Computer Security: The NIST Handbook* [33] proposta pelo *National Institute of Standards and Technology - U.S. Department of Commerce*, entre outras. Para facilitar sua aplicação, as normas de segurança foram divididas em vários controles. Cada controle seria responsável por atender a um dos quesitos da norma. O uso de controles permite uma visão modular da questão da segurança e a aplicação contextualizada das normas às organizações.

À medida em que as organizações cresciam, as redes de computadores e os problemas de segurança cresciam também. Não demorou muito para ficar claro que proteger somente os sistemas operacionais, as rede e as informações que trafegavam por elas não era o suficiente. Com isto, foram criados comitês que tinham como objetivo o desenvolvimento de mecanismos mais eficientes e globais de proteção à informação. Dos quais pode-se destacar o *Comercial Computer Security Centre*, criado pelo governo britânico e que publicaria mais tarde a norma *BS-7799*.

A BS-7799 foi a primeira norma homologada a apresentar soluções para o tratamento da informação de uma maneira mais ampla. Segundo esta norma, todo tipo de informação deve ser protegida, independentemente da sua forma de armazenamento, seja analógica ou digital, e de seu valor para a organização.

No ano de 2000, houve a homologação da primeira parte da *BS-7799* pela *ISO*. Esta homologação originou a Norma Internacional de Segurança da Informação - *ISO/IEC 17799*, sendo composta por 10 macros controles, cada qual subdividido em controles específicos.

Em abril de 2001, a versão brasileira da norma *ISO* foi disponibilizada para consulta pública, e em setembro do mesmo ano a *ABNT* homologou a versão brasileira, que passou a ser denominada *NBR ISO/IEC 17799:2000* [2].

A Norma de Segurança da Informação trouxe mais do que vários controles de segurança. Ela permitiu a criação de um mecanismo de certificação das organizações semelhante as certificações *ISO* já existentes. No entanto, esta nova certificação afirma que a organização certificada manipula os seus dados e os dos clientes de forma segura, independentemente da forma como estão armazenados.

Possuir o certificado *ISO/IEC 17799* é o diferencial que está sendo almejado por

várias instituições. Pois ao ser certificada, a organização mostra-se apta a tratar dados de forma sigilosa. O sigilo e a integridade das informações são os objetos de desejo de todo o mercado consumidor, que está cada vez mais preocupado com a segurança de suas informações.

1.1 Motivação

O uso da *ISO/IEC 17799*, como ferramenta de certificação internacional das organizações, vem atraindo a atenção de vários segmentos econômicos. Pequenas e grandes instituições estão buscando mecanismos que facilitem este processo. Contudo, devido à abrangência e à complexidade da norma, o processo de certificação não é simples e exige um vasto conhecimento do ambiente e do negócio da organização.

Visando facilitar o processo de certificação, vários trabalhos estão sendo desenvolvidos. Entre eles destaca-se uma Ferramenta de Análise de Risco chamada *COBRA*, que é a abreviatura de *Consultive, Objective and Bi-functional Risk Analisis* [13].

O *COBRA* permite a realização de uma extensa gama de análises de segurança em um determinado ambiente. Porém, para que a análise seja realizada, o software faz uso somente de questionários eletrônicos, e, com base nas informações obtidas, o *COBRA* poderá informar quais controles da *ISO* devem ser aplicados, e ainda se foram aplicados corretamente ou não.

Devido ao uso dos questionários, tem-se uma análise passiva do ambiente, e a qualidade desta fica limitada ao conhecimento do indivíduo que o responde. Assim, as avaliações atuais, apesar de úteis, podem em muitos casos, estarem incompletas ou apresentarem um falso nível de segurança.

Deste modo, urge a criação de um modelo que permite realizar a verificação total ou parcial dos controles pertinentes ao ambiente, com o menor nível possível de interferência do seu administrador ou gerente. Este modelo deveria ser independente de qualquer tipo de *hardware* e/ou *software*, bem como livre de implementações proprietárias de qualquer empresa ou órgão.

Um modelo como este talvez não removeria toda a interferência dos administradores e gerentes, mas reduziria ao máximo a sua participação, o que em muitos casos elevaria o nível de veracidade da informação coletada e conseqüentemente o nível de qualidade da análise.

Pois, como é descrito na própria *NBR-ISO/IEC* "aquele que analisa a aplicação de um controle, preferencialmente não deve ser aquele que o implementou", afinal os erros poderiam ser ocultados maliciosamente.

1.2 Objetivos

O Objetivo deste trabalho é propor um Modelo de Verificação Automático da Aderência de um Ambiente à Norma Nacional de Segurança da Informação - *NBR ISO/IEC 17799*, que, ao contrário das soluções existentes, permitirá a coleta e análise das informações que qualificarão o ambiente de forma automática e independente dos gerentes e administradores.

O modelo, para que possa ser aplicado em ambientes heterogêneos, será independente de arquitetura de *hardware* e *software*, de sistema operacional, de fornecedor de serviços, da topologia e da quantidade de hospedeiros da rede da organização. No entanto, este modelo será construído tendo como base a arquitetura de rede *TCP/IP*.

Por prover um processo automático de verificação, o protótipo poderá ser utilizado por pessoas sem prévio conhecimento do ambiente e com um baixo nível de conhecimento técnico, principalmente quanto à segurança. E por realizar a coleta *in-loco* das informações sobre o ambiente, o modelo poderá fornecer um elevado nível de confiança das informações coletadas.

Além de facilitar e agilizar o processo de verificação, o modelo poderá ser utilizado na implementação de ferramentas de diagnóstico e correção das falhas de segurança existentes no ambiente. Contudo, este assunto não será alvo do protótipo em software que será implementado utilizando as descrições do modelo.

1.3 Organização do documento

Este documento é dividido em seis capítulos, tendo ainda dois apêndices, um glossário e as referências bibliográficas.

A Introdução ou primeiro capítulo visa apresentar uma pequena descrição do tema, bem como a motivação para a realização deste trabalho.

No segundo capítulo, será abordada a evolução dos trabalhos na área de segurança da informação, desde o surgimento do Problema Clássico de Segurança até o surgimento e homologação da Norma Nacional de Segurança da Informação - *NBR ISO/IEC 17799*, homologada em 2001. Desde então, vem servindo como base nos processos de certificação de ambientes computacionais em vários países de língua portuguesa. Apresentará também uma breve discussão sobre o processo de utilização da norma de segurança dentro das organizações, bem como ferramentas que podem auxiliar este processo.

No terceiro capítulo, serão descritas as técnicas que podem ser empregadas na coleta de informações sobre os nós da rede. Estas técnicas serão utilizadas pelos Agentes de Verificação a serem propostos no capítulo seguinte.

Ao chegar no quarto capítulo será apresentado o modelo de segurança e os componentes que o constituem, bem como restrições e recomendações para a sua implementação. Será apresentado ainda o protocolo utilizado para a comunicação entre os clientes e servidores que constituem cada um dos agentes.

Já no quinto capítulo, será introduzida a descrição da implementação de um protótipo baseado no modelo proposto.

O sexto capítulo trará as conclusões e sugestões, que formam o resultado do conjunto de esforços de pesquisa e implementação demandados para a construção do modelo e do protótipo.

Em seguida temos o primeiro e segundo apêndice. O primeiro apresenta os níveis de classificação adotados pelo *Orange Book*. O segundo apêndice, apresenta a norma *NBR ISO/IEC 17799*, com os seus 10 (dez) macros controles e seus principais subcontroles de segurança. Cada um destes controles representa um dos quesitos de segurança que deve ser atendido para que o ambiente possa ser considerado “cem

por cento" aderente a norma.

Encerrando o documento há um glossário com os principais termos usados e as referencias bibliográficas.

Capítulo 2

Processo evolutivo da segurança em sistemas de informação

NESTE capítulo será apresentada uma breve descrição do processo evolutivo da segurança das redes e dos ambientes computacionais, iniciando com o surgimento do Problema Clássico de Segurança de Computadores e culminando com a homologação da norma *NBR ISO/IEC-1779:2000*, que é a norma nacional de segurança de informação.

2.1 Surgimento dos primeiros problemas de segurança

Segundo *The NIST HandBook* [33], o surgimento dos primeiros problemas com segurança de computadores foram provenientes da necessidade do compartilhamento do processamento de informações e recursos entre vários tipos de usuários, com vários níveis de confiança. Neste contexto, o compartilhamento de recursos foi um dos incentivos ao desenvolvimento e utilização dos sistemas operacionais de tempo compartilhado (*time-sharing*). Este sistema operacional deve ser implementado de forma a fornecer os meios necessários para minimizar os problemas de segurança gerados pelo compartilhamento de recursos.

Simplificadamente, o problema clássico de segurança de computadores pode ser

apresentado da seguinte forma: “Como fazer com que usuários autorizados possam ter acesso a determinadas informações, ao mesmo tempo em que os usuários não autorizados não possam acessá-las?”

Quando de seu surgimento, o *time-sharing* era visto como uma mera conveniência, mas que, na verdade, criou uma nova forma de se trabalhar com o computador; principalmente pelo fato de que no mesmo período do seu surgimento houve o desenvolvimento de vários tipos de periféricos, tais como: os terminais de acesso, os dispositivos de armazenamento de dados com acesso aleatório, e de *softwares* que foram desenvolvidos para explorar estes novos recursos. Destes *softwares* pode-se destacar os primeiros sistemas operacionais de uso geral, desenvolvidos para gerenciar estes componentes de *hardware*. Esta nova combinação de *hardware* e *software* transformaria o computador em um novo tipo de máquina.

Para um melhor aproveitamento das novas tecnologias, houve a necessidade de se realizar uma classificação das informações. Esta classificação deve ser feita baseada em pelo menos dois níveis hierárquicos onde, cada usuário tem acesso as informações de acordo com o seu nível de liberação. Porém, até meados de 1969 não existia uma metodologia que definisse como combinar informações de vários níveis de acesso em um único sistema do tipo *time-sharing* e como garantir que nenhum usuário poderia ter acesso à algo que ele não possuísse autorização.

2.2 Problema clássico de segurança

No documento *Proof Requeriments in the Orange Book*[38] encontra-se uma definição para, o que ficou conhecido como, o Problema clássico da Segurança de Computador, segundo o qual “... o problema clássico de segurança de computador pode ser caracterizado como a dificuldade de se criar um sistema operacional *time-sharing* que, quando operado em modo multinível, implemente apropriadamente os princípios de nível de segurança e de sistema de liberação”, ou simplesmente “o problema clássico de segurança de computador é o problema de construir um sistema operacional time-sharing multinível seguro ...”.

A solução para o problema clássico seria a construção de sistemas operacionais apropriados que permitissem o compartilhamento seguro dos recursos, bem como o

fornecimento de um conjunto de provas que evidenciassem se o sistema resultante implementava o que se propunha a fazer.

A caracterização do problema clássico de segurança significava a necessidade de um sistema operacional "melhor", com recursos mais aprimorados. Porém, até o presente momento não havia ferramentas para desenvolver tal solução, e esta falta de ferramentas devia-se principalmente ao modo como os softwares eram desenvolvidos.

Como um sistema operacional nada mais é do que um conjunto de *softwares*, a criação do sistema operacional apropriado para a solução proposta requeria o desenvolvimento de *softwares* confiáveis.

Os relatórios de Ware e Anderson que são citados na referencia [38] apresentaram pela primeira vez, quais áreas deveriam ser abordadas para solucionar o problema da construção de *softwares* confiáveis.

Neste período, o desenvolvimento de *softwares* atingia um momento de transformação e aprimoramento, o qual ficou conhecido como a crise do software - *software crisis*. Esse período foi caracterizado pela busca e criação de novas técnicas que descrevessem métodos e conceitos mais precisos de especificação e descrição do comportamento do *software*, bem como melhores técnicas de desenvolvimento, organização e gerenciamento dos processos do projeto (construção e manutenção de *softwares*).

Neste processo duas técnicas se destacaram: a Verificação do Programa e a Programação Estruturada. Porém, elas não forneciam soluções específicas para a definição e desenvolvimento deste novo sistema operacional. Desta forma, houve a procura e o desenvolvimento do que acabou sendo conhecido como *The Doctrine*, que especificaria as funcionalidades que um sistema operacional deveria ter para prover informações para usuários habilitados e bloquear o acesso para usuários que não o devem ter, e conseqüentemente, também foi necessário o desenvolvimento de técnicas para modelar e construir sistemas que pudessem se comportar de acordo com a *Doctrine*.

O processo aqui descrito foi desenvolvido entre os anos de 1961 e 1983 e teve início em meados de 1961 quando o primeiro sistema operacional de propósito geral, em estado experimental, foi criado pelo *MIT*.

2.3 Surgimento das primeiras técnicas e práticas de Segurança

O ano de 1967, foi o ano em que a segurança de computadores passou a ter atenção oficial nos Estados Unidos[38]. No mesmo ano, foi criada uma força tarefa cujo foco era os mecanismos de segurança de computadores, que deveriam ser desenvolvidos para prover a proteção de informações classificadas e do compartilhamento de recursos do sistema; este esforço resultou em um documento denominado *Security Controls for Computer Systems: Report of Defense Science Boad Task Force on Computer Security* editado por W. H. Ware [38]. Este relatório representou o trabalho inicial de identificação e tratamento do problema clássico de segurança de computadores.

Além do Departamento de Defesa dos Estados Unidos (*United States Department of Defense - DoD*), a Agência Central de Inteligência (*Central Intelligency Agency*) também se concentrou na questão da segurança de computadores. Neste período iniciou-se o desenvolvimento do *ADEPT-50* [22], que foi o primeiro sistema operacional de propósito geral a implementar as políticas de segurança do *DoD*.

Em outubro de 1972, J.P. Anderson escreveu um relatório técnico denominado *Computer Security Technologs Planning Study*, no qual ele descreveu todos os problemas envolvidos no processo de fornecimento dos mecanismos necessários para salvaguardar a segurança de computadores.

Entre 1967 e 1968, ocorreu o que ficou conhecido como *software crisis*. O *software crisis* foi largamente tratado na conferência de tecnologia de *software NATO*; planejada em 1967 e realizada em 1969 na Alemanha [38].

Em resposta à crise do *software*, surgiram as primeiras técnicas de verificação de programas. Os primeiros artigos que surgiram sobre o assunto foram: o artigo de R. W. Floyd denominado *Assigning Meanings to Prograns*, apresentado em um simpósio de matemática aplicada (*Mathematical Aspects of Computer Science*) e o outro artigo, de autoria de C. A. R. Hoare, denominado de *An aximatic basic for computer programing*, publicado na revista *Communication* da *ACM* em 1969; ambos artigos influenciaram largamente os trabalhos que o sucederam [38].

A verificação de programas, são mecanismos pelos quais o comportamento do *software* pode ser analisado e verificado durante seu projeto ou desenvolvimento [32]. Durante o desenvolvimento destes mecanismos, ficou claro que a resposta para a necessidade de métodos exatos e conceitos precisos poderia ser encontrada na própria lógica [38].

Segundo [38] a transcrição logo a seguir retirada do artigo *An axiomatica basic for computer programing* que C. A. R. Hoare publicou em 1969, exemplifica claramente os conceitos por trás da verificação do programa - *Program Verification*:

" ... programação de computadores é uma ciência exata, na qual todas as propriedades de um programa e todas as conseqüências de sua execução, em um dado ambiente, pode, a princípio, ser determinada pelo próprio código fonte do programa ... ".

Conforme [38], entre 1971 e 1972, foram publicados os primeiros artigos que definiram os fundamentos da programação estruturada. O primeiro foi *Program development by setp wise refinement*, escrito por Niklaus Wirth e publicado no periódico *Communications* da *ACM* em 1971; e o segundo escrito por D. L. Parnas e denominado de *A technique for software module specification with examples* foi publicado no periódico *Communication* da *ACM* em 1972. Este último apresenta idéias fundamentais, envolvendo a verificação da modelagem de *softwares*.

Neste período, David Parnas iniciou uma série de trabalhos cujo objetivo era o desenvolvimento de mecanismos para melhorar o processo de elaboração, organização do projeto e construção de *softwares*. O seu trabalho estava focado na idéia de se tratar o *software* como vários módulos de programas, que poderiam ser conectados para construir um *software* completo.

Com o desenvolvimento do seu trabalho, Parnas mostra que o processo de projeto e construção de *softwares* requer que o mesmo seja visto sob diferentes formas, de acordo com o nível de descrição do sistema. Esta visão partiria de uma descrição mais abstrata nos níveis mais altos, até uma visão menos abstrata, à medida em que os níveis de representação iam se aproximando da linguagem que poderia ser compreendia pelo computador.

Apesar de não ter respondido a questão de como solucionar o problema clássico de segurança de computadores, o trabalho de Parnas foi de fundamental importância

para o desenvolvimento desta solução, pois como afirma W.H. Ware, em seu artigo intitulado *Security Controls for Computer Systems*, citado por [38]: "... o risco mais sério em um sistema de software é o design incorreto ...". Assim como a modelagem correta é essencial para a segurança, a especificação correta é essencial para a modelagem.

Em paralelo com o trabalho de Parnas, a Força Aérea Norte Americana realizou vários esforços na tentativa de solucionar o problema clássico de segurança. O objetivo da Força Aérea era desenvolver um sistema operacional *time-sharing*, que seria utilizado no Pentágono e que pudesse processar as informações classificadas em ao menos dois níveis de sensibilidade, *Secret* e *Top-Secret*. Desta forma, o computador poderia ser compartilhado por usuários autorizados, os quais só poderiam ter acesso às informações cujo nível de sensibilidade fosse *Secret*.

2.3.1 *Security Kernel*

Ainda neste período, houve o desenvolvimento de várias técnicas e experimentos sobre segurança de computadores. Porém, existe uma controvérsia sobre a real origem deste processo: alguns afirmam que este teve origem com o trabalho de Roger R. Schell na Divisão de Sistemas Eletrônicos - *EDS (Eletronic System Division, Air Force Systens Command)* em 1971; enquanto para outros este processo teve origem com o artigo produzido por Anderson em 1972 [38].

Mas independentemente disto, ao ano de 1973 é atribuído o surgimento do que ficou conhecido como *Security Kernel*, que são os componentes fundamentais para o desenvolvimento e implementação das técnicas descritas por Anderson e Shell no processo de criação de um sistema operacional de uso geral seguro.

Independentemente das controvérsias mencionadas acima, a primeira tentativa de implementar as idéias apresentadas por estes autores, ocorreu com o *Mitre security kernel implementation effort* [46].

Naquele momento, Roger R. Schell possuía dois grandes problemas a serem resolvidos. O primeiro era desenvolver uma técnica que tornasse viável a construção de um sistema operacional multinível e o segundo era que ele deveria tornar esta técnica aplicável e atraente para que outras organizações pudessem se interessar no

seu desenvolvimento.

O primeiro problema foi solucionado quando Schell desenvolveu a idéia do que seria posteriormente conhecido como *Security Kernel*, que segundo o seu idealizador é um subconjunto de *hardware* e *software* suficientes para prover segurança ao sistema, mesmo quando outros componentes, do mesmo sistema, são desenvolvidos por outras entidades.

Anderson, no seu artigo publicado em 1972 e citado por [38], endossou as idéias de Shell quando afirma que o conceito básico no qual os sistemas de segurança de computadores devem ser apoiados é o controle de compartilhamento. Este controle deve ser estabelecido explicitamente sobre cada acesso de usuário a cada recurso compartilhado do sistema, de tal forma que cada usuário e recurso devem ser identificados e interrelacionados de acordo com o nível de acesso, ou acessibilidade, de ambos.

O que Schell tinha em mente quando idealizou o *security kernel* era reduzir o problema de construção do sistema operacional multinível seguro à construção de um componente de *software*, com as funcionalidades descritas anteriormente neste documento.

Porém, o *security kernel* não seria capaz de solucionar a questão do que deveria ser bloqueado ou o que poderia ser acessado por um determinado usuário. Para preencher esta lacuna foram desenvolvidos e produzidos modelos de políticas de controle de acesso, que poderiam ser adaptados para se adequar a política de segurança de uma determinada organização; que por sua vez baseia-se em um conjunto de leis, regulamentações, documentos administrativos e processuais relacionados ao ambiente no qual será aplicado.

2.3.2 Controle de acesso

Relacionado ao processo de controle de acesso, existem alguns termos de relevante importância tais como: classificação,níveis de segurança e liberação. A classificação é a designação, autorizada, do nível de segurança aplicada a uma determinada informação.

O nível de segurança, segundo o *National Computer Security Center*, é definido

como sendo a combinação de uma classificação hierárquica e um conjunto de categorias não hierárquicas que representam a sensibilidade da informação. A categoria por sua vez é um label, ou rótulo, que foi aplicado a uma informação, classificada ou não, visando aumentar o seu nível de proteção, bem como restringir o acesso a ela [33].

Assim, o nível de segurança seria a combinação do nível de sensibilidade e de um conjunto hierárquico de categorias que, juntos, representariam o quão importante ou restrita é uma informação.

A liberação, ou desobstrução, é a atribuição de uma autorização de um nível de segurança a um determinado indivíduo.

2.3.3 Modelos de segurança

Um dos modelos de segurança que pode ser usado como exemplo é o de *Bell / La Padula*. Este modelo pode ser decomposto nos seguintes elementos básicos: estados, regras, axiomas e requisitos (exigências).

Estado é o conjunto completo de informações existentes em um sistema computacional em um dado instante da computação. O processo computacional é a mudança de um determinado estado para outro, segundo um conjunto de regras previamente definidas. Os requisitos definem o que é necessário para que um estado seja considerado seguro.

Os requisitos deste modelo podem ser divididos em duas categorias de segurança: *Mandatory* e *Discretionary*. Aqueles tidos como *Mandatory* são comumente conhecidos pelo slogan *Read down ! Write Up!* [38].

Para compreender o significado desta frase, deve-se conhecer outros conceitos envolvidos no modelo, que distingue os componentes de um estado como sendo objetos ou sujeitos. Os objetos são os componentes que possuem algum tipo de informação e os sujeitos são os componentes que agem sobre os objetos. Quando os sujeitos possuem informações e atuam com outros sujeitos, eles podem se comportar como objetos [38].

Existem várias formas dos sujeitos e dos objetos trabalharem em conjunto e à

estas formas de interação damos o nome de modelo de acesso. Um sujeito que possui acesso de leitura a um objeto está apto a obter informações do mesmo. Já com o acesso de escrita, o sujeito tem o direito de alterar e adicionar informações de um certo objeto.

Este modelo atribui níveis de segurança, tanto aos objetos quantos aos sujeitos. Um nível de segurança aplicado a um sujeito é análogo ao conceito de *clearance* ou liberação, já o nível de segurança aplicado, ou atribuído a um objeto, é o mesmo que o conceito de *classification* ou classificação.

O requisitos *Read Down!*, oficialmente conhecidos como *single security property* garantem que um sujeito pode ter acesso de leitura para um objeto somente se o seu nível de segurança for superior ao nível de segurança do objeto. O requisito *Write UP*, conhecido oficialmente como **-property (star-property)*, garante que um sujeito pode ter acesso de escrita em um objeto somente quando seu nível de segurança for dominado pelo nível de segurança do objeto, ou seja, o objeto possui um nível de segurança maior.

Além dos requisitos do tipo *Mandatory* temos também os *Discretionary*, nos quais cada estado possui uma tabela de decisão. A tabela de decisão descreve o tipo de acesso que cada usuário possui sobre os componentes do tipo sujeito, que por sua vez está associado a um componente do tipo objeto. Logo um usuário somente poderá ter acesso a um objeto, em um determinado estado, se existir na tabela de decisão, uma entrada que referencie o usuário, o objeto e o sujeito.

Somente quando um estado satisfaz todos os requisitos do tipo *Mandatory* e *Discretionary*, ele pode ser considerado um estado seguro [38].

Muitas das implementações que se basearam no *Orange Book* utilizaram como modelo formal de política de segurança o modelo de *Bell / La Padula*, porém, o *Orange Book* não obriga o seu uso.

2.4 *DoD - Computer Security Initiative*

Por volta de 1977, o Departamento de Defesa dos Estados Unidos formulou um plano sistemático para tratar do problema clássico de segurança, este plano deu

origem ao *DoD Computer Security Initiative* [17].

A iniciativa de segurança de computadores desenvolvida pelo Departamento de Defesa Norte Americano tinha basicamente 3 propósitos: permitir ao *Do*D atuar no processo de desenvolvimento e uso de segurança; envolver a indústria no processo de definição dos requisitos necessários para a construção de um sistema confiável e o terceiro propósito era o desenvolvimento de um centro que ficaria sobre o controle do governo e que seria capaz de avaliar o quão bom e seguro eram as soluções produzidas pela indústria.

Somente o terceiro objetivo produziu resultados satisfatórios. Deles nasceram *Computer Security Center*, criado por volta de 1980, bem como o *Orange Book*, que ficou conhecido como a publicação mais importante do centro.

À procura de um lugar para a localização do centro de avaliação iniciou-se em meados do ano de 1979 e culminou em 4 de agosto de 1980, e o seu estabelecimento ocorreu entre setembro de 1980 a 25 de outubro de 1982 quando passou a ter um caráter oficial [38].

Roger Schell foi o primeiro diretor do centro e como ele mesmo afirmava ("I was again an unwilling volunteir") [38], possuía uma função a qual não desejava. Mesmo assim, tinha duas tarefas primordiais sobre a sua responsabilidade: a primeira era prover uma estrutura administrativa segura e confiável e a segunda era prover uma estrutura tecnológica confiável. Quando ele iniciou o seu trabalho, o centro era composto de 35 pessoas sob o seu comando, mas quando deixou o cargo, havia cerca de 200 pessoas trabalhando no centro e uma boa estrutura administrativa e tecnológica .

2.5 Processo de escrita do *Orange Book*

O *Orange Book* foi elaborado para auxiliar o processo de desenvolvimento e classificação de sistemas operacionais. Seu processo de escrita começou em 1978 e se prolongou até o ano de 1985 [17]. O primeiro estágio deste processo foi marcado quando T. M. P. Lee e outros publicaram o documento denominado *Processors, operating systems, and nearby peripherals.* O segundo estágio da produção do *Orange Book* ocorreu com a publicação de *Proposed Technical Evaluation criteria for*

computer systens em 25 de outubro de 1979 que foi escrito por G. H. Nibalde.

A primeira versão oficial produzida pelo Centro foi publicada em 24 de maio de 1982 com o nome de *Trusted Computer System Evaluation Criteria, 1st Draft* [40], este documento por sua vez marcou o terceiro estágio do *Orange Book*.

Revisões sucessivas levaram em 25 de janeiro de 1983 a versão final do *Draft*, que possuía capa verde, representando o quarto estágio do desenvolvimento da norma [40] . Em 15 de agosto do mesmo ano foi publicada a primeira versão não *Draft* do documento, cuja capa já era da cor laranja (*Orange*).

Finalmente em 26 de dezembro de 1995 foi publicado o sexto e último estágio da norma, esta publicação foi realizada pelo *National Computer Security Center - NCSA* e denominada oficialmente de *Trusted Computer System Evaluation Criteria - DoD 5200.28-STD (TCSEC)*, mais comumente conhecida como *The Orange Book* [40].

Os processos de avaliação de sistemas realizados pelo Centro começou entre o final de 1982 e o início de 1983, de um modo informal, porém a primeira avaliação formal ocorreu em 15 de abril de 1983 [38].

Em agosto de 1985 o *Computer Security Center* mudou de nome, passando a se chamar de *National Computer Security Center (NCSC)* e atualmente se encontra subordinado a *National Security Agency (NSA)* dos E.U.A.

Graças aos esforços relacionados com o processo de criação e operação do *NCSC* e ao processo de criação do *Orange Book (TCSEC)*, houve a produção de uma larga quantidade de publicações e de técnicas que se difundiram através da indústria e permitiram o desenvolvimento de sistemas computacionais e produtos mais confiáveis e seguros.

A série de documentos produzidos pelo *NCSC* ficou mundialmente conhecida como *Rainbow Series* [40] [38]. Estas publicações primam pela definição de requisitos de segurança a serem aplicados em ambientes computacionais. Dentre as publicações que fazem parte do Rainbow Series podem ser destacadas:

- O *Red Book*, oficialmente conhecido como *The Trusted Networking Implementation (TNI)*, que fornece os requisitos para a construção de sistemas baseados no *TCSEC* em ambientes de redes bem como a construção de

redes seguras. Sua última atualização foi em agosto de 1990.

- O *Aqua Book*, conhecido oficialmente como *Guide to Understanding Security Modeling in Trusted Systems*, publicado em outubro de 1992.
- E ainda o próprio *Orange Book*.

2.6 Outros esforços relacionados à segurança da informação

Desde o surgimento do problema clássico de segurança de computadores até a homologação da *ISO/IEC 17799:2000*, vários outros esforços nesta mesma área foram desenvolvidos, destes destacam-se:

O Projeto *Protection Analysis*, foi um dos primeiros esforços em automatizar o processo de avaliação da proteção ou segurança de Sistemas Operacionais. Seu objetivo era prover uma avaliação mais eficiente e econômica. Este projeto propôs o uso de métodos que reduziam a necessidade de experiência das pessoas envolvidas no processo de avaliação.

O projeto começou em setembro de 1973 e foi desenvolvido pela *A.R.P.A.* Seu relatório final, denominado *Protection Analysis - Final Report*, foi publicado por Richard Busbey e Dennis Hollingwth em maio de 1978. Para mais informações sobre este projeto consulte seu relatório final [9].

A *Norma ISO/IEC 7498-2 JTC1/SC2*: *Information Tecnologic - Open Systems Interconnection - Part 2: Security Architecture*, é uma extensão do modelo de referencia básica e está focada nos aspectos de segurança necessários aos protocolos de comunicação tratados pela parte 1 da norma.

A primeira versão da norma 7498 foi publicada em 1984. Contudo, após várias revisões ela foi dividida em 4 partes distintas:

Part 1 - *Basic Reference Model: Basic Model*, última versão publicada em 1994;

Serviços	Camadas
Confidencialidade	1,2,3,4,7
Autenticação	3,4,7
Integridade	3,4,7
Controle de Acesso	3,4,7
Não Repudio	7

Tabela 2.1: Serviços de segurança aplicados às camadas da *OSI*

Part 2 - *Security Architecture*, última versão publicada em 1984;

Part 3 - *Basic Reference Model: Naming and Addressing*, ultima versão publicada 1997;

Part 4 - *Basic Reference Model: Management Framework*, ultima versão publicada em 1989.

A *ISO 7498-2* propõe que sejam aplicados serviços de segurança para as diferentes camadas da arquitetura *ISO/OSI*. Os serviços de segurança aqui mencionados são a Confidencialidade, Autenticação, Integridade, Controle de Acesso e Não Repudio. Estes serviços podem ser aplicados as camadas da *OSI* conforme a tabela 2.1 [10].

Para maiores informações sobre esta norma consulte a referencia [25] e a Norma *ISO7498-2* [28]

O *NIST - National Institute of Standards and Technology*, tem produzido uma série de documentos focados na área de segurança e desde 1987 é o responsável pela preparação de normas e guias de segurança a serem utilizadas e aplicadas aos sistemas de informações federais nos Estados Unidos [33].

Dos documentos produzidos pelo *NIST* pode-se destacar o *An Introduction to Computer Security: The NIST Handbook*, que pode ser acessado no próprio site do *NIST* [35].

Este guia além de explicar vários conceitos relacionados com segurança e os in-

terrelacionamentos dos controles de segurança, ajuda o leitor a entender as suas necessidades de segurança e a determinar quais controles usar para alcançar o nível de proteção desejado. E como o autor mesmo afirma: "O guia é portanto útil para qualquer um que necessite aprender o básico sobre segurança de computadores ..." [33]

Em 1988 o Governo Canadense formou o *Canadiam System Security Center (CSSC)* e em maio de 1989 foi publicada a primeira versão *draft* do *CTCPEC*, ou *Canadiam Criteria*, cuja versão mais recente é a 3.0, publicada em janeiro de 1993.

Em 1989 o governo Alemão publicou, através do *West German Information Security Agency*, seu critério de avaliação de sistemas computacionais que, ao contrário do *TCSEC*, não se preocupava somente com questões de segurança do sistema, mas também com a integridade, disponibilidade e segurança da rede.

Ao mesmo tempo o *Britsh Commercial Computer Security Center (CCSC)* desenvolvia o critério Britânico, escrito em uma linguagem clara permitindo aos desenvolvedores de sistemas a fácil compreensão e implementação dos critérios em seus produtos.

Em abril de 1991, o *NCSC*, publicou o documento denominado *Trusted Database (TDI)*, o qual fornecia o critério de avaliação para produtos voltados para o gerenciamento de banco de dados. Inicialmente utilizado para atender os requisitos do governo norte americano, o *TDI* foi posteriormente utilizado por organizações não governamentais que desejam proteger as informações não classificadas, mas sensíveis. O *TDI* utiliza as mesmas 6 classes do *TSEC*.

O governo norte americano em uma tentativa de ampliar o escopo da *TCSEC* iniciou um programa conjunto entre o *NIST* e o *NSA* denominado *Federal Criteria (F.C.) Project*, que teve sua primeira versão *draft* publicada em dezembro de 1992.

No mês de julho do mesmo ano foi homologado o documento *The Information Technology Security Evaluation Criteria - ITESEC*, que foi o resultado da harmonização dos critérios de avaliação de segurança de quatro países europeus. O *ITSEC* desde então, passou a ser o padrão de avaliação de sistemas e produtos na França, na Alemanha, nos Países Baixos e Inglaterra, e posteriormente, em toda a Europa.

Ao contrário da norma americana, *TSEC*, esta trata de forma mais abrangentes questões sobre confidencialidade, integridade e disponibilidade tanto para soluções militares quanto comerciais.

Durante o desenvolvimento da *ITESEC*, a *Oracle* participou ativamente de todos as revisões relacionadas à sistemas de gerenciamento de banco de dados.

O critério canadense - *The Canadian Trusted Computer Product Evaluation Criteria (CTCPEC)* - possui influência tanto da norma *TCSEC* quanto da *ITESEC*.

Em junho de 1993 o *NIST* e o *NSA* decidiram trabalhar em conjunto com o governo Canadense e Europeu para a criação de um critério comum, o qual alinharia as normas *TCSEC*, *ITSEC* e *CTCPEC*.

O *Common Criteria (CC)*, como ficou conhecido o documento *The International Common Criteria for Information Technology Security*, foi o resultado de um esforço conjunto da União Européia, dos Estado Unidos e Canadá para o desenvolvimento de um critério que poderia ser utilizado mundialmente.

Em janeiro de 1996 foi publicada a versão 1.0 da norma *Common Criteria*, esta versão passou por vários ajustes devidos aos comentários recebidos e deu origem a uma segunda versão em maio de 1998.

Em outubro de 1998, Canadá, França, Alemanha, Inglaterra e EUA assinaram um acordo de reconhecimento mútuo (*MRA*) de produtos e sistemas e em maio de 2000 uma segunda versão deste acordo foi assinada, incluindo a Austrália, a Nova Zelândia, a Finlândia, a Grécia, a Itália, os Países Baixos, a Noruega e a Espanha ao grupo original. O *MRA* permitia que sistemas e produtos homologados em um país pudessem ser utilizados em outros países, sem que os mesmos fossem novamente submetidos a um processo de certificação.

Após a homologação de sua segunda versão, a *C.C.* foi utilizada para a criação de uma norma *ISO*, esta passou a ser reconhecida e utilizada mundialmente. A *ISO 15408* foi publicada em primeiro de maio de 1999.

Em julho do mesmo ano, a Comissão das Comunidades Européias emitiu um documento nomeado *Segurança das redes e da informação: Proposta de abordagem de uma política européia.* Este documento apresenta rapidamente o panorama europeu, como as ameaças mais comuns à segurança, os desafios a serem trabalhados

e os elementos básicos que devem fazer parte desta política.

Mais informações sobre esta proposta consulte o texto da própria proposta [12] ou o site em português da União Européia [19].

2.7 Surgimento da Norma Internacional de Segurança da Informação

Foi observado que somente a preocupação com a segurança dos dados, que trafegam por uma rede de computadores não era mais o suficiente no contexto atual da segurança. Logo, o modelo de segurança foi expandido da rede de computadores, para todo e qualquer meio de transmissão, armazenamento e manipulação de informação. Neste contexto, a informação pode se apresentar de várias formas e ser propagada em vários meios, mas independentemente da forma e meio de transmissão, deve ser sempre protegida.

Os esforços relacionados com a busca de melhores mecanismos para salvaguardar a segurança culminaram com a homologação da Norma Internacional de Segurança da Informação denominada *ISO/IEC 17799:2000*. Esta norma trata da segurança das informações e não somente dos dados que trafegam pela rede ou que residem dentro de um sistema computacional.

Esta *ISO* se originou de um esforço do governo britânico, que em 1987, através do *UK DTI (Departamente of Trade Center)* criou o *CCSC (Comercial Computer Security Centre)*, cujo objetivo era a criação de critérios para a avaliação da segurança e de um código de segurança para os usuários das informações, de uma forma geral. No ano de 1989 foi publicada a primeira versão do código de segurança, que na época foi denominado de *PD0003 - Código de Gerenciamento de Segurança da Informação*. [39]

Em 1995 esse código foi revisado, ampliado e publicado como uma norma britânica (*BS*), a *BS7799-1:1995 (Information Technology - Code of pratic for information security management)*. Em 1996, essa norma foi proposta a *ISO* para homologação, mas foi rejeitada. Neste mesmo período uma segunda parte deste documento estava sendo criada e em novembro de 1997 foi disponibilizada para consulta e avaliação do

público. Em 1998 este documento foi publicado como *BS7799-2:1998 (Information Security Management Systems)*[39].

Em Abril de 1999, as duas normas (a de 1995 e a de 1998) foram publicadas, após uma revisão, com o nome de *BS7799-1999*. Neste período, esta norma já estava sendo adotada por outros países como a Austrália, a África do Sul, a República Checa, a Dinamarca, a Coréia, a Suíça e a Nova Zelândia. *BS7799* já foi traduzida para várias línguas das quais pode-se destacar o Francês, o Alemão e o Japonês.

Neste mesmo ano, a primeira parte deste documento foi novamente submetida a *ISO*, para homologação, mas agora, sobre o mecanismo de *Fast Track*. Em maio de 2000 a *BSI* homologou a primeira parte da norma *BS7799*. Em outubro, na reunião do comitê da *ISO* em Tóquio, a norma foi votada e aprovada pela maioria dos representantes.

Os representantes dos países ricos, excluindo a Inglaterra, foram todos contra a homologação mas em primeiro de dezembro de 2000, houve a homologação desta *BS* como *ISO/IEC 17799:2000* [34].

A norma *BS7799-2* foi submetida a um processo de revisão em 2001. Em janeiro de 2002, foi emitido o primeiro *draft*, da mesma, para acesso e avaliação pública. Esta revisão visa ajustar *BS7799-2* com normas internacionais tais como a *ISO9001* e a *ISO14001*, e ainda remover aspectos próprios da lei britânica. Os controles da *ISO/IEC 17799* foram adicionados a um anexo desta versão, permitindo uma correspondência entre a numeração em ambas as normas [18]. A *BS 7799-2:2002* foi publicada no dia 5 de setembro de 2002.

Atualmente, o comitê *ISO/IEC JCT1 SC27* é o mantenedor e o responsável pela revisão da *ISO/IEC 17799:2000*. Devido ao processo de revisão que esta *ISO* está passando, o comitê recebeu recentemente um conjunto de comentários sobre a mesma, e em 24 de abril de 2003 foi realizado um encontro em Quebec, no qual os comentários restantes foram discutidos e uma nova versão da norma foi preparada. A nova versão da *ISO/IEC 17799* está prevista para o final do ano de 2004 ou início de 2005.

2.8 Norma Nacional de Segurança da Informação

A Associação Brasileira de Normas Técnicas (*ABNT*), responsável pelo Fórum Nacional de Normatização, em abril de 2001 [1], disponibilizou para consulta pública o *Projeto 21:204.01-010*, que daria origem a norma nacional de segurança da informação: *NBR ISO/IEC 17799:2000*.

A versão final da *NBR ISO/IEC-17799* que é uma tradução literal da norma Internacional de Segurança da Informação - *ISO/IEC-17799:2000*, foi homologada em Setembro de 2001 e sua publicação inclui oficialmente o Brasil no conjunto de países que, de certa forma, adotam e apóiam o uso da norma de segurança da informação [2]. Esta versão da *ISO/IEC 17799* vem sendo utilizada por vários outros países, como é o caso de Portugal, Angola entre outros.

A norma nacional de segurança de informação é dividida nos 10 macros controles:

Política de Segurança;
Segurança Organizacional;
Classificação e Controle dos Ativos da Informação;
Segurança em Pessoas;
Segurança Física e do ambiente;
Gerenciamento de Operações e Comunicações;
Controle de Acesso;
Desenvolvimento da Segurança de Sistemas;
Gestão da Continuidade do Negócio;
Conformidade.

Cada um destes controles é subdividido em vários outros controles, a *NBR* possui um total de 137 controles de segurança. Os Controles da norma nacional de segurança visam manter e gerir a segurança da informação nas organizações. Maiores detalhes sobre estes controles podem ser obtidos no apêndice Controles da *NBR ISO/IEC-17799*, no final deste documento ou na própria *NBR*[2].

Segundo a *NBR ISO/IEC 17799*, a Segurança de um ambiente é caracterizada pela manutenção de três fatores primordiais: a Confidencialidade, a Integridade e a

Disponibilidade das Informações Críticas [2].

Para a *NBR*, a Informação é "um ativo que, como qualquer outro ativo importante para os negócios, tem um valor para organização e conseqüentemente necessita ser adequadamente protegido".

A Confiabilidade é definida pela *NBR* como sendo a garantia de que a informação só pode ser acessada e manipulada por pessoas autorizadas, ou seja, é restrita a um conjunto de entidades, que podem ser seres humanos ou sistemas eletrônicos.

Já a Integridade implica que toda vez que uma informação é manipulada ela está consistente, ou seja, não foi alterada ou adulterada por um acesso legal ou ilegal.

E por fim, Disponibilidade é a garantia de que uma informação sempre poderá ser acessada, pelas pessoas e processos autorizados, independentemente do momento em que é requisitada e do local no qual está armazenada.

Afirmar que um ambiente é aderente à Norma de Segurança da Informação significa dizer que o mesmo utiliza os recursos adequados para garantir a Disponibilidade, Confidencialidade e a Integridade de suas informações.

Mas para isto devem ser aplicados ao ambiente alguns ou todos os controles existentes na norma de segurança. Contudo, a lista dos controles que devem ser aplicados depende de características do próprio ambiente, como por exemplo: forma e local de armazenamento das informações, valor das informações armazenadas, quem pode acessá-las, quais servidores estão instalados, que tipo de serviços são disponibilizados aos usuários da rede interna, externa e etc.

De acordo com o nível de segurança necessário, um conjunto de controles de segurança deve ser implementado. Mas a *NBR* é composta por 137 controles distintos e o processo de seleção dos controles a ser aplicado nem sempre é fácil de ser realizado. Para facilitar o processo de seleção de controles, podemos utilizar algumas ferramentas, como por exemplo: a análise de risco de um ambiente, a legislação vigente, os objetivos e as necessidades da organização [2].

A análise de risco de um ambiente baseia-se na avaliação do impacto de uma falha de segurança nas atividades da organização, bem como na sua probabilidade de ocorrência. O resultado desta análise apontará os principais pontos a serem trabalhados e a prioridade das informações a serem protegidas.

Sendo o resultado da análise de risco dependente das características atuais do ambiente e como o mesmo não é estático, este resultado representa as melhores ações a serem aplicadas em um determinado momento no ambiente. Conseqüentemente, análises de risco periódicas devem ser realizadas, mesmo quando um ambiente não tenha se modificado. Neste caso, a nova análise é justificada pelo fato de que novas vulnerabilidades e ameaças surgem diariamente.

Contudo, independente do ambiente, dos riscos e das ameaças que foram indicadas pelo resultado da análise de risco, existe um conjunto mínimo de controles, que segundo a *NBR*, sempre devem ser implementados. Este conjunto, segundo a *NBR* [2], engloba:

Política de segurança da informação : este é um documento que descreve quais atividades os usuários estão autorizados a realizar, como e quando podem ser realizadas. É de vital importância que a alta administração apóie o uso da política e demonstre o seu comprometimento com a aplicação das penalidades cabíveis;

Definição das responsabilidades de segurança : este controle visa esclarecer a quem pertence cada ativo da organização, bem como quem deve ser contactado em caso de problemas de segurança relacionados ao ativo em questão;

Processo de treinamento : a melhor forma de evitar o mal uso das informações é educar seus usuários, assim é de vital importância que todo e qualquer usuário passe por um treinamento antes de ter acesso as informações contidas no ambiente.

Relatórios dos incidentes : estes documentos permitem a criação de uma base de conhecimento que poderá ser utilizada para identificar e evitar futuros incidentes de segurança;

Gestão da continuidade das atividades do ambiente: este controle diz respeito ao processo de se manter as informações íntegras, sempre acessíveis mesmo quando parte do ambiente estiver comprometido.

2.8.1 Ferramentas de suporte à aplicação da *NBR*

Todavia, realizar a análise de risco e verificar se o ambiente está atendendo à todos os controles da *NBR* não é uma tarefa fácil. Logo, para agilizar o processo de implementação, verificação e gerência dos controles da *NBR* foram, e ainda estão sendo desenvolvidas ferramentas das quais pode-se destacar o *Consultive, Objective and Bi-funcional Risk Analisis (COBRA).*

O *COBRA - Consultive, Objective and Bi-funcional Risk Analisis*

O *COBRA* consiste em um grupo de ferramentas de análise de risco, consultoria e checagem de segurança. Surgiu como resposta à crescente necessidade de gestão e verificação do nível de segurança das informações por parte das empresas.

Esta ferramenta tem sido largamente utilizada por consultores de segurança durante o processo de homologação de um ambiente à *ISO/IEC 17799*, bem como por administradores de rede ou gerentes de *TI*, que desejam submeter seus ambientes a um teste de conformidade, o qual pode ser seguido de um processo de ajustes e de um novo teste de conformidade. Independentemente de seu uso, a análise feita pelo *COBRA* é tão eficiente quanto a qualidade das informações fornecidas a ele.

O *COBRA* prevê um ambiente completo de análise de risco, baseado em questionários eletrônicos, que faz uso de um sistema simples de perguntas e respostas como um front-end de suas bases de conhecimento.

Os questionários podem ser gerados de forma automática ou manual, mas para a geração destes questionários de forma automática é necessário completar um pré-questionário (*buisiness/impact*). Assim, sem que o operador tenha prévios conhecimentos sobre segurança é possível identificar quais são os pontos fracos a serem trabalhados.

A partir dos questionários, há a avaliação das ameças, das vulnerabilidades e conseqüentemente da geração de relatórios, cada qual contendo recomendações e soluções apropriadas.

O processo de análise é realizado em três fases distintas:

Geração dos questionários : que pode ser automática ou manual. A geração manual permite uma maior flexibilidade ao processo de análise. Durante a geração dos questionários, é possível escolher quais módulos serão utilizados. Cada módulo abrange uma área particular da análise.

Exame dos riscos: nesta fase, cada módulo do questionário deve ser respondido pela pessoa responsável.

Geração dos relatórios finais: com as informações inseridas no passo anterior, uma análise completa do ambiente é realizada e no final as informações decorrentes desta análise podem ser convertidas em vários tipos de relatórios técnicos e não técnicos. Estes relatórios são compostos basicamente de:

- *Soluções e sugestões de controles adicionais*;
- *Análise e o indice de risco para cada categoria de risco*;
- *Análise de impacto*;
- *Relacionamento entre os riscos e as implicações financeiras.*

Apesar de útil, a resposta fornecida pelo *COBRA* é questionável, pois se baseia no conhecimento do(s) usuário(s) que respondeu(ram) os questionários e não no ambiente real.

O ideal seria que o mesmo realizasse ao menos um levantamento das informações de maneira ativa, ou seja, realiza-se uma varredura da rede na qual o software foi instalado e a coleta da impressão digital das máquinas detectadas pela varredura. Maiores informações sobre este processo podem ser obtidas no Capítulo 3.

Para mais informações sobre o *COBRA* e análise de risco consulte: *Security Review Products* [47], *Introduction to Security Risk Analysis*[26], *ISO 17799 World* [27], *Window to Security Risk Analysis*[53], *BS 7799 Compliance* [11] e o próprio site do *COBRA* [13].

Check-up Tool

O *Check-up Tool*, é uma ferramenta de gestão de segurança utilizada pela equipe da Módulo, para a análise de riscos para ativos tecnológicos e não-tecnológicos.

Esta ferramenta permite a verificação da aderência aos requisitos da *BS7799*, *Cobit*, *Sarbanes-Oxley* e *Basiléia II* através de uma estrutura centralizada.

Esta solução permite a coleta de informações baseada em um sistema *offline* de resposta de *checklists*, em um sistema *on-line* de entrevista dos usuários através de um módulo *WEB* e através de coletores automáticos. Contudo, somente existem coletores automáticos para os sistemas operacionais da família *Windows (MS Windows 2000 Professional, MS Windows 2000 Server, MS Windows XP Professional)* e para alguns outros produtos da *Microsoft (MS Internet Information Services e MS SQL Serve)*. Os coletores estão sendo desenvolvidos através de uma parceria da Módulo com o Laboratório de Engenharia de Software da PUC-Rio, que estão utilizando a linguagem de programação LUA para o desenvolvimento destes aplicativos.

Apesar de ser uma ferramenta que implementa uma coleta de dados mais abrangente do que o *COBRA*, a solução da Módulo foi desenvolvida especificamente para a plataforma *Microsoft Windows* e utilizando ferramentas proprietárias, como é o caso da *MS SQL Server* 2000, *IIS* 6.0 e *.Net Framework* 1.1, mas por outro lado o software em si foi desenvolvido utilizado-se da linguagem *C++*, que é uma linguagem multiplataforma.

Para mais informações sobre o *Checkup Tool* consulte seu site ou a referência [15]

Tamanduá-Mirim

Esta ferramenta é o resultado da implementação parcial do modelo de verificação proposto neste documento. E ao contrário do *COBRA*, que realiza todo o trabalho de análise meramente usando questionários eletrônicos, esta ferramenta coleta várias informações sobre o ambiente de forma automática antes de iniciar o processo de verificação da aderência do mesmo à norma de segurança.

A coleta inicial de informações realizada por este produto, se baseia na varredura da rede em que foi instalado. Esta varredura irá determinar os endereços dos nós de rede que estão atualmente ativos. Com estes dados será possível obter informações do sistema operacional instalado na máquina e de serviços ativos.

O sistema operacional será determinado utilizando-se a técnica da coleta da

impressão digital - fingerprint e os serviços ativos serão determinados através da técnica de varredura das portas do nó da rede. Estas técnicas são descritas com maiores detalhes no Capítulo 3.

Uma vez obtidas estas informações o protótipo será capaz de determinar a melhor forma de realizar a análise dos nós da rede e posteriormente de todo o ambiente.

Contudo, as duas ferramentas, *COBRA* e Tamanduá-Mirim, podem ser utilizadas em conjunto, para garantir uma melhor análise do ambiente, através do cruzamento das informações fornecidas por elas.

No Capítulo 4, é possível obter todas as informações sobre o modelo proposto e no Capítulo 5, o protótipo (Tamanduá Mirim) será descrito com mais detalhes.

Para permitir uma coleta automática das evidencias de aderência, o modelo se utiliza do ambiente de rede existente. Todavia antes de iniciar a verificação propriamente dita o modelo precisar realizar uma coleta inicial das informações sobre as máquinas que estão ativas na rede. Para este fim serão utilizadas técnicas específicas de varredura de rede e de coleta do *fingerprint* das máquinas. O próximo capítulo, com o objetivo de esclarecer sobre estas técnicas, é dedicado a este assunto.

Capítulo 3

Coleta de informações

PARA que o modelo proposto possa realizar a análise do nível de aderência do ambiente, ele deve realizar uma coleta inicial das informações sobre o mesmo. A partir das informações coletadas, será possível determinar quais nós devem ser analisados e quais componentes devem ser enviado aos mesmos.

Conforme observado no final do Capitulo 2, supõe-se a existência de um ambiente de rede no qual estão conectados as máquinas a serem analisadas, assim a coleta dos dados será realizada através de uma varredura nesta rede, o objetivo desta é identificar os endereços *IP*s ativos e os serviços de rede que estão sendo executados, bem como derterminar o sistema operacional de cada nó.

Pelos motivos acima, este capítulo irá descrever algumas das técnicas que podem ser empregadas no processo de varredura da rede, coleta de dados sobre os nós e determinação de seu sistema operacional.

3.1 Varredura da rede

Para realizar a varredura da rede, características específicas do seu protocolo devem ser exploradas. Como o protocolo de rede utilizado na especificação do modelo é o *TCP/IP*, as técnicas de varredura aqui descritas são aquelas que se baseiam nas características desta pilha de protocolo.

Sobre a pilha do *TCP/IP* existem três tipos básicos de varreduras: uma baseada

no seu protocolo de controle (*ICMP*), outra no protocolo *TCP* e a terceira no protocolo *UDP*. Dentre as varreduras, a mais utilizada é a *TCP*, isto se deve ao elevado nível de exatidão que esta coleta de dados propicia.

3.1.1 Varredura *ICMP - Internet Control Message Protocol*

Uma das técnicas mais simples de varredura é a varredura via *ICMP*. O protocolo *ICMP* é normalmente utilizado para o envio e recebimento de informações sobre o funcionamento dos nós, como por exemplo, data e hora.

Através deste protocolo é possível implementar vários métodos de varreduras, como por exemplo: envio de *ICMP ECHO* e *ECHO Reply*; análise de mensagens de erro geradas por um nó, uso de *Trace Route* com *ICMP ECHO* [4] e etc.

***ICMP ECHO* e *ECHO Reply* :** Esta varredura consiste no envio de pacotes do tipo *ICMP ECHO* ao nó alvo e aguardar por uma resposta. Todo nó ativo deve responder a esta mensagem com um pacote do tipo *ICMP ECHO Reply*. Quando um nó não responde aos pacotes *ICMP ECHO* dentro de um intervalo de tempo *(time-out)*, o mesmo pode estar desconectado ou sendo filtrado por algum *firewall* ou *gateway* de borda, como mostra a Figura 3.1.

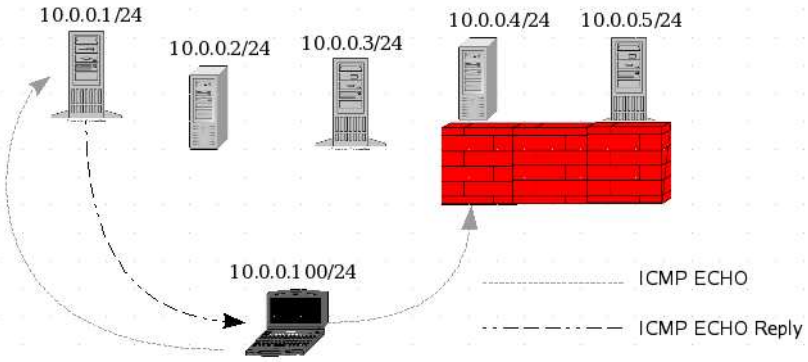

Figura 3.1: Varredura *ICMP ECHO*

Um exemplo de varredura via *ICMP* é a utilização do comando *ping* [1]

[1] O *ping* é um alicativo padrão *TCP/IP* que está presente em várias arquiteturas é permite o envio de pacotes do tipo *ICMP ECHO*.

para verificar se um nó está ativo. Porém, esta forma de verificação não é indicada para redes com mais do que poucas dezenas de nós. O *ping* sempre espera a resposda de um nó ao pacote enviado, ou até que ocorra o *time-out*, antes de enviar um outro pacote.

Uma alternativa ao *ping* é o aplicativo *fping*[2] que utiliza a mesma técnica, entretanto realiza o envio dos pacotes *ICMP* de forma simultânea para vários endereços, acelerando o processo de varredura da rede. Outra funcionalidade deste aplicativo é a possibilidade de se configurar o *time-out* da conexão. Para mais informações sobre este comando consulte a referência [20].

O uso do endereço de *broadcast* e de rede são duas alternativas que podem ser utilizadas neste tipo de varredura. Neste caso, todos os nós ativos devem responder ao nó que originou a mensagem com um pacote *ICMP ECHO Reply*. Desta forma, com um único pacote é possível varrer todo um segmento. Entretanto, esta técnica só pode ser aplicada para nós baseados no *Unix*, uma vez que máquinas com sistema operacional da *Microsoft* não respondem a solicitação *ICMP* quando enviada ao endereço de *broadcast* ou de rede[44], como pode ser observado na Figura 3.2.

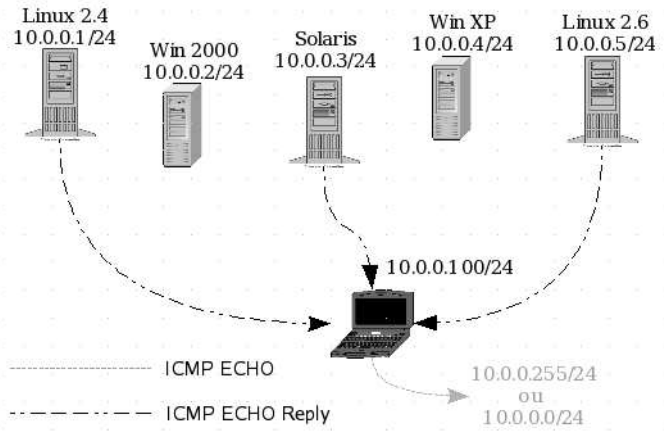

Figura 3.2: Varredura *ICMP* via *broadcast*

[2] O *fping* é utilitário de rede disponível para as plataformas *Unix Like* e que funciona de forma análoga ao *ping*, mas com o envio de pacotes de forma paralela

Pois segundo a *RFC1122* [44], se um pacote *ICMP ECHO* é enviado via endereço de *broadcast* ou de rede, o mesmo pode ser silenciosamente descartado pelos nós que o recebe.

A varredura utilizando endereço de *broadcast* ou rede, pode provocar uma negação de serviço (*D.o.S - Denial-of-Service*) no nó responsável pelo envio dos pacotes *ICMP ECHO*. Esta negação de serviço ocorre devido à grande quantidade de pacotes de resposta que ele recebe para cada pacote *ICMP* gerado.

Non-ECHO ICMP **:** Além da mensagem *ICMP ECHO*, o protocolo *ICMP* implementa vários outros tipos de mensagens que podem ser utilizadas na varredura e em alguns casos no próprio mapeamento de dispositivos de rede. Dentre as principais mensagens *ICMP* pode-se destacar: *ECHO* (*Request - Tipo 8* , *Reply - Tipo 0*); *Time Stamp* (*Request - Tipo 13* , *Reply - Tipo 14*); *Information* (*Request - Tipo 15* , *Reply - Tipo 16*) e *Address Mask* (*Request - Tipo 17* , *Reply - Tipo 18*).

O envio de mensagens do tipo *ICMP Time Stamp Request* é utilizado normalmente, para questionar a um nó a sua hora e data atual. Receber uma resposta *ICMP Time Stamp Reply*, significa que o nó está ativo. Contudo, as mensagens *ICMP Time Stamp Request* não costumam ser respondidas por nós com um dos sistemas operacionais da *Microsoft*. O comportamento dos sistemas da *Microsoft* se deve à forma como a *RFC* 1122 [44] foi interpretada pelos seus desenvolvedores.

A mensagem *ICMP Address Mask Request* é utilizada por uma máquina sem disco, para obter a máscara de rede durante seu processo de boot e também pode ser utilizada para realizar a varredura.

Quando uma solicitação deste tipo é feita através do endereço de *broadcast* todos os nós que estão configurados para responder a esta mensagem envia um *reply* com o endereço solicitado. Normalmente este tipo de mensagem é respondida por *gateways*, roteadores e em alguns casos por

máquinas, que funcionam como agentes autorizados para o fornecimento de máscara de rede.

***ICMP Error Messages*:** Este método consiste no envio de mensagens formatadas de tal modo que ao serem recebidas pelo nó produzirão uma mensagem de erro do protocolo *ICMP*.

Uma das formas de se provocar uma mensagem de erro é através do envio de pacotes *IP* que possuem em seu cabeçalho campos incorretos. Pacotes deste tipo normalmente geram uma mensagem de erro do tipo *ICMP Parameter Problem Error*, que normalmente possui o campo *Code* configurado para zero (0). Esta varredura permite identificar nós ativos, independentemente dos protocolos (*TCP/UDP/ICMP*) implementados por eles, que também pode ser empregada sobre roteadores. Contudo, roteadores realizam uma verificação mais relaxada dos campos do cabeçalho *IP*, como afirma a *RFC1812* [45].

Outra vantagem é que, esta técnica pode ser aplicada mesmo se o nó está sendo filtrado por algum *firewall*, uma vez que as mensagens forjadas podem ser enviadas para portas específicas, como por exemplo, as portas *TCP*: 21, 25, 80 e UDP 53.

***ICMP Fragment Reassembly Time Exceeded* :** Quando um nó recebe um *datagrama* fragmentado em vários pacotes e ao remontá-los percebe que estão faltando alguns, ele os aguarda por um determinado período de tempo, após o qual irá descartar todos e enviará ao nó que os originou, uma mensagem de erro do tipo: *ICMP Fragment Reassembly Time Exceeded*.

***ICMP Fragmentation Needed and Don't Fragment Bit was Set* :** Quando um roteador recebe uma mensagem, ele verifica o valor do campo *PMTU*, se este for maior do que o valor que ele está utilizando, o roteador pode retornar ao nó, que gerou o pacote, uma mensagem de erro do tipo *ICMP Fragmentation Needed and Don't Fragment Bit was Set*.

Utilizando esta técnica, é possível determinar a existência de roteadores entre a máquina alvo e a máquina atual, bem como informações específicas que podem ajudar à determinar a suposta marca e modelo deste equipamento.

3.1.2 Varredura de portas

Vários gateways de segurança impedem o envio e o recebimento de pacotes *ICMP* entre redes distintas, para inibir este tipo de varredura.

Quando as varreduras baseadas no protocolo *ICMP* não podem ser utilizadas, a vVarredura de portas é uma alternativa eficiente, uma vez que sempre existem portas através das quais um *gateway* permite o tráfego de entrada e saída, como é o caso das portas: *21 (FTP), 22 (Telnet), 23 (SSH), 25 (SMTP), 80 (HTTP), 110 (POP) e 443 (HTTPS).*

As varreduras de porta são normalmente baseadas ou no protocolo *TCP* ou no protocolo *UDP*. Devido à suas características, o protocolo *TCP* é aquele que apresenta a maior variedade de tipos de varredura.

Varredura de portas *TCP*

A varredura de portas *TCP* é feita pelo envio de pacotes à porta alvo e aguardando-se pelas respostas. Quando a porta alvo responde, os pacotes recebidos terão alguns de seus campos analisados e, com base nesta análise, será possível determinar se a porta está aberta e o nó ativo.

Atualmente, existem vários tipos de varredura *TCP*, algumas das quais são descritas a seguir.

Varredura de Conexão *TCP* : Nesta varredura, a máquina de origem tenta estabelecer uma conexão válida com a porta alvo, durante a qual deve ocorrer o *3 way handshake* do protocolo *TCP*.

Como mostra a Figura 3.3, este processo consiste no envio de um pacote *SYN* à porta alvo, no recebimento de um pacote *SYN/ACK* e finalmente

no envio do pacote *ACK* à porta alvo. Caso a conexão seja completada com sucesso, é possível afirmar que a porta está aberta e o nó ativo.

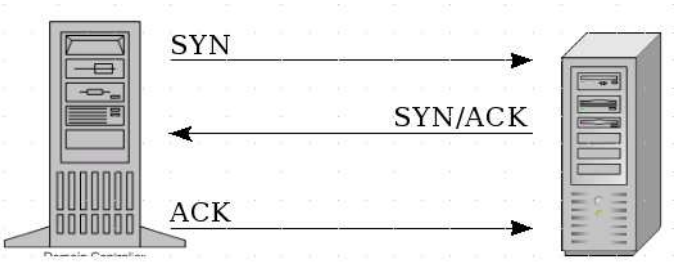

Figura 3.3: *TCP/IP: 3 way Handshake*

Por realizar uma conexão completa, esta varredura pode ser facilmente percebida por ferramentas de detecção de intrusão e por essa razão raramente é utilizada com finalidade furtiva.

Varredura *TCP SYN* : esta varredura também é conhecida como varredura semi-aberta, pois ao receber o pacote *SYN/ACK* vindo da porta alvo, o nó, que originou a primeiro pacote, não responde com o pacote *ACK*. Ao receber o pacote com os bits *SYN/ACK* setados, pode-se afirmar que a porta está aberta. Contudo, se for recebido um pacote com os bits *RST/ACK*, a porta provavelmente não está no estado de escuta.

Varredura *TCP FIN* : este técnica é muito utilizada contra sistemas baseados em *UNIX*, pois quando estes recebem um pacote *FIN* destinado à porta fechada, eles enviam de volta um pacote *RST*, contudo o mesmo não acontece com sistemas *Windows* que ignoram o pacote. Este padrão é documentado na *RFC793* [42].

Varredura *TCP Xmas Tree* : neste caso ocorre o envio à porta alvo de um pacote com os bits *FIN*, *URG* e *PUSH* ligados e segundo a *RFC793* [42], toda porta fechada deve responder com um pacote *RST*

Varredura *TCP NULL* : Quando uma porta alvo recebe um pacote com todos os flags desligados (setados como 0), ela deve responder com um

pacote *RST*, caso esteja fechada.

Varredura *TCP ACK* : Esta técnica é usada para mapear regras de um determinado *firewall*, permitindo determinar se o *firewall* é apenas um filtro de pacote ou um *firewall* de estados.

Varredura *TCP Windows* : Esta técnica analisa o tamanho do campo janela do protocolo *TCP* e permite determinar se a porta está aberta, filtrada ou não filtrada. Esta análise é possível, pois alguns *gateways* de segurança alteram o valor deste campo quando filtram determinadas portas.

Varredura *TCP RPC* : Esta técnica permite identificar qual é a versão dos aplicativos *RPC*[3] que estão ativos em um nó *Unix Like*. Alguns dos serviços que fazem uso do *RPC* são o *NFS (Network File System)* e o *NIS (Network Information Service)*.

Varredura de portas *UDP*

Apesar de existirem vários tipos de varreduras *TCP*, existe somente uma baseada no protocolo *UDP*. Esta única varredura consiste no envio de *datagramas* com zero (0) *bytes* de dados à porta alvo. A resposta pode ser uma mensagem *ICMP port unreachable* indicando que a porta está fechada, caso não ocorra nenhuma resposta, a porta pode estar aberta ou sendo filtrada.

Quando *datagramas* são enviados a várias portas de uma máquina e um grande número delas não os responde, estas provavelmente estão em estado de escuta. Por outro lado, se nenhuma porta responde, isto pode indicar que ou o nó está fora do ar ou o mesmo está sendo filtrado.

Uma forma alternativa de análise consiste na escolha de uma porta *UDP* que, segundo a lista de porta definida pela *IANA*[29], não deve estar em estado de escuta,

[3] O serviço *RPC (Remote Procedure Call)* é especifico do sistema operacional *UNIX*.

e no envio de pacotes, se a mesma não responder com uma mensagem *ICMP Port Unreachable Error* é porque ela está sendo filtrada [4].

Existe uma grande quantidade de ferramentas disponíveis para se realizar a varredura de redes de computadores, no entanto, apenas algumas se destacam pelas suas funcionalidades, dentre elas:

Varredura de *ICMP*: *Ping, Fping, Hping, Nmap, Icmpenum e Icmpquery.*

Varredura de Portas: *Nmap, Strobe, Udp_scan e Netcat.*

3.2 Coleta do sistema operacional dos nós remotos

Utilizando-se das técnicas de varredura, o modelo pode realizar o mapeamento de todo um segmento de rede, identificando nós ativos e portas abertas, porém estas técnicas não permitem determinar, com exatidão, o sistema operacional ativo em um nó remoto e portanto é necessário o uso de uma outra metodologia para este fim.

A metodologia comumente utilizada para determinar o sistema operacional de um nó se baseia na coleta de seu *fingerprint* ou sua impressão digital. Quando os desenvolvedores dos vários sistemas operacionais implementam a pilha do protocolo *TCP/IP*, eles normalmente interpretam de forma distinta as recomendações feitas pelas *RFCs* e produzem pilhas com características próprias. Estas características formam a impressão digital de um nó.

Analisando o comportamento da pilha de uma determinada versão de um sistema operacional e mapeando o resultado dentro de um banco de dados, é possível utilizar estes dados para a verificação do sistema operacional de outros nós.

A impressão digital de um nó pode ser coletada de forma ativa, através do envio de pacotes ao nó alvo e da análise do pacotes recebidos, e de forma passiva, pela análise de pacotes pertencentes a conexões já estabelecidas e coletados diretamente pela interface de rede.

3.2.1 Coleta ativa da impressão digital

A coleta ativa da impressão digital consiste no envio de pacotes préviamente formatados para o nó alvo e na análise de campos específicos dos pacotes recebidos como resposta. Neste tipo de coleta, a interface de rede não necessita ser colocada em modo promíscuo.

A coleta ativa possui um elevado nível de exatidão, mas falha quando as características da pilha *TCP/IP* são alteradas pelo seu administrador. Isto normalmente ocorre quando o administrador realiza otimizações no nó ou quando ele implementa uma política baseada em obscuridade, na qual a impressão digital é alterada para forjar um outro sistema operacional.

A coleta ativa da impressão digital de um nó pode ser realizada através de várias análises simples, tais como:

Sonda *FIN:* segundo a *RFC973* [42], quando uma porta aberta recebe um pacote *FIN* não deve responder, entretanto algumas pilhas *TCP/IP*, como é o caso da pilha do *Windows NT*, responde com uma mensagem *FIN/ACK*;

Sonda de *flag* falso : consiste no envio de um pacote *SYN* com o *flag* indefinido configurado como "1", dependendo do sistema operacional. Este pacote pode ser respondido através de um pacote com o mesmo *flag* setado em "1". Este procedimento é implementado pelo *Linux* e por outros sistemas *Unix Like*;

***Bit* não fragmentar :** para permitir um melhor desempenho do protocolo, *TCP/IP* de alguns sistema operacionais setam este campo com o valor "1" e outros, como é o caso do *Solaris*, não permitem o seu uso;

Tamanho da janela *TCP* : o tamanho deste campo é normalmente dependente do sistema operacional. Por exemplo: o *AIX* é o único sistema operacional a utilizar o valor Ox3F25 para este campo [21];

Numero seqüencial do *ACK* : esta técnica consiste em observar o valor do número seqüencial do pacote *ACK*. Alguns sistemas respondem ao pacote com o mesmo número seqüencial daquele que o originou, outros incrementam este valor em uma unidade antes de respondê-lo;

Limitação das mensagem de erro *ICMP*: alguns sistemas operacionais, seguindo as recomendações da *RFC1812* [45], limitam a taxa de envio das mensagens de erro *ICMP* a um determinado valor por intervalo de tempo. Sendo este valor padronizado, basta enviar vários pacotes *UDP* à uma porta fechada e aguaradar a mensagem *ICMP port unreachable*. Esta técnica normalmente não é muito utilizada, pois torna a coleta da impressão digital lenta [21];

Tipo de serviço *(TOS)* : a maioria dos sistema operacionais setam este campo para zero (0) quando enviam uma mensagem *ICMP port unreachable*, apesar de alguns sistemas operacionais utilizarem valores diferentes.

Opções do *TCP* : as opções básicas do protocolo *TCP* são determinadas pela *RFC793* [42] e pela *RFC1323*, já as opções avançadas são definidas pela *RFC1323*. Realizando o envio de vários pacotes com valores específicos nestes campos e analisando a resposta, pode-se supor o sistema operacional ativo, visto que nem todos os sistemas implementam todas as opções descritas nestas *RFCs*.

Existem ainda outras técnicas que não se utilizam de pacotes *TCP* ou *UDP*, ao invés disto, fazem uso de mensagens de *ICMP*. Estas técnicas foram utilizadas por Ofir Arkin na construção da ferramenta *Xprobe*. Para maiores informações sobre estas técnicas consulte as referências: [5], [6], [7], [8].

3.2.2 Coleta passiva da impressão digital

A coleta passiva consiste na captura de pacotes, pertencentes às conexões ativas, por um *sniffer* [4] e da análise dos mesmos. Esta coleta necessita que a interface de rede esteja operando em modo promíscuo[5], não gerando assim tráfego na rede.

A coleta passiva possui a grande vantagem de ser indetectável pelo nó alvo. Por outro lado, caso existam pacotes forjados circulando pela rede, a coleta passiva pode ser induzida a determinar de forma errônea o sistema operacional de alguns nós.

A análise dos pacotes coletados pela interface de rede tem por finalidade determinar as idiossincrasias nas implementações das pilhas *TCP/IP* de cada sistema operacional.

Os principais campos utilizados para identificar o sistema operacional de um nó são: o *TTL*, o *Window Size*, o *DF* e o *TOS*, pois possuem valores tabelados de acordo com o sistema operacional e sua versão.

Durante uma análise, o primeiro campo verificado é o campo *TTL*. Quando o valor deste campo é igual à 64, o pacote provavelmente pertence a um nó com Linux ou FreeBSD.

O segundo campo analisado é o *Window Size*, que normalmente possui valor igual a 0x7D78 quando a máquina possui uma versão do Windows ou do Linux instalado. O terceiro campo verificado é o *flag DF*, que indica se o pacote está fragmentado ou não. Poucos são os sistemas que setam este bit como zero, mas o SCO e o OpenBSD utilizam este padrão. O ultimo campo verificado é o *TOS*, que na maioria das mensagens está setado para zero.

Por exemplo: suponha que um dos pacotes capturados pelo *sniffer* apresente as características da Tabela 3.1. Cruzando os dados coletados com as informações acima, pode-se concluir que o nó provavelmente roda uma versão do *kernel* do *Linux*.

Outros campos também podem ser utilizados na detecção do sistema operacional

[4] *Sniffers* são utilitários de rede que operam em modo promíscuo e permitem a coleta dos dados das conexões de outras estações de trabalho, sem que ocorra a alteração da mesma ou que a ação seja detectada

[5] No modo promíscuo todo o tráfego de dados pode ser capturado por uma interface de rede, independentemente do endereço de destino

TTL	64
Window Size	0x7D78
DF bit	1
TOS	0x0

Tabela 3.1: Exemplo de dados coletados por um *sniffer* de rede

de forma passiva, mas softwares como o *Siphon*, *p0f* e o *Cheops* utilizam somente estes quatro campos para gerar e verificar as assinaturas dos sistemas operacionais.

Para maiores informações sobre esta técnica consulte as referências: [48], [36] e [16].

Capítulo 4

Modelo de verificação do nível de aderência

ESTE capítulo descreve o modelo de verificação proposto, com os seus componentes, justificativas de uso e implementação. O modelo fará uso de tecnologias e técnicas não proprietárias, e sempre que possível soluções *Open Source*, as quais irão auxiliar a sua independência de *hardware* e *software*.

4.1 Descrição do modelo

O modelo tem como base a arquitetura de rede *TCP/IP* e faz uso do modelo cliente/servidor na especificação dos agentes de software que realizarão da varredura da rede, a análise dos nós e verificação da aderência do ambiente à Norma Nacional de Segurança da Informação.

Este modelo é composto por três agentes de software (Agente de Coleta de Dados, Agente Primário de Verificação e Agente Secundário de Verificação), cada qual contém um conjunto particular de aplicativos cliente/servidor e um componente de controle e gerência dos anteriores.

Para que os agentes possam analisar o ambiente, os controles da *NBR* devem ser traduzidos nos *scripts* de verificação. Os *scripts* de verificação bem como as informações relevantes da análise são armazenadas em um repositório de dados.

Desta forma, os quatro componentes básicos do modelo de verificação são:

Agente de Coleta de Dados (A.C.D.): sua função é realizar o levantamento do perfil dos nós e a coleta de informações, através da execução dos *scripts* de verificação. O perfil será utilizado para determinar quais controles da *NBR* devem ser verificados e quais *scripts* devem ser executados.

A obtenção de dados baseada no perfil do nó garante que as informações coletadas caracterizam o ambiente real, o que nem sempre acontece quando a mesma se baseia no uso de formulários eletrônicos.

Agente Primário de Verificação (*A.P.V.*): este controla o processo de coleta das informações e realiza a verificação da aderência do ambiente à *NBR-ISO/IEC 17799*. O *A.P.V.* é o responsável pelo envio de todo e qualquer componente de software necessário para o funcionamento dos demais agentes, pelo envio dos *scripts* de verificação aos *A.C.D.s* e pela gerência da análise.

Agente Secundário de Verificação (*A.S.V.*): este é instalado nos nós que estão conectados a mais de um segmento de rede. Sua função é realizar a análise dos segmentos de rede que não podem ser verificados pelo *A.P.V.*

Devido a possibilidade da existência de vários segmentos de rede no mesmo ambiente, vários *A.S.V.* podem estar ativos, porém não existirá mais de um *A.S.V.* monitorando a mesma rede ou sub-rede.

Repositório de Dados: este é o componente responsável pelo armazenamento de todos os dados e informações necessárias para o funcionamento dos demais componentes, bem como pelo armazenamento dos *script* de verificação e das informações coletadas sobre o ambiente.

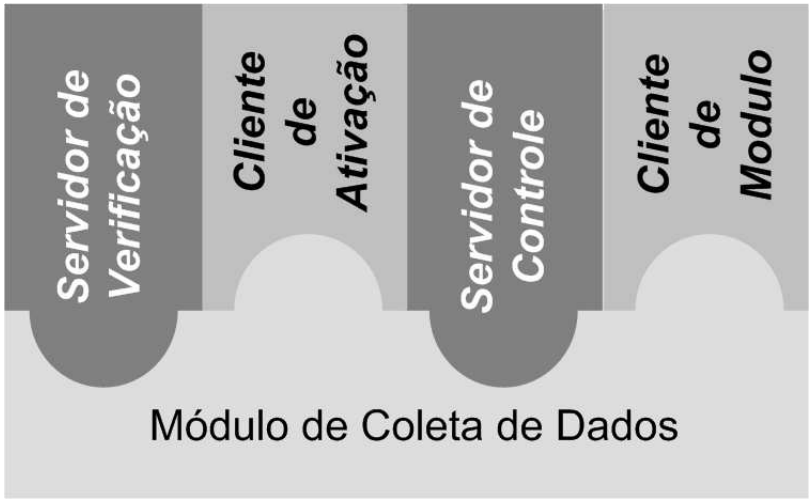

Figura 4.1: Agente de Coleta de Dados

4.1.1 Agente de Coleta de Dados - *A.C.D.*

O agente de coleta de dados é o primeiro componente do modelo que será instalado nos nós a serem analisados. Sua função é coletar as informações necessárias para a verificação da aderência do ambiente à *NBR*.

Por realizar uma coleta de dados diretamente no nó, ele evita a manipulação de informações incorretas, inconsistentes ou incompletas durante a análise, algo que poderia ocorrer em uma coleta baseada em questionários eletrônicos, como ocorre com o *COBRA*. Sem os *ACDs* não seria possível a verificação automática do nível de aderência de todo o ambiente.

O Agente de Coleta de Dados é formado pelos quatro componentes da Figura 4.1, cada um destes componentes será descrito nas próximas sessões.

Estes agentes somente entram em funcionamento após a ativação dos agentes de verificação e da execução da varredura do segmento da rede onde serão instalados.

Como mostra a Figura 4.2, o ciclo de vida de um Agente de Coleta de Dados é dividido em três fases bem distintas. A primeira vai do estado 1 ao estado 4, representando sua ativação. A segunda vai do estado 5 ao estado 9 e representa sua fase de execução. Já os estados de 10 a 12 representam sua fase de encerramento.

Fase de ativação

Durante a inicialização dos Agentes de Coleta, o primeiro elemento a ser ativado é o Módulo de Coleta de Dados, que é o responsável pelo gerenciamento, ativação e encerramento dos demais componentes.

Após o Módulo de Coleta ter sido iniciado, ele executa os servidores e solicita a ativação do *A.C.D.* ao Agente de Verificação (*A.V.*). Este processo visa informar ao Agente Primário de Verificação que existe um novo *A.C.D* ativo e pronto para receber os *scripts* de verificação.

Para realizar esta ativação, o *A.C.D.* fecha com o *A.V.* uma conexão na qual será informado o seu endereço *IP* e a sua Chave de Ativação. Em resposta ao pedido de ativação, o *A.C.D.* deve receber a confirmação de ativação e a Chave de Comunicação, ou um código de erro informando que o *A.C.D.* já estava ativo.

A Chave de Ativação é criada pelo *A.P.V.* durante a geração do pacote de instalação do agente de coleta. A Chave de Comunicação e o endereço *IP* da máquina servirão para identificar univocamente o *A.C.D.* Sem sua chave de comunicação um *A.C.D.* não pode ser contactado ou contatar um *A.V.*

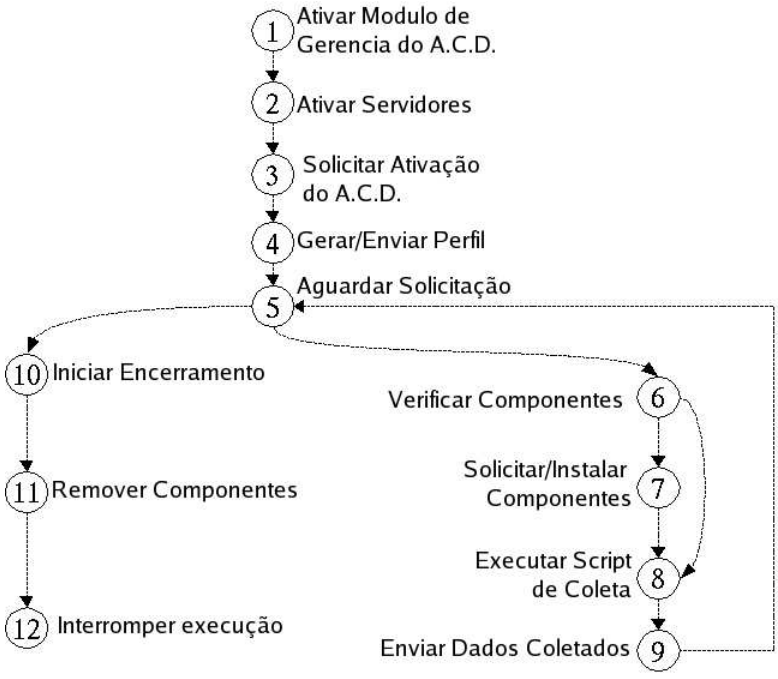

Figura 4.2: Ciclo de Vida do Agente de Coleta de Dados

Após a fase de ativação, ocorre o processo de coleta do perfil da máquina. Esta tem por objetivo adquirir as informações que caracterizam a máquina, dentre elas: o(s) endereço(s) *IP*(s) disponível(is); seu nome; nome e versão do sistema operacional; lista dos serviços que estão sendo executados; lista dos aplicativos instalados; quais portas *TCP/IP* estão abertas; entre outros dados.

O perfil tem função vital no processo de coleta de dados, a partir das informações coletadas será possível definir quais controles da *NBR-ISO/IEC 17799* devem ser testados no nó. Dependendo do seu perfil, scripts de verificação específicos serão enviados ao Agente de Coleta.

Fase de execução

Esta fase é caracterizada pelo recebimento, execução dos *scripts* de verificação e envio das informações coletadas ao agente de verificação. O primeiro *script* de verificação a ser recebido é o que solicita o envio do perfil da máquina.

Feito o envio do seu perfil ao Agente de Verificação, o Agente de Coleta entra em um ciclo de recebimento e execução de vários *scripts*, cada qual gerando uma saída que deve ser enviada ao Agente de Verificação, para posterior análise.

Para cada *script* recebido, o Agente de Coleta deve verificar se os componentes necessários para executá-lo estão disponíveis, caso haja componentes faltando, devem ser solicitados ao agente de verificação e posteriormente instalados. Somente após a verificação e instalação dos componentes é que um *script* pode ser executado.

O agente de coleta irá receber tantos *scripts* quanto forem necessários para verificar todos os controles da norma de segurança, uma coleta somente termina quando não há mais *scripts* a serem enviados.

Fase desativação

A fase de execução é encerrada de forma normal quando não há mais *scripts* de verificação a serem enviadas para o *A.C.D.*, neste caso o agente apenas recebe um sinal do Agente de Verificação indicando que deve ser iniciada a Fase de Desativação.

Ao receber o sinal de desativação, o *A.C.D.* se incumbe de restaurar o nó ao

seu estado original, removendo todo e qualquer módulo extra instalado, assim como qualquer modificação realizada pelo *A.C.D.* nos arquivos de configuração e/ou configurações em memória.

Para permitir a desinstalação dos módulos e remoção das alterações nas configurações, o *A.C.D.* deve possuir um banco de dados local, onde armazenará todas as modificações realizadas no sistema.

4.1.2 Agente Primário de Verificação - *A.P.V.*

O Agente Primário de Verificação (*A.P.V.*) é o nó central da rede de verificação, como em uma rede com topologia estrela, é o responsável pela comunicação e gerenciamento do processo de troca de mensagem do ambiente.

O *A.P.V.* é o primeiro agente a ser acionado e está hospedado na mesma máquina que o Repositório de Dados, com os **scripts** e os módulos utilizados na verificação.

O *A.P.V.* enviará aos Agente de Coleta os scripts de verificação e receberá de volta os *logs* de saída destas rotinas. Ao receber um *log* de saída, o *A.P.V.* analisa o seu conteúdo e com base nela será determinado quais controles da Norma de Segurança estão sendo implementados e quais não estão.

Findo o processo de coleta e análise dos dados de todos os nós do ambiente, o Agente Primário de Verificação será capaz de informar seu nível de aderência à Norma Nacional de Segurança.

Os seis módulos que formam o Agente Primário de Verificação podem ser observados na figura 4.3, e cada um deles será descrito detalhadamente nas próximas sessões.

Como mostra a Figura 4.4, o ciclo de vida do *A.P.V.* também é dividido em três fases: a fase de ativação, representada pelos estados de 1 à 5, a fase de execução, composta pelos estados de 6 ao 16 e a fase de encerramento, composta pelos estados 17, 18 e 19.

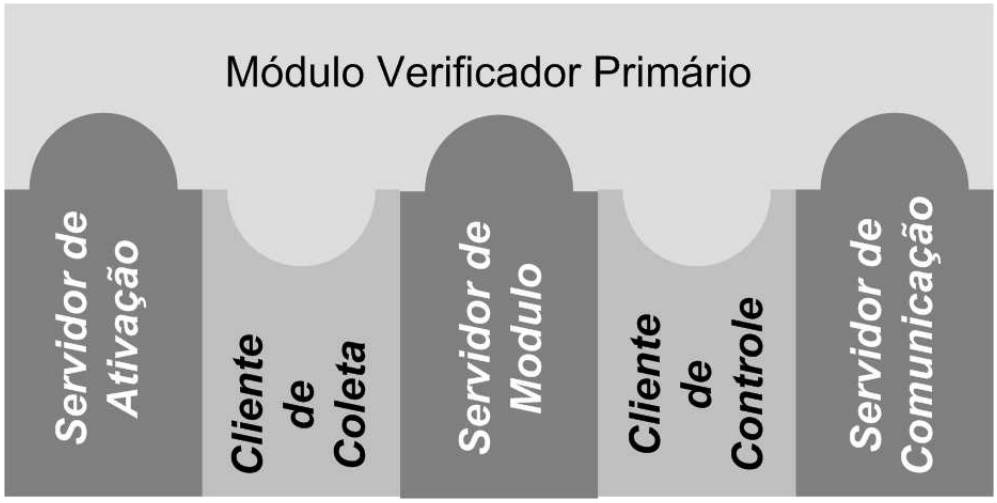

Figura 4.3: Agente Primário de Verificação

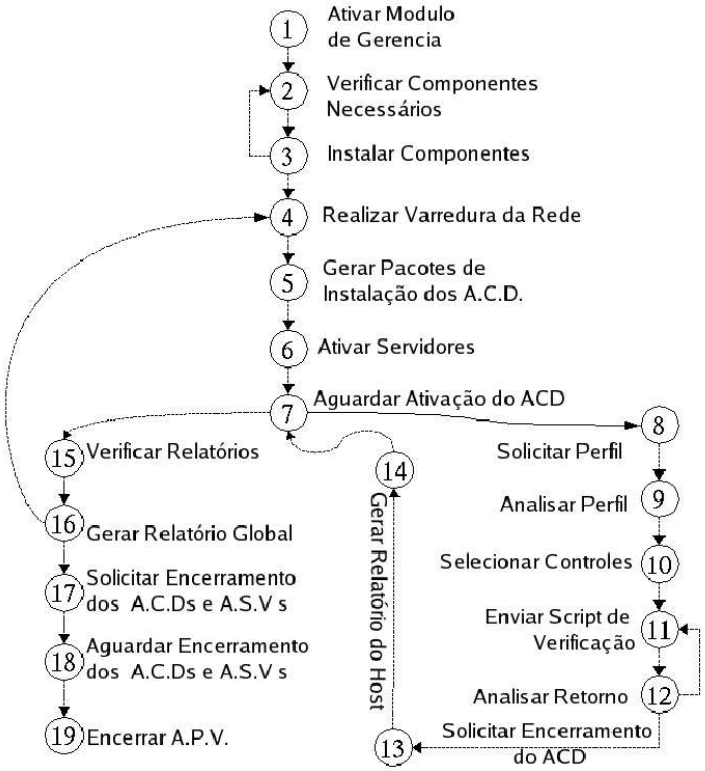

Figura 4.4: Ciclo de Vida do Agente Primário de Verificação

Fase de ativação

O primeiro elemento deste agente a ser iniciado é o Módulo de Verificação, que realizará uma vistoria da máquina em busca dos elementos necessários para a execução dos demais componentes, na ausência de um elemento, o mesmo será extraido do Repositório de Dados e instalado.

Finda a verificação inicial do nó, inicia-se o processo de coleta do perfil. Este perfil conterá: o endereço(s) *IP*(s) do nó, a(s) rede(s) na(s) qual(is) está conectado, serviços ativos, sistema operacional, versão (do *kernel*) além de outras informações importantes.

Com o endereço *IP* e a máscara de rede da máquina onde está instalado, o módulo realizará a varredura deste segmento da rede no qual o *A.P.V.* está conectado.

A varredura visa determinar quais máquinas / endereços *IP*s estão ativos e a partir destas informações determinar quais os serviços e sistema operacional esta sendo executados em cada nó. [1]

Ao obter a lista de *IP*s ativos e dos sistemas operacionais, o Agente de Verificação determinará em quais nós os *A.C.D.* devem ser instalados e qual versão deve ser aplicada.

A existência de várias versões para o *A.C.D.* se deve ao fato de diferentes arquiteturas de *hardwares* e diferentes sistemas operacionais necessitarem de *softwares* diferentes, mesmo quando seu código fonte é idêntico. Situação análoga acontece com os *scripts* de verificação, que também são dependentes da arquitetura de *hardware e software* adotada na máquina a ser analisada.

Ao término da varredura do segmento de rede, o *A.P.V.* deve armazenar, para cada nó detectado, as seguintes informações no Repositório de Dados: seu endereço *IP*, o seu nome, a versão do seu sistema operacional, sua Chave de Ativação e seu estado [2].

Após o armazenamento das informações citadas acima, serão gerados pacotes de instalação dos *A.C.D.* Estes pacotes de instalação são compostos pelos executáveis

[1] Mais informações sobre o processo de varredura da rede pode ser obtido no capítulo anterior.

[2] Normalmente os nós quando adicionados ao repositório de dados estão no estado "0" (*A.C.D* inativo)

que formam o *A.C.D*, pela chave de ativação e por alguns scripts de gerência.

A chave acima citada, deve ser única para cada nó e quando combinada ao endereço *IP* do nó, serve para autenticar *A.C.D.* durante o seu processo de ativação.

Fase de execução

O início desta fase é identificado pela inicialização dos Servidores de Ativação, de Módulo e de Comunicação.

Nesta fase o *A.P.V.* aguarda a solicitação de ativação dos *A.C.D.*s, ocorrendo esta solicitação, o *A.P.V.* autenticará o *A.C.D.* tendo como base o endereço *IP* do *A.C.D.* e sua chave de ativação, caso ocorra uma autenticação positiva, o *A.P.V.* altera o estado do *A.C.D.* para um (1) no repositório de dados, gera a chave de comunicação e envia ao *A.C.D.* uma mensagem contendo a nova chave e a confirmação de ativação.

Ao perceber a ativação de um *A.C.D.*, o Módulo de Verificação solicitará o perfil do agente de coleta, ao receber este perfil o módulo analisará as características daquele nó e determinará quais *scripts* de Verificação devem ser enviados ao *A.C.D.*

Para cada *script* de verificação enviado, o *A.P.V.* recebe um arquivo contendo o resultado da coleta, que será analisado para determinar se o controle da Norma de Segurança pode ou não ser homologado.

O processo de coleta de dados de um *A.C.D.* chega ao fim somente quando todos os *scripts*, previamente selecionados, já tiverem sido analisados. Pela análise dos arquivos enviados por um *A.C.D.*, será possível determinar o nível de aderência deste nó à *NBR-ISO/IEC 17799*.

Quando um *script* de verificação requer um módulo que não está instalado, cabe ao *A.C.D.*, através do Cliente de Módulos, realizar a sua solicitação ao agente de verificação e posteriormente instala-lo.

Quando não houver mais nenhum *A.C.D.* a ser verificado, o *A.P.V.* emitirá um relatório informando quais controles da *NBR ISO/IEC 17799* foram homologados no ambiente e conseqüentemente seu nível de aderência à norma. Para os controles que não foram certificados será emitido um aviso indicando o motivo da não homologação.

Com base nas informações emitidas, o auditor ou o gerente do ambiente poderá realizar as ações necessárias para a correção dos problemas e em seguida poderá submeter o ambiente a uma nova verificação.

Fase de desativação

Quando o relatório final tiver sido apresentado, será solicitado ao usuário a confirmação do fim da análise. Com a confirmação, será enviado um sinal para todos os *A.C.D.*s e *A.S.V.*s para que sejam encerrados. Somente após o encerramento dos demais componentes é que o *A.P.V.* pode iniciar sua fase de desativação.

Durante a fase de desativação, todo e qualquer componente adicionado e qualquer alteração realizada pelo *A.P.V.* nas configurações da máquina devem ser removidas, esta remoção deve retornar a máquina ao seu estado original.

Somente quando todos os componentes já tiverem sido removidos e todas as configurações restauradas, o *A.P.V.* será descarregado da memória.

4.1.3 Agente Secundário de Verificação - *A.S.V.*

O Agente Secundário de Verificação (*A.S.V.*), é instalado nas máquinas que possuem o *A.C.D.* ativo e que estejam conectadas a dois, ou mais, segmentos de rede. O primeiro segmento, já possuindo um processo de análise em andamento, e os demais serão gerenciados por este *A.S.V.* que acaba de ser acionado

Como o *A.S.V.* não possui acesso direto ao repositório de dados instalado no mesmo nó que o *A.P.V.*, toda e qualquer solicitação de módulo e/ou script de verificação deve ser repassada ao *A.P.V.* através do cliente de comunicação. Quando um *A.S.V.* recebe uma informação vinda do *A.P.V.*, esta deverá ser armazenada localmente até que o *A.S.V.* receba a confirmação de que a informação chegou ao seu destino.

O Agente Secundário de Verificação, como mostra a Figura 4.5, é formado por oito componentes, os quais serão descritos nas próximas sessões.

Os servidores secundários funcionam de forma semelhante aos servidores do A.P.V., mas em vez de tratarem as informações recebidas, eles as repassam ao

A.P.V. O Agente Primário de Verificação, ao receber uma solicitação de um A.S.V, processa a solicitação e a envia de volta ao A.S.V., que por sua vez a encaminha para o Agente que fez a requisição.

Um *A.S.V.* funciona como uma ponte de comunicação entre os *A.C.D.*s instalados na nova rede com o *A.P.V.*, que está localizado no nível mais alto da rede de verificação.

Como alguns ambientes são formados por vários segmentos de rede, mais do que um único *A.S.V.* pode ser necessário para cobrir todo o ambiente. Contudo, um novo *A.S.V.* somente será necessário se um dos *A.C.D.*s estiver conectado a uma rede que ainda não está sendo monitorada. As informações sobre quais redes estão sendo monitoradas é armazenada no repositório de dados do *A.P.V.*

Em um ambiente com vários segmentos de rede, o modelo pode criar uma grande rede hierárquica de comunicação, na qual o *A.P.V.* é o nó central que controla todos os demais agentes, e os *A.S.V.*s funcionam como concentradores de comunicação nas camadas inferiores da estrutura, como pode ser observado na Figura 4.6.

O ciclo de vida em um *A.S.V.*, como mostra a Figura 4.7, é composto pelas mesmas fases dos agentes anteriores. A fase de ativação é representada pelos estados

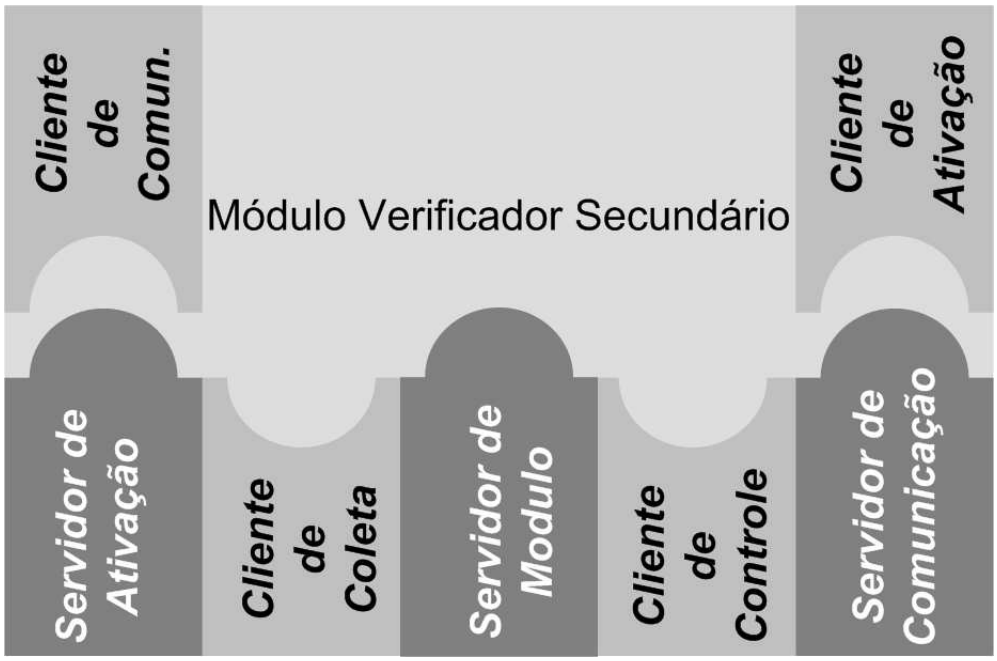

Figura 4.5: Agente Secundário de Verificação

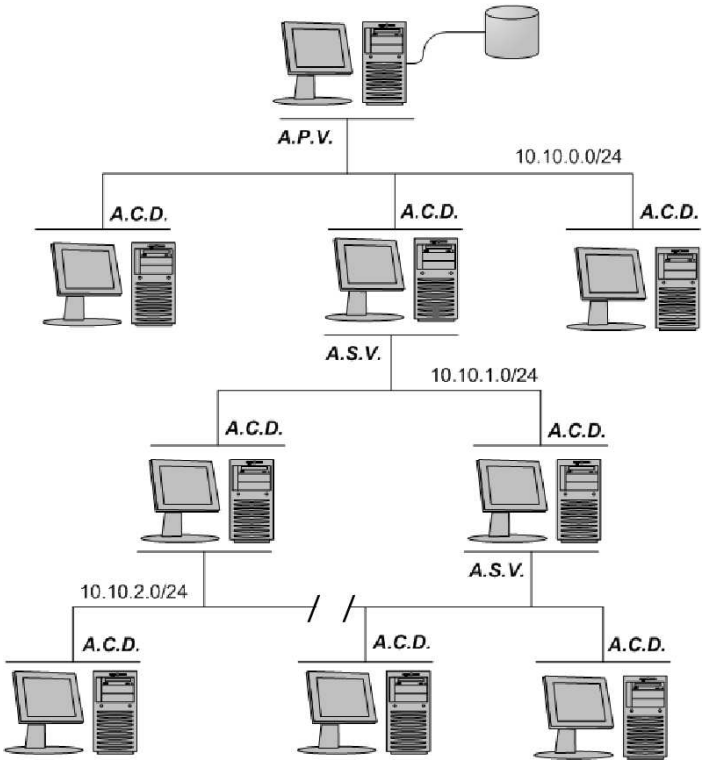

Figura 4.6: Rede hierárquica de comunicação

de 1 até 4, a fase de execução pelos estados de 5 até 16 e a fase de encerramento pelos estados de 17 até 20.

Fase de ativação

Quando o *A.P.V.* recebe o perfil de um *A.C.D.* que está conectado a mais de uma rede e pelo menos uma delas não está sendo monitorada, ele solicita ao *A.C.D* que realize a instalação e ativação do *A.S.V.*, permitindo assim a análise deste novo segmento.

Durante a fase de ativação, o Agente Secundário verifica a disponibilidade dos componentes necessários para a sua execução. Ocorrendo a falta de módulos, estes serão solicitados ao *A.P.V.*

Com todos os componentes instalados e funcionando, o *A.S.V.* realiza a varredura dos novos segmentos de rede. Este processo tem o mesmo objetivo da varredura realizada pelo Agente Primário. Encerrada a varredura, o *A.S.V* . envia uma cópia da mesma ao *A.P.V.* que ao recebê-la determinará em quais nós devem ser instalados

os novos Agentes de Coleta.

Fase de execução

Esta fase é iniciada pela inicialização dos servidores de ativação, comunicação e de módulo, com estes servidores ativos o agente passa a gerenciar o novo segmento. Como o *A.S.V.* não possui acesso direto ao repositório de dados do *A.P.V.*, toda e qualquer solicitação de informação ou componente de *software* deve ser repassada ao *A.P.V.* Quando o *A.S.V.* recebe os componentes solicitados, ele os repassa ao agente correspondente.

Cada componente recebido pelo *A.S.V.* será temporariamente armazenado em uma *cache* local, assim, antes de realizar a solicitação de um novo componente, o agente secundário deve fazer uma busca na sua cache, deste modo, evita-se gerar tráfego desnecessário na rede.

Quando um *A.C.D.*, subordinado ao *A.S.V.* realiza a sua ativação, o agente

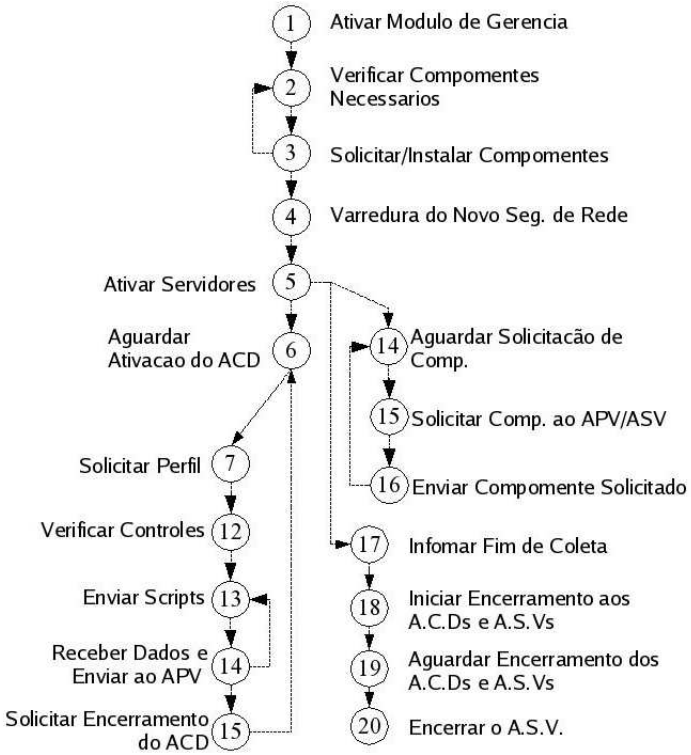

Figura 4.7: Ciclo de vida do Agente Secundário de Verificação

secundário se comporta de forma análoga ao agente primário, mas repassa todas as solicitações e informações ao *A.P.V.*.

Fase de desativação

Quando não existem mais *A.C.D.*s para serem analisados, o *A.S.V.* informa o fim da coleta de dados ao *A.P.V.* e aguarda pelo sinal de encerramento.

Ao receber a solicitação de encerramento o *A.S.V.* repassa esta solicitação aos agentes que estão subordinados a ele e, somente quando todos os agentes subordinados tiverem sinalizado seu encerramento, é que o *A.S.V.* inicia a restauração das configurações do nó ao seu estado original. Finda a restauração do nó, o Agente Secundário informa ao *A.P.V.* que está sendo desativado. Confirmada a solicitação, ele finaliza a sua execução.

4.1.4 Repositório de Dados

O repositório de dados é o banco de dados principal do modelo de verificação, este deve ser armazenado na mesma máquina onde o Agente Primário está ativo. Sem este banco de dados não seria possível o processo de verificação e análise da aderência do ambiente à norma *NBR ISO/IEC 17799*.

Neste banco estão as tabelas com as informações sobre a Norma Nacional de Segurança e seus controles; os *scripts* de verificação; a estrutura básica para a instalação dos agentes (*A.C.D. , A.S.V e A.P.V.*); as informações coletadas sobre cada nó do ambiente; os *softwares* necessários para o funcionamento dos agentes, execução dos *scripts* e para a coleta do perfil de cada nó, assim como o resultado da análise.

A representação do diagrama de entidade e relacionamento (*DER*) do repositório de dados pode ser observada na Figura 4.8, nesta temos as 12 tabelas do banco de dados e a representação gráfica dos relacionamentos. Em seguida, para complementar as informações desta figura, há uma breve descrição das 12 tabelas do banco.

Os símbolos: $\vdash\!\!\!\vdash$ e $\dashv\!\!\!\triangleleft$, encontrados na Figura 4.8, representam a cardinalidade

dos relacionamentos do banco de dados, o primeiro elemento indica o relacionamento com um único elemento e o segundo indica o relacionamento com um ou vários elementos.

A tabela **Controles** é composta pela descrição dos controles que fazem parte da *NBR ISO/IEC 17799:2000*. Esta tabela é utilizada para indicar quais *scripts* de verificação estão subordinados a um determinado controle e seus campos podem ser observados na Tabela 4.1.

O campo *cd_controle* é utilizado para identificar de forma única o controle, o campo *nm_controle* representa o nome do controle da *NBR*, e o campo *ds_controle* é um breve descrição do mesmo.

Campo	Tipo de Dados
cd_controle	Int
nm_controle	Varchar (25)
ds_controle	Varchar (100)

Tabela 4.1: Estrutura da Tabela de Controles

A tabela ***Script*** é a responsável por armazenar todos os *script* de verificação. Cada *script* está relacionado à um único controle da norma de segurança e será

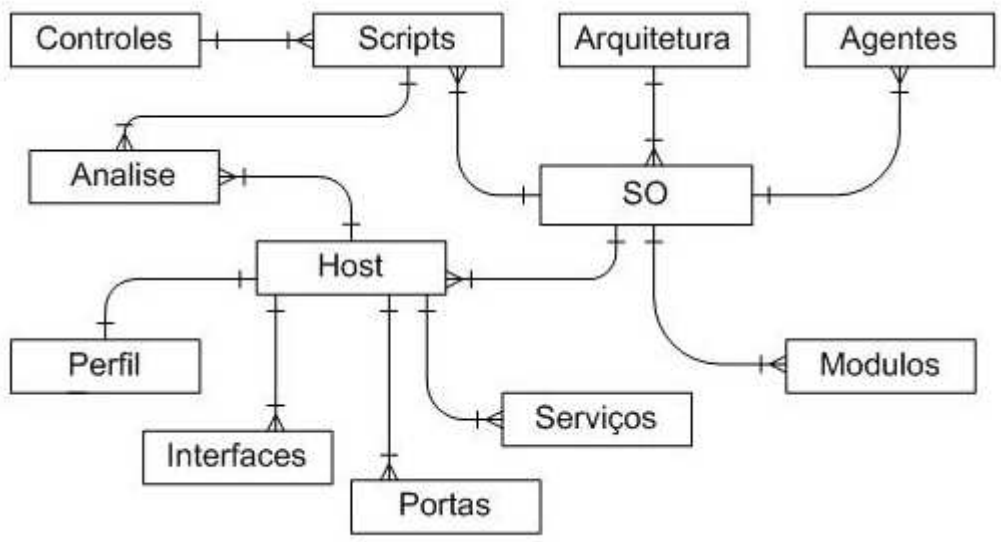

Figura 4.8: Diagrama de Entidade e Relacionamento do repositório de dados.

enviado aos Agentes de Coleta de acordo com o seu perfil, permitindo desta forma, realizar a análise do ambiente. Sua estrutura pode ser observada na Tabela 4.2. O campo cd_script é utilizado na identificação única do elemento, cd_controle associa o *script* a um controle previamente cadastrado, cd_so informa para qual sistema operacional o *script* foi desenvolvido e finalmente ds_script contem uma breve descrição do mesmo.

Campo	Tipo de Dados
cd_script	Int
cd_controle	Int
cd_so	Int
ds_script	Varchar (100)

Tabela 4.2: Estrutura da Tabela de *Scripts*

A tabela **Módulos** hospeda os componentes de *software* que podem ser necessários para a execução dos clientes, servidores, módulos de controle e *scripts* de verificação, utilizados pelo modelo. E ainda possui os campos apresentados na Tabela 4.3. O campo cd_modulo é utilizado para identificar de forma única cada elemento, o cd_so identifica o sistema operacional no qual ele pode ser utilizado, ds_modulo é uma breve descrição do módulo e id_modulo é um pequena identificação textual do módulo.

Campo	Tipo de Dados
cd_modulo	Int
cd_so	Int
ds_modulo	Varchar (100)
id_modulo	Varchar (25)

Tabela 4.3: Estrutura da Tabela de Modulo

Já a tabela **Agentes** armazena a estrutura básica de instalação de cada um dos agentes que compõem o modelo. Para complementar as informações contidas nesta tabela, uma chave de ativação é gerada sempre que um pacote de instalação é criado. A tabela tem sua estrutura descrita na Tabela 4.4. O campo cd_agente identifica

de forma única o elemento dentro do banco de dados, o cd_so informa para qual sistema operacional o agente foi desenvolvido, nm_agente corresponde ao nome do agente e ds_agente é uma breve descrição do mesmo.

Campo	Tipo de Dados
cd_agente	Int
cd_so	Int
nm_agente	Varchar (25)
ds_agente	Varchar (100)

Tabela 4.4: Estrutura da Tabela de Agentes

A tabela ***Hosts*** possui as informações básicas sobre cada nó existente no ambiente. Sua estrutura pode ser observada na Tabela 4.5. Seu campo vl_ip identifica o endereço *IP* do nó, ds_estado informa se existe algum agente ativo, vl_chave_at é o valor da chave de ativação do nó e finalmente vl_chave_com é o valor da chave de comunicação.

Campo	Tipo de Dados
vl_ip	Varchar(15)
ds_estado	Varchar(1)
vl_chave_at	Varchar(10)
vl_chave_com	Varchar(10)

Tabela 4.5: Estrutura da Tabela de *Hosts*

A tabela **Perfil**, como o próprio nome diz, armazena as informações sobre o perfil de cada nó existente no ambiente e sua estrutura é mostrada na Tabela 4.6. O campo vl_ip identifica o endereço *IP* do nó, nm_host é o nome do nó, ds_dist é o nome do sistema operacional instalado, ds_kernel é a identificação do seu *kernel* ou sua versão, ds_arq informa qual é a arquitetura do nó, qt_mem informa a quantidade de memória instalada, ds_proc_vel armazena a velocidade do processador, ds_proc_mod representa o modelo do processador, ds_mod é o modelo do nó, ds_aut é o tipo de autenticação de usuário utilizada, vl_dns1 e vl_dns2 são os valores de seu servidor de *DNS* primário e secundário, vl_gateway é o endereço do roteador ou *gateway* de borda utilizado, vl_proxy é o endereço do servidor de *proxy*.

Campo	Tipo de Dados
vl_ip	Varchar(15)
nm_host	Varchar (40)
ds_dist	Varchar (20)
ds_kernel	Varchar (10)
ds_arq	Varchar (10)
qt_mem	Varchar (6)
ds_proc_vel	Varchar (6)
ds_proc_mod	Varchar (10)
ds_mod	Varchar (100)
ds_aut	Varchar (40)
vl_dns1	Varchar (15)
vl_dns2	Varchar (15)
vl_gateway	Varchar (15)
vl_proxy	Varchar (15)

Tabela 4.6: Estrutura da Tabela de Perfil

Para complementar as informações contidas, na tabela perfil existem quatro tabelas auxiliares. A primeira se chama **Sistema de Arquivo**, e possui a estrutura exibida pela Tabela 4.7, e contém informações sobre as partições e pontos de montagens ativos nas máquinas.O campo vl_ip identifica o endereço *IP* do nó, ds_particao identifica a partição, ds_montagem informa onde a partição está sendo montada, vl_tipo especifica o tipo de sistema de arquivo utilizado e ds_perm guarda as informações sobre as permissões de acesso da partição.

Campo	Tipo de Dados
vl_ip	Varchar(15)
ds_particao	Varchar (15)
ds_montagem	Varchar (100)
vl_tipo	Int
ds_sist_arq	Varchar (10)
ds_perm	Varchar (100)

Tabela 4.7: Estrutura da Tabela de Sistema de Arquivo

A Segunda é a tabela **Interfaces**, que contém informações sobre as interfaces de rede que o nó possui, bem como seus endereços *IP*s. Sua estrutura pode ser observada na Tabela 4.8. O campo vl_ip identifica o endereço *IP* principal do nó, ds_nome especifica o nome da interface de rede, vl_ip2 é o endereço secundário do nó, ds_mascara é a mascara de rede utilizada nesta interface.

Campo	Tipo de Dados
vl_ip	Varchar(15)
ds_nome	Varchar (15)
vl_ip2	Varchar (15)
vl_mascara	Varchar (15)

Tabela 4.8: Estrutura da Tabela das Interfaces de Rede

A terceira é a tabela **Serviços**, que armazena a lista dos serviços que estão sendo executados e oferecidos por um determinado nó, às demais máquinas conectadas a rede. Sua estrutura pode ser vista na Tabela 4.9. O campo vl_ip identifica o

endereço *IP* do nó, ds_nome é o nome do serviço em execução, ds_usuario é a identifica do usuário que iniciou o serviço.

Campo	Tipo de Dados
vl_ip	Varchar(15)
ds_nome	Varchar (30)
ds_usuario	Varchar (20)

Tabela 4.9: Estrutura da Tabela dos Serviços

A última tabela relacionada o perfil das máquinas é a tabela **Portas**. Esta armazena todas as portas que estão abertas em um determinado nó e sua estrutura é representada pela Tabela 4.10. O campo vl_ip identifica o endereço *IP* do nó, vl_nome é o número da porta que está aberta, ds_usuario é a identifica do usuário que iniciou o serviço associado a porta.

Campo	Tipo de Dados
vl_ip	Varchar(15)
vl_porta	Varchar (6)
ds_usuario	Varchar (20)

Tabela 4.10: Estrutura da Tabela de Portas

Para gerenciar o processo de análise dos nós existe a tabela **Análise**, que possui as informações sobre o resultado de execução dos *scripts* de verificação em todos os nós do ambiente. Sua estrutura pode ser vista na Tabela 4.11. O campo cd_analise é utilizado para identificar únicamente o resultado de uma análise, cd_script informa qual foi o *script* analisado, vl_ip informa qual foi a máquina analisada, ds_saida é a saída produzida pelo *script* e ds_resultado é o resultado obtido com a análise da saída do *script*.

Como vários componentes são dependentes da arquitetura na qual serão executadas, existe uma última tabela denominada SO, que irá identificar o sistema operacional para o qual estes foram desenvolvidos. A estrutura desta é apresentada na Tabela 4.12. O campo cd_so é utilizado para identificar de forma única um sistema operacional e o campo nm_so representa o nome do sistema operacional.

Campo	Tipo de Dados
cd_analise	Int
cd_script	Int
vl_ip	Varchar(15)
ds_saida	longtext
ds_resultado	Varchar (20)

Tabela 4.11: Estrutura da Tabela de Análise do Ambiente

Campo	Tipo de Dados
cd_so	Int
nm_so	Varchar(20)

Tabela 4.12: Estrutura da Tabela SO

4.2 Descrição dos clientes/servidores

Nesta sessão será abordado o funcionamento dos clientes e servidores utilizados na implementação dos agentes que compõem o modelo proposto e a estrutura das mensagens trocadas entre eles.

Os clientes e servidores fazem uso da arquitetura *TCP/IP* e de *Stream Sockets* para se comunicarem. A troca de mensagens entre estes é feita por meio de portas não privilegiadas que segundo a **RFC1700**, [43] não estão reservadas para uso.

4.2.1 Cliente/Servidor de Ativação

O Servidor de Ativação é parte integrante dos Agentes de Verificação e o Cliente faz parte dos Agentes de Coleta de Dados. O Servidor de Ativação deve ser acessado através da porta 7600/*TCP* considerada uma porta não privilegiada como comentado no item anterior.

Através destes dois componentes, é possível determinar: quais máquinas hospedam os Agentes de Coleta e os Agentes Secundários de Verificação; gerenciar as chaves de ativação e de comunicação utilizadas na troca de dados entre os Agentes; e de-

terminar o estado de todos os nós da rede de verificação.

As chaves mencionadas acima, são dois números aleatórios associados ao endereço *IP* de cada *A.C.D.* Estas chaves servem como um mecanismo de autenticação dos clientes e servidores. A chave de ativação é criada quando o pacote de instalação do *A.C.D.* é construído, e utilizada quando o *A.C.D.* realiza seu processo de ativação.

Ao passar pelo processo de ativação, o *A.C.D.* utiliza o seu Cliente de Ativação para se conectar ao *A.V.* e informar que já está pronto para receber os *scripts* de verificação. Ao confirmar a ativação do *A.C.D.*, o Servidor de Ativação envia uma nova mensagem contendo a chave de comunicação, que será utilizada nas comunicações futuras com o Agente de Verificação.

A cada nova conexão, o Servidor de Ativação recebe e emite duas mensagens, cada qual com uma formatação própria e um significado específico. A mensagem recebida pelo servidor é a *Mensagem de Conexão*. Esta mensagem é formada pelos campos da Figura 4.9 e descritos a seguir.

```
struct conect.in   {
                   char   ip       [15]
                   char   status   [1]
                   char   chave1   [10]
                   }
```

Figura 4.9: Cliente/Servidor de Ativação: Mensagem de Conexão

IP **:** campo formado por uma seqüencia de 15 caracteres alfanuméricos e que representa o endereço *IP* da máquina que está realizando a conexão, normalmente um *A.C.D.*;

CHAVE 1 : campo formado por uma seqüencia de 10 caracteres numéricos. Quando esta conexão fizer parte do processo de ativação do *A.C.D.*, o valor deste campo será o valor da chave de ativação, caso o *A.C.D.* já esteja ativo e utilizando

esta conexão para uma alteração de estado, o conteúdo deste campo será igual ao valor da sua chave de comunicação.

STATUS **:** este campo é formado por um único caracter numérico e utilizado para sinalizar qual é a mudança de estado que está sendo solicitada. Ele pode assumir um dos seguintes valores:

0 : Desativar *A.C.D.*;

1 : Ativar *A.C.D.*;

2 : Desativar *A.S.V.*;

3 : Ativar *A.S.V.*

Em resposta à Mensagem de Conexão, o servidor emite a Mensagem de Confirmação, cuja estrutura é apresentada na Figura 4.10 e cada é campo descrito logo a seguir.

```
struct conect.out   {
                    char   ip       [15]
                    char   status   [1]
                    char   chave1   [10]
                    char   chave2   [10]
                    }
```

Figura 4.10: Cliente/Servidor de Ativação : Mensagem de Confirmação

IP **:** este campo é formado por uma seqüencia de 15 caracteres alfanuméricos, que representam o endereço *IP* da máquina que realizou a solicitação (*A.C.D.*);

CHAVE1 : campo formado por uma seqüência de 10 caracteres numéricos. Caso faça parte de um processo de ativação, seu valor será igual a chave de ativação, caso contrário seu valor será o mesmo da chave de comunicação associada ao endereço *IP*;

CHAVE2 : campo formado por uma seqüencia de 10 caracteres numéricos. Somente é utilizado quando da ativação do *A.C.D.*, nos demais caso o seu valor será sempre NULO.

***STATUS* :** indica se a alteração do estado do *A.C.D.* ou *A.S.V.* foi permitida ou não. Este campo é formado por um único caracter numérico e pode assumir os seguintes valores:

0 : Confirmada a desativação do *A.C.D.* e *A.S.V.*;

1 : Confirmada a ativação do *A.C.D.*;

2 : Confirmada a ativação do *A.C.D.* e desativação do *A.S.V.*;

3 : Confirmada a ativação do *A.S.V.*;

4 : *A.C.D.* já estava ativo no endereço informado;

5 : *A.S.V.* já estava ativo no endereço informado;

6 : *A.S.V.* não pode ser ativado antes do *A.C.D.*;

7 : *A.C.D.* não pode ser desativado antes do *A.S.V.*;

8 : *A.C.D.* já estava inativo;

9 : *A.S.V.* Já estava inativo;

A : A chave de ativação não confere;

B : A chave de comunicação não confere;

C : O endereço *IP* não está cadastrado;

D : Erro desconhecido.

Caso o servidor receba uma conexão com o valor do endereço *IP* inválido, ele responde a mensagem apenas trocando o valor do *STATUS* para "C", indicando que endereço *IP* não é conhecido.

Ao receber uma conexão com *STATUS* igual a "0", ou seja uma solicitação de encerramento do *A.C.D.*, o servidor pode retornar a mensagem com o campo *STATUS* igual a:

0 : Quando o *A.C.D.* pode ser desativado com sucesso;

7 : Caso o *A.S.V.* ainda esteja ativo. Dado que o *A.C.D.* não pode ser desativado antes do *A.S.V.*;

B : Caso o valor do campo CHAVE1 não confere com o valor da chave de comunicação armazenada no repositório de dados;

Quando o Servidor de Ativação recebe uma conexão solicitando a ativação de um *A.C.D.*, ele pode responder esta mensagem atribuindo ao campo *STATUS* um dos seguintes valores:

A : se a chave de ativação não confere;

4 : quando o *A.C.D.* já estava ativo;

1 : quando o *A.C.D.* pode ser ativado. Neste caso o campo CHAVE2 irá receber o valor da chave de ativação.

Ao receber uma mensagem com o valor do campo *STATUS* igual a "2", o servidor pode responder esta mensagem de entrada contendo uma outra, com o *STATUS* igual:

B : se a chave de comunicação não confere;

5 : quando o *A.S.V.* já estava ativo;

2 : quando o *A.S.V.* pode ser acionado.

Quando o servidor recebe uma mensagem com o campo *STATUS* igual a "3" e com endereço de um *IP* válido, o servidor verifica se o valor do campo CHAVE1 é igual ao valor da chave de comunicação e responde a mensagem com o campo *STATUS* alterado para:

B : se a chave de comunicação não confere;

8 : quando o *A.C.D.* não está ativo;

5 : quando o *A.S.V.* já estava ativo;

3 : confirmando a possibilidade de ativação do *A.S.V.*

4.2.2 Cliente/Servidor de Coleta

O Servidor de Coleta é um dos componentes do *A.C.D.* Este servidor recebe os *scripts* de verificação, enviados pelo Agente de Verificação, executa-os e retorna para o Cliente de Coleta o arquivo de saída.

O Servidor de Coleta deve ficar escutando a porta 7601/*TCP*, não privilegiada, que conforme mencionado no item 4.2 pode ser utilizada pelo servidor. Já no Cliente de Coleta, a atribuição da porta de comunicação será feita de forma automática, não havendo necessidade de reserva ou especificação prévia.

A cada nova conexão, o servidor recebe uma *Mensagem de Solicitação* com a estrutura apresentada na Figura 4.11, seus campos são descritos a seguir:

```
struct conect.out   {
                    char   ip       [15]
                    char   chave    [10]
                    char   status   [1]
                    char   hash     [32]
                    char   *dados
                    }
```

Figura 4.11: Cliente/Servidor de Coleta : Mensagem de Solicitação

IP **:** formado por uma seqüência de 15 caracteres alfanuméricos, que representam o endereço *IP* da máquina com a qual se deseja estabelecer a conexão (*A.C.D.*);

CHAVE : este campo representa o valor da chave de comunicação do nó que hospeda o servidor e formado por uma seqüência de 10 caracteres numéricos.

STATUS **:** este define a ação a ser executada sobre as informações que estão sendo transmitidas no campo dados. Ele é formado por um caracter alfanumérico e os possíveis valores que este campo pode assumir são:

0 : executar arquivo;

1 : copiar arquivo;

2 : envio do perfil da máquina;

3 : re-envio de dados.

***HASH* :** campo formado por uma seqüência de 32 caracteres alfanuméricos, que representam o valor do *hash MD5* das informações transmitidas no campo Dados.

DADOS : campo formado por uma seqüência de caracteres alfanuméricos de tamanho variados. Este campo contém os dados que serão tratados pelo agente de coleta.

Em resposta à Mensagem de Solicitação, o Servidor de Coleta envia a Mensagem de Dados. Esta mensagem contém normalmente a saída dos *scripts* de verificação, possui a estrutura da Figura 4.12 e seus campos são descritos a seguir.

```
struct conect.out   {
                    char   ip       [15]
                    char   chave    [10]
                    char   status   [1]
                    char   hash     [32]
                    char   *dados
                    }
```

Figura 4.12: Cliente/Servidor de Coleta : Mensagem de Dados

***IP* :** campo formado por uma seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.C.D.*(onde está o servidor);

CHAVE : campo formado por uma seqüência de 10 caracteres numéricos que representam o valor da chave de comunicação do agente de coleta;

STATUS **:** campo formado por um único caracter alfanumérico cujo valor define o resultado da ação executada sobre as informações enviadas na Mensagem de Conexão. Os possíveis valores para este campo são:

0 : o comando anterior foi executado com sucesso;

1 : o arquivo solicitado não estava disponível;

2 : o perfil não pode ser enviado ou já havia sido;

3 : falha na execução do comando anterior;

4 : o *hash MD5* dos dados enviados não confere;

HASH **:** campo formado por uma seqüência de 32 caracteres alfanuméricos, que representam o valor do *hash MD5* das informações transmitidas no campo dados.

DADOS : campo de tamanho e conteúdo variado, que contém as informações geradas pela saída de um *script* de verificação ou do perfil do nó, em alguns casos, este campo pode estar vazio.

Para finalizar a conexão, o Cliente envia ao Servidor a Mensagem de Confirmação, cujo formato é o mesmo da Figura 4.13 e seus campos são descritos a seguir.

```
struct conect.out   {
                    char   ip       [15]
                    char   chave    [10]
                    char   status   [1]
                    }
```

Figura 4.13: Cliente/Servidor de Coleta - Mensagem de Confirmação

IP **:** campo formado por uma seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* da máquina com a qual se deseja estabelecer a conexão (*A.C.D.*);

CHAVE : campo formado por uma seqüência de 10 caracteres numéricos que representam o valor da chave de comunicação do servidor.

***STATUS* :** campo formado por um caracter alfanumérico cujo valor informa se as informações transmitidas pela mensagem anterior foram recebidas ou não com sucesso. Os possíveis valores para este campo são:

0 : Os dados foram recebidos com sucesso;

1 : O *hash MD5* não confere;

4.2.3 Cliente/Servidor de Módulo

Este é conjunto de componentes opcionais, mas seu uso se justifica pela existência de nós que não possuem as ferramentas necessárias para o funcionamento dos Agentes de Coleta e Agentes Secundários de Verificação.

O Servidor de Módulo é parte integrante dos Agentes de Verificação. Sua função é enviar aos Agentes de Coleta os *softwares* necessários para garantir o funcionamento dos seus componentes e ainda permitir a execução dos *script* de coleta. O Cliente de módulo, por sua vez, faz parte dos Agentes de Coleta de Dados e dos Agentes Secundários de Verificação.

O Servidor de Módulo deve ficar escutando a porta $7602/TCP$, que é uma porta não privilegiada, conforme mencionado no item 4.2. No lado do cliente a atribuição da porta de comunicação será feita de forma automática.

Ao se conectar com o Servidor de Módulo, o Cliente envia a Mensagem Solicitação cuja estrutura pode ser observada na Figura 4.14. Seus campos são descritos logo a seguir:

***IP* :** campo formado por uma seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.C.D.*;

CHAVE : campo formado por uma seqüencia de 10 caracteres numéricos que representam o valor da chave de comunicação do nó que hospeda o *A.C.D.*

```
struct conect.in   {
                   char   ip          [15]
                   char   chave       [10]
                   char   idmodulo    [25]
                   }
```

Figura 4.14: Cliente/Servidor de Modulo : Mensagem de Solicitação

IDMODULO : campo formado por uma seqüência de 25 caracteres que representam a identificação do módulo que está sendo solicitado. Normalmente o valor deste campo é o nome do *software* desejado.

O Servidor de Módulo, ao receber a Mensagem de Solicitação, envia ao cliente a Mensagem de Dados que contém o módulo solicitado e que possui o formato apresentado pela Figura 4.15, e seus campos são descritos abaixo:

```
struct conect.res  {
                   char   ip          [15]
                   char   chave       [10]
                   char   idmodulo    [25]
                   char   status      [1]
                   char   hash        [32]
                   bin    *dados
                   }
```

Figura 4.15: Cliente/Servidor de Módulo : Mensagem de Resposta a Solicitação

***IP* :** formado por uma seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.C.D.* (cliente);

CHAVE : campo formado por uma seqüência de 10 caracteres numéricos que representam valor da chave de comunicação do *A.C.D.*;

IDMODULO : formado por uma seqüência de 25 caracteres que representam a identificação do módulo que está sendo enviado.

***STATUS* :** formado por um único caracter, que permite informar se o módulo foi localizado no repositório de dados e se está sendo enviado. Ele pode assumir os seguintes valores:

0 : O módulo está sendo enviado;

1 : Não foi possível localizar o módulo;

***HASH* :** formado por uma seqüência de 32 caracteres que representam o *hash MD5* das informações transmitidas no campo DADOS;

DADOS : campo de formato binário e de tamanho variado que possui o módulo solicitado pelo cliente ao servidor.

Ao receber a mensagem anterior, o cliente responde com a Mensagem de Confirmação, informando se os dados foram recebidos ou não com sucesso. Esta mensagem possui a estrutura da Figura 4.16 e seus campos descritos a seguir:

```
struct conect.con   {
                    char   ip          [15]
                    char   chave       [10]
                    char   idmodulo    [25]
                    char   status      [1]
                    }
```

Figura 4.16: Cliente/Servidor de Modulo : Mensagem de Confirmação

***IP* :** campo de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.C.D.*;

CHAVE : campo de 10 caracteres numéricos que representam o valor da chave de comunicação do *A.C.D.*

IDMODULO : campo de 25 caracteres que representam a identificação do módulo que foi enviado pelo servidor.

***STATUS* :** formado por um único caracter. Este informa se o módulo foi recebido ou não com sucesso e pode assumir os seguintes valores:

0 : O módulo foi recebido com sucesso;

1 : O *hash MD5* do módulo não confere.

4.2.4 Cliente/Servidor de Comunicação

Estes dois componentes são necessários para o funcionamento do agente secundário de verificação. Esse agente não possui acesso ao Repositório de Dados, e não realiza a análise dos arquivos de saída recebidos dos agentes de coleta; assim necessita de um canal de comunicação com o Agente Primário de Verificação. Este canal de troca de dados será criado utilizando o Servidor e o Cliente de Comunicação.

O Servidor de Comunicação deve ficar escutando a porta 7603/*TCP*, que é uma porta não privilegiada, conforme mencionado no item 4.2. No lado do Cliente de Comunicação, a atribuição da porta será feita de forma automática durante a conexão com o servidor.

Sem a arquitetura provida pelo Cliente/Servidor de Comunicação, não seria possível ao Agente de Verificação Primário controlar a análise de um ambiente multinível como mostrado na Figura 4.17.

O módulo servidor faz parte dos agentes primário e secundário de verificação e permite que estes agentes se comuniquem com os níveis inferiores da estrutura de comunicação. Os clientes por sua vez fazem parte apenas dos *A.S.V.* Estes módulos são utilizados para enviar ou solicitar dados ao *A.P.V*, como mostra a Figura 4.17.

A comunicação entre estes clientes e servidores é semelhante às demais existentes no modelo. Ao se conectar com o Servidor de Comunicação, o Cliente envia uma mensagem com a sintaxe da Figura 4.18. Esta mensagem tem seus campos descritos logo a baixo:

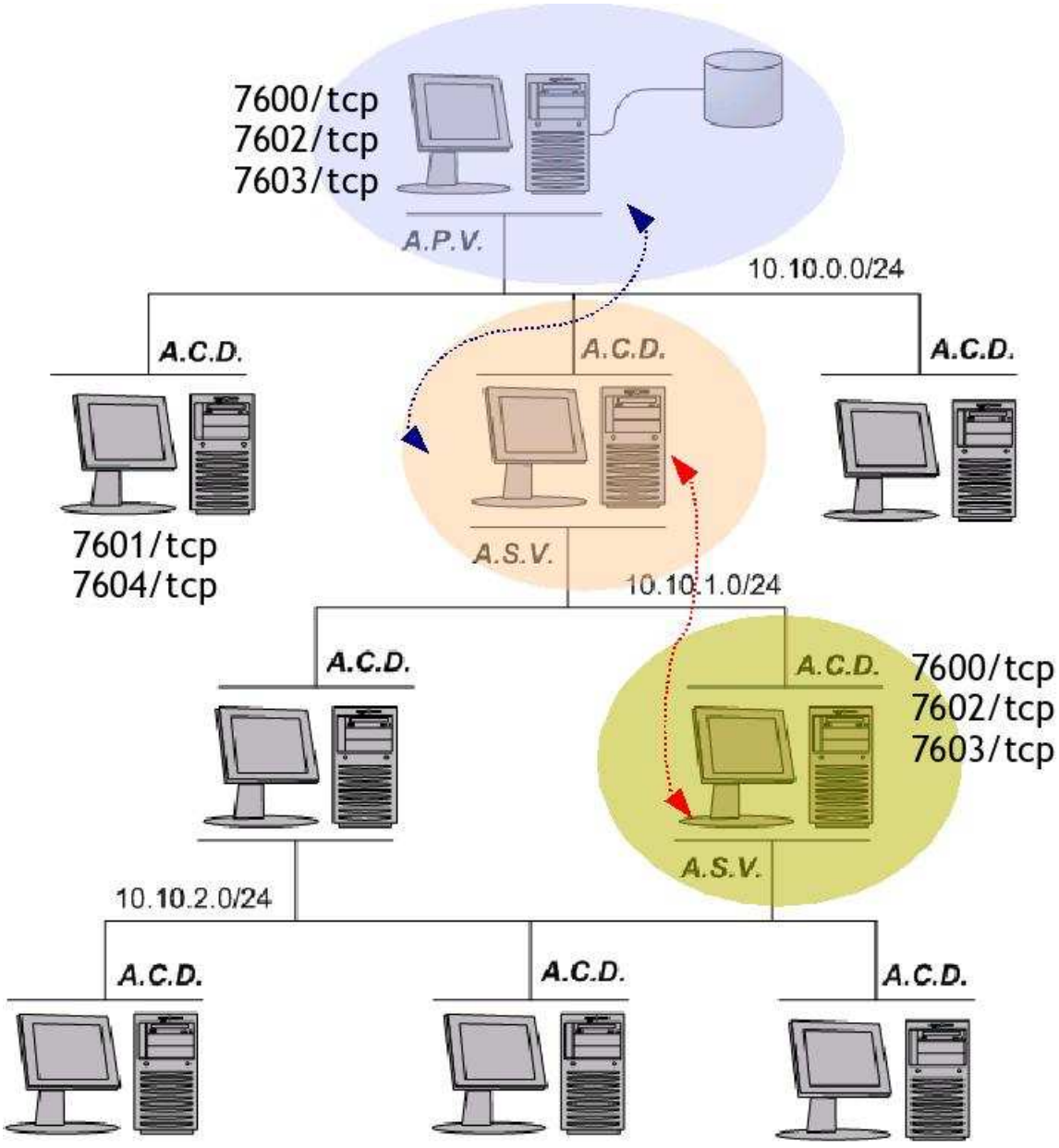

Figura 4.17: Processo de comunicação entre os *A.S.V.* e *A.P.V.*

IP **:** formado por uma seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.S.V.*;

CHAVE : formado por uma seqüência de 10 caracteres numéricos que representam o valor da chave de comunicação do nó que hospeda o *A.S.V.*

HASH **:** formado por uma seqüência de 32 caracteres alfanuméricos que representam o *hash* das informações enviadas no campo DADOS.

STATUS **:** formado por um único caracter e utilizado para indicar o tipo de informação que está sendo transmitida. Ele pode assumir um dos seguintes valores:

0 : arquivo de varredura da rede;

1 : perfil de um *A.C.D.*;

2 : resultado da coleta;

3 : outro arquivo;

4 : solicitação de modulo;

DADOS : campo de tamanho variado que contém as informações que estão sendo transmitidas ou solicitadas ao *A.P.V.*

Ao receber uma Mensagem de Conexão, o servidor envia ao cliente a Mensagem de Envio, que pode conter os dados solicitados na mensagem anterior ou a confirmação dos dados recebidos. O conteúdo desta mensagem é dependente da mensagem

```
struct conect.in   {
                   char   ip       [15]
                   char   chave    [10]
                   char   status   [1]
                   char   hash     [32]
                   char   *dados
                   }
```

Figura 4.18: Cliente/Servidor de Comunicação : Mensagem de Solicitação

anterior mas seu formato é fixo e apresentado na Figura 4.19. Seus campos são descritos logo a seguir:

```
struct conect.env   {
                    char   ip       [15]
                    char   chave    [10]
                    char   status   [1]
                    char   hash     [32]
                    char   *dados
                    }
```

Figura 4.19: Cliente/Servidor de Comunicação : Mensagem de Envio

IP **:** campo de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.S.V.*;

CHAVE : campo de 10 caracteres numéricos que representam o valor da chave de comunicação do nó que hospeda o *A.S.V.*

STATUS **:** campo formado por um único caracter que determina o tipo de informação que está sendo transmitida no campo DADOS ou se os dados transmitidos na mensagem anterior chegaram com sucesso. Este campo pode assumir um dos seguintes valores:

0 : não há mais solicitaõeso a serem enviada ao *A.C.D.*;

1 : pacotes de instalação (*A.C.D. / A.S.V.*);

2 : lista dos controles que devem ser verificados no *A.C.D.*;

3 : o *hash* dos dados enviados anteriormente não confere.

HASH **:** campo de 32 caracteres alfanuméricos que representa o *hash* dos dados enviados na mensagem;

DADOS - campo de tamanho variado que contém as informações que estão sendo transmitidas do Servidor para o Cliente. Nem sempre este campo será utilizado.

Quando o cliente recebe a Mensagem de Envio, ele responde com a Mensagem de Confirmação. Esta mensagem indica se os dados chegaram ou não com sucesso. A estrutura desta mensagem é apresentada pela Figura 4.20 e seus campos descritos logo abaixo:

```
struct conect.con   {
                    char   ip       [15]
                    char   chave    [10]
                    char   status   [1]
                    }
```

Figura 4.20: Cliente/Servidor de Comunicação : Mensagem de Confirmação

IP **:** seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.S.V.*;

CHAVE : seqüência de 10 caracteres numéricos que representam o valor da chave de comunicação do nó que hospeda o *A.S.V.*;

STATUS **:** único caracter que informa se a Mensagem de Envio foi recebida com sucesso. Os possíveis valores deste campo são:

0 : A mensagem foi recebida com sucesso;

1 : O *hash* dos dados não confere;

4.2.5 Cliente/Servidor de Controle

Quando um Agente de Coleta de Dados executa um *script* enviado pelo agente de verificação, rotinas mal implementadas podem travar o *script*. Neste caso, cabe ao agente de verificação interromper o funcionamento do mesmo.

O cliente de controle pode iniciar, parar e/ou reiniciar serviços e aplicações nos agentes da sub-rede de verificação do Agente Verificação no qual está instalado. O servidor, por sua vez, está instalado nos agentes de coleta aguardando por uma solicitação do cliente de controle

O servidor de controle é parte integrante dos Agentes de Coleta de Dados e quando acionados pelos clientes, instalados nos agentes de verificação, permitem a gerência e execução de processos de forma remota.

O Servidor de Controle deve ficar escutando a porta 7604/*TCP*, conforme mencionado no item 4.2. Já no Cliente, a atribuição da porta de comunicação será feita de forma automática durante seu processo de conexão com o servidor.

O cliente troca com o servidor dois tipos de mensagem. A primeira é a Mensagem de Solicitação, que é recebida pelo servidor quando ocorre a necessidade de gerenciar de forma remota os processo do sistema operacional. Esta mensagem tem a estrutura da Figura 4.21 e seus campos descritos a seguir:

```
struct conect.in   {
                   char   ip          [15]
                   char   chave       [10]
                   char   processo    [100]
                   char   sinal       [3]
                   }
```

Figura 4.21: Cliente/Servidor de Controle : Mensagem de Solicitação

IP **:** seqüência de 15 caracteres alfanuméricos que representam o endereço *IP* do *A.C.D.* que está sendo conectado;

CHAVE : campo formado por um a seqüência de 10 caracteres numéricos que representam a chave de comunicação do *A.C.D.*;

PROCESSO : formado por no máximo 100 caracteres alfanuméricos que especificam o nome do processo a ser tratado;

SINAL : campo formado por 2 caracteres alfanuméricos, cujo valor indica a ação a ser realizada sobre o processo e pode assumir os seguinte valores:

1 : recarregar (*HANGUP*) o processo;

9 : matar (*KILL*) o processo;

2 : interromper o funcionamento do processo;

14 : despertador, acorda um processo que havia sido interrompido;

16 : executar um comando de forma remota.

Ao receber esta mensagem, o servidor envia ao cliente uma segunda mensagem, denominada Mensagem de Confirmação que informa se a solicitação recebida anteriormente pode ser executada. Esta mensagem possui a estrutura da Figura 4.22. Seus campos são descritos logo abaixo:

```
struct conect.out   {
                    char   ip       [15]
                    char   chave    [10]
                    char   status   [3]
                    }
```

Figura 4.22: Cliente/Servidor de controle : Mensagem de Confirmação

***IP* :** campo formado por uma seqüência de 15 caracteres alfanuméricos, que representam o endereço *IP* do *A.C.D.* contactado;

CHAVE : campo formado por um a seqüência de 10 caracteres numéricos, que representam a chave de comunicação do *A.C.D.*

SINAL : campo formado por 2 caracteres alfanuméricos, que informa o resultado da ação realizada sobre o processo indicado pela mensagem anterior. Este campo pode assumir os seguinte valores [3]:

[3] A definição dos possíveis valores para o campo SINAL, que é utilizado nas Mensagem de Solicitação e Confirmação do Servidor/Cliente de Controle, foi baseada na lista de sinais utilizados no Sistema Operacional Linux, e obtidos a partir do livro: Linux, O Guia Essencial [49].

0 : Ação executada com sucesso;

4 : Instrução ilegal;

12 : Argumento ruim;

15 : Término normal;

4.2.6 Módulo Verificador Primário

Este módulo é parte integrante do Agente Primário de Verificação e responsável pelo gerenciamento dos demais componentes deste agente (clientes, servidores e repositório de dados). Quando o *A.P.V.* é acionado, o módulo verificador é o primeiro a ser ativado.

Ao ser iniciado, ele verifica se todos os aplicativos necessários para a execução dos demais componentes do *A.P.V.* estão disponíveis. Cada *software* que não está disponível é extraído do repositório de dados e instalado pelo módulo verificador.

Finda a instalação dos aplicativos necessários, ele realiza o processo de varredura do segmento de rede no qual o nó está conectado. Para a realização deste processo, pode ser utilizado uma das técnicas descritas no Capítulo 3.

Ao término do processo de varredura, este módulo armazena os dados coletados no repositório de dados, seleciona os nós que serão analisados, gera os pacotes de instalação dos agentes de coleta de dados e para cada nó será gerado uma chave de ativação, a ser utilizada quando o *A.C.D.* for acionado.

Depois de ter realizado o levantamento das informações sobre os nós ativos e de ter gerado os pacotes de instalação dos agentes de coleta, o módulo verificador realiza a ativação dos servidores que compõem o Agente Primário de Verificação. Nesta fase o *A.P.V.* já está totalmente funcional e aguardando pela solicitação de ativação dos *A.C.D.*s.

Durante esta fase, é função do Módulo Verificador gerenciar os demais componentes, o processo de coleta de dados e de análise dos dados coletados. A análise de cada nó será a base da verificação da aderência do ambiente à *NBR ISO/IEC 17799*.

Finda a coleta e análise das informações sobre o ambiente, pode ser solicitado ao

módulo a geração de um relatório contendo as informações sobre quais os controles foram homologados nas máquinas, bem como o nível de aderência de todo o ambiente à Norma Nacional de Segurança da Informação.

4.2.7 Módulo Verificador Secundário

Assim como o módulo verificador primário, este é o primeiro componente de software a ser carregado quando o *A.S.V.* é iniciado, e é o responsável pelo gerenciamento e controle dos demais componentes deste agente.

Ao ser iniciado, o módulo verifica a disponibilidade dos *softwares* necessários para a execução do *A.S.V.* Caso esteja faltando algum *software*, ele o solicita ao Agente Primário de Verificação, utilizando-se o cliente de módulo.

Ao término da instalação de todos os *softwares* que estavam faltando, o módulo verificador ativa os servidores que compõem o Agente Secundário de Verificação e utiliza o cliente de ativação para confirmar a ativação do *A.S.V.*, no nó em que está instalado.

Este módulo é responsável pelo redirecionamento de qualquer solicitação, vinda da sub-rede que está monitorando, para o agente de verificação responsável pelo monitoramento da camada superior da rede hierárquica de verificação. Este processo é repetido até que a solicitação chegue ao Agente Primário de Verificação, que ao receber, processar e atender a solicitação envia uma mensagem que deve percorrer o caminho inverso.

Quando este módulo verifica que um dos servidores parou de funcionar, é sua função reativá-lo, através do uso do cliente/sevidor de controle, caso contrário o processo de análise do ambiente poderá ser comprometido.

4.2.8 Módulo de Coleta de Dados

O Módulo de Coleta de Dados faz parte do *A.C.D.* Este componente é o primeiro a ser acionado quando o *A.C.D.* entra em funcionamento, ele é o responsável pelo bom funcionamento dos demais componentes.

Ao ser acionado, este módulo verifica a disponibilidade dos *softwares* necessários para o funcionamento dos demais componentes do *A.C.D.* Havendo a falta de um componente, este será solicitado ao *A.P.V.* e posteriormente instalado.

Feita a instalação dos componentes que faltam, o módulo de coleta aciona os servidores que fazem parte do *A.C.D.* e utiliza o cliente de ativação para realizar o acionamento do agente de coleta junto ao agente de verificação.

Após a ativação do *A.C.D.*, o módulo de coleta fica aguardando as solicitações que chegam pelo servidores e as atende.

Quando um dos servidores falha e deixa de funcionar, é função deste módulo realizar o seu re-carregamento e garantir que o mesmo esteja funcionando corretamente.

Quando não há mais scripts a serem enviados ao Agente de Coleta de Dados, ocorre o processo de encerramento do *A.C.D.* Durante este processo, é função do módulo de coleta realizar todas as ações necessárias para encerrar o funcionamento dos demais componentes e remover qualquer alteração realizada pelo *A.C.D.* na máquina em que foi instalado.

4.3 Quadro comparativo

A Figura 4.23, apresenta de forma resumida uma comparação do protótipo com outras soluções existentes.

O *COBRA* é uma ferramenta que realiza a análise de um ambiente totalmente baseado em questionários eletrônicos. O *Checkup Tool*, faz uso de questionários e também de coletores automáticos. O modelo por sua vez faz uso apenas de agentes de coleta.

Como pode ser observado tanto o *COBRA* quanto o *Checkup Tool*, necessitam de uma máquina com pelo menos o *Windows* instalado, em ambos os casos nesta máquina será instalado o módulo que irá realizar a verificação. Mas o *A.P.V.* do modelo pode ser instalado em qualquer sistema operacional.

Além do modelo a única solução que trabalha com coletores automáticos é a

Checkup Tool, mas esta somente realizar a coleta de dados em máquinas baseadas em *Windows 2000* ou superior, e tais módulos estão destinados somente aos servidores. O modelo além de realizar a coleta de forma automática ele faz uso da própria rede do ambiente para realizar este coleta, sendo que ele não restringe a coleta somente aos servidores, as estações clientes também serão analisadas. E o sistema operacional das máquina a serem analisadas não necessita ser somente versões do *windows*, podem ser implementados agentes de coleta para todas as plataformas de *hardware* e *software*.

O modelo realiza uma análise prévia do ambiente para que seja possível obter o perfil das máquinas a serem analisadas e em seguida analisa cada nó detectado.

	COBRA	*Checkup Tool*	*Modelo Proposto*
Plataforma de Hardware	Intel(i386)	Intel(i386)	Independente
Plataforma de Software	Microsoft	Microsoft	Independente
Sistema Operacional	Win 98 e sup	Win 2000 XP Serv	Independente
Normas	BS7799/ ISO	ISO/IEC 17799, Cobit, Brasileia	ISO/IEC 17799
Forma de Coleta			
Questionários	Sim	Sim	Não
Coletores Automáticos	Não	Sim	Sim
Coleta pela Rede	Não	Não	Sim
Baseadano Perfil	Não	Não	Sim
Análise Prévia do Ambiente	Não	Não	Sim

Figura 4.23: Tamanduá : Quadro comparativo

Este capítulo descreveu todos os componentes do modelo proposto, suas funcionalidades e seu processo de interação, baseado nestas informações foi implementado um protótipo que permitiu verificar a viabilidade do uso do modelo. Este protótipo será tratado com mais detalhes no próximo capítulo.

Capítulo 5

Implementação do protótipo: Tamanduá-Mirim

Este capítulo tem por finalidade apresentar o processo de desenvolvimento, instalação e utilização do protótipo denominado Tamanduá-Mirim, que é a implementação parcial da proposta apresentada no Capítulo 4.

Também será descrito o processo de configuração dos componentes do sistema operacional necessários para o perfeito funcionamento do *A.P.V.*

O código fonte do protótipo pode ser acessado a partir dos endereços: *http://www.projetotamandua.rg3.net* ou *http://www.projetotamandua.cjb.net*.

5.1 Objetivo

O objetivo do Tamanduá-Mirim é permitir a verificação da aderência de um ambiente aos controles 7.2 - Gerenciamento de Acesso do Usuário, 7.4 - Controle de Acesso a Rede e 7.5 - Controle de Acesso ao Sistema Operacional da Norma Nacional de Segurança da Informação. Para tal ele implementa a estrutura básica do modelo de verificação proposto no Capítulo 4 e faz uso apenas de *softwares* de código aberto ou gratuitos.

O Ambiente utilizado para o desenvolvimento do Tamanduá-Mirim foi o sistema

operacional *Slackware Linux* e as linguagens de programação C, *PHP* e *Shell Script*; o servidor *http* utilizado foi o *Apache* (família 1.3) e o servidor de banco de dados foi o *MySQL*.

Baseado nas informações do Capítulo 4 foram implementados o *A.P.V*, o *A.C.D*, mas não foi implementado o *A.S.V*, pois este protótipo visa apenas validar a viabilidade da verificação automática dos controles da Norma Nacional de Segurança em ambiente com um único segmento de rede. Uma vez validado este processo de análise, ele pode ser expandido para as sub-redes de comunicação.

O Agente Primário de Verificação do protótipo é composto pelo Módulo Verificador Primário, pelo Repositório de Dados, pelo Servidor de Ativação, pelos Cliente de Coleta e de Controle. O Agente de Coleta de Dados é formado pelo Cliente de Ativação, pelo Módulo de Coleta de Dados, pelos Servidores de Controle e de Coleta.

Apesar de não ser uma implementação completa do modelo, o Tamanduá-Mirim permitiu: validar o funcionamento do modelo, identificar várias dificuldades na análise automática do ambiente e identificar algumas melhorias que podem ser feitas e que são descritas no Capítulo 6.

5.2 Plataforma adotada

Como um dos objetivos do modelo é prover um ambiente de verificação multiplataforma, que faça uso somente de aplicativos *Open Source*, nenhuma solução proprietária foi utilizada e todos os *software* empregados possuem um elevado grau de portabilidade entre as plataformas de *hardware* e *software* existente atualmente.

É listado abaixo os aplicativos e utilitários usados no desenvolvimento do Tamanduá-Mirim:

Sistema Operacional : *Slackware Linux*

O *Slackware Linux*, mantido pelo *The Slackware Linux Project*, foi uma das primeiras distribuições *Linux* desenvolvidas, tendo seu primeiro *release* liberado em Abril de 1993, ele é tido como a distribuição mais *Unix-Like*, atualmente, disponível.. O *Slackware Linux*, prima por fornecer os pacotes de

software mais estáveis, graças a estes fatores, ele tem se tornado uma das distribuições mais estáveis e populares para servidores de rede e estações de trabalho, justificando assim sua adoção como sistema operacional base do Tamanduá-Mirim.

Gerenciador de Banco de Dados : *MySQL*

O *MySQL* que é um dos gerenciadores de banco de dados mais utilizado pelos usuários de *Linux*, apresenta uma *API* de fácil interação, facilitando assim a manipulação dos dados a partir de linguagens como o C, C++, *Java* e o *PHP*. Este *software* ainda possui suporte a transação e uma robusta estrutura de segurança, além de ter sido portado para várias arquiteturas de *hardware* e sistemas operacionais. Este é o aplicativo utilizado na construção do repositório de dados do modelo proposto

Servidor Web : *Apache Web Server*

O *Apache Web Server*, *software Open-Source*, multiplataforma que possui suporte à várias linguagens *CGI*s, dentre as quais: o *Perl* e o *PHP*, é um dos servidores de *http* mais populares e estáveis já desenvolvidos. Este servidor tem por finalidade suportar a execução dos *script*s *PHP* que formam o *front-end* do protótipo com o usuário.

Segundo o *Netcraft Web Server Survey* [37] em abril de 2004 mais de 68% dos servidores *web* da Internet possuem este aplicativo instalado, como pode ser observado na Figura 5.1.

Linguagem de programação adotada nas páginas : *PHP*

A linguagem *PHP*, que é uma linguagem de *script server-side*, que desde seu surgimento tem chamado atenção devido sua flexibilidade e portabilidade. Atualmente esta linguagem é suportada pela grande maioria dos servidores *web* incluindo: *Apache*, M*icrosoft Internet Information Server*, *Personal Web Server* e etc. Como mencionado anteriormente, esta foi a linguagem escolhida para a construção do *front-end* do protótipo.

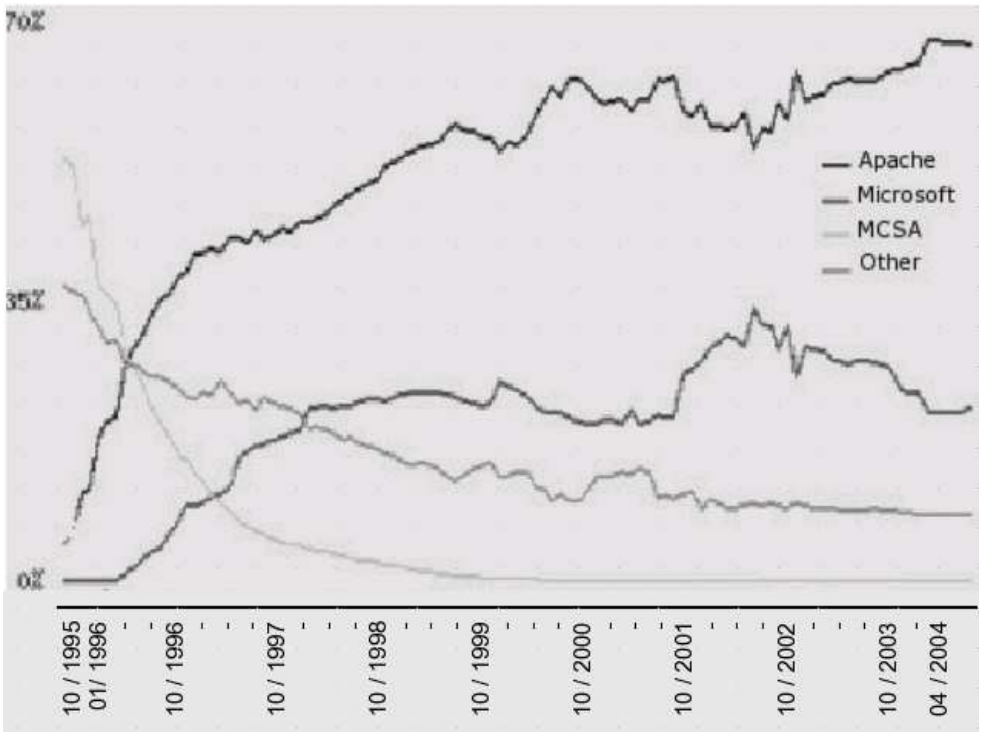

Figura 5.1: *Netcraft Web Server Survey* - Abril de 2004

Segundo o *Apache Module Report* [3], 52.75% dos servidores *web* baseados no *Apache* possuem o *PHP* instalado.

Esta linguagem possui uma sintaxe muito parecida com a da linguagem C, permite o desenvolvimento de aplicativos complexos com poucas linhas de código e possui suporte nativos a vários gerenciadores de banco de dados, do quais podemos destacar o *MySQL*, o *PostgreSQL*, o *IBM DB2*, o *InterBase*, o *Oracle* e etc. Sendo extremamente útil no processamento de texto e no desenvolvimento de *scripts* de linha de comando.

Linguagem utilizada nos *scripts* de coleta de dados : *Shell Script*

O *Shell Script*, que é uma linguagem de programação fornecida pelo *shell* do sistema operacional *Linux*, permite o desenvolvimento de ferramentas personalizadas, as quais podem ser utilizadas na administração e gerência das máquinas. Esta linguagem possui suporte a várias estruturas de repetição, controle de fluxo e a vetores.

Graças ao *shell Bash*, foi possível o desenvolvimento dos *scripts* de coleta de

dados utilizando-se os comandos básicos de uma distribuição *Linux*, evitando a necessidade do desenvolvimento de novos aplicativos.

5.3 Requisitos para a instalação e uso

A seguir são descritos os requisitos de *software* para a instalação e operação do protótipo, bem como o processo de instalação e configuração de algum deles.

5.3.1 Softwares requeridos

Sistema Operacional :

- ***Slackware Linux* 10.0** ou superior
- *Site*: http://www.slackware.org/

Gerenciador de Banco de Dados :

- ***MySQL*** - versão 4.0 ou superior
- *Site*: http://www.mysql.com/

Servidor *Web* :

- ***Apache HTTP Server*** - versão 1.3.29 ou superior;
- *Site*: http://www.apache.org/

- ***mod_ssl*** - versão 2.8.16 for *Apache* 1.3.29 ou superior (compatível como o *Apache*);
- *Site*: http://www.modssl.org/

- **Linguagem *PHP*** - versão 4.3.6 ou superior
- *Site*: http://www.php.net/

Scanner* de Portas *TCP/UDP :

- ***Nmap - Network Mapper*** - versão 3.50 ou superior;

- *Site*: http://www.insecure.org/nmap/

Varredura da Rede :

- ***Fping*** - versão 2.4b2_to-ipv6 ou superior;
- *Site* : http://www.fping.com/

Gerencia de execução de comandos :

- ***Sudo ("superuser do")*** - versão 1.6.7p5 ou superior;
- *Site*: http://www.sudo.ws/sudo/sudo.html

Shell :

- ***Bash*** - *Bourne Again Shell*;
- *Site*: http://www.gnu.org/software/bash/bash.html

5.4 Instalação das dependências

5.4.1 *MySQL*

Script **de instalação**

```
#!/bin/bash
    # Verifica se o usuário tem permissão para instalar o pacote
    if [[ $USER != "0" ]]
    then
        echo " Este script deve ser executado através do usuário root"
        exit
    fi
    # == --------------
    # Baixando pacote
    # == --------------
    wget http://www3.linuxpackages.net/packages/Slackware-9.1/ \
    Daemon/mysql/mysql-4.0.18-i486-1jim.tgz
    # == --------------
```

```
# Instalando pacote
# == --------------
installpkg mysql-4.0.18-i486-1jim.tgz
# == --------------
# Instalando banco de dados
# == --------------
mysql_install_db
mysql_setpermission
# == --------------
# Alterando permissões das tabelas
# == --------------
chown -R mysql.mysql /var/lib/mysql/
# == --------------
# Iniciando o mysql
# == --------------
mysqld_safe &
echo 'mysqld_safe &' >> /etc/rc.d/rc.local
```

Configurações pós instalação

Após a instalação do servidor de banco de dados *MySQL*, o mesmo deve sofrer alguns ajustes em suas configurações. Estas modificações visam permitir a execução do tamanduá em um contexto separado e seguro.

Dentre as modificações a serem executadas as mais importantes são a atribuição de uma senha ao usuário *root* dentro do gerenciador de banco de dados, a remoção de contas de acesso que não possuem senha, a criação do banco de dados / tabelas utilizadas pelo Tamanduá-Mirim e a criação de um usuário com permissão de acesso ao banco de dados do protótipo.

Atribuir senha ao usuário *root* :

Para atribuir uma senha ao usuário *root*, dentro de uma *shell* execute o comando a seguir:

mysqladmin -u root password 'tamandua'

***Script* de criação do Banco de Dados** :

O banco de dados e as tabelas necessárias para o funcionamento do protótipo, podem ser construídas com o uso de um *script* como este que aparece a seguir.

```
CREATE DATABASE ‘tamandua‘ ;
use ‘tamandua‘;
# --------------------------------------------------------
# Estrutura da tabela ‘agentes‘
# --------------------------------------------------------
CREATE TABLE ‘agentes‘ (
  ‘cd_agente‘ int(11) unsigned NOT NULL auto_increment,
  ‘cd_so‘ int(11) unsigned default NULL,
  ‘ds_agente‘ varchar(100) default ’’,
  ‘ds_dados‘ longblob,
  PRIMARY KEY  (‘cd_agente‘)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
# --------------------------------------------------------
# Estrutura da tabela ‘analise‘
# --------------------------------------------------------
CREATE TABLE ‘analise‘ (
  ‘cd_analise‘ int(11) unsigned NOT NULL auto_increment,
  ‘vl_ip‘ varchar(15) default ’0’,
  ‘cd_script‘ int(11) unsigned default ’0’,
  ‘ds_resultado‘ varchar(20) default ’’,
  ‘ds_saida‘ longtext,
  PRIMARY KEY  (‘cd_analise‘)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
# --------------------------------------------------------
# Estrutura da tabela ‘controles‘
# --------------------------------------------------------
CREATE TABLE ‘controles‘ (
  ‘cd_controle‘ int(11) unsigned NOT NULL auto_increment,
  ‘ds_controle‘ varchar(100) default ’’,
  ‘nm_controle‘ varchar(25) default NULL,
  PRIMARY KEY  (‘cd_controle‘)
```

```
) TYPE=MyISAM AUTO_INCREMENT=1 ;
# --------------------------------------------------------
# Estrutura da tabela `hosts`
# --------------------------------------------------------
CREATE TABLE `hosts` (
  `vl_ip` varchar(15) default '000.000.000.000',
  `ds_estado` char(1) default '0',
  `vl_chave_at` varchar(10) default '0000000000',
  `vl_chave_com` varchar(10) default '0000000000',
  `ds_coleta` int(1) default '0',
  `ds_analise` int(1) default '0',
  PRIMARY KEY  (`vl_ip`)
) TYPE=MyISAM  ;
# --------------------------------------------------------
# Estrutura da tabela `interfaces`
# --------------------------------------------------------
CREATE TABLE `interfaces` (
  `vl_ip` varchar(15) default '000.000.000.000',
  `ds_nome` varchar(15) default '',
  `vl_ip_placa` varchar(15) default '000.000.000.000',
  `vl_mascara` varchar(15) default '000.000.000.000'
) TYPE=MyISAM;
#--------------------------------------------------------
# Estrutura da tabela `modulo`
# --------------------------------------------------------
CREATE TABLE `modulo` (
  `cd_modulo` int(11) unsigned NOT NULL auto_increment,
  `cd_so` int(11) unsigned default '0',
  `ds_modulo` varchar(100) default '',
  `id_modulo` varchar(25) default '',
  `ds_dados` mediumblob,
  PRIMARY KEY  (`cd_modulo`),
) TYPE=MyISAM AUTO_INCREMENT=1 ;
# --------------------------------------------------------
```

```
# Estrutura da tabela ‘perfil‘
# --------------------------------------------------------
CREATE TABLE ‘perfil‘ (
   ‘vl_ip‘ varchar(15) default  ’000.000.000.000’,
  ‘nm_host‘ varchar(40) default ’’,
  ‘ds_dist‘ varchar(20) default ’’,
  ‘ds_kernel‘ varchar(10) default ’’,
  ‘ds_arq‘ varchar(10) default ’’,
  ‘qt_mem‘ varchar(6) default ’’,
  ‘ds_proc_vel‘ varchar(6) default ’’,
  ‘ds_proc_mod‘ varchar(10) default ’’,
  ‘ds_mod‘ varchar(100) default ’’,
  ‘ds_aut‘ varchar(40) default ’’,
  ‘vl_dns1‘ varchar(15) default ’’,
  ‘vl_dns2‘ varchar(15) default ’’,
  ‘vl_gateway‘ varchar(15) default ’’,
  ‘vl_prox‘ varchar(15) default ’’,
  PRIMARY KEY  (‘vl_ip‘)
) TYPE=MyISAM;
# --------------------------------------------------------
# Estrutura da tabela ‘portas‘
# --------------------------------------------------------
CREATE TABLE ‘portas‘ (
  ‘vl_ip‘ varchar(15) default ’000.000.000.000’,
  ‘vl_porta‘ varchar(6) default ’’,
  ‘ds_usuario‘ varchar(20) default ’’,
  ‘tp_porta‘ char(3) default ’’,
  ) TYPE=MyISAM;
# --------------------------------------------------------
# Estrutura da tabela ‘scripts‘
# --------------------------------------------------------
CREATE TABLE ‘scripts‘ (
  ‘cd_script‘ int(11) unsigned NOT NULL auto_increment,
  ‘cd_controle‘ int(11) unsigned default ’0’,
```

```
  `cd_so` int(11) unsigned default '0',
  `ds_script` varchar(100) default '',
  `ds_dados` varchar(255) default '',
  PRIMARY KEY  (`cd_script`)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
# --------------------------------------------------------
# Estrutura da tabela `servicos`
# --------------------------------------------------------
CREATE TABLE `servicos` (
  `vl_ip` varchar(15) default '000.000.000.000',
  `ds_nome` varchar(30) default '',
  `ds_usuario` varchar(20) default ''
) TYPE=MyISAM;
# --------------------------------------------------------
# Estrutura da tabela `sisarq`
# --------------------------------------------------------
CREATE TABLE `sisarq` (
  `vl_ip` varchar(15) default '000.000.000.000',
  `ds_particao` varchar(15) default '',
  `ds_montagem` varchar(100) default '',
  `vl_tipo` int(11) default '0',
  `ds_sist_arq` varchar(10) default '',
  `ds_perm` varchar(100) default ''
) TYPE=MyISAM;
# --------------------------------------------------------
# Estrutura da tabela `so`
# --------------------------------------------------------
CREATE TABLE `so` (
  `cd_so` int(11) NOT NULL auto_increment,
  `nm_so` varchar(20) NOT NULL default '',
  PRIMARY KEY  (`cd_so`)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
# --------------------------------------------------------
# Estrutura da tabela `usuarios`
```

```
# --------------------------------------------------------
CREATE TABLE `usuarios` (
  `cd_usr` int(11) NOT NULL auto_increment,
  `nm_usr` varchar(20) default '',
  `ds_senha` varchar(255) default '',
  `ds_validade` int(11) default '0',
  `ds_travado` char(1) default '',
  PRIMARY KEY  (`cd_usr`)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
```

Removendo as contas do *MySQL* que estão sem senha :

Para remover todos os usuários que não possuem uma senha cadastrada no gerenciador de banco de dados, pode-se utilizar os comandos a seguir:

```
mysql -p -u root mysql
mysql> delete from user where password='';
mysql> flush privileges;
mysql> \q
```

Criando o usuário tamanduá no servidor *MySQL* :

Os comandos a seguir permitem a criação do usuário que irá acessar as tabelas do protótipo, bem como fornecer tais permissões ao mesmo.

```
% mysql -u root mysql -p
mysql> GRANT ALL PRIVILEGES ON tamandua.* TO tamandua@'%'
IDENTIFIED BY 'tamandua' WITH GRANT OPTION;
mysql> FLUSH PRIVILEGES;
mysql> quit
```

5.4.2 *Apache* com suporte à *PHP*, *MySQL* e *mod_ssl*

Após a instalação e configuração do gerenciador de banco de dados, deve ser executado o processo de instalação e configuração do servidor *web* juntamente com seus módulos. O objetivo desta instalação é prover a infra-estrutura para a execução do *front-end* do protótipo, que foi implementado, utilizando-se a linguagem *PHP*.

Script de instalação

Este script realizará o *download* e a instalação dos pacotes necessários para o funcionamento do servidor *web* requerido pelo protótipo.

```
#!/bin/bash
# === ----------------------------------------------------
# Verifica se o usuário tem permissão para instalar o Apache
# === ----------------------------------------------------
if [[ $USER != "0" ]]
then
    echo " Este script deve ser executado através do usuário root"
    exit
fi
# === ----------------------------------------------------
# Download de Arquivos necessários para a instalação
# === ----------------------------------------------------
wget http://apache.usp.br/httpd/apache_1.3.29.tar.gz
wget http://www.modssl.org/source/mod_ssl-2.8.16-1.3.29.tar.gz
wget http://br.php.net/distributions/php-4.3.6.tar.bz2
# === ----------------------------------------------------
# Desempacotando arquivos
# === ----------------------------------------------------
tar -xvzf apache_1.3.29.tar.gz
tar -xvzf mod_ssl-2.8.16-1.3.29.tar.gz
bunzip2 php-4.3.6.tar.bz2
tar -xvf php-4.3.6.tar
# === ----------------------------------------------------
```

```
# Pré compilando o Apache
# === ----------------------------------------------------
cd apache_1.3.29
./configure
# === ----------------------------------------------------
# Compilando o PHP com suporte ao Apache
# === ----------------------------------------------------
cd ../php-4.3.6
./configure --with-apache=../apache_1.3.29 --with-mysql
make & make install
cp php.ini-dist  /usr/local/lib/php.ini
cd ..
# === ----------------------------------------------------
# Compilando o mod_ssl com suporte ao Apache
# === ----------------------------------------------------
cd mod_ssl-2.8.16-1.3.29
./configure  --with-apache=../apache_1.3.29 --with-ssl
cd ..
# === ----------------------------------------------------
# Compilando e instalando o Apache
# === ----------------------------------------------------
cd ../apache_1.3.29 --activate-module=src/modules/php4/libphp4.a,\
--with-layout=RedHat,--enable-module=ssl
make
make certificate
make install
```

Ajustes dos arquivos de configuração:

Para que seja possível executar alguns dos scripts do protótipo, é necessário realizar várias modificações nos arquivos de configuração dos componentes do servidor *web* instalado. Das quais pode-se destacar:

/etc/httpd/httpd.conf - Apache :

Para que o *Apache*, funcione perfeitamente com o protótipo, altere as seguintes linhas do arquivo *httpd.conf*:

```
# === ----------------------------------------
# Acessando scripts em php
# === ----------------------------------------
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
# === ----------------------------------------
# Definindo o index.php como a página de inicio
# === ----------------------------------------
DirectoryIndex index.php index.htm index.html
# === ----------------------------------------
# Definição do Grupo e do Usuário
# === ----------------------------------------
User tamandua
Group tamandua
# === ----------------------------------------
# Permitindo o uso de paginas pessoais
# === ----------------------------------------
<IfModule mod_userdir.c>
    UserDir public_html
</IfModule>
<Directory /home/*/public_html>
    AllowOverride FileInfo AuthConfig Limit
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    <Limit GET POST OPTIONS PROPFIND>
        Order allow,deny
        Allow from all
    </Limit>
    <LimitExcept GET POST OPTIONS PROPFIND>
        Order deny,allow
        Deny from all
    </LimitExcept>
</Directory>
```

php.ini - PHP :

Já o arquivo de configuração do *PHP* (*/usr/local/lib/php.ini*) deve sofrer as seguintes alterações:

```
# === ----------------------------------------
#  Habilita o uso de variáveis como nomes sem
#  prefixos
# === ----------------------------------------
register_globals = On
track_vars = On
```

5.4.3 Instalação do *Nmap*

Para a instalação do *scanner* de portas *NMAP* pode ser executado o seguinte *shell script*:

```
#!/bin/bash
   # === ----------------------------------------------------
   # Verifica se o usuário tem permissão para instalar o Nmap
   # === ----------------------------------------------------
   if [[ $USER != "0" ]]
   then
     echo "Este script deve ser executado através do usuário root"
     exit
   fi
   # === ----------------------------------------
   # Download do pacote de instalação
   # === ----------------------------------------
   wget  http://www3.linuxpackages.net/packages/Slackware-9.1/pcxz/ \
   nmap/nmap-3.50-i686-1.tgz
   # === ----------------------------------------
   # Instalando o arquivo
   # === ----------------------------------------
   installpkg nmap-3.50-i686-1.tgz
```

```
# === ---------------------------------------
#  Habilitando o setuid, para que o Tamanduá
# possa executar o levantamento das informações
# sobre os hosts
# === ---------------------------------------
chmod 6755 /usr/bin/nmap
```

5.4.4 Instalação do *Fping*

A ferramente de varredura de rede *Fping*, pode ser instalada utilizando-se o *script* a baixo:

```
#!/bin/bash
# === ---------------------------------------------------
# Verifica se o usuário tem permissão para instalar o Fping
# === ---------------------------------------------------
if [[ $USER != "0" ]]
then
    echo " Este script deve ser executado através do usuário root"
    exit
fi
# === ---------------------------------------
# Download do pacote de instalação
# === ---------------------------------------
wget http://www2.linuxpackages.net/packages/Slackware-10.0\
/kaneda/fping/fping-2.4b2-i486-1kan.tgz
# === ---------------------------------------
# Instalando o arquivo
# === ---------------------------------------
installpkg fping-2.4b2-i486-1kan.tgz
# === ---------------------------------------
#  Habilitando o setuid, para que o Tamanduá
# possa realizar a varredura da rede.
# === ---------------------------------------
chmod 6755 /usr/sbin/fping
```

5.4.5 *SUDO*

Script de instalação do *SUDO*

Para o *download* e instalação do *SUDO* pode ser utilizado o *shell script* abaixo:

```
#!/bin/bash
# === ----------------------------------------------------
# Verifica se o usuário tem permissão para instalar
# === ----------------------------------------------------
if [[ $USER != "0" ]]
then
    echo "Este script deve ser executado através do usuário root"
    exit
fi
# === ---------------------------------------
# download do arquivo
# === ---------------------------------------
wget http://www3.linuxpackages.net/packages/Slackware-9.1/ \
Console/SuDo/sudo-1.6.7p5-i486-1ok.tgz
# === ---------------------------------------
# Instalando o arquivo
# === ---------------------------------------
installpkg sudo-1.6.7p5-i486-1ok.tgz
```

Configuração do *SUDO*

Para que o usuário tamandua possa gerenciar alguns serviços na máquina onde o protótipo foi instalado, adicione a seguinte linha ao arquivo */etc/sudoers*

tamandua ALL=(ALL) NOPASSWD: ALL

E ainda, altere a posse do arquivo */etc/sudoers* para o usuário e grupo *root* e sua permissão para que ele somente possa ser lido pelo *root* e pelo seu grupo.

chown root.root /etc/sudosers

chmod 440 /etc/sudoers

5.5 Instalação do protótipo

Findo o processo de instalação dos componentes necessários para o funcionamento do protótipo, pode-se iniciar a instalação do protótipo em si, que pode ser executado com o uso do *shell script* que segue abaixo.

***Script* de instalação** :

```
#!/bin/bash
# === ----------------------------------------------------
# Verifica se o usuário tem permissão para instalar
# === ----------------------------------------------------
if [[ $USER != "0" ]]
then
    echo "Este script deve ser executado através do usuário root"
   exit
fi
# === ----------------------------------------------------
#  Criando o usuário tamandua
# === ----------------------------------------------------
useradd tamandua -g tamandua  -d /home/tamandua  -s /bin/bash
passwd tamandua
# === ----------------------------------------------------
#  Configurando o Mysql
# === ----------------------------------------------------
mysqladmin -u root passwd 'tamandua'
mysql -h 127.0.0.1 -u root -p  < cria_banco.sql
mysql -h 127.0.0.1 -u root -p  < cria_usuario.sql
mysql -h 127.0.0.1 -u root -p  < remove_usuarios_sem_senha.sql
# === ----------------------------------------------------
#  Alterando as configurações do SUDO
```

```
# === ----------------------------------------------------
echo 'tamandua  ALL=(ALL) NOPASSWD: ALL' >> /etc/sudousers
chown root.root /etc/sudousers
chmod 440 /etc/sudousers
# === ----------------------------------------------------
# Instalando os arquivos do protótipo
# === ----------------------------------------------------
cp -Rvf /home/tamandua
chown -Rvf tamandua.tamandua /home/tamandua
# === ----------------------------------------------------
#  (Re)iniciando serviços
# === ----------------------------------------------------
apachectl stop
sleep 2
apachectl start
# === ----------------------------------------------------
#  Ativando o Servidor de Ativação durante a inicialização
#           e o agendador de tarefas do tamandua
# === ----------------------------------------------------
echo "/home/tamandua/apv/serv-ativa&" >> /etc/rc.d/rc.local
echo "/home/tamandua/cron/tamandua-cron.bash&" >>/etc/rc.d/rc.local
```

Obtendo o pacote :

O Pacote necessário para a instalação e uso do Tamanduá-Mirim, pode ser obtido no *www.projetotamandua.cjb.net*. Uma forma alternativa de se obter o instalador do protótipo é através do comando abaixo.

wget http://www.projetotamandua.cjb.net/download/tamanduamirim.tgz

Desempacotando o pacote de Instalação :

Uma vez obtido o pacote, os arquivos necessários para instalação do protótipo podem ser extraídos utilizando-se o comando:

tar -xvzf tamanduamirim.tgz

Executando o Instalador :

Uma vez que os arquivos foram extraídos, o instalador pode ser executado utilizando-se os comandos:

cd tamanduamirim
./instalar

5.6 Estrutura de diretórios

Esta sessão visa descrever a estrutura de diretório, bem como a finalidade de cada um dos arquivos disponíveis no Agente Primário de Verificação

Ao ser instalado na máquina que funcionará como *A.P.V.*, o Tamanduá-Mirim criará a seguinte estrutura de diretórios:

/home/tamandua/ : raíz da árvore de diretório do Agente Primário de Verificação, contendo todos os componentes necessários para o seu funcionamento e dos demais componentes do protótipo;

/home/tamandua/acd/ : neste diretório estão os arquivos utilizados para gerar os pacotes de instalação dos Agentes de Coleta de Dados. Este diretório possui os seguintes arquivos:

acd-start.bash : *script* responsável por iniciar o Servidor de Coleta, informar ao Agente Primário de Verificação que *A.C.D.* está ativo e que já pode receber os *script* de verificação. Este é executado logo após a instalação do pacote contendo o *A.C.D*;

acd-stop.bash : este *script* realiza o encerramento de todos os componentes do *A.C.D.*, bem como a remoção dos arquivos instalados;

serv-coleta : este é o aplicativo responsável por receber os *script* de verificação, executá-los e retornar ao *A.P.V.* o arquivo de saída (*saida.log*), contendo as informações coletadas;

cli-ativa : aplicativo responsável por informar ao *A.P.V.* as mudanças de estado ocorridas no *A.C.D.*

/home/tamandua/apv/ : este diretório contém os aplicativos necessários para a comunicação com os demais agentes existentes no ambiente, sem os quais o processo de análise não seria possível. Os arquivos deste diretório são:

cli-coleta : aplicativo responsável pelo envio dos *script*s de verificação de dados aos *A.C.D.*;

serv-ativa : este aplicativo é o responsável pelo recebimento de informações sobre o estado de funcionamento dos *A.C.D.*, somentes após o recebimento da confirmação de ativação de um *A.C.D.* é que o mesmo poderá receber os *script*s de verificação;

/home/tamandua/cron/ : este diretório contém os *script*s que gerenciam o funcionamento do *A.P.V.* É composto pelos seguintes arquivos:

tamandua-pacote-acd.bash : *script* responsável pela criação dos pacotes de instalação dos *A.C.D.*s;

tamandua-cron.bash : este é o responsável por iniciar e parar os servidores do APV e por agendar a execução de outros processos;

serv-ativa-stop : *script* responsável pelo encerramento do servidor de

ativação;

serv-ativa-start : *script* responsável pela inicialização do servidor de ativação;

/home/tamandua/hosts/ : diretório contendo os arquivos de saída, dos *script*s de coleta de dados, enviados pelos *A.C.D.*, e seus respectivos pacotes de instalação. Cada subdiretório é o endereço *IP* de um nó ativo e descoberto durante a varredura da rede.

/home/tamandua/scripts/ : este diretório contem os *script*s de coleta de dados e de varredura da rede;

/home/tamandua/scripts/controles/ : neste diretório estão os *script*s de coleta utilizados para obter os dados que permitirão informar o nível de aderência do ambiente aos controles da Norma Nacional de Segurança da Informação. Os arquivos deste diretório são:

721.sh : *script* responsável pela verificação do controle 7.2.1 - Registro de usuário;

722.sh : *script* responsável pela verificação do controle 7.2.2 - Gerência de privilégios;

723.sh : *script* responsável pela verificação do controle 7.2.3 - Gerenciamento de senha dos usuários;

743.sh : *script* responsável pela verificação do controle 7.4.3 - Autenticação para conexão externa do usuário;

752.sh : *script* responsável pela verificação do controle 7.5.2 - Procedimentos de entrada no sistema;

753.sh : *script* responsável pela verificação do controle 7.5.3 - Iden-

tificação e autenticação do usuário;

754.sh : responsável pela verificação do controle 7.5.4 - Sistema de gerenciamento de senhas;

755.sh : *script* responsável pela verificação do controle 7.5.5 - Uso de programas utilitários;

757.sh : *script* responsável pela verificação do controle 7.5.7 - Desconexão do terminal por inatividade;

758.sh : *script* responsável pela verificação do controle 7.5.8 - Limitação do tempo de conexão;

/home/tamandua/scripts/perfil/ : o único arquivo deste diretório é o arquivo perfil.bash, responsável pelo levantamento do perfil dos nós que possuem o *A.C.D.* instalado;

/home/tamandua/scripts/varredura/ : este diretório contém o *script* de varredura da rede bem como o seu resultado. Com as informações deste diretório será possível determinar em quais máquinas devem ser instalado o Agente de Coleta de Dados. Ele é composto dos seguintes arquivos e diretório:

/home/tamandua/scripts/varredura/resultado/ : diretório que contém o arquivo *varredura.dat*, com as informações sobre os nós ativos no momento da varredura;

descobre-rede.sh : *script* responsável por determinar as informações do endereçamento *IP* da máquina onde o *A.P.V.* está instalado;

varredura-da-rede.sh : *script* responsável pelo levantamento dos endereços *IP*s ativos, pela varredura de portas e pela determi-

nação do sistema operacional destes nós;

/home/tamandua/public_html/ : neste local são armazenadas as páginas que formam o *front-end* do Tamanduá-Mirim com o usuário, bem como alguns *script*s responsáveis pelo gerenciamento dos demais componentes do modelo. Os principais arquivos deste diretório são:

tamandua_analisa_arquivos.php : este *script* é invocado toda vez que ocorre a verificação dos arquivos de análise de um nó;

tamandua_analisa_7.5.8.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.5.8 da *NBR*;

tamandua_analisa_7.5.7.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.5.7 da *NBR*;

tamandua_analisa_7.5.5.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.5.5 da *NBR*;

tamandua_analisa_7.5.4.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.5.4 da *NBR*;

tamandua_analisa_7.5.3.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.5.3 da *NBR*;

tamandua_analisa_7.5.2.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.5.2 da *NBR*;

tamandua_analisa_7.4.3.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.4.3 da *NBR*;

tamandua_analisa_7.2.3.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.2.3 da *NBR*;

tamandua_analisa_7.2.2.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.2.2 da *NBR*;

tamandua_analisa_7.2.1.php : este *script* é invocado toda vez que ocorre a verificação de aderência ao controle 7.2.1 da *NBR*;

principal.php : contém a estrutura de frames do *front-end*;

index.php : página de abertura que realiza a autenticação do usuário no Tamanduá-Mirim;

inc_usr.php : este *script* contém as rotinas necessárias para o acesso ao banco de dados assim como as principais variáveis de controle utilizadas pelo protótipo;

conteudo_usu_travar.php : *script* utilizado para bloquear uma conta de usuário dentro do protótipo;

conteudo_usu_senha.php : *script* utilizado para alterar a senha de uma conta de usuário dentro do protótipo;

conteudo_usu_remover.php : *script* utilizado para remover a conta de um usuário dentro do protótipo;

conteudo_usu_criar.php : *script* utilizado para criar uma conta de usuário dentro do protótipo;

conteudo_tam_script.php : *script* utilizado para gerenciar os *script* de coleta de dados utilizados no protótipo;

conteudo_tam_remover.php : *script* utilizado para remover os dados coletados durante a ultima análise;

conteudo_tam_desinstala.php : *script* utilizado para desinstalar o protótipo, removendo todos os arquivos e alterações realizadas na máquina durante a instalação e uso do mesmo;

conteudo_an_status_host.php : este *script* informa sobre o andamento do processo de análise de um determinado nó;

conteudo_an_status.php : este *script* informa sobre o estado de funcionamento dos servidores que compõem o Agente Primário de Verificação de Dados;

conteudo_an_relatorios_hosts.php : este *script* informa sobre o processo de análise dos nós do ambiente;

conteudo_an_relatorios_controles.php : este *script* descreve quais controles já foram analisados e os respectivos resultados;

conteudo_an_relatorios.php : este *script* apresenta os relatórios sobre a análise do ambiente;

conteudo_an_perfil_remover.php : este *script* remove um nó do processo de análise;

conteudo_an_perfil_novo_host.php : *script* utilizado para adicionar um novo nó ao processo de análise;

conteudo_an_perfil_info.php : *script* utilizado para exibir as informações contidas no perfil dos nós que atualmente fazem parte do processo de análise;

conteudo_an_perfil_gera_todos_pacote.php : *script* utilizado para gerar os pacotes de instalação dos Agentes de Coleta de Dados para

todas as máquinas que fazem parte da análise atual;

conteudo_an_perfil_gera_pacote.php : *script* utilizado gerar o pacote de instalação do Agentes de Coleta de Dados para um dos nós que é alvo da analise;

conteudo_an_discovery.php : *script* utilizado exibir os dados colidos durante a ultima varredura da rede;

conteudo_an_discovery_novo.php : *script* utilizado realizar uma varredura da rede.

5.7 Utilização do protótipo

O protótipo, após ser instalado, pode ser acessado através de um navegador, como por exemplo o *Mozilla*, para isto informe o endereço da máquina onde ele foi instalado seguido de sua localização, como por exemplo: *http://localhost/ tamandua/*. Feito isto uma tela de *login*, semelhante a Figura 5.2, será exibida.

Figura 5.2: Tamanduá: Tela de *Login*

Uma vez feito o *login* o usuário terá acesso a tela de abertura, com uma breve descrição do projeto e do protótipo, como mostra a figura 5.3.

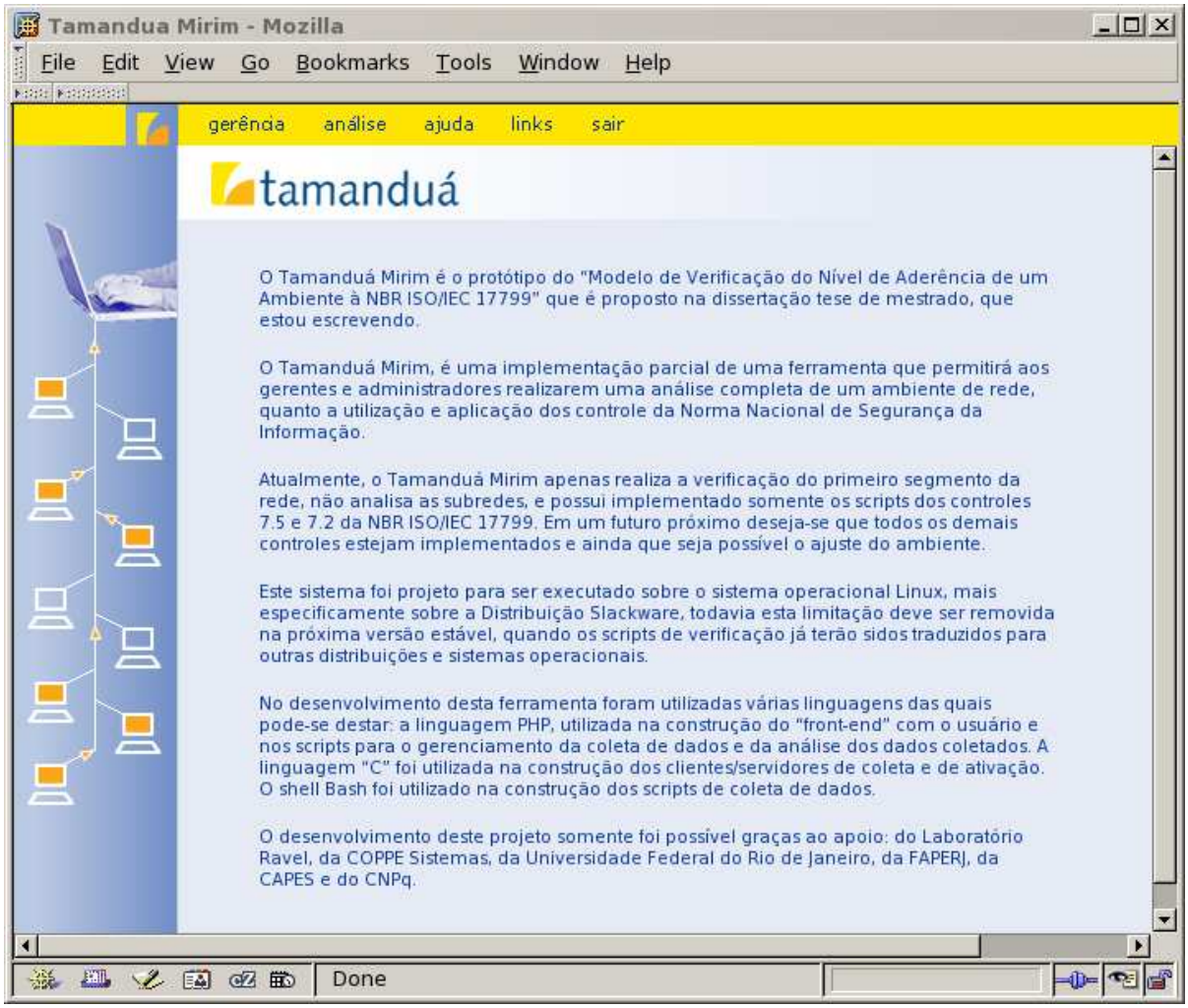

Figura 5.3: Tamanduá: Tela de Abertura

O primeiro passo no uso do protótipo é realizar o cadastro dos usuários que poderão ter acesso ao tamandua, para isto clique no link Gerência, em seguida no menu que aparece do lado esquerdo, clique em Criar, informe os dados do usuário, confirme e pronto.

Além de realizar este cadastro é possível alterar a senha de um usuário e bloquear o seu acesso ao protótipo.

Uma vez cadastrados os usuários, deve-se realizar a inclusão dos controles da *NBR ISO/IEC 17799* a serem verificados, para isto selecione a opção Controles disponível a partir do menu Gerência, feito isto surgirá uma tela semelhante a Figura 5.4.

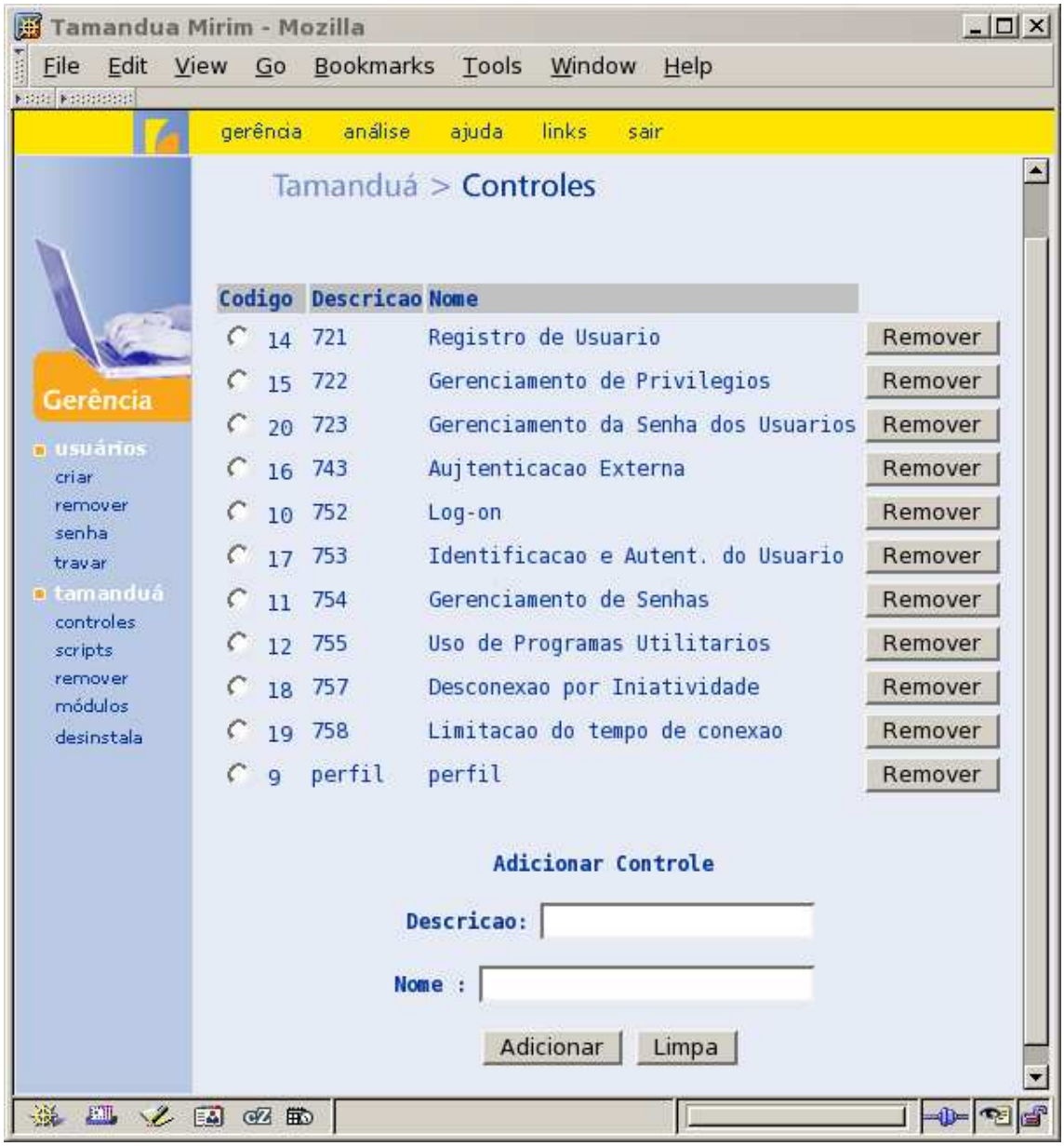

Figura 5.4: Tamanduá: Tela de Cadastro de Controles

Após o cadastro de todos os controles, deve-se realizar o cadastro dos *scripts* de verificação, esta ação pode ser realizada através da opção *Scripts* do menu *Gerência*. A tela de cadastro dos *scripts* pode ser vista na Figura: 5.5.

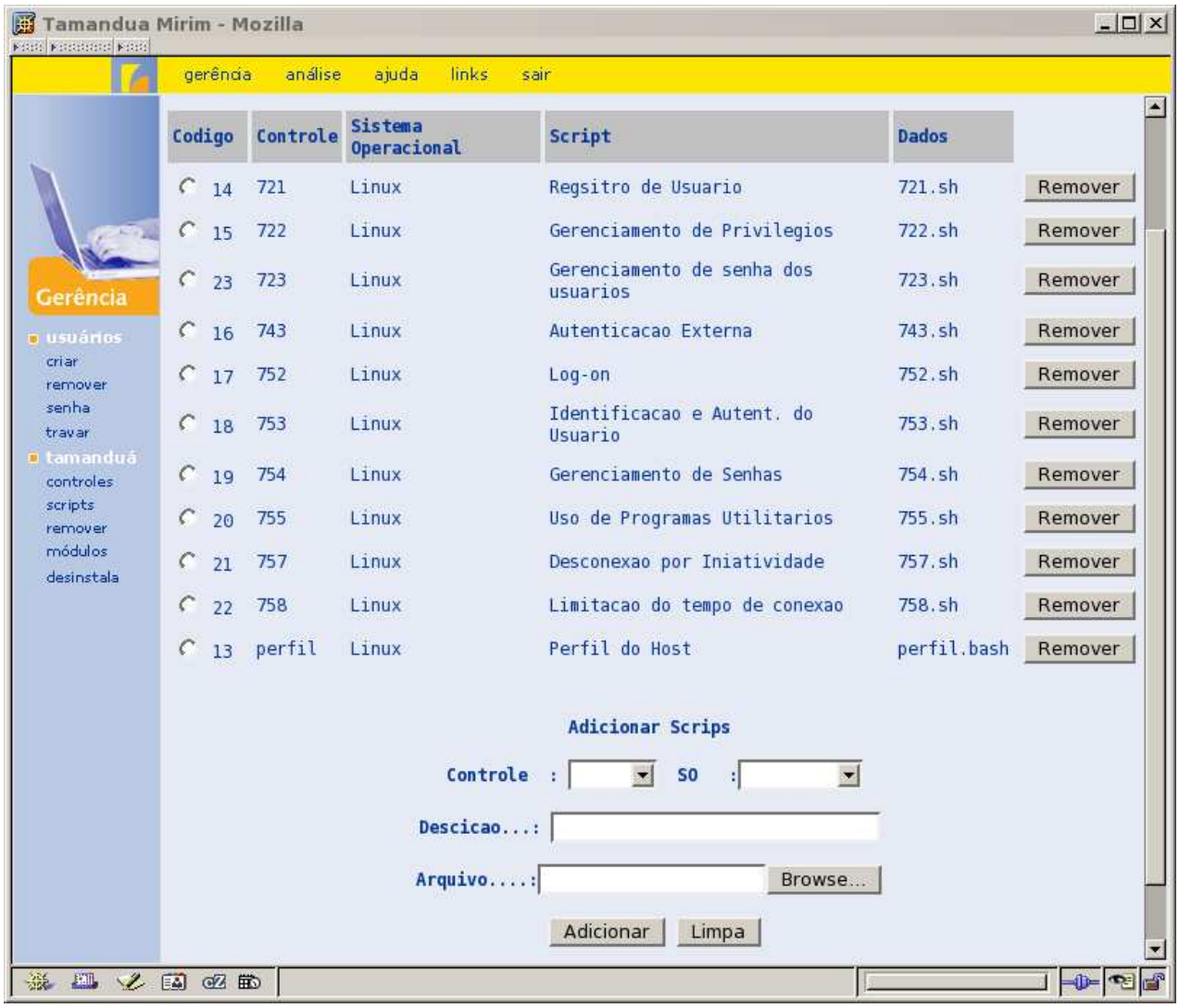

Figura 5.5: Tamanduá: Tela de Cadastro de *Scripts*

Antes de se iniciar um novo processo de análise, os dados da ultima análise devem ser removidos deste modo, para excluir os dados armazenados no banco de dados do Tamanduá-Mirim, deve ser utilizado a opção Remover, disponível através do menu Gerência. Uma vez selecionada esta opção será apresentada uma tela na qual o usuário deverá confirmar a exclusão destas informações.

Outra opção importante do menu Gerência é a Desinstala, que irá desinstalar todos os componentes do Tamanduá-Mirim, deixando a máquina livre deste aplicativo

e de suas configurações.

Clicando no meu Análise tem-se acesso as opções relacionadas ao processo de análise de aderência do ambiente à *NBR ISO/IEC*, este é composto pelas seguintes opções:

Discovery **:** utilizada para a realização da coleta inicial de informações sobre o ambiente. Ela executará a varredura da rede e a determinação do sistema operacional atualmente disponível nas máquinas. Esta opção abre uma tela na qual é possível acessar os *Links*: Nova Varredura e Informações sobre os *Hosts*, tendo na Figura 5.6 um exemplo de sua aparencia.

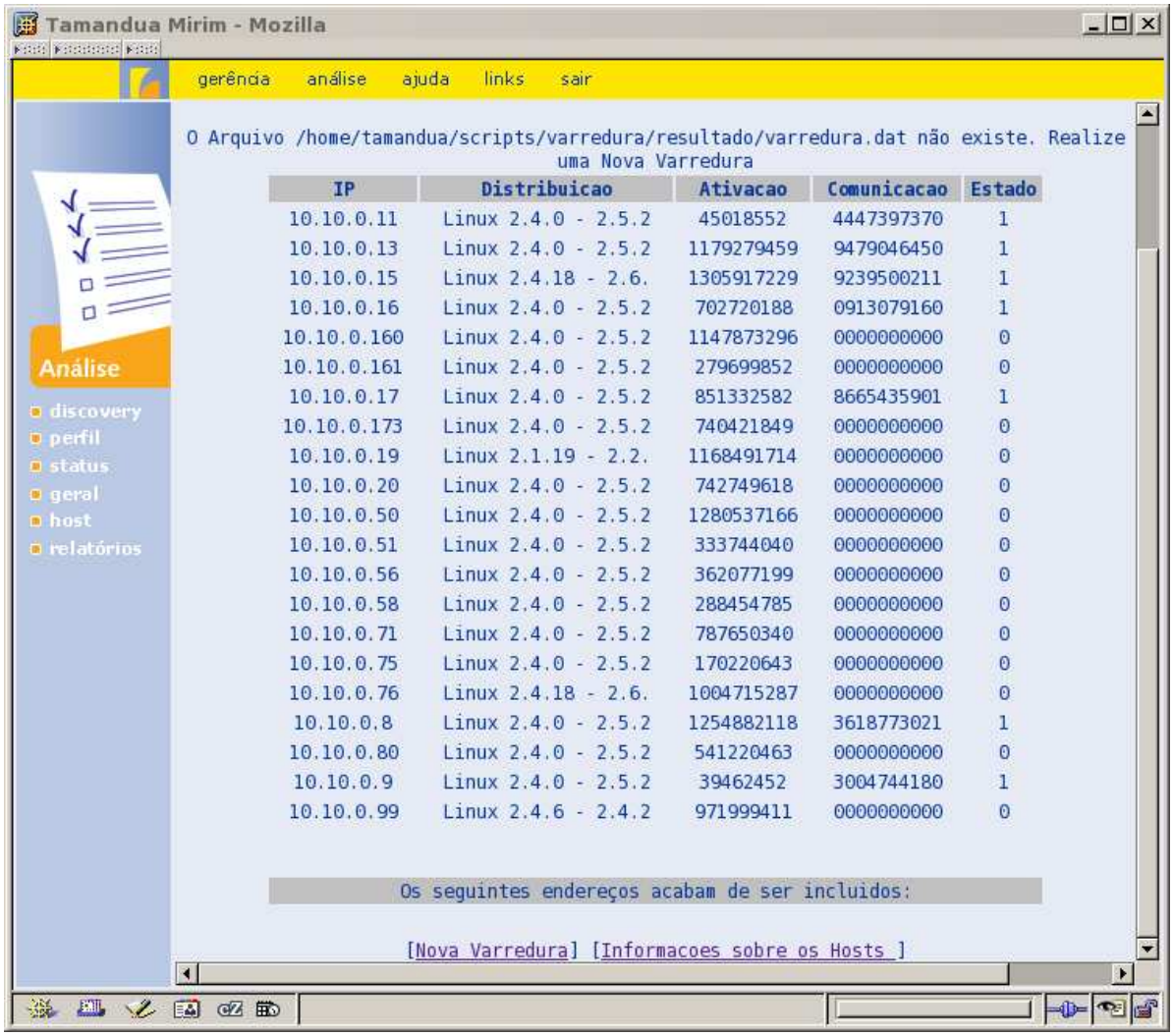

Figura 5.6: Tamanduá: Tela de Varredura

Perfil : esta opção permite visualizar o perfil de todas as máquinas detectadas pela varredura de rede, realizada anteriormente; permite Adicionar e Remover máquinas à análise; gerar o pacote de instalação do Agente de Coleta de Dados de uma única máquina ou de todas as máquinas cadastradas. Um exemplo desta tela pode ser observado na Figura: 5.7.

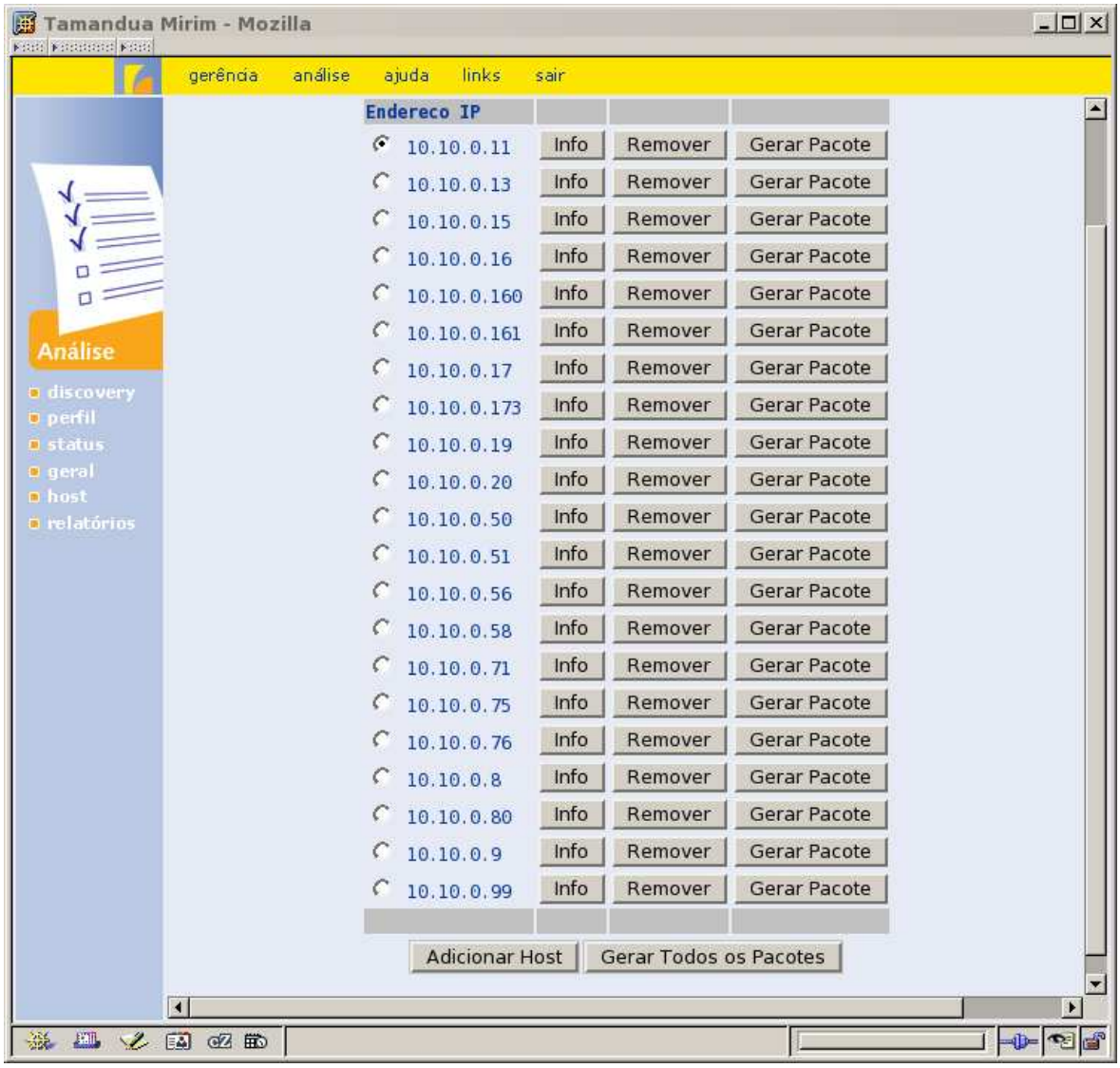

Figura 5.7: Tamanduá: Tela de Perfil

Status **:** a partir desta opção é possível gerenciar os Servidores que formam o *A.P.V.* e que estão rodando na máquina que hospeda o Tamanduá-Mirim. A Figura 5.8 é um exemplo da tela que será exibida quando esta opção for selecionada.

Figura 5.8: Tamanduá: Tela de *Status*

Geral : utilizando esta opção é possível obter uma visão geral do processo de análise do ambiente. Como mostra a figura 5.9, a tela desta opção informa o total de máquinas detectadas, quantas possuem um *A.C.D.* ativo, em quantas destas a coleta de dados já começou e em quantas a análise já terminou.

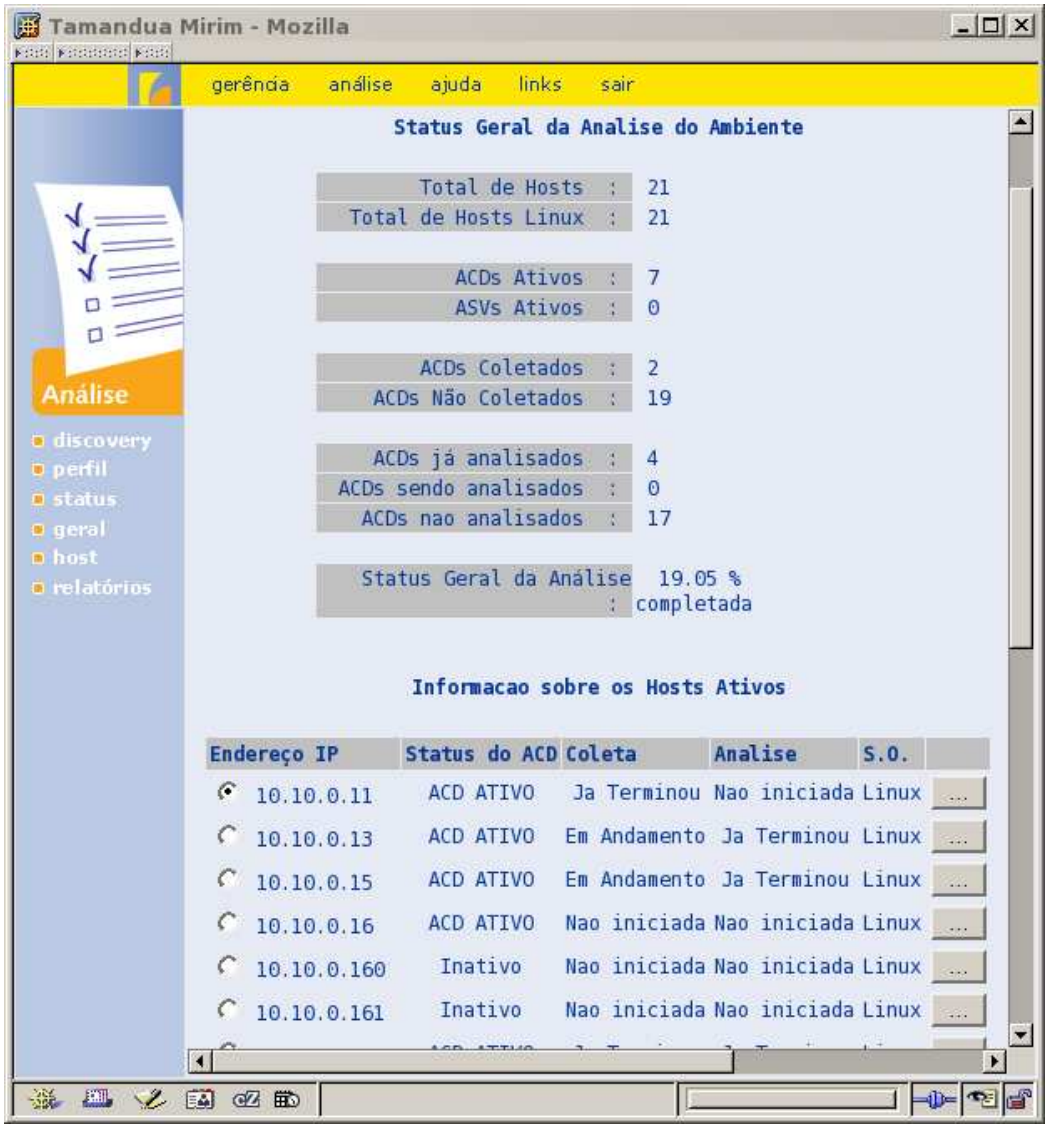

Figura 5.9: Tamanduá: Tela com o Status Geral da Análise

Host : esta opção permite obter uma visão mais específica do processo de análise do ambiente, como mostra a figura 5.10. Selecionando-se uma das máquinas é possível observar o andamento de sua análise, a figura 5.11 contém um exemplo das informações que podem ser obtidas.

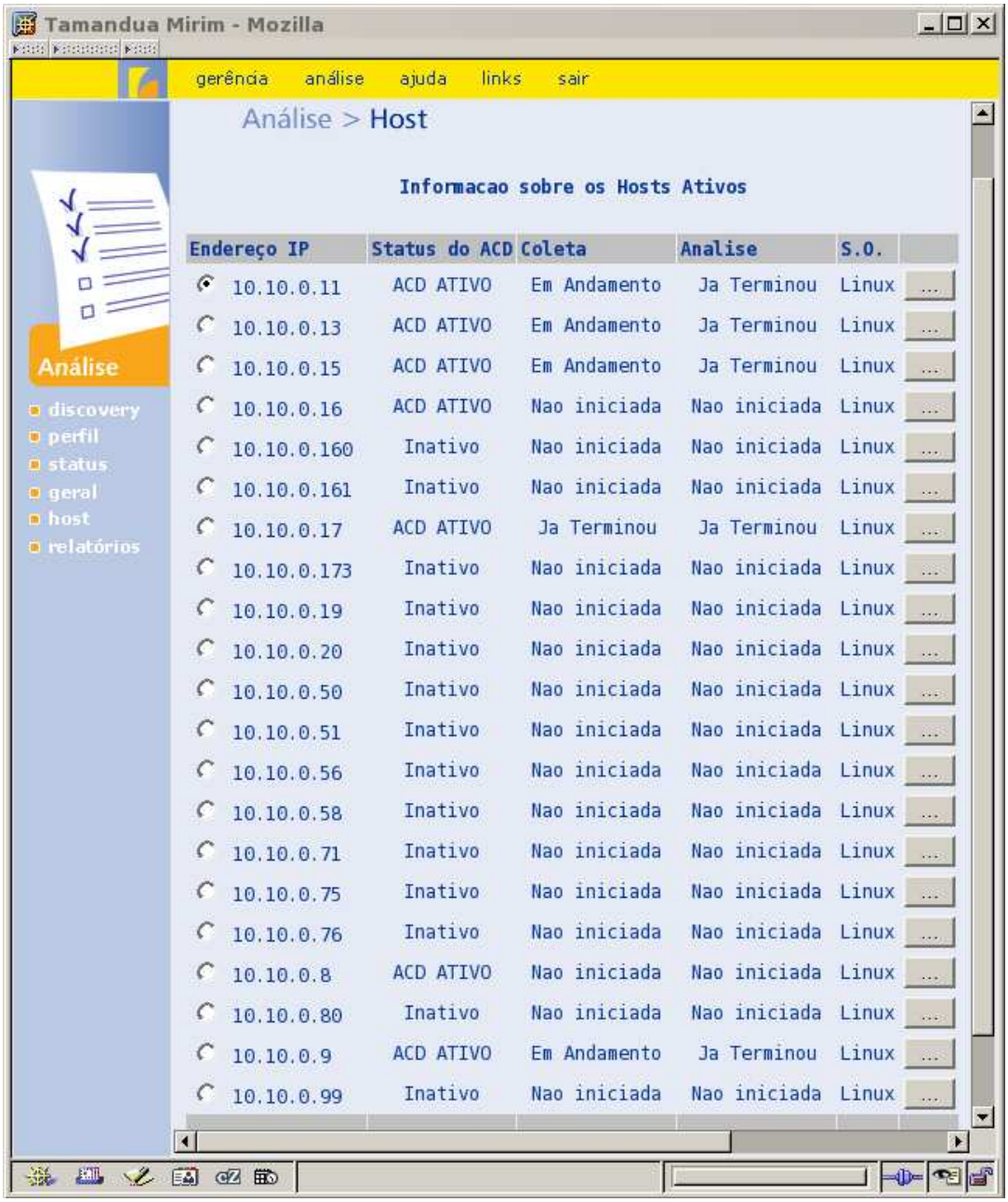

Figura 5.10: Tamanduá: Tela para Seleção da Máquina a ser verificada

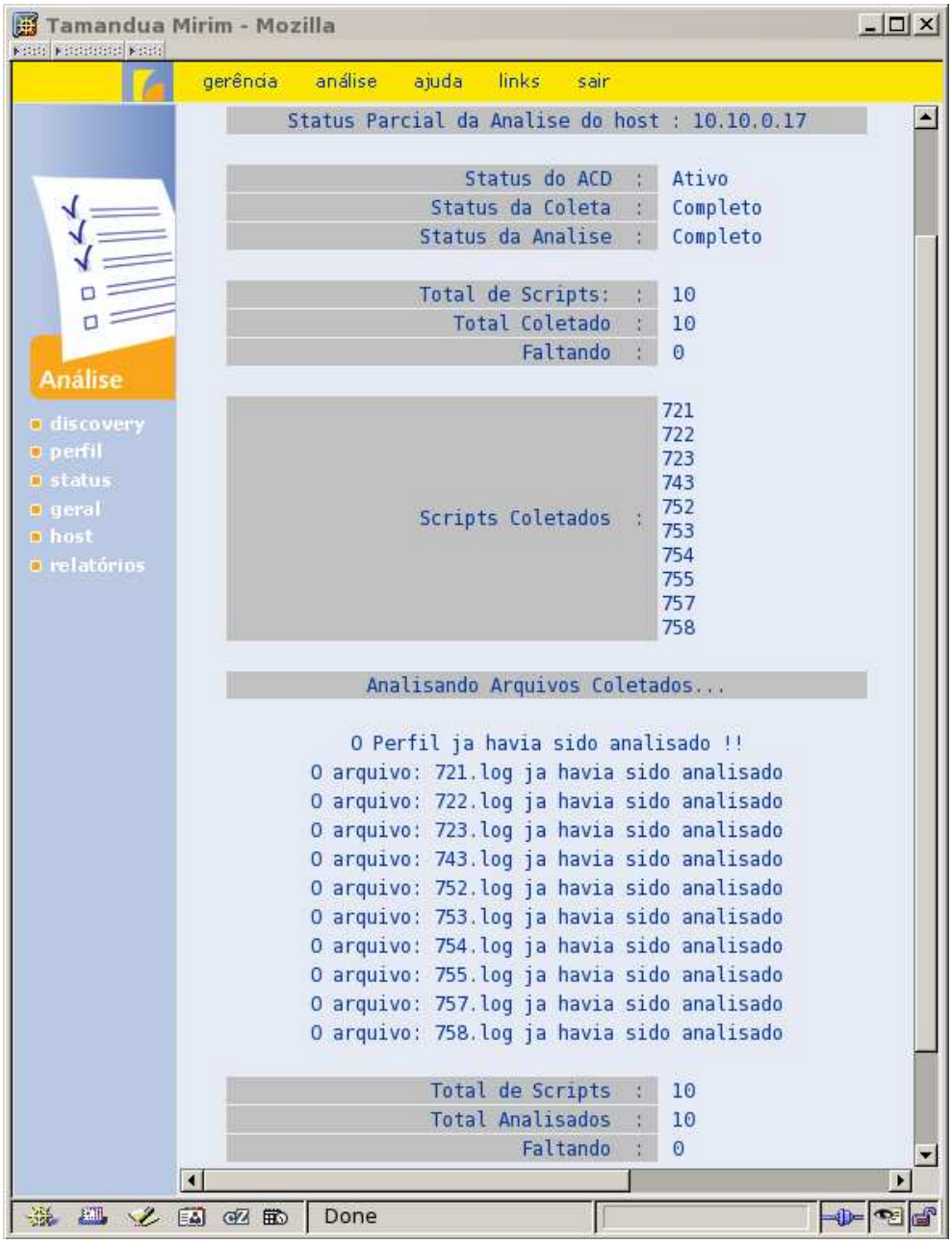

Figura 5.11: Tamanduá: Tela do *Status* da Análise em uma Máquina

Relatórios : através desta é possível obter acesso à um dos vários relatórios gerados pelo Tamanduá-Mirim. Atualmente os únicos três relatórios disponíveis são o Relatório Geral da Análise, o Resultado da Análise dos Controles e o Resultado da Análise dos Hosts.

Maiores informações sobre o uso e funcionamento do protótipo podem ser obtidas através do menu Ajuda, o qual descreve rápidamente todas as opções disponíveis no Tamanduá-Mirim.

5.8 Exemplo de aplicação

Uma vez implementado o protótipo, foram aplicados alguns testes de uso. Um deles foi realizado no Laboratório de Redes de Alta Velocidade (RAVEL) da Universidade Federal do Rio de Janeiro (UFRJ). Os dados obtidos neste teste serão utilizados como um exemplo de aplicação do protótipo em um ambiente real.

5.8.1 Ambiente analisado

O ambiente analisado, como pode ser visto na Figura 5.12 era formando por um conjunto de aproximadamente 21 máquinas da plataforma *Intel* com os sistemas operacionais *Windows* e *Linux* instalados. Todas as máquinas estão conectadas a *internet* através de um enlace de dados gerenciado por um *gateway* de segurança. O ambiente analisado, era formado por somente um segmento de rede (sub-rede 10.10.0.0/24). As informações que se seguem dizem respeitos a análise deste ambiente.

5.8.2 Resultados obtidos

Ao contrário do *COBRA*, que forneceria como resultado um relatório, baseado em questionários eletrônicos e informações percentuais sobre o nível de aderência, vide Figura 4.23. O uso do protótipo permitiu obter informações mais precisas sobre o processo de homologação, foi possível, por exemplo, saber quais controles foram homologados em um determinado nó, em quais nós um controle foi certificado,

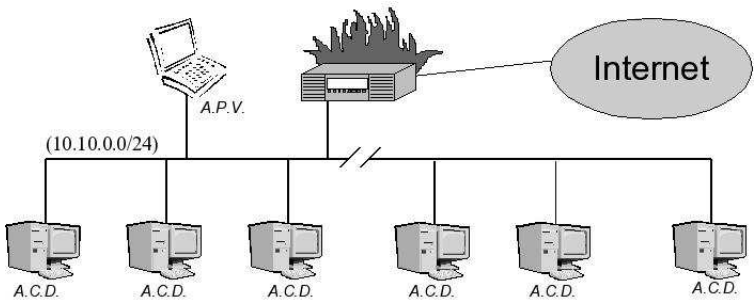

Figura 5.12: Tamanduá: Rede analisada

a quantidade de nós já analisados e quantos dos nós analisados estão totalmente aderentes. Logo protótipo forneceu mais detalhes sobre os problemas encontrados nos nós da rede do que seriam fornecidos pelo *COBRA*.

Com o uso do protótipo foi possível ter acesso a três relatórios. O primeiro traz uma visão geral da análise até aquele momento. O segundo é baseado nos controles e informa quais os nós da rede estão aderentes, para aqueles que não estão aderentes ele apresenta uma breve descrição da justificativa da não homologação. Já o terceiro se baseia nas máquinas e informa quais controles foram homologados, para aqueles que não foram certificados é apresentada uma descrição do motivo da não aderência.

Estes relatórios apresentam de forma objetiva os motivos que levaram os nós da rede e o ambiente a não estarem aderentes a *NBR*, facilitando assim o processo de ajuste dos nós e consequentemente de todo o ambiente. Ao contrário dos relatórios gerados pelo protótipo, os relatórios do *COBRA* apresentam uma visão geral do ambiente, sem um detalhamento das conformidades e não conformidades específicas de cada nó do ambiente.

Infelizmente não foi possível uma comparação do protótipo (Tamanduá-Mirim) com a ferramenta da Módulo (*Checkp Tool*), pois esta empresa não fornece cópias de avaliação de seu produto.

Os resultados exibidos a seguir foram coletados a partir da análise do ambiente descrito acima, e fornecem uma referência dos relatórios anteriormente mencionados, bem como das informações que podem ser obtidas com o uso da implementação atual do protótipo.

A Figura 5.13 é o exemplo de relatório que pode ser obtido durante a análise

de um ambiente, ele nos fornece uma visão geral do processo de verificação. Este relatório informa: a quantidade total de nós do ambiente, o sistema operacional dos nós, em quantos nós a análise já chegou o fim, em quantos a análise ainda está sendo executada e em quantos a análise ainda não começou. Em seguida é informado quais nós apresentam não conformidades aos controles examinados, e finalmente o estado geral a análise dos *scripts*, com a quantidade total enviada, a quantidade não homologada e homologada.

A Figura 5.14 é o exemplo da tela a ser utilizada para se obter um relatório sobre a análise do ambiente onde as informações estão organizadas de forma a exibir o resultados agrupados pelos controles da *NBR ISO/IEC 17799*, uma vez selecionado um dos controles uma nova tela será aberta com as informações sobre a coleta deste controle.

A Figura 5.15 é o exemplo da tela obtida quando é solicitado o relatório de análise baseado em um determinando controle. A tela atual foi construída tendo como base as informações obtidas a partir da análise do controle 7.2.1 (Registro de Usuário).

:: Dados Gerais da Analise ::

Atualmente existem 21 endereços IPs ativos no ambiente, destes 21 estão rodando o sistema operacional Linux e são o alvo da análise atual.

Destas, um total 5 já foram completamente analisadas, 0 ainda estão sendo analisadas e em 16 a análise inda não começou.

As 5 máquinas nas quais a análise já começou e seus estados são:

10.10.0.9	Com restrição
10.10.0.11	Com restrição
10.10.0.15	Com restrição
10.10.0.17	Com restrição
10.10.0.13	Com restrição

Logo, há um total de 5 máquinas com restrições, estas as impediram de serem homologas, e neste mesmo ambiente há 0 máquina que estão aderentes a norma NBR/ISO-IEC 17799.

De um total de 50 arquivos, com as saidas dos scripts de coleta, analisados, 30 possuem alguma restrição que os impediu a homologação do conrole correspondente e 20 arquivos permitiram a homologação de seus controles.

Figura 5.13: Tamanduá: Resultado geral da análise

Figura 5.14: Tamanduá: Tela para seleção do controle

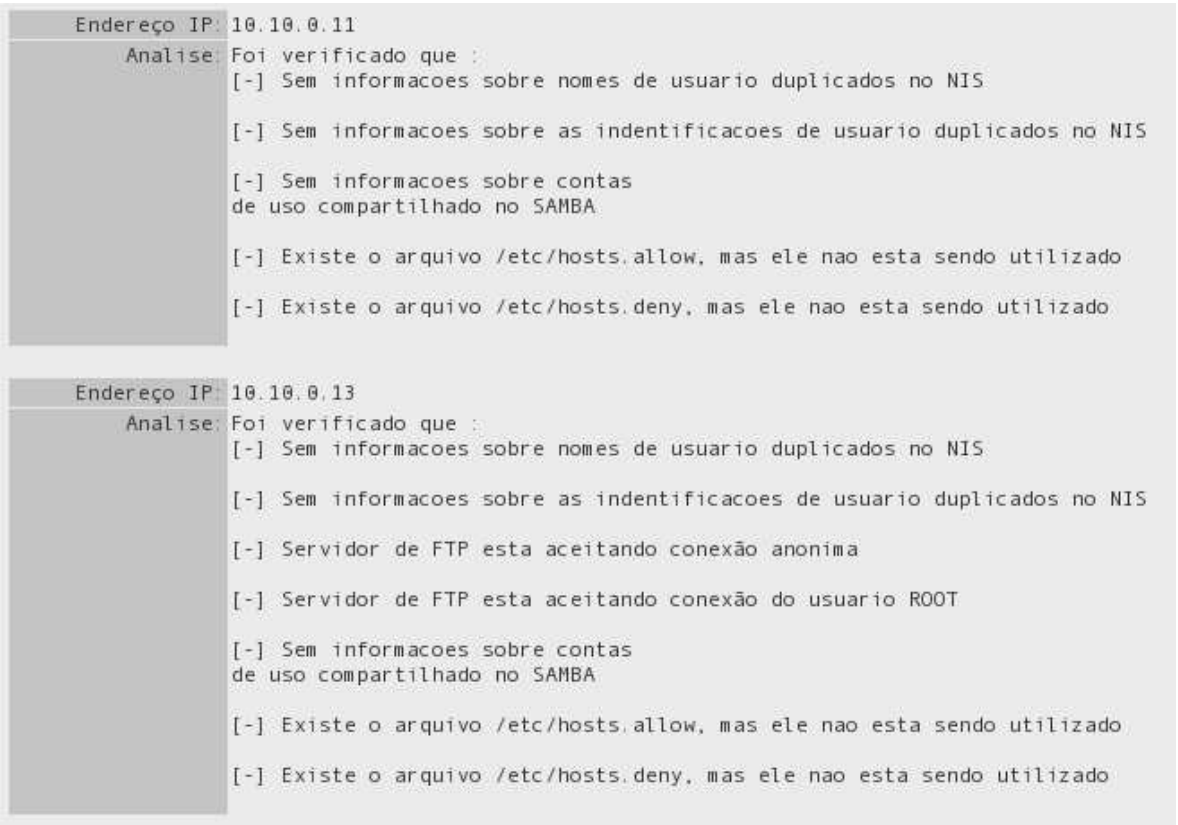

Figura 5.15: Tamanduá: Relatório baseado nas informações de um controle

O último tipo de relatório é aquele cujo exemplo é mostrado na figura 5.16. Neste é possível ter acesso ao resultado da análise de todos os *scripts* de coleta já enviados ao nó. Esta tela além de informar quais controles já foram analisados apresenta alguns comentários sobre a análise da cada controle.

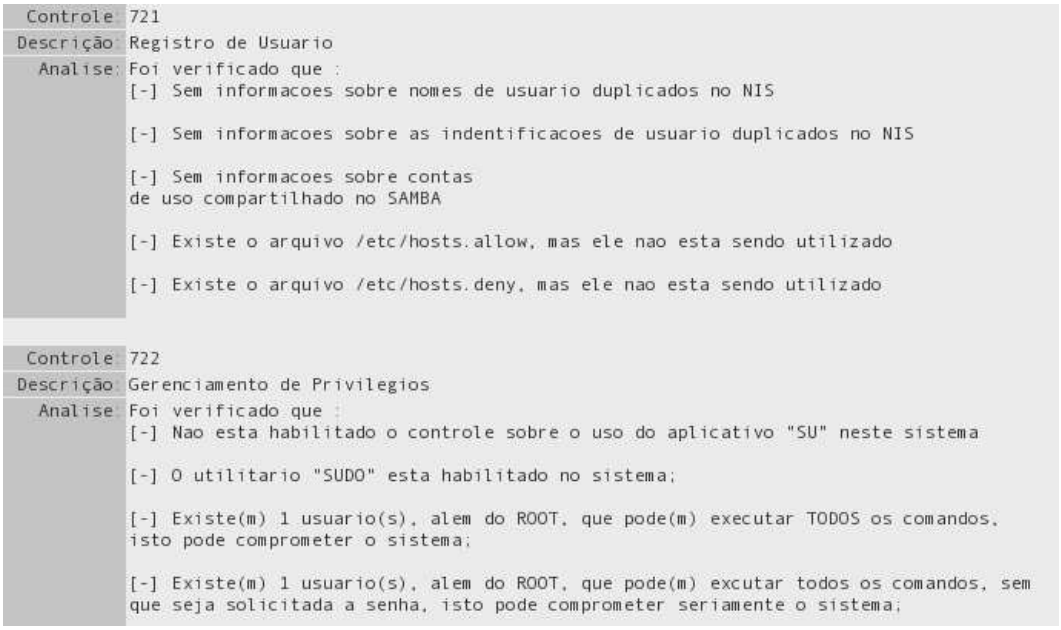

Figura 5.16: Tamanduá: Relatório baseado nas informações de um nó

Capítulo 6

Conclusão

Este capítulo encerra o trabalho apresentando os resultados obtidos com a implementação do protótipo, as descrições das contribuições do projeto, as sugestões para trabalhos futuros e os possíveis esforços a serem aplicados para a evolução do Tamanduá-Mirim.

6.1 Conclusões

O principal objetivo da tese foi alcançado, o qual era propor um modelo que permitisse uma coleta automática das evidências de que um ambiente computacional está aderente, ao menos parcialmente, a norma nacional de segurança; modelo este que poderá ser utilizado na construção de várias ferramentas de segurança.

O uso da técnica de varredurra de rede foi de vital importância no processo de análise do ambiente, através da qual foi possível determinar em quais máquinas os *A.C.D.* deveriam ser instalados, e com estas informações foi possível gerar os pacotes de instalação dos mesmos.

Após a instalação dos *A.C.D.*, o uso dos clientes e do servidor de ativação, permitiu determinar o momento certo do início da coleta de dados de cada *A.C.D.*, a qual procedeu de forma normal na maior parte dos casos.

Graças ao uso dos servidores e dos *scripts* de coleta de dados, foi possível obter as informações que evidenciem a implementação dos controles da *NBR-ISO/IEC*

17799 nas máquinas que possuíam o sistema operacional *Linux* instalado. Como os *scripts* de coleta foram escritos utilizando-se apenas os comandos básicos do *Linux*, eles podem ser facilmente adaptados para funcionarem em mais de uma distribuição *Linux*, alguns deles já foram portados para a distribuição *SuSe Linux*.

Como não foram implementados os clientes e o servidor de módulo no protótipo, a coleta dos *scripts* que usavam os comandos *stat* e *awk* não retornaram todas as informações previstas, uma vez, que algumas máquinas não possuíam estas ferramentas instaladas. Estes problemas foram contornados com a instalação manual dos pacotes e com uma nova submissão das máquinas ao processo de coleta de dados, o qual foi finalizado com sucesso. Estes incidentes deixaram claro a necessidade da implementação destes componentes na próxima versão do protótipo.

Uma dificuldade detectada no processo de implementação do protótipo foi a tradução dos controles nos *scripts* de coleta. Este processo foi um dos mais complexos, já que envolve conhecimentos avançados da arquitetura de *hardware* e *software* a ser analisado. O que demonstra a necessidade do estudo de uma forma mais simples, independente de *hardware* e *software* para a escrita destes *scripts*.

A medida que o protótipo foi sendo construído observou-se que o modelo proposto pode ser utilizado na implementação de várias ferramentas de segurança e gerência de um ambiente de rede, as quais serão discutidas logo a seguir.

6.2 Contribuições

A primeira contribuição a ser destacada é a possibilidade da construção de uma ferramenta que, além de se basear no uso de coletores automático de dados e de realizar uma coleta de informações quase sem interferência dos usuários, permitiria a análise, o gerenciamento e a adaptação de todo um ambiente computacional à um conjunto de regras préviamente determinadas. A única restrição existente é que o ambiente de rede esteja utilizando o protocolo *TCP/IP*, não importando se o mesmo possui subredes ou outros segmentos interconectados.

O modelo proposto pode ser expandido para atender a necessidade de outras normas e até mesmo o desenvolvimento de uma política de segurança que atenda as necessidades específicas de uma organização. Outra expansão possível é a inclusão

de mecanismos através dos quais a coleta automática possa ser complementada por um processo automático, porém autorizado, de adaptação do ambiente aos controles que não foram homologados.

O modelo proposto também pode ser utilizado para construção de outros tipos de ferramentas. Uma delas seria uma ferramenta de gerência das máquinas de uma rede e subredes a partir de um ponto central ou através de um módulo *web*.

6.3 Sugestões para trabalhos futuros

Devido a dificuldade na construção dos *scripts* de coleta, um dos trabalhos a serem desenvolvidos é a definição de uma linguagem mais intuitiva para a construção dos mesmos. Esta pseudo linguagem deve ser independente do sistema operacional utilizado nos agentes de coleta, mas seu uso exigirá a criação de um novo componente para a interpretação, tradução e execução dos *scripts*.

Outro passo é a implementação dos demais controles da norma nacional de segurança que não foram contemplados, bem como a inclusão de novas normas e controles ao processo de análise.

Pode-se ainda alterar o modelo, para que ele permita o ajuste de forma automática do ambiente aos controles a que foi submetido, este ajuste apesar de ser automático, deve ser autorizado pelo administrador ou gerente do ambiente.

Pode-se fazer uso do modelo na implementação de uma ferramenta de gerência de rede que permita a instalação, atualização e ajustes das máquinas de forma simples e guiada por um conjunto de coletores de informações.

Realizar a implementação dos agentes para outras plataformas, como por exemplo: para o sistema operacional *Windows, Solaris, Aix, Qnx, OpenBSD, NetBSD, FreeBSD* e para outras arquiteturas de hardware como *Sparc, PowerPC, Apple* e outras.

Implementar uma rotina de *update* automático dos *scripts* e das normas disponíveis a partir de um ponto central. Assim, os usuários do Tamanduá poderão manter a base de conhecimento utilizada na análise do ambiente sempre atualizada.

Calcula-se que para a tradução e implementação de todos os controles tecnológicos existentes na NBR nos *scripts* de coleta de dados para os sistemas operacionais *Linux* e *Windows* por um especalista nestes sistemas e com largo conhecimento em programação nas linguagens C, C++ e *shell script* serão necessários pelo menos 24 meses de trabalho. Já para a implementação completa do modelo em um protótipo completamente usual há a previsão da necessidade de mais 18 meses.

Apêndice A

Níveis de classificação adotados pelo Orange Book

Esta apêndice tem por objetivo descrever rapidamente os principais níveis de classificação utilizados pelo Orange Book e seus requisitos. Para maiores informações sobre o Orange Book consulte as referências: DoD 5200.28-STD [17], Proof Requeriments in the Orange Book[38] e Rainbow Series Library [40].

Níveis de Classificação

De acordo com o Orange Book [17], os principais requisitos de segurança a serem satisfeitos por um sistema computacional são:

- Estabelecer uma Política de segurança;
- Rotular os objetos relacionados com a Política;
- Identificar os usuários;
- Contabilizar os eventos relevantes à segurança;
- Usar mecanismos de hardware e software para o fornecimento de um nível

de confiabilidade, o qual garanta que os requisitos anteriores possam ser satisfeitos;

- Garantir a proteção do sistema contra alterações não autorizadas.

Com base nos seis requisitos de segurança anteriores, o TCSEC realiza a classificação dos sistemas computacionais, utilizando uma estrutura hierárquia de Divisões e Classes. As Divisões e Classes utilizadas na classificação podem ser observadas logo abaixo:

A.0.1 Divisão "D- Proteção Mínima

Esta divisão possui uma única classe (D1), a qual representa o menor nível de segurança. É atribuída a sistemas tidos como não confiáveis. Sistemas classificados como D1 não possuem nenhuma proteção para o Hardware, bem como não realizam a autenticação dos usuários que utilizam os recursos do sistema operacional.

A.0.2 Divisão "C- Proteção Arbitrária

Esta apresenta duas classes: C1, C2. Sistemas nesta divisão permitem ao usuário especificar quais usuários podem acessar os seus dados e definir a capacidade de contabilizar os eventos relacionados com a segurança.

Classe "C1- Segurança Arbitrária

Sistemas nesta classe implementam a separação de usuário e dados e ainda a identificação e autenticação dos usuários. O mecanismo de controle de acesso - Discretionary Access Control (DAC) - deve fornecer aos proprietários dos arquivos um controle, de forma individual ou coletiva, de quem poderá acessá-los.

As principais características desta classe são: Uso opcional da proteção de arquivo através de mecanismos como Listas de Controle de Acesso (ACL); proteção por senha e/ou banco de dados de autorização seguro; checagem da integridade do TCB; documentação da segurança para usuários, para administradores do sistema e para testes de segurança.

As primeiras versões do Unix eram avaliadas como sendo desta classe.

Classe "C2": Acesso Controlado

Esta classe apresenta um controle de acesso mais refinado do que a classe anterior (C1). Esta prevê a contabilização das ações da cada usuário de forma independente, através do processo de login de auditoria de eventos e isolamento de recursos.

Dentre as características de um sistema Classe "C2", pode-se destacar:

- Acesso restrito a usuários autorizados;
- Processo obrigatório de identificação de usuário;
- Proteção do re-uso de objetos;
- Auditoria de eventos;
- Operação em modo protegido;
- Melhor processo de documentação sobre análise de auditoria.

Algumas versões, atuais, do Unix e o VMS são classificados como "C2".

A.0.3 Divisão B - Proteção Obrigatória

Se caracteriza pelo uso de rótulos (label) para a classificação das informações, segundo o seu grau de sensibilidade. Devido ao uso extensivo dos rótulos, um dos requisitos desta divisão é manter a integridade dos mesmos.

Esta divisão é subdividida nas seguintes classes B1, B2 e B3.

Classe B1 - Segurança Rotulada

Esta possui os mesmos requisitos da C2 e ainda: suporte a rotulação dos dados; controle da integridade dos rótulos; controle de acesso aos objetos do sistema;

controle de acesso sobre recursos e usuários,; disponibilidade de mecanismos de especificação do nível de segurança das informações que estão na fila de impressão.

Esta classe prevê que para cada elemento do sistema deve haver um rótulo associado e para cada usuário um nível de "acesso". Baseado no seu nível de acesso, o usuário pode ou não obter direito de uso de um elemento do sistema.

Classe B2 - Proteção Estruturada

Esta classe requer um sistema de segurança baseado em um modelo de política de segurança formal, no qual todos os tipos de controles de acesso devem ser implementados e aplicados a todos os usuários e recursos disponíveis; um sistema de compartilhamento seguro, no qual há a separação dos dados em unidades classificadas (rotuladas), devendo permitir a comunicação segura entre o sistema e o usuário, prover um processo de notificação dos níveis de segurança, um processo de análise e auditoria de atualizações e de correções das versões dos componentes do sistema. Em adição a estes controles, todos aqueles pertencente a classe B1 também devem ser implementados

Esta classe possu tambémi um conjunto de testes de segurança melhorados. Os sistemas categorizados como B2 são tidos como "*Relativamente resistentes à penetração*".

Classe B3 - Domínios de Segurança

Como nos casos anteriores, esta possui as mesmas característica da classe B2 e ainda: implementação de um monitor de referência para controle de todos os acessos dos usuários aos recursos disponibilizados pelos sistema; procedimentos voltados para a recuperação do sistema e documentação do processo de recuperação do sistema; sinalização de eventos importantes que ocorrem no sistema; mecanismos de análise de segurança automática e análise de auditoria de eventos seguros.

Os sistemas classificados como B3 são tidos como "*Altamente resistentes a penetrações*"

Divisão A - Proteção Verificada

Esta se caracteriza da seguinte forma: utilização de métodos formais para a verificação da segurança. Estes visam garantir que o controle de acesso realmente pode proteger toda e qualquer informação sensível existente no sistema; além do uso de documentação formal, que demonstra que os requisitos de segurança são atendidos em todas as fases do projeto, desenvolvimento e implementação do sistema. Esta divisão possui somente uma classe.

A Classe A1 apresenta as mesmas características da classe B3, porém faz uso de técnicas formais de especificação e verificação usadas na fase de projeto, assim como o elevado grau de garantia com o qual o sistema de segurança foi implementado corretamente. Esta classe faz uso de métodos formais para a especificação da TCB.

Apêndice B

Controles da NBR ISO/IEC-17799

Esta apêndice se dedica a descrever de uma maneira breve os 10 macros controles e vários subcontroles de primeiro nível da norma de segurança NBR ISO/IEC 17799. Os 10 macros controles existentes nesta norma são:

1. Política de Segurança ;
2. Segurança Organizacional;
3. Classificação e Controle dos Ativos da Informação;
4. Segurança em Pessoas;
5. Segurança Física e do ambiente;
6. Gerenciamento de Operações e Comunicações;
7. Controle de Acesso;
8. Desenvolvimento da Segurança de Sistemas;
9. Gestão da Continuidade do Negócio;
10. Conformidade.

B.1 Política de segurança

Este é formado principalmente pela Política de Segurança da Informação, que visa definir a linha mestra na gestão de risco e segurança da informação, definindo: padrões a serem seguidos e ações a serem tomadas; descreve os vários processos envolvidos com o trabalho de segurança e a responsabilidades sobre os mesmos. O documento da política de segurança deve ser claro e de fácil entendimento.

É fundamental que a alta administração mostre o seu apoio a este documento, bem como, haja um processo de treinamento e de divulgação sobre o seu conteúdo.

Todos os funcionários devem estar cientes da sua implementação, das implicações em suas atividades e também demonstrar a aceitação quanto ao uso e cumprimento da política. Esta demonstração pode ser feita oficialmente através da assinatura de um documento, no qual o usuário afirma que é de seu conhecimento o conteúdo de toda a política de segurança e que se compromete em respeitá-la e cumprí-la.

Toda política deve possuir um gestor, o qual deve ser responsável pela verificação do seu nível de conformidade com o ambiente. É fundamental que o responsável pela política de segurança promova análises periódicas da mesma.

É indicado que conste na política de segurança, referências aos documentos que serviram de base para a sua construção, como por exemplo: a própria NBR ISO/IEC 17799, a Constituição Brasileira, o Estatudo da Organização e etc.

B.2 Segurança Organizacional

Este macro controle é divido nos seguintes controles:

1. Infra-estrutura de Segurança da Informação;
2. Segurança de Acesso a Terceiros;
3. Terceirização.

Cada um destes controles será abordado a seguir:

Infra estrutura de segurança da informação

Visa a criação de uma estrutura de gerenciamento da segurança da informação, da definição de responsabilidades, da definição de fontes externas de informações e ajuda, as quais serão utilizadas no processo de implantação da segurança da informação.

É de vital importância que as responsabilidades sejam claramente definidas e que a política de segurança seja um guia sobre a aplicação de regras e responsabilidades.

Sempre que necessário, esta política deve ser complementada com orientações específicas de acordo com o ambiente (Hardware, Aplicativos, Sistema Operacional e etc...)

Com o objetivo de melhorar o processo de gerência, além do gestor geral da política de segurança, devem existir gestores específicos e responsáveis pela sua gerência e segurança de cada ativo.

Como novos recursos trazem consigo novas vulnerabilidades, a inclusão de novos recursos ao ambiente e o uso de recursos pessoais de processamento devem ser autorizados pelo gestor responsável pelo sistema de segurança.

Por causa da complexidade no processo de gerenciamento, deve haver um consultor, interno ou externo, que possa ser invocado sempre que necessário, principalmente após suspeitas de incidentes ou violações na segurança.

Segurança de acesso a terceiros

O Acesso a terceiros ao ambiente deve ser controlado. Para que haja um controle, uma análise prévia de risco deve ser realizada. Esta análise definirá os controles necessários para garantir a segurança do ambiente. Tais controles devem levar em conta tanto os acessos lógicos quanto físicos, bem como o valor da informação que poderá ser acessada pelos usuários.

Antes que um terceiro tenha acesso ao ambiente, ele deve assinar um documento contendo a discriminação de todas as normas e restrições as quais o uso do ambiente está sujeito.

Quando houver a prestação de serviços por terceiros, é indicado a existência de uma cláusula contratual relacionada com o pagamento de indenizações, quando da violação da política de segurança.

Terceirização

Este controle visa ajudar a estabelecer as responsabilidades e os riscos quando parte ou todo o processo da informação é terceirizado.

Inicialmente, este processo requer que ocorra a definição e a assinatura de um contrato que irá reger a interação de ambas as partes (organização e terceirizados). Este contrato, por sua vez, deve permitir que os requisitos e procedimentos sejam expandidos.

B.3 Classificação e Controle dos ativos da informação

Este terceiro macro controle visa realizar uma identificação e associação de um *"valor"*, ou nível de *"importância"*, a um determinado ativo da organização.

Este macro controle é composto dos seguinte controles de segundo nível: Contabilização dos Ativos e Classificação da Informação.

Contabilização dos ativos

Este procedimento tem por objetivo identificar todos os ativos da informação existentes na organização e em seguida atribuir para cada ativo um *"proprietário"*, que será o responsável pelo seu gerenciamento e manutenção.

Este processo ajuda a evidenciar a importância ou *"valor"* de um determinado ativo para a organização. Associar um nível de segurança ao ativo, facilita a determinação dos controles de segurança a serem aplicados, uma vez que aplicação de controles deve ser proporcional ao valor do que se deseja proteger.

Uns dos controles que podem ser aplicados após a definição do *"valor"* de um ativo é a especificação da sua localização dentro da organização, bem como o seu nível de exposição.

Classificação da informação

A classificação da informação visa definir o seu nível de importância. Normalmente, este nível não é estático podendo, desta forma, variar com o passar do tempo e com o modelo do ambiente.

Para a classificação das informações, é necessário definir um conjunto de regras e níveis de classificação. Estes devem levar em consideração o nível de compartilhamento/restrição das informações e o impacto desta classificação nas atividades do ambiente.

Deste modo, o rótulo usado na classificação irá definir o valor e a sensibilidade de um ativo no contexto atual da organização.

Dentre os vários fatores relacionados a este processo, um dos mais críticos é a definição das categorias de classificação a serem utilizadas pela organização; bem como a interpretação dos rótulos de classificação, provenientes de outras organizações.

A revisão da classificação e a manutenção de um ativo é de inteira responsabilidade do seu proprietário ou do rotulador.

Devido à possibilidade das informações estarem em dois estados distintos, ou seja, Físico ou Lógico, devemos levar em consideração a existência destes dois tipo de rótulos.

Os rótulos físicos estarão, de certa forma, fixados nos próprios ativos. Já os rótulos lógicos não estarão diretamente aplicados aos ativos, gerando alguns problemas, os quais deverão ser tratados de forma conveniente. Os problemas gerados pela rotulação lógica estão relacionados com o armazenamento, a recuperação e a manutenção destes rótulos de forma segura.

B.4 Segurança em pessoas

Este macro controle visa prover os recursos necessários para o gerenciamento dos fatores de segurança que envolvem as atividades humanas na organização bem como o processo de treinamento e conscientização quando do uso correto da política de segurança. Este macro controle é subdividido em : Segurança na Definição e nos Recursos de Trabalho; Treinamento dos Usuários; Respondendo a Incidentes de Segurança e Mal Funcionamento.

Segurança na definição e nos recursos de trabalho

Este visa reduzir os riscos de segurança provenientes das atividades humanas em uma determinada organização, como por exemplo: os problemas de roubo e venda de informações confidenciais.

Dentre os procedimentos indicados para este fim pode-se destacar:

- Todo funcionário deve estar ciente da política de segurança e deve ter em seu contrato de trabalho cláusulas relacionadas à política de segurança, as quais ele deve se comprometer, seguir e respeitar. Destas cláusulas, uma pode ser o acordo de sigilo que impede que o funcionário faça menção a qualquer tipo de informação, tida como crítica para organização, a pessoas não autorizadas;

- Quando da contratação de recursos humanos, via uma agência de empregos, deve haver um contrato entre a agência e a organização, especificando como deve ser feita a seleção, qual é a responsabilidade da agência sobre o candidato selecionado e como proceder caso não haja a obtenção de um candidato adequado às exigências.

- Para que alguém tenha acesso a áreas tidas como restritas, faz-se necessário que o indivíduo assine um acordo de sigilo ou confidencialidade que o impedirá legalmente de revelar para pessoas não autorizadas as informações as quais ele teve acesso.

Treinamento dos usuários

Este item tem por finalidade fornecer a todos os usuários, das informações provenientes de uma organização, uma visão completa das ameaças e das preocupações que a organização possui sobre a segurança da informação.

O treinamento deve capacitar os usuários a apoiarem o bom funcionamento da política de segurança.

É indicado que todos os usuários antes de terem acesso ao ambiente passem por um treinamento completo

Respondendo a incidentes de segurança e mal funcionamento.

Este controle, além de procurar reduzir os danos causados por falhas no ambiente, visa permitir que se aprenda com as falhas já ocorridas.

Para tal, existe a necessidade da conscientização dos usuários quanto ao processo de notificação de incidentes, mesmo quando estes incidentes não passarem de simples suspeitas.

A conscientização irá ajudar para que o processo de notificação de incidentes seja feito através dos meios e às pessoas corretas, o mais rápido possível.

O quanto antes um incidente for notificado, melhor ele poderá ser tratado. Para apoiar o processo de notificação, deve haver um sistema formal de notificação de incidentes, bem como de resposta aos incidentes.

Muitas vezes o processo de notificação pode dar ao usuário a falsa impressão de que ele está habilitado a tratar o incidente por ele notificado. Por este motivo deve-se informar a todos os usuários que eles não estão autorizados a tratar qualquer tipo de incidente.

Caso um usuário tente tratar algum incidente de segurança, esta ação pode ser interpretada como uso impróprio do sistema, conseqüentemente, sendo cabível a aplicação de *"penalidades"* préviamente definidas na política de segurança.

No registro dos incidentes, informações como tempo de paralisação das atividades, custo, danos e o processo adotado para a sua correção, são de vital importân-

cia, pois estas informações devem ser utilizadas no processo de melhoria da política de segurança, bem como no processo de resposta a incidentes.

Normalmente, o processo de registro de incidentes dá origem ao que conhecemos como "kwonledgebase"de incidentes.

B.5 Segurança física e do ambiente

Este macro controle apresenta três subcontroles: Áreas de Segurança; Equipamentos de Segurança e Controles Gerais; todos relacionados a aspectos físicos da segurança da informação.

Áreas de segurança

A finalidade deste controle é inibir o acesso não autorizado às áreas onde se encontram informações vitais para organização. A proteção aplicada a uma determinada área deve ser proporcional aos riscos identificados pela análise de riscos.

Os perímetros de segurança devem ser indicados de forma bem clara.

Visitantes e terceiros só podem ganhar acesso às áreas específicas da organização, em especial aquelas que estão dentro de perímetros de segurança, quando acompanhados de um funcionário responsável e com a devida autorização.

É indicado a utilização de mecanismos de autenticação, que gerenciem a entrada e saída de pessoas e material das áreas de segurança.

O direito de acesso aos perímetros de segurança somente deve ser concedido quando for estritamente necessário, assim como deve passar por um período de verificação e validação. A verificação dos direitos de acesso visa garantir que autorizações não mais necessárias possam ser removidas.

Sempre que possível, deve ser feito o uso de identificadores visuais, como por exemplo os crachás, que permitem facilmente visualizar a identidade do usuário bem como o seu nível de acesso ao ambiente.

Os identificadores devem ser mantidos de forma que possam ser visualizados

facilmente, assim os funcionários e usuários autorizados podem ser instruídos a comunicarem a existência de pessoas estranhas no ambiente.

Salas que são tidas como vazias ou que não se encontram em uso, devem ser constantemente monitoradas, para evitar que as mesmas tenham um uso indevido. As áreas de processamento, gerenciadas por terceiros, devem ser isoladas fisicamente das gerenciadas pela organização.

Sempre que possível deve ser inibido o uso de câmeras fotográficas e gravadores de aúdio/vídeo, trazidas por funcionários ou visitantes, impedindo assim que detalhes sobre o ambiente sejam facilmente obtidos.

As áreas de expedição e recebimento de material jamais devem possuir comunicação direta com as áreas de segurança, por motivos mais que óbvios.

Equipamentos de segurança

Este controle visa prover dos mecanismos necessários para evitar a paralisação das atividades da organização ocasionada por danos em seus equipamentos. Para tal, devem ser adotados procedimentos que busquem a proteção física contra ameaças e perigos existente no próprio ambiente.

Um problema freqüente e tratado por este controle é o acesso não autorizado aos dispositivos de processamentos. Para inibir este tipo de problema, devem ser empregados mecanismos físicos que evitem o acesso desnecessário à área de processamento das informações.

Outro fator de risco encontrado em áreas de processamento é o consumo de bebidas, alimentos e fumo. Tal padrão de comportamento tende a gerar vários tipos de danos aos equipamentos, portanto, o consumo de alimentos, bebidas e fumo deve ser tratado pela política de segurança da organização a fim de inibir a sua ocorrência nas áreas de processamento.

Problemas gerados pela falta ou alteração da tensão da corrente elétrica também devem ser levados em consideração. Quando houver o uso de mecanismos que visem evitar ou minimizar este problema, os mesmos devem ser periódicamente testados e submetidos a manutenções preventivas, visando com isto garantir o seu perfeito

funcionamento.

Mesmo implementando-se um sistema contra problemas de tensão elétrica, deve haver um plano de contingência previamente documentado e testado para ser acionado quando da falha dos mecanismos de proteção. Este tipo de plano deve abranger muito mais que os fatores elétricos.

Os cabeamento da rede elétrica e da rede de dados devem ser protegidos contra danos e interceptações. Sempre que possível, deve se utilizar passagens subterrâneas, evitando, desta maneira, passá-los de forma aérea e por vias públicas.

Dutos compartilhados onde rede de dados e rede elétrica convivem juntos devem ser substituídos por dutos individuais, não sendo indicado a combinação destes dois cabeamentos no mesmo duto de transporte.

Devido a questão de segurança e gerenciamento, somente pessoas previamente autorizadas podem realizar manutenções e instalações nos equipamentos de processamento da informação, pelo mesmo motivo deve existir um registro completo de todos os processos de manutenção.

Equipamentos só podem ser retirados da organização ou utilizados fora das instalações com uma autorização explicita fornecida pela administração da organização, uma vez que informações importantes podem ser expostas por descuido dos usuários.

Controles gerais

Estes objetivam a redução da possibilidade de divulgação e roubo de informações que pertencem à organização.

Um exemplo de medida a ser adotada é a política da *"tela limpa e mesa limpa"*. Mesmo sendo simples, esta política evita que usuários não autorizados tenham acesso às informações que estão sendo manipuladas em um determinado momento no ambiente.

Os terminais de trabalho devem ser mantidos desconectados quando não estiverem em uso ou protegidos por algum sistema de senha ou tranca física. Terminais de trabalho que forem abandonados por um determinado período de tempo devem realizar a desconexão de automática do usuário conectado.

B.6 Gerenciamento de operações e comunicações

Este visa prover os recursos necessário para facilitar o processo de gerenciamento dos mecanismos de troca de informações dentro e fora da organização e é composto dos seguintes subcontroles:

1. Procedimentos e Responsabilidades Operacionais;
2. Planejamento e Aceitação dos Sistemas;
3. Proteção contra softwares maliciosos;
4. Housekeeping;
5. Gerência de Rede;
6. Segurança e Manuseio de Mídias;
7. Troca de Informações e Softwares.

Procedimentos e responsabilidades operacionais

Visando gerir as operações de processamento das informações e inibir a ocorrência de erros, existe a necessidade de que os procedimentos operacionais a serem executados estejam sempre claramente especificados e documentados, assim como a definição das responsabilidades pela execução e gerência dos procedimentos.

Na especificação da documentação, deve-se levar em conta a complexidade do processo que está sendo o foco da atenção, pois alguns necessitarão ser especificados com um maior nível de detalhe, como é caso dos procedimentos de backup de dados e de contingência da rede elétrica.

Toda modificação no ambiente de processamento da informação, assim como a definição das responsabilidade pelas modificações devem ser formalizados. Procedimentos para o retorno ao estado original do sistema também devem ser preparados, testados e devidamente documentados.

Deve ser mantida uma "trilha de auditoria", para que seja possível a realização de análises de violações de segurança, assim como a definição da causa desta violação.

Na existência de um ambiente de desenvolvimento e teste, este deve ser isolado do ambiente de produção, pois o não isolamento permitiria o uso indevido ou até mesmo a fraude das informações vitais da organização.

Planejamento e aceitação dos sistemas

Para reduzir a possibilidade de falhas do ambiente computacional, um processo de gerenciamento de recursos e de cargas das máquinas deve ser aplicado.

A demanda ou carga dos sistemas deve ser constantemente monitorada, a fim de facilitar o processo de projeção de cargas futuras. O aumento da carga de um sistema pode ser gerado pelo uso de um novo recurso ou pela atualização de algum já existente.

A análise de carga permite prever possíveis gargalos e pontos que podem comprometer a segurança dos sistemas e das informações que são manipuladas.

Quando ocorrer a necessidade da inclusão de novos recursos ou a atualização de algum já existente, um estudo do seu impacto sobre a carga do sistema deve ser executado, e recursos só poderão entrar em operação quando for verificado a satisfação de critérios previamente estabelecidos.

Proteção contra softwares maliciosos

Este controle busca impedir ou inibir a ação de softwares que possam comprometer a integridade das informações, como é o caso dos vírus, worms e cavalos de Tróia.

Almejando o sucesso deste controle, é de fundamental a realização do trabalho de conscientização dos usuários quanto ao risco do uso de softwares não autorizados, visto que os mesmos podem conter algum tipo de código malicioso, que ao se proliferar pelo ambiente poderia ocasionar perda ou dano das informações.

É função dos gestores a utilização de todos os mecanismos possíveis e cabíveis para o controle da ação de softwares maliciosos. Estes controles poderiam ser aplicados desde as estações de trabalho até os servidores de rede. Normalmente são os

servidores a porta de entrada dos softwares maliciosos.

Housekeeping

Este controle visa garantir a integridade e a disponibilidade dos serviços necessários para o processamento da informação. Um dos mecanismos adotados para esta finalidade é a realização de cópias periódicas dos dados e sistemas essenciais para o funcionamento do ambiente. As cópias, por medida de segurança, devem ser testadas periódicamente e guardadas por um período de tempo previamente determinado. Normalmente, há o armazenamento cíclico de três versões diferentes das cópias de segurança.

Além do processo de realização de cópias de segurança, há a necessidade de procedimentos de recuperação do ambiente após algum tipo de *"catástrofe"*. Estes procedimentos devem ser periodicamente testados para verificar se os mesmos permitem a recuperação do ambiente dentro de um tempo aceitável para a continuidade das atividades do ambiente, sem gerar grandes "perdas".

A existência de um sistema de registro das falhas, bem como o registro do processo de resolução é altamente indicado, mas em paralelo com este, um segundo registro deve gerenciar o armazenamento de todas as atividades realizadas no sistema.

Gerenciamento de rede

Este controle tem seu foco sobre a segurança das informações que trafegam pela rede de computadores, seja ela interna ou externa à organização.

A *"segurança da rede"* não se aplica somente às informações que trafegam por ela, todavia, há a necessidade de procedimentos que garantam a disponibilidade dos serviços prestados pela rede, como o caso dos serviços de e-mail, de http e https.

Segurança e manuseio das mídias

Este se preocupa com o processo de manipulação das informações armazenadas em qualquer tipo de mídia, seja ela eletrônica ou não. Independentemente do seu tipo, o acesso e manuseio das mídias deve ser controlado de tal modo que todo e qualquer usuário, antes de retirar qualquer mídia removível da organização, receba uma autorização explícita para tal. O registro de retirada e devolução de mídias é utilizado quando uma auditoria se faz necessária.

Toda e qualquer mídia que não for mais utilizada deve ser descartada de forma segura. O descarte deve ser feito, preferencialmente, por meio da destruição completa da mídia.

Independente da forma como a informação se apresenta, devem existir procedimentos que definam como deve ser feito o seu manuseio e armazenamento. Este procedimento deve sempre levar em consideração a classificação da informação e o meio de armazenamento.

Para a documentação de sistemas é indicado um tratamento de forma especial, visto que a mesma contém dados altamente sigilosos, por conseguinte é aconselhável que haja um controle de todas as pessoas que tenham acesso à esta informação. O número de pessoas que têm acesso a documentação deve ser sempre o menor possível. Quando a documentação é mantida para acesso público, ela necessita ser protegida de forma apropriada.

Troca de informações e softwares

Este controle tem por finalidade proteger informações trocadas entre organizações. Uma das formas de realizar tal proteção é possuir um controle rígido sobre o tipo de informação que está sendo trocada, bem como manter esta troca baseada em contratos legítimos, firmados entre as organizações.

Devido ao uso cada vez maior, as suas características e vulnerabilidades, o Comércio Eletrônico também devem ser alvo de atenção. As atividades de Comércio Eletrônico deve sempre prover dos mecanismos necessários para assegurar a confidencialidade e a integridade das informações trocadas. Várias técnicas de criptografia

podem ser empregadas como uma forma de auxiliar tal processo.

Como os servidores de comércio eletrônico são constantemente alvos de vários tipos de ataques, toda e qualquer comunicação dos mesmos com o restante da rede da organização deve ser avaliada, buscando averiguar o impacto na segurança.

Ainda associado à questão da troca de informações, há o serviço de e-mail, que assim como o comércio eletrônico, é alvo de várias vulnerabilidades, das quais podemos destacar: a possibilidade de acesso não autorizado ao conteúdo da mensagem; a modificação de seu conteúdo; dificuldade em comprovar e provar o envio e o recebimento de mensagens, assim como as questões relacionadas ao acesso remoto por parte dos usuários às caixas de mensagens.

Nos dias de hoje, é comum que as organizações possuam sistemas que estão disponíveis para o acesso ao público. Estes sistemas, normalmente acessíveis pela WEB, devem ser estruturados de tal forma que impeçam a alteração e a divulgação indevida de informações críticas. Deste modo, toda e qualquer informação, antes de ser publicada deve ser verificada e sua publicação autorizada. Estes aplicativos devem ser implementados de forma a impedir que a estrutura privada da organização fique exposta aos acessos feitos pelo sistema público.

Informações críticas, quando expostas pelos sistemas públicos, requerem vários mecanismos de proteção, como por exemplo, o uso de canais de comunicação seguros e o processo de autenticação de quem o acessa.

Logo, devem ser criadas políticas claras e eficientes, quanto à utilização de e-mail, do comércio eletrônico e da distribuição de informações por meio de sistemas públicos.

B.7 Controle de acesso

Este macro controle pode ser decomposto nos seguintes controles de segundo nível:

1. Requisitos do negócio para controle de acesso;
2. Gerência de acesso dos usuários;

3. Responsabilidade dos usuários;
4. Controle de Acesso à rede;
5. Controle de Acesso ao Sistema Operacional;
6. Controle de Acesso às aplicações;
7. Notificação do uso e acesso ao sistema;
8. Computação móvel e trabalho remoto;

Um dos objetivos destes controles é inibir os problemas de segurança gerados pelo acesso lógico, não autorizado, de usuários às informações da organização.

Requisitos do negócio para controle de acesso

Para controlar o acesso às informações, políticas específicas devem ser desenvolvidas e implementadas, que devem prover os meios para gerenciar o processo de autorização e de distribuição das informações. Estas políticas baseiam-se em uma documentação formal, contendo os direitos de acesso de cada usuário e de cada grupo de usuário. No controle de acesso, deve ser levado em consideração não somente as informações mas também as aplicações, além dos sistemas e os serviços que irão manipular estas informações.

Visando facilitar o processo de gerência do controle de acesso, pode ser criado e utilizado um sistema de *"perfil de liberação"*, que poderá ser atribuído a vários usuários de acordo com as suas atividades.

Independentemente do tipo e do mecanismo de acesso, deve haver um processo formal de gerenciamento dos direitos que os usuários possuem quando acessam o ambiente.

É expressamente indicado pela NBR ISO/IEC o uso de regras baseadas na premissa *"Tudo deve ser proibido a menos que expressamente permitido"*, ao invés da premissa "*Tudo é permitido a menos que expressamente proibido"*.

A primeira por ser mais restritiva, garante um nível maior de segurança, porém tende a aumentar a complexidade do gerenciamento do ambiente.

Gerenciamento de acesso dos usuários

Este controle visa impedir que pessoas não autorizadas consigam acessar os sistemas de informação. Para tal, procedimentos de controle na concessão do direito de acesso, privilegiados ou não, devem ser implementados, bem como procedimentos para gerenciar o registro de inclusão e remoção de usuários e de direitos de um ou vários usuários.

Visando facilitar a gerência do acesso, cada usuário deve possuir um identificador único em toda organização. Os identificadores de grupo só deverão ser utilizados quando estritamente necessários. Para garantir a unicidade do identificador verificações periódicas devem ser executadas, que permitirão que ocorra a remoção de identificadores duplicados.

Um usuário só poderá ter acesso a um sistema após ter recebido uma autorização de acesso ao mesmo. Todos os usuários devem ter conhecimento de seus direitos de acesso, bem como de suas responsabilidades.

Deve existir um documento contendo a assinatura do usuário. Neste documento o usuário confirma possuir conhecimento pleno sobre os seus direitos de acesso e que irá respeitar o seu nível de ingresso ao ambiente.

Quanto ao processo de concessão de privilégios, este deve ser altamente controlado. Preferencialmente, deve haver um processo formal para o fornecimento e gerenciamento dos privilégios de um usuário. Privilégios jamais devem ser globais, ou seja, para cada sistema o usuário deve ter um nível de acesso próprio, o que corresponde a um tipo de privilégio no sistema. Os privilégios de acesso não devem ser fornecidos antes que ocorra um processo de autorização formal.

O gerenciamento das senhas dos usuários é mais um fator culminante nesta estrutura, pois somente usuários que possuam uma senha válida poderão ter acesso aos sistemas da organização. Por este motivo, as senhas são de caráter pessoal e jamais devem ser compartilhadas, a não ser as senhas de grupos de trabalho.

Quando houver o fornecimento de uma senha de caráter temporário, é função do usuário a sua alteração o quanto antes possível.

Em intervalos regulares de tempo deve ocorrer uma verificação dos direitos de

acesso, privilegiados ou não, pois funcionários que não estão mais desempenhando uma determinada função, talvez não necessitem do tipo de acesso ao qual estão atrelados.

Responsabilidades dos usuários

Este controle tem por finalidade conscientizar cada funcionário sobre as suas responsabilidades no que diz respeito ao processo de controle de acesso.

O primeiro ponto a ser trabalhado é a criação e o uso de senhas. O usuário é o único responsável pelas suas senhas e, por este motivo, deve estar ciente sobre a necessidade do uso de senhas fortes. Segundo a NBR, uma *"boa"* senha teria pelo menos seis caracteres e composta de caracteres alfanuméricos, maiúsculos e minúsculos(ela deve ser fácil de ser lembrada, porém difícil de ser descoberta).

Para garantir o sigilo da senha, ela deve ser trocada periodicamente, sendo evitado ao máximo a re-utilização de senhas antigas.

Como já foi dito, os usuários jamais devem *"emprestar"* as suas senhas e devem evitar de deixá-las expostas em locais de fácil localização, como em *"post-its"* colados debaixo do teclado, ou ao lado do monitor das estações de trabalho.

O uso de senhas fortes e a sua troca periódica não garantem que um usuário mal intencionado irá ganhar acesso ao sistema como um usuário válido. Este tipo de ação pode ocorrer quando os usuários esquecem de fechar as conexões que estavam sendo utilizadas ou abandonam os seus terminais de trabalho, deixando-os ligados e *"logados"*. Para evitar estes tipos de problemas, todas as conexões que não estão sendo utilizadas devem ser encerradas.

Um terminal que está ligado e conectado jamais deve ser abandonado sem que o mesmo possa ser protegido por algum mecanismo de "lock", como é o caso da tecla de bloqueio e a senha de acesso.

Controle de acesso à rede

Todo o acesso aos recursos da rede deve ser controlado. Este controle deve levar em consideração o processo de autenticação de usuários. Deve haver um controle formal do fornecimento de acesso às redes, no qual deve ser especificado quais serviços podem ser acessados por quais os usuários.

A interconexão entre redes deve possuir o maior nível de preocupação quando ocorrer a conexão da rede interna da organização com a que provê acesso público ou quando usuários necessitam acessar os recursos computacionais a partir de redes inseguras.

Sempre que possível, deve ser adotado o uso de rotas forçadas e de verificação em vários pontos desta rota, assim como uso de *"firewall"* e o isolamento do ambiente em sub-redes lógicas.

Quando o ambiente estiver sendo acessado por um usuário que está fora dos domínios da rede, este deve sofrer um processo de autenticação mais forte, como por exemplo o uso da criptografia. As informações, quando acessadas de fora da rede, não devem trafegar em texto plano. Preferencialmente, devem ser criptografadas antes que seja realizada a sua transferência. Sempre que possível, a autenticação do *"nó"*, que está se conectando com a rede, deve ser realizada. Este processo permite a autenticação de um grupo de usuários remotos.

Muitos equipamentos de rede possuem portas de diagnóstico que devem ser protegidas contra acessos não autorizados.

Devido ao cenário econômico atual, várias organizações constantemente fecham processos de parcerias. Há, em muitos casos, a necessidade de compartilhamento de recursos e informações por parte destas organizações, o que gera a necessidade de segregar a rede das organizações em vários grupos, dos quais podemos destacar: grupo de serviços de informação, grupos de usuários e de sistemas de informação.

A segregação da rede permite a criação de domínios lógicos os quais podem ser vistos como perímetros de seguranças. Este processo pode ser implementado com o uso de *"gateway"* de borda, o qual deve ser configurado para realizar um filtro dos pacotes que entram e saem dos perímetros. Para este tipo de função geralmente é utilizado um *"firewall"*.

Toda a conexão da rede externa deve ser filtrada e possuir um controle que limite o consumo de banda e o nível de acesso ao ambiente. As regras que definem este controle devem ser baseadas na política de controle de acesso.

Controle de acesso ao sistema operacional

Este controle visa permitir que somente os usuários autorizados tenham acesso aos recursos computacionais. Normalmente, este controle de acesso baseia-se no uso de senhas, mas em alguns casos é indicado a identificação automática de terminais. Este tipo de identificação é utilizada quando existem serviços que somente podem ser executados em determinados terminais.

O processo de *"logon"* deve evitar ao máximo o acesso não autorizado ao sistema, assim como deve fornecer o mínimo de informações a quem está tentando entrar no sistema. Toda vez que um usuário se conectar com sucesso, ele deve ser informado da ocorrência da tentativa fracassada de acesso a sua conta e quando foi o último acesso bem sucedido.

Para que haja um controle eficaz, cada usuário deve ter um identificador que seja único no sistema. Este identificador será utilizado para registrar todas as atividades do usuário, registro este que pode ser utilizado em um processo de auditoria.

No caso do uso de um identificador, que deve ser claramente documentado, para um grupo de usuários, bem como sua listagem, deve ser constantemente atualizada e conhecida pelos gestores.

É possível o uso de outras técnicas de controle, além da senha de acesso, como da autenticação via *"smart card"* ou até mesmo autenticação biometria. Mesmo com técnicas mais modernas, o sistema de senha ainda é o mais utilizado.

Um sistema operacional deve fornecer um conjunto mínimo de facilidades para o gerenciamento das senhas, bem como a garantia de que as senhas usadas são de boa qualidade.

Para garantir a segurança no gerenciamento do sistema, todo e qualquer acesso aos utilitários devem ser monitorados e restritos a um grupo específico de usuários, assim como utilitários desnecessários devem ser removidos do ambiente.

Devido ao fato de ser comum que usuários deixem seus terminais de trabalhos abandonados, existe a necessidade de que as conexões possuam um tempo máximo de inatividade. Findo este período, a conexão será suspensa. Assim como deve ser implementada uma política de tela limpa.

Controle de acesso as aplicações

Este controle visa evitar o acesso às informações armazenadas na forma dos sistemas de informação. Para isto, ele prevê a necessidade de mecanismos nos quais o acesso às informações e sistemas seja baseado em uma política de acesso focada no negócio da organização.

O acesso só deve ser permitido ao *"dono"* da informação e a outros usuários previamente conhecidos e autorizados.

No caso dos sistemas de aplicação, o ingresso dos usuários deve seguir uma política de controle de acesso préviamente definida. É importante que seja possível controlar quais funções estarão disponíveis ao usuário de acordo com os seus direitos de acesso.

Sistemas de aplicações sensíveis devem ser executados a partir de um computador dedicado e que compartilhe informações e serviços somente com outros sistemas tidos como confiáveis.

Monitoração do uso e acesso ao sistema

Este controle visa monitorar sistemas em busca de evidências que indiquem alguma violação das políticas acordadas. Este monitoramento é feito normalmente através de um sistema de *"log"* que registra as principais atividades que acorrem em um ambiente.

Este processo é fundamental para a análise de risco, uma vez que permite verificar se os usuários estão realizando as funções para as quais eles foram autorizados ou não.

Os resultados da monitoração devem ser constantemente verificados para que

medidas corretivas ou preventivas sejam executadas. Não é indicado que o verificador dos *"logs"* seja uma das pessoas que tenham suas atividades monitoradas.

Assim como outros recursos computacionais, o sistema de monitoramento também está sujeito a ameaças, das quais podemos destacar a adulteração de registros e a desativação das atividades. Por este motivo, procedimentos específicos devem ser executados para salvaguardar o sistema.

Um processo de sincronização dos relógios dos computadores deve ser levado em conta, possibilitando assim uma maior precisão e confiança no registro das atividades e uma melhor análise dos vários logs do ambiente.

Computação móvel e trabalho remoto

Este controle visa garantir a segurança quando da existência de mecanismos de computação móvel e trabalho remoto.

Para que este requisito seja alcançado, deve haver o estabelecimento de uma política formal que leve em consideração os riscos de se trabalhar em um ambiente desprotegido.

Devido às suas características, este controle normalmente faz uso de algum nível de criptografia no ambiente móvel ou remoto. Esta camada visa evitar que as informações sejam acessadas ou roubadas. Outro fator a ser considerado, além do uso da criptografia, é a questão do controle de acesso.

Como normalmente o trabalho remoto requer acesso externo à infra estrutura da organização, este acesso pode tornar o ambiente vulnerável. Todo trabalho remoto deve ser previamente autorizado e controlado pelo gestor responsável, assim como mecanismos que garantam a proteção dos dados que trafegam entre o ambiente interno da organização e o local de trabalho a ser executado.

B.8 Desenvolvimento de segurança de sistemas

Este controle fornece os critérios necessários para o desenvolvimento de aplicativos mais consistentes com a norma de segurança da organização e pode ser subdi-

vidido nos subcontroles:

1. Requisitos de segurança de sistemas;
2. Segurança nos sistemas de Aplicação;
3. Controles de criptografia;
4. Segurança do sistema de Arquivo;
5. Segurança dos processos de Desenvolvimento e Suporte.

Requisitos de segurança de sistemas

A *"segurança"* deve ser parte integrante de todo sistema computacional que manipule as informações da organização. A preocupação com a questão da segurança deve surgir na fase de modelagem do sistema.

Um sistema que é projetado para ser seguro possui várias vantagens sobre aqueles que devem ser adaptados ou modificados para atenderem a esta necessidade. Todo controle inserido na fase de projeto é evidentemente mais barato e eficiente do que aqueles aplicados quando o sistema já está em desenvolvimento ou completamente desenvolvido.

Segurança nos sistemas de aplicação

Este controle visa prevenir que os sistemas de aplicação comprometam a integridade e a confidencialidade dos dados. Tais sistemas devem prover de mecanismos para o controle das atividades, que poderão ser utilizados em processos de auditoria e verificação do ambiente.

Devido à necessidade de manter a integridade das informações, toda e qualquer entrada de dados deve ser validada em busca de dados inconsistentes ou incompletos. Mesmo com a validação da entrada de dados, os processos internos de um sistema podem comprometer a consistência destes, levando a perda da integridade dos mesmos. Conseqüentemente, verificações periódicas devem ser executadas para validar a consistência dos dados que são manipulados pelo sistema. A entrada de dados e

a saída de informações providas por um sistema também devem ser validadas antes de serem repassada ao usuário.

Quanto aos programas, devem ser executados em uma ordem pré estabelecida, devem existir controles que garantam que os mesmos são executados na ordem correta, visto que a inversão desta ordem pode ocasionar erro no processamento. Quando uma execução fora de ordem for observada, o controle irá executar o cancelamento do processamento ou outra ação apropriada.

Controles de criptografia

Dependendo do nível de sensibilidade da informação ou criticidade da mesma, existe a necessidade da aplicação de controles que aumentem o nível de segurança dos dados. Um dos controles que podem ser utilizados é a criptografia, mas o seu uso deve ser feito baseado no resultado de uma análise de risco, uma vez que tais controles não devem comprometer, de forma alguma, o processamento da informação.

Quanto ao uso da criptografia, deve ser observado fatores técnicos como por exemplo: o tipo de algoritmo e o tamanho da chave a ser utilizada, além de outros não técnicos como é o caso das regulamentações e restrições nacionais do seu uso.

Em muitos casos, pode-se fazer uso de serviços de não repúdio que comprovem a ocorrência, ou não, de uma determinada atividade, como por exemplo o envio de mensagens eletrônicas

Segurança do sistema de arquivo

Este controle visa assegurar a integridade dos arquivos associados às aplicações existentes em um ambiente. Todo e qualquer acesso aos dados armazenados em um sistema de arquivo deve se controlado, assim como um *"log"* de alterações e atualizações deve ser fornecido.

Como o processo de atualização de softwares pode gerar falhas nos ambientes, toda versão anterior deve ser guardada como uma medida de contingência, assim como os "patches"de softwares devem ser aplicados quando os mesmos visarem um aumento no nível de segurança e/ou de facilidade de interação dos aplicativos com

os usuários.

Segurança dos Processos de Desenvolvimento e Suporte

Todo ambiente de desenvolvimento e suporte deve ser fortemente controlado devido ao elevado grau de conhecimento que os usuários possuem sobre o ambiente e principalmente pela possibilidade da criação de brechas de segurança.

Um controle rígido na implementação e ativação de mudanças deve ser realizado. Alterações nos sistemas podem levar a mudanças no plano de contingências. Conseqüentemente, antes que as modificações nas aplicações se tornem funcionais, um novo plano de contingência deve ser elaborado e testado.

B.9 Gestão da continuidade do negócio

Este controle se relaciona com os mecanismos necessários para o perfeito funcionamento do ambiente, mesmo quando parte deste apresenta falhas.

Aspectos da gestão da continuidade do negócio

Este controle visa garantir a continuidade da operação do ambiente mesmo em caso de falha parcial.

Alcançar o que é chamado de contingência total, ou seja, não haver paralisação do ambiente em nenhum momento de sua operação é quase impossível. Por outro lado, um trabalho amplo deve ser realizado para que o tempo de paralisação do ambiente, em caso de alguma falha, seja tido como aceitável para o negócio da organização.

A definição deste tempo *"aceitável"* é dependente do modelo do ambiente e deve ser determinado durante a análise de risco do mesmo.

O plano de contingência deve abranger toda a organização e levar em conta os riscos pelos quais o ambiente está exposto, assim como o impacto que uma paralisação pode causar no mesmo.

O plano de contingência deve gerar uma documentação que detalhe os objetivos

e prioridades deste controle dentro da organização. O nível de documentação e o tempo de indisponibilidade devem estar consistentes com o nível de exigência existente no ambiente.

Devido ao caráter mutante dos ambientes das organizações, os planos de continuidade devem ser constantemente testados e atualizados bem como a identificação dos riscos e dos impactos devem ser feitos mediante ao uso de uma ferramenta, como a análise de risco de segurança.

Um plano de continuidade pode ser dividido em cinco partes: a identificação dos procedimentos, a implementação dos procedimentos, a documentação do plano, o processo de treinamento, os testes de conformidade e as atualizações.

O plano de continuidade deve informar:

- Quando um plano de ação deve ser ativado;
- Quais os procedimentos devem ser executados após a ocorrência de um incidente;
- Quais os procedimentos devem ser tomados quando da reativação dos processos;
- E quando e como o plano será testado.

B.10 Conformidade

Este macro controle abrange os aspectos legais relacionados ao uso de determinados softwares; quando e como deve ser realizado as revisões da política de segurança e como deve ocorrer o processo de auditoria.

Conformidade com requisitos legais

Este controle visa inibir violações legais no uso e manuseio da informação, independentemente de seu tipo, dentro da organização

De acordo com sua localização, a organização está sujeita a estatutos e leis de âmbito municipal, estadual, federal e às vezes mundial, que em muitos casos variam de localidade para localidade. Por este e outros motivos é altamente indicado que a organização tenha, sempre que possível, uma consultoria para assuntos legais.

Procedimentos devem ser implantados nos processos de regulamentação de tal modo que toda e qualquer cláusula relevante seja claramente explicada, assim como procedimentos próprios devem ser implantados para garantir que questões referentes aos direitos autorais sejam respeitadas.

Devido ao sistema de distribuição de alguns softwares, existe a necessidade da implantação de uma política sobre os direitos de propriedade de cada software. Esta política deve possuir instruções que indiquem quais atitudes devem ser tomadas quando ela for violada por um determinado usuário.

Em muitos casos, existe a necessidade de manter um registro sobre determinados tipos de informações, como por exemplo as informações contábeis. Para cada tipo de informação, há um tempo de *"preservação"* diferente. Conhecer este período é fundamental para a escolha das mídias, nas quais as informações serão mantidas, bem como onde serão armazenadas.

Outro problema a ser observado é a questão do uso indevido dos recursos computacionais de uma organização. Este problema tem sido uma constante em muitas organizações. A forma como ele pode ser tratado varia de acordo com a localidade onde a organização está sediada. Uma questão bem clara é que para resolver o problema do uso indevido, um processo de monitoramento deve ser iniciado, mas em determinados países existem leis que restringem este tipo de controle. O melhor caso é quando o usuário está ciente e concorda com tal processo de monitoramento.

O uso de controles de criptografia está sendo largamente utilizado, porém, a sua utilização está normalmente vinculada a determinadas leis ou estatutos. Devido a isso, convém que uma consultoria em assuntos legais seja requerida antes do uso de qualquer controle deste tipo.

Devido principalmente ao crescente número de crimes digitais, é indicado que exista na organização uma política e procedimentos de coleta de evidências que devem ser implementados de tal forma a servir de apoio em processos legais que envolvem questões disciplinares internas ou quando envolver leis federais.

Revisão da política de segurança e da conformidade técnica

É importante que todos os gestores garantam que a política e as normas de segurança sejam seguidas. Mesmo assim, revisões periódicas devem ser executadas para garantir o nível de conformidade com as normas. A periodicidade destas revisões está diretamente relacionada com a sua criticidade para o ambiente.

Periodicamente, a verificação de conformidade deve ser executada de forma manual, e sempre que possível, apoiada em softwares que possam gerar relatórios. O teste de conformidade pode fazer uso de ferramentas que realizem verificações de possíveis vulnerabilidades, até mesmo testes de invasão podem ser realizados, porém este tipo de teste somente deve ser realizado por pessoas préviamente autorizadas, visto que o mesmo pode comprometer a integridade da segurança do sistema.

Considerações sobre a auditoria de sistemas

Os processos de auditoria devem ser planejados de tal forma que a sua execução não interfira no funcionamento normal do ambiente

O nível da análise deve ser acordado antes de sua realização, assim como os acessos diferentes de *"leitura"* não devem ser feitos nos arquivos originais, mas sim, em cópia dos mesmos, as quais devem ser removidas posteriormente ao teste.

O acesso às ferramentas de auditoria deve ser controlado para impedir que um usuário malicioso tenha acesso às informações sobre o sistema.

Apêndice C

Glossário

A.V.: Os Agentes de Verificação são os componentes em software responsáveis pelo monitoramento e gerencia dos agentes que realizam a coleta de dados de um ambiente;

ABNT: Este á abreviatura de Associação Brasileira de Normas Técnicas, que é o órgão responsável pela homologação e desenvolvimento de normas e técnicas utilizadas em todo o território nacional e outros países de língua portuguesa;

ACD: Os Agentes de Coleta de Dados, são os componentes responsáveis por enviar ao Agentes de Verificação, todos os dados coletados nas máquinas que estão conectadas à rede do ambiente que está sendo analisado;

ACL: As Access Control List, são utilizadas para a construção listas de restrição, que podem controlar o acesso a determinados recursos disponíveis em um ambiente;

API: este termo descreve as interfaces de programação forneceidas por uma determinada linguagem ou software;

ASV: Os Agentes Secundários de Verificação, são os componentes do modelo que fazem o gerenciamento da coleta de dados nas subredes de verificação, estes normalmente são instalados em máquinas conectadas à mais de uma rede.

Aqua BOOK: É um glossário de termos relacionados a segurança, que teve sua ultima atualização no dia 21 outubro de 1988;

BS7799: Esta é a norma britânica de segurança da informação e que posteriormente deu origem a ISO/IEC 17799;

BS7799-1: Esta é a primeira parte da BS7799, que teve sua primeira versão homologada no ano de 1995

BS7799-2: Esta é a segunda parte da BS7799, que teve sua primeira versão homologada no ano de 1999

BSI: Fundada em 1901, foi o primeiro órgão britânico a emitir normas à nível nacional. Ela é um dos membros fundadores da Organização Internacional de Normalização (International Organization for Standardization - ISO), a BSI facilitou e publicou as primeiras normas comerciais em sistemas de gestão da qualidade, sistemas de gestão ambiental, sistemas de gestão da saúde e segurança ocupacionais e gestão de projeto.

Basiléia II: Esta é a nova versão de um conjunto de *"princípios essenciais"* desenvolvidos pelo Comitê de Supervisão Bancária da Basiléia, este documento é uma referência básica para a supervisão bancária eficaz;

CC: O Common Criteria (CC) é uma iniciativa para padronizar os critérios de segurança do ITSEC, CTCPEC (Canadian criteria) e US Federal Criteria (FC). Usado na validação de produtos, de sistemas e para iniciar a padronização dos critérios de segurança dentro da ISO.

CCSC: O Britsh Commercial Computer Security Center, é o centro comercial britânico de pesquisa sobre segurança da informação, e foi um dos membros do desenvolvimento da BS7799;

CCSC : O Comercial Computer Security Centre, foi criado em 1987 pelo DTI (Department Of Trade Centre) com o objetivo de auxiliar as companhias britânicas que comercializavam produtos para Segurança de Tecnologia da Informação através da criação de critérios para avaliação da segurança.

Chave de Ativação : Esta chave é um valor aleatório, utilizado para confirmar que um agente de coleta de dados está sendo instalado na máquina correta;

Chave de Comunicação: Esta chave também é um valor aleatório, utilizada para autenticar a identidade do ACD e permitir a troca de informações entre ele e seu Agente de Verificação;

Cobit : CobiT é um guia para a gestão de TI orientado ao negócio da organização;

Criptografia : é um conjunto de técnicas cujo objetivo é ocultar as informações de pessoas não autorizadas;

D.o.S : O Denial of Service é um tipo de ataque de segurança, que visa impedir o fornecimento de um serviço a um usuário legítimo;

DoD 5200.28-STD : vide Orage Book;

Firewall : Hardware ou software utilizado para aumentar o nível de segurança de uma rede;

Gateways : São normalmente hardwares utilizados na interconexão de duas ou mais redes distintas;

ISO A International Organization for Standardization, é uma entidade internacional responsável pela emissão de normas técnicas e pelo processo de certificação de organizações;

ISO 14001: Norma internacional, que estabelece parâmetros de implementação e operação do sistema de gestão ambiental;

ISO 15408: Norma voltada para a segurança lógica das aplicações e para o desenvolvimento de aplicações seguras;

ISO/IEC 17799: Norma internacional de segurança da informação;

ITSEC : O The Information Technology Security Evaluation Criteria, foi um padrões de segurança desenvolvido em 1980 pela a Inglaterra, Alemanha, França e Nova Zelândia;

NBR-ISO/IEC 17799 : Esta é a norma nacional de segurança da informação, cujo conteúdo é uma tradução literal da norma internacional de segurança da informação (ISO/IEC 17799)

NCSC : O National Computer Security Center, é parte da estrutura da agência nacional de segurança dos Estados Unidos. Este centro e o responsável pelos testes de segurança;

NFS : O Network File System, é um dos sistemas de arquivos de rede mais utilizados no mundo Linux;

NIS : O Network Information Service, é um dos principais sistema de autenticação de usuários utilizados em redes baseadas no sistema operacional Linux;

NIST O National Institute of Standards and Technology, é um dos órgão norte amaricanos responsável pela emissão de normas técnicas e de testes de homologação.

NSA : O National Security Agency, este é a agencia norte americana de segurança, a qual vem publicando guias de segurança para vários sistemas operacionais, incluindo windows e Linux;

Nmap : Esta é uma das ferramentas *"hackers"* mais famosa, sendo largamente utilizada na varredura e coleta de dados sobre uma determinada rede ou máquina conectada a ela;

Orange Book : vide TCSEC;

PD0003 : Este foi o nome dado a primeira versão do documento que originou a norma de segurança BS7799;

Perfil da Máquina : Este termo normalmente é utilizado na especificação de um conjunto de informações sobre uma máquina, como por exemplo o sistema operacional instalado, serviços prestados por ela, seus endereços IPs e etc;

RFC : As Request for Comments, são os documentos utilizados na padronização de serviços e funcionalidades fornecidas por vários protocolos de rede;

RFC 1122 : Esta é a RFC que especifica as camadas do modelo TCP/IP;

RFC 1323 : Esta especifica várias extensão do protocolo TCP;

RFC 1700 : Ela é a responsável pela alocação dos número de porta aos serviços e protocolos implementados sobre a arquitetura TCP/IP;

RFC 1812 : Esta especifica os requisitos que os roteadores baseados no protocolo IP versão 4 (IPV4) devem satisfazer;

Rainbow Series Library : Este é o conjunto de documentos produzidos pelo Departamento de Defesa dos EUA (DoD);

Sniffers : São programas de rede normalmente utilizados para coleta e análise de pacotes de redes, sem que seu emissor ou receptor tenham consciência desta ação;

TCSEC : O Trusted Computer System Evaluation Criteria, foi o primeiro conjunto de critérios desenvolvido para análise de segurança, este critério é popularmente conhecido como "Orange Book";

Referências Bibliográficas

[1] Associação Brasileira de Normas Técnicas; Projeto 21:204.01-010 - Tecnologia da Informação - Código de Práticas para a gestão da Segurança da Informação; Abril de 2001.

[2] Associação Brasileira de Normas Técnicas; NBR-ISO/IEC 17799 - Tecnologia da Informação - Código de Práticas para a gestão da Segurança da Informação; Setembro de 2001.

[3] Apache Module Report - http:\\www.securityspace.com\s_survey\ data\man.200403\apachemods.html. Acessado em 26 de abril de 2004

[4] Arkin, Ofir; ICMP Usage in Scanning; http:\\www.sys-security.com\index.php?page=papers; Acessado em 24 de fevereiro de 2005

[5] Arkin, Ofir; ICMP Usage in Scanning, The Complete Know-How; Versão 3.0;http:\\www.sys-security.com\archive\papers\ICMP_Scanning_v3.0.pdf; Acessado em 24 de fevereiro de 2005.

[6] Arkin, Ofir e Yarochkin, Fyodor; X remote ICMP based OS Fingerprinting techniques; http:\\www.sys-security.com\index.php?page=xprobe; Acessado em 24 de fevereiro de 2005 .

[7] Arkin, Ofir; A remote active OS fingerprinting tool using ICMP; http:\\www.sys-security.com\archive\articles\login.pdf; Acessado em 24 de fevereiro de 2005.

[8] Arkin, Ofir e Yarochkin, Fyodor; The Present and Future of Xprobe2, The Next Generation of Active Operation System Fingerprinting; http:\\archive\www.sys-security.com\archive\papers\Present_and_Future_Xprobe2-v1.0.pdf; Acessado em 24 de fevereiro de 2005.

[9] Bisbey, Richard e Hollingworth Dennis; Protestion Analysis: Final Report; http:\\csrc.nist.gov\publications\history\bisb78.pdf; Acessado em 24 de fevereiro de 2005.

[10] Buddenberg, Rex; Information Security; Naval Postgraduate School, Abril de 2002.

[11] BS 7799 Compliance - How To Achieve It - http:\\www.securityrisk.co.uk\. Acessado em 20 de fevereiro de 2003.

[12] Comissão das Comunidades Europeias; Segurança das redes e da Informação: Proposta de abordagem de uma política europeia; http:\\europa.eu.int\information_society\eeurope\2002\news_library\pdf_files\netsec_pt.pdf; Acessado em 24 de fevereiro de 2005.

[13] COBRA Risk Consultant - http:\\www.securitypolicy.co.uk\risk.htm. Acessado em 20 de fevereiro de 2003.

[14] Comer, Douglas; Internetworking with TCP/IP, Volume 1, Ano 1997, Prentice-Hall

[15] Check-up Tool : http\\www.modulo.com.br\checkuptool\; Acessado em 23 de agosto de 2004.

[16] Cid, Daniel B.; Identificação Passiva de Sistemas Operacionais; http:\\www.honeypot.com.brr\repositorior\documentosr\pass.pdf; Acessado em 23 de agosto de 2004.

[17] Department of Defense; Orange Book - DoD 5200.28-STD - Trusted Computer System Evaluation Criteria; 15 de Agosto de 1983.

[18] DRAFT BS 7799-2:2002, Information Security Management - Part2: Specification for Information Security Management System. BSI, Novembro de 2001.

[19] EUROPA - Documentos da União Europeia, http:\\europa.eu.int\documents\index_pt.htm. Acessado em 15 março de 2003.

[20] Fping - http:\\www.fping.com\. Acessado em 02 de Janeiro de 2004.

[21] Fyodor; Remote OS detection via TCP/IP Stack FingerPrinting; http:\\www.insecure.org\nmap\nmapfingerprintingarticle.html; 18 de Outubro de 1998.

[22] History of Computer Security - http:\\csrc.nist.gov\publications\history\. Acessado em 2 de Dezembro de 2003

[23] Hall, Brian Beej; Beej's Guide to Network Programming - Using Internet Sockets; October 8, 2001.

[24] Hunt, Craig; TCP/IP Network Administration, 3rd Edition, O'Reilly

[25] Ib7498-2 - http:\\www.csi.map.es\csi \caibi\ibst\estandar\1\ib7498.htm. Acessado em 04 de fevereiro de 2003.

[26] Introduction to Security Risk Analysis - http:\\www.security-risk-analysis.com\. Acesso em 20 de fevereiro de 2003

[27] ISO 17799 World - http:\\www.iso-17799-security-world.co.uk\. Acesso em 20 de fevereiro de 2003

[28] ISO/IEC 7498-2, Information processing systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture, 1989

[29] IANA: List of assigned ports; ftp:\\ftp.isi.edu\in-notes\iana\assignments\port-numbers; Acessado em 10 de dezembro de 2003

[30] Casanar, Alex Delgado Gonçalves; Impacto da Implementação da norma de segurança NBR ISO/IEC 17799: Código de Prática para a Gestão da Segurança da Informação nas empresas; UFSC: Universidade Federal de Santa Catarina.

[31] Júnior, Sebastião dos Sandos; Segurança de Sistema Operacionais; http: \\www.casadaseguranca.hpg.ig.com.br\. Acessado em 21 de fevereiro de 2003.

[32] Kurshan, Robert P.; Program Verification; Notices of the AMS, vol. 47, no. 5, pages 534-544.

[33] NIST - National Institute of Standards and Technology; An Introduction to Computer Security: The Nist HandBook; http:\\csrc.nist.gov\publications\nistpubs\800-12\; Acessado em 24 de fevereiro de 2005.

[34] Nascimento, Neide Landim Teixeirense do; ISO 17799. UNEB - COPEX; Setembro de 2001.

[35] NIST - National Institute of Standards and Technology; http:\\csrc.nist.gov\publications\index.html; Acessado em 11 de setembro de 2003

[36] Nazario, Jose; Passive System Fingerprinting using Networking Client Applications; http:\\www.crimelabs.net\docs\passive.html; 27 de Novembro de 2000

[37] Netcraft Web Server Survey - April 2004 Web Server Survey - http:\\news.netcraft.com\archives\2004\04\01\april_2004_web_server_survey.html. Acessado em 26 de abril de 2004

[38] Pottinger, Carrel; Proof Requeriments in the Orange Book: Origns, Implementation, and Implications;Cornell University; 11 de Fevereiro de 1994.

[39] Portal ISO 17799 - http:\\www.iso17799.hpg.ig.com.br\. Acesso em 20 de fevereiro de 2003

[40] Rainbow Series Library - http:\\www.radium.ncsc.mil \tpep\library\rainbow\. Acessado em 21 de fevereiro de 2003.

[41] Ramos, Fabio Furtado; NBR-ISO/IEC 17799: Benefícios e Aplicações; Março de 2002.

[42] RFC793 - TRANSMISSION CONTROL PROTOCOL; DARPA INTERNET PROGRAM, PROTOCOL SPECIFICATION; Defense Advanced Research Projects Agency. September 1981

[43] RFC1700 - Assigned Numbers; Network Working Group; October 1994

[44] RFC 1122: Requiriments for Internet Hosts - Communication Layes; http:\\www.ietf.org\rfc\rfc1122.txt; Acessado em 24 de fevereiro de 2005 .

[45] RFC 1812: Requiriments for IPv4 Routers; http:\\www.ietf.org\rfc\rfc1812.txt; Acessado em 24 de fevereiro de 2005.

[46] Schiller, W.L.; The Design and Specification of a Security Kernel for the PDP-11/45; MITRE Tchenical Report - MTR-2934; Março de 1975

[47] Security Review Products - http:\\www.securitypolicy.co.uk\secrevie.htm. Acessado em 20 de fevereiro de 2003.

[48] Smith, Craing; Know Your Enemy: Passive Fingerprinting - Identifing remote hosts, without then khnowing; http:\\project.honeynet.org\papers\finger\; 4 de Março de 2002

[49] Siever,Ellen e Equipe da O'Reilly; Linux, O Guia Essencial, Tradição da Segunda Edição, Editora Campus - 2000

[50] Stevens, W. Richard; TCP/IP Illustrated, Volume 1, Addison-Wesley

[51] Uri's TCP/IP Resources List - http:\\www.private.org.il\tcpip_rl.html - Acessado em 23 de Outubro de 2003

[52] UNIX Socket FAQ - http:\\www.ibrado.com\sock-faq\. Acessado em 09 de julho de 2003

[53] Window to Security Risk Analysis; ISO 17799, Information Security Policies, Audit and Business Continuity - http:\\www.riskserver.co.\. Acessado em 20 de fevereiro de 2003.