Análise das Falhas de Segurança dos Protocolos de Comunicação do Windows NT

Marcus Cunha Granado

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

Análise das Falhas de Segurança dos Protocolos de Comunicação do Windows NT

Marcus Cunha Granado[1]

Maio de 2001

Banca Examinadora:

- Célio Cardoso Guimarães
  Instituto de Computação, UNICAMP (Orientador)
- Paulo Lício de Geus
  Instituto de Computação, UNICAMP
- Ricardo Dahab
  Instituto de Computação, UNICAMP
- Marco Aurélio Amaral Henriques
  Faculdade de Engenharia Elétrica e de Computação, UNICAMP

[1] **Projeto "Segurança em Sistemas Corporativos Baseados em Windows NT", financiado pela Compaq Brasil através do Termo Aditivo n.4 do Convênio de Cooperação Tecno-Científica entre a UNICAMP e a Compaq**

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Granado, Marcus Cunha

G762a Análise das falhas de segurança dos protocolos de comunicação do windows NT / Marcus Cunha Granado -- Campinas, [S.P. :s.n.], 2001.

Orientador : Célio Cardoso Guimarães

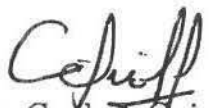
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Redes de computadores - Protocolos. 2. Windows NT (Sistema operacional de computador). I. Guimarães, Célio Cardoso. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Análise das Falhas de Segurança dos Protocolos de Comunicação do Windows NT

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Marcus Cunha Granado e aprovada pela Banca Examinadora.

Campinas, 17 de junho de 2001.



Célio Cardoso Guimarães
Instituto de Computação, UNICAMP
(Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 04 de maio de 2001, pela Banca Examinadora composta pelos Professores Doutores:

Prof. Dr. Marco Aurélio Amaral Henriques
FEEC - UNICAMP

Prof. Dr. Paulo Lício de Geus
IC - UNICAMP

Prof. Dr. Ricardo Dahab
IC - UNICAMP

Prof. Dr. Célio Cardoso Guimarães
IC - UNICAMP

Agradecimentos

Eu gostaria de agradecer a todos que contribuíram para a criação desta dissertação. Muitos foram os que ajudaram, mas o espaço não permite enumerar o nome de cada um. Isto não significa que a contribuição foi menos importante. Em especial, eu gostaria de agradecer:

Aos meus pais, por todo o incentivo despendido durante todos esses anos;

Ao meu orientador, por todos os conselhos e amizade;

À minha namorada, por todas as horas de convívio subtraídas;

Aos meus professores, colegas e amigos do Instituto de Computação da UNICAMP, pela amizade e discussões esclarecedoras;

Às professoras e aos amigos do Laboratório de Ensino de Matemática Universitária do Instituto de Matemática, Estatística e Computação Científica (IMECC/UNICAMP), pela amizade, suporte e equipamentos;

À Compaq Brasil, pelo suporte financeiro e equipamentos.

Sumário

Lista de Tabelas

Lista de Figuras

Resumo

Esta dissertação de mestrado visa analisar a segurança dos protocolos de comunicação do sistema operacional Windows NT da Microsoft, especialmente quando usados em ambientes corporativos, onde é importante determinar se informações sigilosas estão adequadamente protegidas. Várias falhas importantes, de difícil correção, encontradas dispersas na bibliografia, foram apontadas e agrupadas. Novas e importantes vulnerabilidades foram descobertas, mostrando que mesmo o Windows 2000, a última versão deste sistema operacional, está sujeita a ataques eficientes que permitem sobrepujar esses protocolos de comunicação para obter informações sigilosas.

Abstract

This master thesis aims to assess Microsoft Windows NT operating system communication protocols security, mainly when used in corporation environments, where it is important to ascertain if sensitive information is adequately secured. Several important security faults, which are difficult to correct and are dispersed in bibliography, were pointed and grouped. Important new vulnerabilities were discovered, showing that even Windows 2000, the last version of this operating system, is subject to efficient attacks that allow supplanting those communication protocols in order to obtain sensitive information.

Capítulo 1

Introdução

Esta dissertação de mestrado visa analisar a segurança dos protocolos de comunicação do sistema operacional Windows NT da Microsoft. Recentemente, o Windows NT vem se tornando pervasivo em ambientes corporativos. Era necessário, portanto, um estudo atual de sua capacidade de garantir a segurança das informações confidenciais e valiosas das empresas.

Aqui o termo "Windows NT" é usado como referência à família de sistemas operacionais da Microsoft derivada do Windows NT 3.1 original (Windows NT 3.5, Windows NT 3.51, Windows NT 4.0 e Windows 2000). Quando alguma característica for particular a algum membro desta família, o nome específico da versão do sistema operacional será citado.

Esta introdução está dividida em duas partes. Inicialmente, a motivação para esta pesquisa é mostrada. Em seguida, os principais resultados são apresentados, com referências aos capítulos que os descrevem. Entre os resultados estão uma visão unificada dos protocolos do Windows NT relevantes ao assunto e a análise original das particularidades do novo protocolo de autenticação Kerberos do Windows 2000.

A partir do capítulo 5, supõe-se que o leitor conheça fundamentos básicos de criptografia, como texto claro, texto cifrado, ataques de dicionários, funções de *hash*, e cifradores de bloco. Para aqueles que desejam entender melhor estes conceitos, recomenda-se ler [Sch96].

De um modo geral, os primeiros capítulos introduzem o sistema operacional e a arquitetura de rede. Logo a seguir, uma análise dos principais protocolos de segurança do sistema é realizada, mostrando resultados originais. E finalmente, um estudo de ferramentas de segurança disponíveis para automatizar o processo de análise de segurança do sistema em ambientes corporativos é mostrado.

1.1 Motivação

O estudo de falhas de segurança em protocolos de comunicação é uma área interessante de pesquisa. A atual tendência de interligação entre as mais diversas máquinas através de redes torna imperativa a análise crítica de novos protocolos quanto à segurança.

Inicialmente a interligação entre essas máquinas restringia-se a redes pequenas e isoladas umas das outras, onde todos eram - a princípio - confiáveis. A segurança dos protocolos usados na comunicação entre as máquinas que compunham a rede, não era portanto um requisito muito importante. Seguiu-se um processo de interligação dessas pequenas redes, às vezes através de redes públicas, como a Internet, permitindo a troca de informações entre uma quantidade enorme de máquinas. No entanto, esse processo resultou em alguns efeitos colaterais nefastos:

- a inclusão, na rede, de máquinas que não eram necessariamente confiáveis
- a transmissão das informações por linhas - públicas ou privadas - que a princípio podem ser lidas e/ou modificadas por qualquer um que tenha acesso a elas.

Os protocolos de comunicação que foram desenvolvidos considerando a situação inicial de redes confiáveis não deveriam ser usados nessa nova configuração em que as informações poderiam ser lidas por qualquer um comprometendo a integridade delas. No entanto, em vários contextos, protocolos que não consideravam inicialmente a segurança como requisito importante continuaram a ser usados nessas redes não confiáveis.

1.1.1 O Windows NT

O sistema operacional Windows NT da Microsoft está sendo amplamente adotado nos últimos anos, inclusive em empresas que possuem grande quantidade de informações confidenciais valiosas. No entanto, as deficiências em seus protocolos de comunicação são notórias quanto à segurança, autenticação, integridade e ocultamento de informações sensíveis. Esse sistema é um desses contextos onde protocolos antigos, que não priorizavam a segurança, continuaram a ser usados, mesmo quando tornou-se importante para o público-alvo desse sistema operacional a interligação com a Internet, de forma a divulgar informações para a maior quantidade possível de pessoas.

As deficiências se originam tanto da ingenuidade da concepção original dos protocolos (desenvolvidos para um ambiente onde a rede seria composta apenas de máquinas confiáveis), quanto do ocultamento das especificações destes protocolos, que não eram submetidos a nenhum fórum público de análise para descobrir falhas antes de serem implementados. A Microsoft sistematicamente oculta especificações de certos protocolos-chaves. Por exemplo, os protocolos NTLM para autenticação de usuário no Windows NT

4.0 via rede eram desconhecidos dos usuários em geral até 1997. Esse procedimento da Microsoft está longe do ideal.

A engenharia reversa da especificação de um grande número desses protocolos da Microsoft já foi realizada, e revelou que a razão do ocultamento era devida ao fato dos mesmos estarem muito aquém do nível de segurança exigido por um sistema operacional. A partir de 1997, tornou-se público que vários protocolos usados no Windows NT eram incrivelmente ingênuos em matéria de segurança, dando a impressão de terem sido desenvolvidos visando seu uso apenas em redes confiáveis. Diversas análises começaram a ser publicadas explorando as fraquezas dos protocolos desse sistema e sistematicamente descobrindo falhas importantes. Por exemplo, desde 1997 são amplamente conhecidas as graves falhas do protocolo NTLM de autenticação baseado em 'desafio-resposta criptografada', usado no Windows NT 4.0 e anteriores [Hob97]. Estas falhas básicas permitem a descoberta da senha dos usuários utilizando ataques otimizados, como os mostrados no capítulo 6.

Um dos resultados mais visíveis do esforço de engenharia reversa é o pacote SAMBA do Unix [SMB01], um pacote contendo código fonte de domínio público que permite a comunicação entre o UNIX e o Windows NT utilizando-se do protocolo de comunicação SMB (*Server Message Block protocol*) do Windows NT. Esse protocolo é a base de funcionamento do sistema "LAN Manager" de compartilhamento de arquivos distribuídos e impressoras. É possível, utilizando o SAMBA, ou algum outro pacote que consiga lidar com SMB, simular um cliente (ou servidor) Windows NT malicioso e, explorando fraquezas na autenticação do protocolo SMB, conseguir até mesmo a senha do usuário ou outra informação sensível. Outro resultado visível é a engenharia reversa do protocolo para chamada de procedimentos remotos (RPC, de *Remote Procedure Calls*) em cima do SMB [KL00]. Estas chamadas são amplamente utilizadas por ferramentas administrativas do Windows NT para acesso e configuração do sistema, e são repletas de falhas de segurança, como pode ser visto na seção 3.3.7.

Novas versões do Windows NT, como o Windows 2000, apareceram com novas características de segurança associadas aos protocolos de comunicação, como o difundido protocolo de autenticação Kerberos, utilizado preferencialmente no lugar do antigo NTLM. O Kerberos destina-se a autenticar, ou seja, assegurar que as partes envolvidas em uma comunicação realmente são quem alegam ser, e a permitir o intercâmbio de uma chave criptográfica entre essas partes antes da conexão ser feita, de modo a permitir criptografar a informação trocada durante a conexão. Esse protocolo, ao contrário do NTLM, foi bastante escrutinizado em fóruns públicos quanto a segurança (ver, por exemplo, [BM91], [KNT94], [Gon95], [Jas96] e [KPS95]) durante os anos '90. No entanto, uma pesquisa de segurança envolvendo a implementação específica do Kerberos no Windows 2000, como a mostrada no capítulo 8, em que várias falhas são apontadas, ainda não havia sido feita.

A implementação pela Microsoft de protocolos de segurança tem se revelado um desastre após o outro. Por exemplo, a implementação Microsoft do *Point to Point Tunneling Protocol* (PPTP) do Windows NT não é robusta o suficiente do ponto de vista criptográfico, possuindo várias falhas inerentes ao defasado método de autenticação baseado em 'desafio-resposta criptografada' do NTLM e geração incorreta de chaves criptográficas ([Sch98] e [Sch99]), como pode ser visto na seção 6.6.

Em um sistema operacional que a princípio deveria prezar a segurança como característica mais importante, pesquisas nesta área são fundamentais para determinar quais protocolos usados continuam com deficiências, e se novos protocolos sendo implementados ou desenvolvidos pela Microsoft para as versões atuais e futuras de seu sistema operacional estão aquém do que seria considerado razoável.

1.2 Organização da Dissertação

Inicialmente um extenso estudo da história do Windows NT foi realizado, com ênfase na arquitetura de rede e protocolos de segurança. Um dos requisitos era entender a modularidade da arquitetura de rede feita para facilitar a incorporação de quaisquer protocolos de transporte e autenticação. Outro requisito era entender os protocolos originais compatíveis com o produto "LAN Manager" utilizado no MS-DOS e OS/2 da IBM. Um esforço foi feito para fazer uma descrição unificada de todas as versões do Windows NT. Este estudo, mostrado nos capítulos 2 e 3, permitiu localizar o Windows NT dentro do contexto dos sistemas operacionais e de protocolos de segurança dos anos '80 onde este se originou. Por exemplo, o antigo produto "LAN Manager", da IBM e Microsoft, para compartilhamento de arquivos e impressoras numa rede local usando MS-DOS, OS/2 ou Windows 3.1 for Workgroups, utilizava o protocolo SMB para comunicação, e um protocolo de autenticação baseado em enviar a senha do usuário em texto claro ou usando o método de 'desafio-resposta criptografada' - usado hoje no protocolo de autenticação NTLM do Windows NT. Estes protocolos são utilizados até hoje e constituem ainda um importante subsistema de comunicação (e de falhas de segurança) do Windows NT. As novas características de segurança incorporadas ao Windows NT são discutidas no capítulo 4.

O principal objetivo da dissertação era verificar a segurança (ou falta dela) nos protocolos de comunicação do Windows NT. Certos protocolos já tinham falhas conhecidas em outros sistemas que não se aplicavam ao Windows NT (como o Kerberos versão 4) e outras que se aplicavam (como a autenticação LAN Manager original). Algumas falhas já haviam sido apontadas (como no PPTP). Outros ainda não haviam sido devidamente escrutinizados no Windows 2000 (Kerberos versão 5, SNTP) e constituem o principal resultado dessa dissertação.

Verificou-se que no Windows NT os protocolos seguros de comunicação e transporte de dados dependem de um pequeno número de protocolos de autenticação iniciais que asseguram a identidade das partes envolvidas, trocam chaves criptográficas secretas entre elas, e permitem a implementação de segurança via criptografia. Se houver uma falha na autenticação inicial, esta falha pode ser explorada de modo a obter as chaves criptográficas secretas. De posse destas chaves, o ocultamento e integridade das informações trocadas entre as partes podem ser comprometidas. A segurança destes protocolos de autenticação básicos, portanto, é essencial.

Desta maneira, ao lado de analisar falhas de segurança em protocolos de transporte de dados típicos do Windows NT, como o SMB, o NBT (SMB sobre TCP/IP), o RPC sobre SMB, o PPTP (ver capítulos 3 e 6), e outros como o protocolo de sincronização de relógio *Simple Network Time Protocol* (SNTP), usado no Windows 2000 - ver capítulo 7, também é parte integrante o estudo das falhas de segurança envolvidas nos protocolos de autenticação associados, como o NTLM e o Kerberos (ver capítulos 6 e 8).

Uma classificação dos diversos métodos de autenticação disponíveis nas décadas de '80, '90 e 2000 em diante é realizada no capítulo 5. Os protocolos de autenticação do Windows NT são enquadrados nessas classes. De acordo com essa classificação, o Windows NT ainda não oferece um sistema de autenticação seguro, mesmo com o novo protocolo Kerberos: o NTLM oferece métodos obsoletos de autenticação do início dos anos '80, e o Kerberos novos métodos mais seguros do fim dos anos '80 e início dos '90. Mesmo assim, a implementação do Kerberos do Windows 2000 da Microsoft tem falhas na autenticação inicial, permitindo a utilização de ataques similares aos já disponíveis no protocolo NTLM do Windows NT 4.0 para descoberta da senha do usuário, acesso não-autorizado a recursos e comprometimento do sistema. Novos métodos seguros de autenticação de senhas, surgidos a partir dos anos '90 - baseados em trocas de chaves aleatórias públicas/privadas temporárias, criadas e usadas apenas na autenticação e depois descartadas - não são utilizados no Windows NT.

Com base nas diversas classes de protocolos de autenticação, várias recomendações para evitar as deficiências encontradas nesses protocolos são sugeridas, baseadas na substituição do protocolo ou no acoplamento da implementação já existente com os novos métodos seguros de autenticação de senhas dos anos '90. As análises detalhadas das falhas nos protocolos de autenticação NTLM e Kerberos podem ser vistas nos capítulos 6 e 8, respectivamente. O capítulo 6 foi apresentado em outubro de 1999 na "2a Conferência de Redes de Computadores de Portugal (CRC99)", em um artigo entitulado "Aspectos Criptográficos no Windows NT" [Gra99]. O capítulo 8 foi apresentado no WSeg'2001 (Workshop em Segurança de Sistemas Computacionais), realizado nos dias 5 e 6 de março de 2001 em Florianópolis, SC, com o título *Aplicando Ataques de Dicionário no protocolo Kerberos do Windows 2000* [Gra01a], e uma variação vai ser apresentada em 16 de

maio de 2001 na "10a Conferência Internacional em Administração de Sistemas, Redes e Segurança (SANS2001, de *Tenth International Conference on System Administration, Networking and Security*)", em um artigo entitulado *Attacking Windows 2000 Kerberos Password Security* [Gra01b].

Por fim, como último objetivo da dissertação, uma análise das principais ferramentas de segurança existentes para Windows NT e sua classificação segundo um critério de utilidade para apontar falhas nas configurações das máquinas rodando Windows NT é mostrada no Apêndice A. Estas ferramentas são importantes por proporcionarem um acompanhamento atualizado de todas as vulnerabilidades conhecidas do sistema, muitas vezes com aplicação automática das últimas atualizações e correções.

Capítulo 2

Arquitetura do Windows NT

De acordo com a Microsoft ([Cus93], [Sol98] e [Sol00]), o Sistema Operacional Windows NT foi inicialmente projetado para ser um sistema operacional robusto, portável, que deveria ser flexível e ter manutenção fácil. Este capítulo explica o projeto, sua história, seus principais objetivos e sua arquitetura subjacente, de modo a estabelecer um contexto que os capítulos subsequentes utilizarão.

2.1 Uma visão Histórica do Windows NT

OS/2, IBM e Microsoft

O Windows NT é o resultado de uma cisão entre a Microsoft e a IBM no seu programa de co-desenvolvimento do sistema operacional OS/2, iniciado na década de 1980.

Em outubro de 1988, a Microsoft contratou David N. Cutler, um conhecido arquiteto de sistemas de minicomputadores[1], para liderar um novo esforço de desenvolvimento: criar o sistema operacional da Microsoft "para os anos '90", em substituição ao DOS. David Cutler montou um time de engenheiros para projetar o sistema operacional de Nova Tecnologia (NT, de *New Technology*) da Microsoft.

Coincidentemente, a IBM e a Microsoft introduziram o sistema operacional *OS/2 Presentation Manager* em outubro de 1988. Enquanto David Cutler desenvolvia o NT, a Microsoft continuou a trabalhar com a IBM para melhorar o OS/2, que era pré-requisito para rodar o sistema de rede LAN Manager da Microsoft e a versão para *computador pessoal* (PC) do sistema de gerenciamento de base de dados relacional SQL Server da Sybase, apresentada em 1988 como Microsoft SQL Server.

[1] Antes de seu trabalho na Microsoft, David N.Cutler trabalhou na Digital Equipment Corporation e por 17 anos desenvolveu vários sistemas operacionais e compiladores, incluindo o VAX/VMS, o Micro VAX I, o RSX-11M em máquinas PDP-11, e os compiladores VAX PL/1 e VAX C.

Os produtos *LAN Manager* da Microsoft e *LAN Server* da IBM derivaram do produto de rede ponto-a-ponto (*peer-to-peer*) MS-NET da Microsoft para DOS, vendido pela IBM como PC-NET. O LAN Manager, inicialmente usando apenas o protocolo de transporte NetBEUI (*Network Basic Input/Output System (NetBIOS) Extended User Interface*), introduzido pela IBM em 1985, nunca se destacou no mercado de Sistemas Operacionais de Rede (NOS, de *Network Operating Systems*). No início de 1990, ele detinha 10% do mercado de NOS. O produto NetWare da empresa Novell era o líder, com cerca de 60% do mercado.

A Microsoft lançou o Windows 3.0 em 22 de maio de 1990. O Windows 3.0 foi um sucesso absoluto de vendas, ofuscando o nascente mercado para o OS/2 nas máquinas PCs. Em 17 de setembro de 1990, a Microsoft e a IBM anunciaram uma divisão de responsabilidade para o desenvolvimento do OS/2. A IBM, sozinha, iria trabalhar nas versões 1.x e 2.x, enquanto a Microsoft iria ser responsável pelas versões avançadas 3.x. O OS/2 3.0 foi especificado para ser um sistema 32 bits, portável, multiprocessável, preemptivo e multitarefa com características de segurança avançadas.

Em setembro de 1991, com o Windows assegurando um papel dominante no mundo dos PCs, a Microsoft iniciou o rompimento com a IBM: Steve Ballmer, o vice presidente de marketing da Microsoft, declarou que o OS/2 "estava morto" se a IBM não concordasse em usar o *kernel* do Windows NT como a fundação do OS/2 3.x. A IBM, que mesmo em 1991 estava chamando sua versão beta do OS/2 2.0 "melhor DOS que DOS, melhor Windows que Windows", não desejava adotar o *kernel* de um sistema operacional centrado no Windows para o OS/2 3.0. Separadamente, a IBM iniciou o desenvolvimento do que no fim se tornou a primeira versão do OS/2 Warp, um sistema operacional 32 bits que era compatível com aplicações Windows 16 bits e não oferecia nem portabilidade entre processadores e arquiteturas, nem capacidade de multiprocessamento.

Windows NT 3.1

A Microsoft introduziu o *Windows NT 3.1* e *Windows NT 3.1 Advanced Server* para processadores Intel e MIPS em agosto de 1993. O número de versão veio do Windows 3.1, cuja interface gráfica (GUI, de *Graphical User Interface*) foi copiada para o Windows NT. A versão *Advanced Server*, projetada para substituir o sistema LAN Manager rodando em cima do OS/2 da Microsoft, suportava o protocolo NetBEUI e incluía uma pilha primitiva de TCP/IP. Os recursos exigidos pelo sistema eram imensos para a época: 16 MB de memória RAM, e 100 MB de disco rígido apenas para o sistema. O custo associado ao *hardware* e o fato de ainda ser um sistema imaturo contribuíram para a baixa aceitação do sistema.

Windows NT 3.5 e 3.51

O Windows NT 3.5, a primeira grande atualização ao sistema operacional, apareceu em 21 de setembro de 1994. Durante esta época, havia muita atenção devotada ao lançamento dos novos sistemas operacionais Windows 95 e IBM OS/2 Warp. O time de desenvolvimento reduziu a memória necessária para 12 MB, e portou o Windows NT para o processador RISC Alpha AXP da Digital.

Os melhoramentos mais significantes foram a adição do NWlink (a pilha IPX/SPX da Microsoft para o NetWare da Novell), e melhoramentos na pilha TCP/IP. A Microsoft mudou o protocolo de rede padrão do NetBEUI da IBM para o IPX/SPX da Novell, mas foi a inclusão de características TCP/IP, como WINS (*Windows Internet Naming Service*), DNS (*Domain Name Service*), DHCP (*Dynamic Host Configuration Protocol*), e suporte para PPP (*Point-to-Point Protocol*) e SLIP (*Serial Line Internet Protocol*) em cima de RAS (*Remote Access Service*), junto com a queda do preço do *hardware* necessário, que chamou a atenção da indústria para considerar seriamente a utilização do Windows NT 3.5 como um servidor de produção, bem como um servidor de base de dados. Junto com o Windows NT 3.5, a Microsoft lançou a versão 1.0 de seu pacote de produção BackOffice.

Uma atualização pontual, a Microsoft anunciou o Windows NT 3.51 em 12 de junho de 1995. O objetivo era oferecer compatibilidade com as bibliotecas de controles e diálogos da interface gráfica do Windows 95, de modo a rodar aplicações "Projetadas para Windows 95" no Windows NT. Novas adições incluíam a possibilidade de compactar seletivamente arquivos em partições NTFS, e suporte para tecnologia PC Card (PCMCIA). Outros melhoramentos incluíam suporte à tecnologia "NetworkOLE 0.9" (o precursor da tecnologia DCOM do Windows NT 4.0), que deixa desenvolvedores criarem aplicações distribuídas para sistemas de base de dados cliente/servidor em três camadas (*three-tier*). O protocolo de rede padrão, também, mudou do IPX/SPX para o TCP/IP.

Windows NT 4.0 e a a atualização da interface gráfica

A Microsoft supôs corretamente que depois que uma pessoa usasse o Windows 95 por um período de tempo, ela consideraria antiquada a interface gráfica do Windows NT 3.51. Para resolver este problema, a Microsoft liberou uma versão alfa não suportada da *Atualização da Interface Gráfica* (SUR, de *Shell Update Release*) para o Windows NT 3.51 no fim de 1995, com planos de liberar a SUR definitivamente na forma de um *Service Pack* (atualização) para o Windows NT 3.51.

Embora a Microsoft insistisse que a versão 4.0 era reservada para a "próxima grande versão do Windows NT" (na época, Cairo, a ser lançado por volta de 1998), a primeira versão beta da SUR tinha o nome de Windows NT 4.0 já no início de 1996. A versão final do Windows NT 4.0 foi liberada para produção em 31 de julho de 1996. Ela in-

cluía várias inovações, as mais importantes relacionadas à utilização da nova tecnologia DCOM (*Distributed Common Object Model*) - antiga NetworkOLE, e um melhoramento no DNS para TCP/IP. A Microsoft precisava da DCOM para implementar seus planos de distribuir controles *ActiveX* (antigo OLE) e documentos via Internet e para implementar computação cliente/servidor em três camadas utilizando *Automation* (antigo *OLE Automation*). A Microsoft planejava usar o DNS como a base de seus serviços de diretório para a próxima versão do Windows NT. Outras adições incluem o protocolo de tunelamento PPTP (*Point-to-Point Tunneling Protocol*) e a API de Telefonia (TAPI) 1.0.

Cairo

Os objetivos iniciais da Microsoft em 1995 para Cairo estavam centrados principalmente na criação do OFS (*Object File System*). O OFS era para ser um sistema de arquivos distribuído, orientado a objetos que incorporaria um repositório de diretório[2]. As entradas no repositório de diretório seriam baseadas num sistema de nomes usando nomes de objetos e valores de atributos (tais como autor e data de criação), ao invés da sintaxe UNC (*Uniform Naming Convention*, do tipo \\Servidor\Compartilhamento\Arquivo), de modo a permitir a busca por nomes de arquivos e também por atributos.

No entanto, a enorme demanda comercial por produtos relacionados à Internet e a defasagem da Microsoft nesta área ocasionaram a "Iniciativa Internet" da Microsoft, em dezembro de 1995, resultando em uma mudança de planos para a implementação do OFS e o repositório de diretório. Em março de 1996 a Microsoft anunciou que as características do OFS seriam incorporadas ao NTFS e que o repositório iria ser implementado em uma base de dados derivada da arquitetura de armazenamento de mensagens do Microsoft Exchange Server 4.0. Também, as convenções de DNS seriam suportadas, permitindo o uso de nomes DNS para especificar árvores de diretórios e para habilitar a inclusão de URLs de intranets e da Internet no repositório.

Esta mudança de estratégia significou um atraso na liberação da próxima grande versão do Windows NT, e a parada definitiva na produção do Cairo. Até lá, a estratégia seria acrescentar cada vez mais facilidades de interação com a Internet através de atualizações periódicas (*Service Packs*) ao Windows NT 4.0.

Windows 2000

Em outubro de 1998, a Microsoft anunciou que a próxima versão do Windows NT, até então informalmente chamada Windows NT 5.0, chamar-se-ia Windows 2000. O Windows

[2]Um repositório de diretório é uma base de dados que contém registros para as árvores de diretórios de todos os servidores em um domínio. O repositório oferece uma visão lógica da rede que não depende de sua implementação física, agrupando classes de objetos distintos como usuários, arquivos, impressoras, configurações, dispositivos etc.

2000, lançado em 17 de fevereiro de 2000, é mais uma etapa da Microsoft no desenvolvimento do Windows NT, com duas grandes adições ao sistema. A primeira é o novo sistema de serviço de diretório X.500 (*Microsoft ActiveDirectory*, no jargão do Windows NT), destinado a armazenar de maneira distribuída todas as configurações do sistema, como usuários, senhas, e informações de serviços e dispositivos. A segunda é o protocolo de autenticação Kerberos, que substitui o obsoleto protocolo de autenticação NTLM do Windows NT 4.0 e anteriores. Outras adições incluem um sistema de arquivos criptografado (EFS, de *Encrypted File System*), e uma política muito estrita de permissão de acesso aos arquivos do sistema e outros recursos, para dificultar a modificação indevida de bibliotecas do sistema e dificultar a disseminação de Cavalos de Tróia e vírus. A política padrão de acesso agora assemelha-se muito à do UNIX, onde o usuário normal pode escrever apenas no seu diretório de trabalho.

O Windows NT foi projetado para executar em uma variedade de arquiteturas de *hardware*, como sistemas CISC Intel e RISC de diversos fabricantes. A versão inicial (Windows NT 3.1) suportava as arquiteturas Intel i386, i486 e MIPS R4000. O suporte para o processador Alpha AXP da Digital Equipment Corporation (DEC) foi adicionado seis semanas após o lançamento do Windows NT 3.1. Após o lançamento do Windows NT 3.5, no fim de 1994, foi adicionado suporte para o processador PowerPC. Devido a demandas de mercado, porém, o suporte às arquiteturas MIPS e PowerPC foi retirado antes que o desenvolvimento do Windows 2000 iniciasse. Posteriormente, a Compaq retirou o suporte para a arquitetura Alpha AXP. Desta forma, a arquitetura Intel x86 é a única que oferece suporte ao Windows 2000.

Windows NT de 64 bits e Além

A próxima arquitetura que uma versão futura do Windows NT irá oferecer suporte é a nova família de processadores Intel Itanium de 64 bits, desenvolvida pela Intel e Hewlett-Packard, chamada IA-64 (Intel Architecture 64). Esta nova versão do Windows NT suportará diversas melhorias relacionadas ao gerenciamento de memória, como suporte a até 16 TB de espaço de endereçamento virtual para processos, ao invés dos usuais 4 GB da arquitetura de 32 bits. O tamanho máximo do arquivo de paginação de memória em disco irá aumentar de 16TB para 512TB. Devido ao aspecto modular do Windows NT, esta mudança não necessariamente implica grandes mudanças estruturais na arquitetura do sistema, com exceção, é claro, do módulo de Gerenciamento de Memória.

Conforme David Cutler costuma afirmar, o projeto do Windows NT é um projeto ainda em andamento. O Windows 2000 é a versão mais trabalhada, testada e avançada de todas, o resultado de mais de 11 anos de trabalho, mas "há mais por vir e estamos todos ocupados trabalhando na próxima versão."

2.2 O projeto do Windows NT

Quando o time de desenvolvimento do Windows NT foi formado em 1989, ele tinha uma missão clara: projetar e construir um sistema operacional para computadores pessoais (PCs) que deveria satisfazer as necessidades atuais e futuras para a plataforma PC. Para atingir este objetivo, no início de 1989, Bill Gates e estrategistas da Microsoft se encontraram para revisar as especificações do sistema operacional que o time de David Cutler havia definido. Eles identificaram as seguintes necessidades de mercado e objetivos iniciais do projeto:

Robustez O sistema deve ser robusto e proteger-se ativamente de malfuncionamento interno e externo (acidental ou deliberado);

Eficiência O sistema deve ser eficiente, rápido e otimizado;

Portabilidade Fornecer portabilidade fácil para outras arquiteturas de 32 bits (RISC e CISC) com um mínimo de mudança de código;

Escalabilidade Fornecer escalabilidade e suporte a multiprocessamento (vários processadores); o sistema deve satisfazer as necessidades de produtores independentes de computadores pessoais (OEMs, de *Original Equipment Manufacturers*) e da Microsoft: deve acomodar mudanças e adições aos conjuntos de interfaces de programação originais (APIs, de *Application Program Interface*).

Interoperabilidade Suportar computação distribuída, permitindo a múltiplos computadores acessar recursos compartilhados;

POSIX Oferecer as APIs necessárias ao sistema POSIX (*Portable Operating System Interface for UNIX*, IEEE Standard 1003.1-1988), conforme requisitado pelas agências governamentais dos EUA nos contratos de computação com o governo, no fim da década de 1980;

Nível C2 Oferecer pelo menos os requisitos do nível de segurança C2 (Proteção por Acesso Controlado) do Departamento de Defesa do governo dos EUA, de acordo com o critério de avaliação definido em [DoD85] (*Orange Book*, ou TCSEC, de *U.S. DoD Trusted Computer System Evaluation Criteria*), além de ter um caminho para a classe B1 e acima. Os níveis de segurança começam do D (Mínima Proteção) até o A (Projeto Verificado por Análise Formal), com vários subníveis. O nível D está reservado para sistemas que não se enquadraram em nenhum nível de segurança. O nível de segurança mais baixo, acima do D, é o C2. No nível C2, atualmente, além do Windows NT, estão classificados sistemas operacionais como OpenVMS VAX e

Alpha Versão 6.1, IBM AS/400 com OS/400, IBM RS/6000, componentes como Novell NetWare 4.11, e aplicações como Oracle 7, Microsoft SQL Server 2000 versão 8.0, e Sybase SQL Server versão 11.0.6, entre outros.

Um sistema com nível de segurança C2 (Proteção por Acesso Controlado), entre outras coisas, deve ser capaz de permitir ou negar uso e/ou acesso a recursos do sistema para certos usuários ou grupos de usuários; garantir que quando um bloco de memória é liberado, seu conteúdo é explicitamente sobreescrito antes de ser designado a outro processo; identificar todos os usuários, habilitar um rastro de auditoria em qualquer usuário, e atribuir todas as ações do rastro de auditoria ao usuário que as realizou; e proteger-se de modificações a arquivos e componentes do sistema.

Em 31 de julho de 1995, o *Windows NT 3.5 com Service Pack 3*, submetido para avaliação ao Departamento de Defesa dos EUA (DoD, de *Department of Defense*), foi formalmente incluído na lista de produtos seguros no nível de segurança C2 [NSA95]. Em novembro de 1999, o *Windows NT 4.0 com Service Pack 6a e Atualização C2* foi incluído na mesma lista, com o mesmo nível de segurança [NSA99]. Há planos de elevar o nível de segurança do Windows NT e incluí-lo no nível de segurança B2 (Proteção Estruturada) [SH96]. Nesta categoria está apenas o sistema operacional Trusted XENIX 3.0 e 4.0, da Trusted Information Systems, Inc.

2.3 Os Componentes do Sistema

A arquitetura do Windows NT agrupa os atributos de um *sistema operacional em camadas* com aqueles de um *sistema operacional modular, com microkernel* (ver figura 2.1):

HAL A camada mais profunda do sistema é a *Camada de Abstração de Hardware* (HAL, de *Hardware Abstraction Layer*), que oferece uma abstração comum de software em cima de dispositivos tais como relógios, controladores de memória, adaptadores de periféricos, funções de multiprocessamento simétrico (SMP, de *symmetric multiprocessing*), e barramentos de sistema (*system buses*). Um exemplo de abstração é o tamanho da página de memória virtual do processador, que Windows NT determina dinamicamente na hora de inicialização (*boot*) do sistema.

MicroKernel Acima está o *microkernel*, o coração do sistema operacional, que oferece as mais básicas funções do sistema operacional, tais como manipulação de interrupção, gerenciamento de *threads*, e sincronização de primitivas.

Executivo Um *módulo executivo (executive)*, executando no modo privilegiado/supervisor do processador (*kernel mode, ring 0 mode* ou *supervisor mode*), oferece serviços de

sistema.O executivo fornece um *único* ponto de entrada para o sistema, e é um sistema operacional completo, de baixo nível, auto-contido, faltando apenas a interface com o usuário.

Subsistemas Protegidos Uma série de servidores não-privilegiados chamados *subsistemas protegidos* (*protected subsystems*) executa no modo não-privilegiado/usuário (*user mode* ou *ring 3 mode*[3]), acima do executivo. Eles englobam subsistemas isolados para aplicações Windows de 32 bits, 16 bits, DOS, OS/2 e POSIX, mais as aplicações clientes em cada uma destas cinco categorias.

O Windows NT tem um *microkernel* bem definido que roda em modo privilegiado, mas não é um sistema *microkernel* "puro": vários outros módulos do sistema também executam no modo privilegiado por motivos de otimização de desempenho, o que também é chamado de "*microkernel* modificado" ou *macrokernel*. Todos estes módulos do sistema são confiáveis, e compõe a chamada *Base de Computação Confiável* (TCB, de *Trusted Computing Base*). A figura 2.1 ilustra a arquitetura básica do Windows NT até sua versão 3.51 e na versão 4.0. Notar que apenas alguns subsistemas são mostrados na figura.

2.3.1 Os Componentes do Executivo

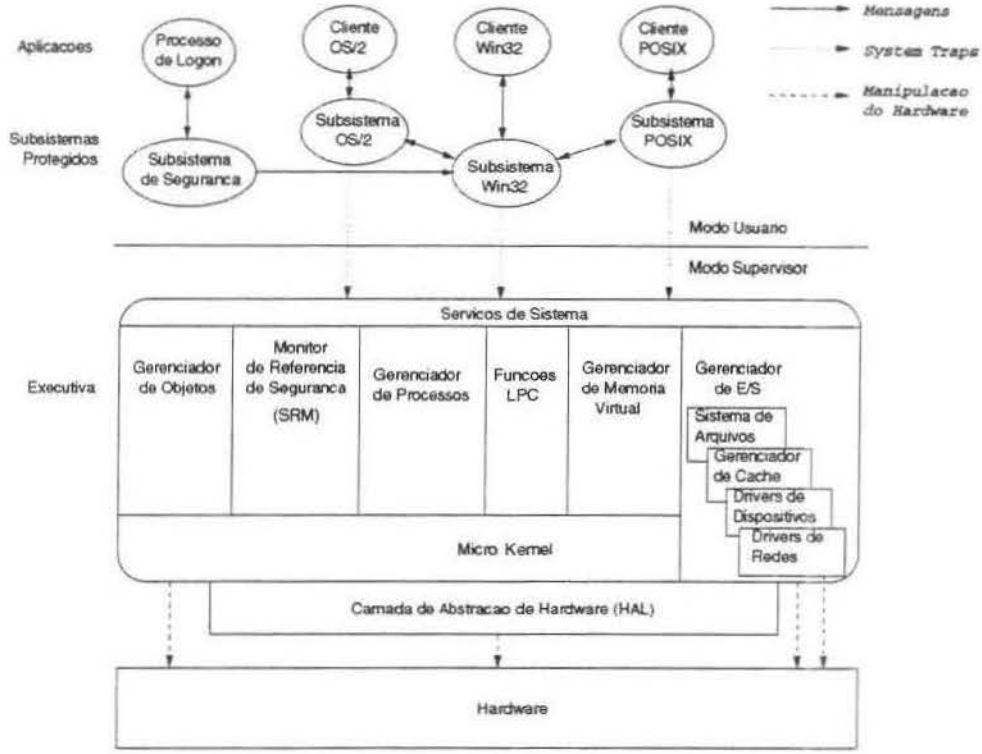
A executiva do NT é um nome genérico para uma série de componentes de serviço do sistema, que são expostos através de um conjunto de serviços de sistema na forma de APIs:

Gerenciador de Processos (PM, de *Process Manager*) Responsável por criar e destruir processos, *threads* e recursos associados. Lida com a alocação das *threads* aos processadores quando o multiprocessamento está em uso.

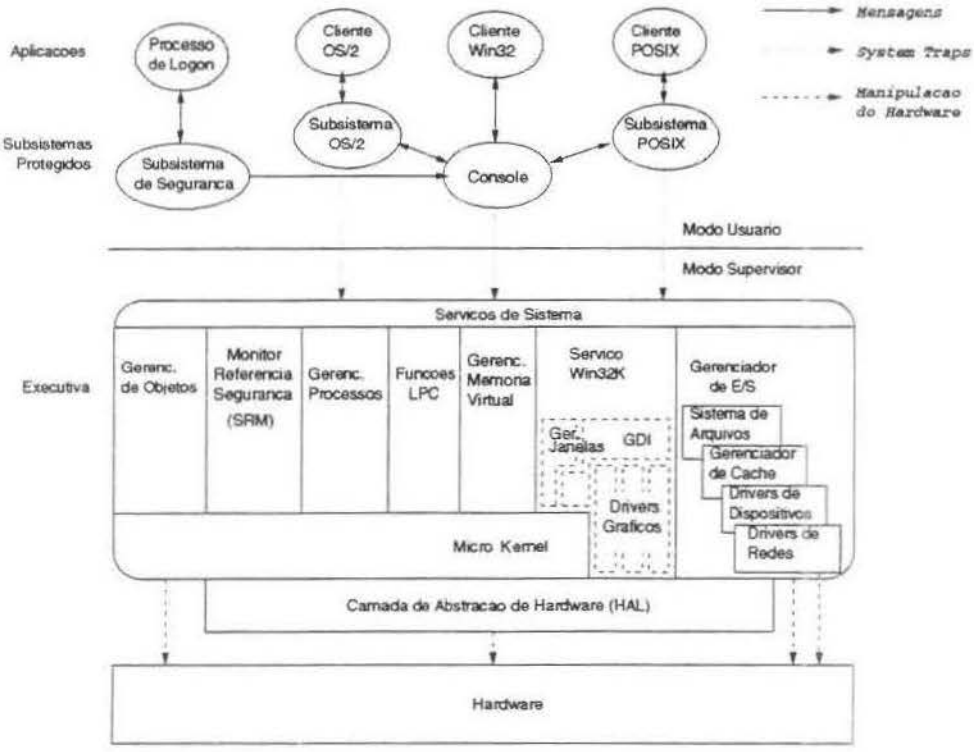
Módulo LPC (Local Procedure Call Facility) Fornece comunicação entre aplicações (clientes) e subsistemas protegidos (servidores). É uma versão local, mais leve, da comunicação RPC (*Remote Procedure Call*).

Monitor de Referência de Segurança (SRM, de *Security Reference Monitor*) Fornece as características de segurança básicas, tais como criação de *token* de acesso e algoritmos de controle de acesso aos recursos.

[3]Os processadores CISC Intel, suportados pelo Windows NT, desde o 80386 fornecem *níveis de privilégios* (PL, de *privilege levels*) ou anéis de proteção (*rings*). O Windows NT utiliza apenas o PL=0 (modo mais privilegiado) e o PL=3 (modo menos privilegiado), de modo a manter portabilidade de código através de arquiteturas RISC, onde usualmente existem apenas dois níveis de privilégios.



(a) A arquitetura basica do Windows NT 3.51 e anteriores



(b) A arquitetura do Windows NT 4.0

Figura 2.1: Comparação das arquiteturas de versões do Windows NT 3.51 e 4.0

Gerenciador de Objetos (OM, de *Object Manager*) Responsável por criar, acessar e destruir objetos pertencentes a classes de recursos do sistema, tais como Processos, *Threads*, Arquivos, Portas LPCs, *Tokens* de Acesso, Eventos, Semáforos, Temporizadores, Links Simbólicos de Objetos etc. Todos os objetos possuem um conjunto comum de dados e métodos de acesso, e outro específico a cada classe. O OM é também responsável por criar e gerenciar internamente no executivo um espaço de nomes de objetos unificado, com uma estrutura hierárquica semelhante a arquivos e diretórios, onde sub-hierarquias inteiras podem ser montadas (*mounted*) e controladas por outros componentes da executiva. Como todo acesso aos objetos é feito através do OM, ele unifica o controle de acesso a qualquer objeto do sistema, redirecionando a validação do acesso ao SRM, e, caso necessário, roteando a responsabilidade de execução da ação requerida ao componente do executivo responsável pela sub-hierarquia particular do espaço de nomes do OM.

Gerenciador de Memória Virtual (VMM, de Virtual Memory Manager) Aloca memória protegida a cada processo. Quando a necessidade de memória excede a quantidade de RAM disponível, o VMM armazena parte da RAM no arquivo de paginação do disco rígido, e a recupera quando novamente necessária (*swapping*). Alguns objetos vitais são guardados num *espaço não-paginável* (*non-paged pool*) que nunca é paginado ao disco.

Gerenciador de E/S (IOM, de I/O Manager) Atua nos arquivos e dispositivos que manipulam arquivos, incluindo componentes de rede. Inclui todos os *Sistemas de Arquivos* disponíveis (FAT, NTFS, NPFS etc), o *Gerenciador de Cache*, *Network Drivers* (componentes *Redirector* e *Server* do ambiente de rede LAN Manager) e *Drivers de Dispositivos* para placas de rede, teclados, *mouse*, adaptador gráfico, discos etc.

Um subsistema protegido executa no modo não-privilegiado como um processo regular, com seu próprio espaço protegido de memória. O subsistema pode ter privilégios estendidos se comparado a uma aplicação, mas não é considerado parte do executivo e, portanto, não pode suplantar a arquitetura de segurança do sistema ou corromper o sistema. Os subsistemas comunicam com seus clientes e entre si usando chamadas de procedimento local de alta performance (LPCs, de *Local Procedure Call*s), e com o executivo através de chamadas de sistema.

Os subsistemas que oferecem funcionalidade adicional podem ser adicionados ao sistema sem interferir na sua base. Novos subsistemas podem ser adicionados sem modificações ao executivo, em adição aos inicialmente oferecidos subsistemas para ambientes Win32, DOS (também chamado VDM, de *Virtual DOS Machine*), Win16 (WOW, de *Windows*

on Win32), OS/2 (suporte a console apenas) e POSIX. O subsistema Win32 fornece a interface de usuário (teclado, *mouse*, janelas etc) para todos os outros subsistemas. Todos estes subsistemas podem usufruir das características de segurança oferecida pelo executivo. O próprio executivo é modular em projeto: seus componentes internos são independentes entre si, e se comunicam através de APIs bem definidas, possibilitando substituir um componente sem impactar significativamente o sistema.

No Windows NT 4.0 (figura 2.1), muito da interface gráfica do usuário (GUI, de *Graphical User Interface*) do subsistema Win32 - o Gerenciador de Janelas (*Window Manager*), Interface de Dispositivos Gráficos (GDI, de *Graphics Device Interface*), e os *drivers* relacionados - foram movidos de um corpo de código que executava no processo CSRSS.EXE (subsistema Win32) para um *driver de dispositivo* (*device driver*) WIN32K.SYS em modo privilegiado no executivo (serviço Win32, na figura). Porções relacionadas ao console de texto e controle de erros permaneceram no modo não-privilegiado. Esta mudança aumentou significantemente a performance do sistema gráfico, diminuiu os requisitos de memória, e não teve impacto para desenvolvedores de aplicações [TNet00b]. As aplicações agora acessam os subsistemas GUI da mesma forma que outros serviços do sistema - via executivo - tais como entrada/saída e gerenciamento de memória. Esta mudança profunda no sistema, sem impacto para desenvolvedores e usuários, é muito citada como uma demonstração da mantenabilidade e flexibilidade do projeto modular do Windows NT.

O executivo do Windows 2000 acrescentou suporte à tecnologia *Plug and Play*, opções para controle de gasto de energia (*Power Options*), e uma extensão ao modelo de *drivers* do Windows NT chamada *Windows Driver Model* (WDM). O Windows 2000 é compatível com antigos *drivers* do Windows NT 4.0. O modelo da arquitetura do Windows 2000 pode ser visto na figura 2.2.

2.3.2 Componentes Não-Privilegiados

Existe uma imensa quantidade de componentes não-privilegiados do sistema, como banco de dados de configuração e processos, que executam em cima da executiva para oferecer serviços. Entre eles, destaca-se o Registro do Sistema.

Registro do Sistema (Registry)

O Registro do Sistema é uma base de dados de configuração do sistema local que oferece ao cliente uma visão hierárquica, composta de *chaves* e *valores*, similar à encontrada num sistema de arquivos, com diretórios e arquivos. As informações contidas no Registro são usadas pelo sistema e por aplicativos.

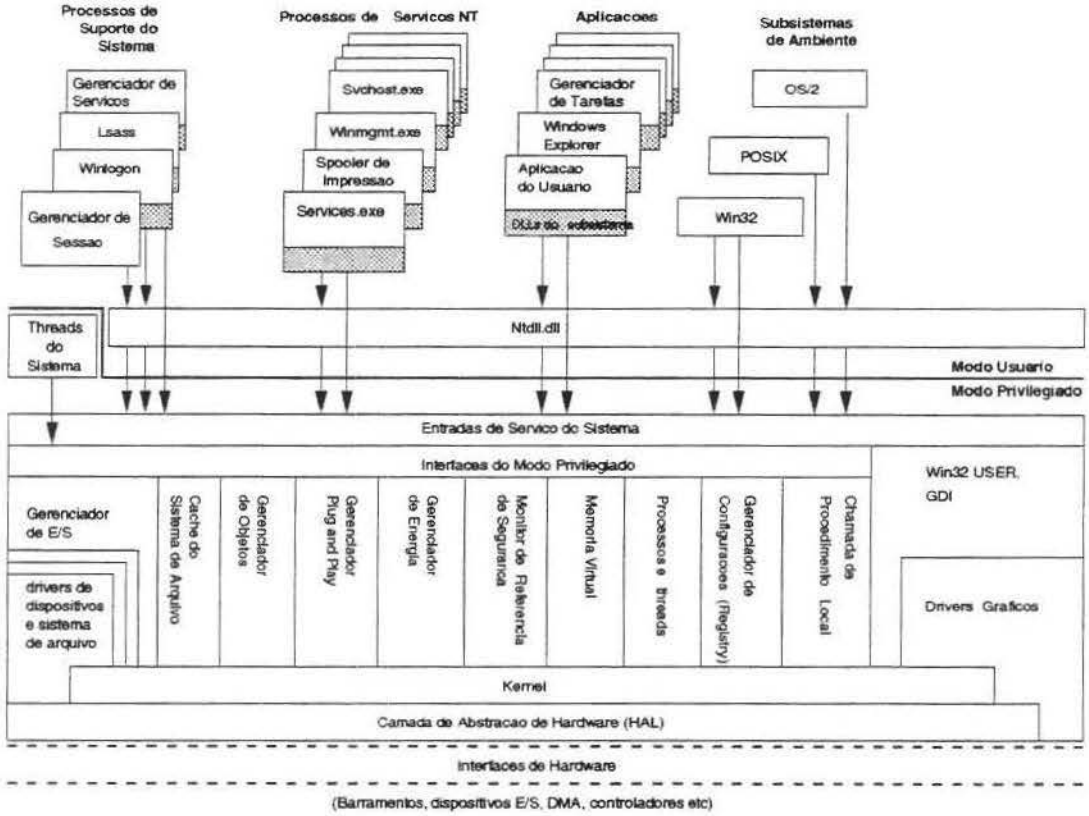


Figura 2.2: A arquitetura básica do Windows 2000

Partes deste Registro são associadas à máquina (subchave HKLM), e partes ao usuário (subchave HKCU). A grande maioria das configurações dos usuários ficam na subchave HKCU, que os acompanha a qualquer máquina da rede, e inclui principalmente configurações de aplicativos. As configurações da máquina ficam na subchave HKLM, e inclui configurações dos serviços, de dispositivos, de protocolos e a base de dados de segurança e de usuários locais.

No Windows 2000, as informações de configuração do Domínio de Rede NT (como, por exemplo, enumeração das máquinas e usuários do Domínio), que antes ficam armazenadas no Registro da máquina Controladora de Domínio, foram transportadas para o ActiveDirectory, o serviço de diretório distribuído baseado em X.500, que oferece maior escalabilidade. As informações de configuração da máquina local (incluindo os usuários existentes apenas na máquina local) permanecem armazenadas no Registro do Sistema da máquina local.

Capítulo 3

A Arquitetura de Rede

3.1 A Arquitetura de Rede no Windows NT

Uma diferença significativa entre o Windows NT e o OS/2 1.x e 2.x é que o módulo de rede no Windows NT foi construído a partir do zero. Com o DOS e OS/2, a rede foi acrescentada em cima do sistema operacional. Isto significa que os projetistas do Windows NT tiveram a oportunidade de projetar seus componentes dentro do contexto de um sistema operacional que ainda estava sendo definido.

Os objetivos originais da rede, levando em conta portabilidade e segurança, eram:

- Fornecer serviços de rede transparentes às aplicações. O compartilhamento de arquivos e impressoras deveria ser parte de cada máquina Windows NT.
- Fornecer uma plataforma para aplicações distribuídas. A comunicação entre processos (IPC, de *Interprocess Comunication*) no nível de Aplicação deveria estar disponível para o desenvolvimento de aplicações cliente/servidor.
- Fornecer uma plataforma de rede expansível para outros componentes de rede.

Como em outros componentes da arquitetura do Windows NT, a arquitetura de rede é constituída de camadas (ver figura 3.1). A arquitetura de camadas do Windows NT espelha bastante o modelo de referência OSI/ISO (*Open Systems Interconnection*) [ISO/IS 7498]. A camada de Apresentação é entre fina e não-existente, dependendo do protocolo e sistema utilizado. O modelo OSI vai ser utilizado como uma parâmetro para a descrição da arquitetura.

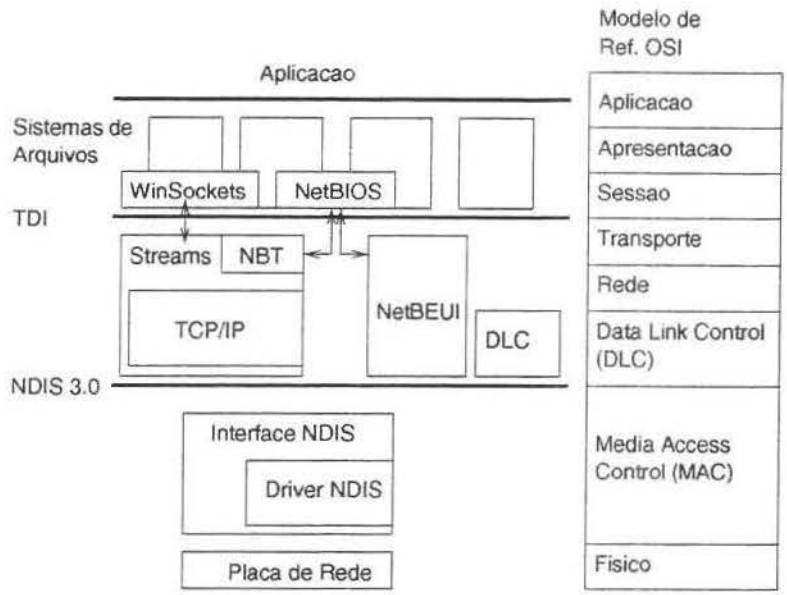


Figura 3.1: Arquitetura de rede do Windows NT e o modelo OSI

3.1.1 NDIS

A camada mais profunda é o *driver* da placa de rede. O Windows NT suporta *drivers* de dispositivos escritos para a especificação NDIS 3.0 (*Network Device Interface Specification*) . A NDIS 3.0 é baseada no padrão utilizado pelo OS/2 (NDIS 2.0). A NDIS 3.0 atende aos requisitos do Windows NT: interface em chamadas de funções C, *drivers* em código 32 bits, portável e segurança em ambientes multiprocessados. Esta interface permite que as camadas superiores sejam independentes da placa de rede utilizada.

3.1.2 Protocolos de Transporte

Acima da camada NDIS estão os drivers de protocolos de transporte. O Windows NT originalmente possuía três transportes: o NetBEUI da IBM, oferecendo compatibilidade com instalações compatíveis com os protocolos LAN Manager, LAN Server, e MS-NET; o TCP/IP (*Transmission Control Protocol/Internet Protocol*) oferecendo um protocolo popular para WANs (*Wide Area Networks*); e o DLC (*Data Link Control*) oferecendo uma interface para acesso a *mainframes* IBM e impressoras com interface de rede. Outros protocolos de transporte, como o NWLink (IPX/SPX) da Novell, o AppleTalk e *Remote Access Services* (RAS) foram incorporados posteriormente [Mic00f]. Os protocolos AppleTalk e DLC existem apenas na versão Servidor do Windows NT.

O protocolo DLC (Data Link Control)

Este protocolo da IBM não é um protocolo de transporte completo conforme as definições OSI. Sua interface de cima está na camada de ligação (*data link*). O DLC é usado para conversas rápidas, com conexão simples ou sem conexão, a algumas impressoras com interfaces de rede ou *mainframes*, como um componente do *System Network Architecture* (SNA) da IBM.

O protocolo NetBEUI e a interface NetBIOS

O NetBEUI 3.0 (*NetBIOS Extended User Interface*) é fornecido no Windows NT para manter compatibilidade com instalações LAN Manager, LAN Server e MS-NET. Este protocolo é rápido, com baixa sobrecarga (*overhead*), ou número de bytes extras, por *frame* de dados transmitido. O protocolo, no entanto, não pode ser roteado entre redes. Assim, o NetBEUI é mais apropriado para pequenas redes, de 20 a 200 máquinas.

NetBIOS vem de *Network Basic Input/Output System*, um padrão de rede desenvolvido para a IBM pela Sytek, em 1983. Este padrão definia uma interface de acesso na camada de sessão OSI/ISO (controlada através de comandos submetidos via *Network Control Blocks* (NCBs)) e um protocolo de transporte de dados chamado *NetBIOS Frames Protocol* (NBFP).

O NetBEUI é comumente refererido como o protocolo da camada de transporte, separadamente da interface de programação NetBIOS, na camada de sessão. Implementações anteriores do MS-DOS e OS/2 implementavam a interface NetBIOS como parte do *driver* de transporte. No entanto, no Windows NT a interface de programação NetBIOS foi separada do protocolo de transporte NetBEUI para incrementar a flexibilidade na arquitetura em camadas. A separação destes dois componentes permite a reutilização do mesmo código NetBIOS para transportes múltiplos como NetBEUI e TCP/IP.

Uma aplicação que usa a API da interface NetBIOS para comunicação via rede pode operar com qualquer transporte que exponha a interface NetBIOS.

Nas versões de rede do Windows 16 bits, MS-DOS e OS/2, os transportes que não implementam NBFP (por exemplo, TCP/IP e IPX/SPX) deviam expor a interface NetBIOS e possuir um meio de mapear cada comando NetBIOS para alguma sequência de seu protocolo nativo.

Já no Windows NT, os protocolos de transportes não expõem a interface NetBIOS. Ao invés disso, eles expõem a interface TDI (*Transport Driver Interface*), mais flexível. Os transportes que não incluem NBFP implementam uma camada de compatibilidade NetBIOS para resolver endereços NetBIOS aos endereços nativos do protocolo de transporte, e para mapeamento de mensagens. A camada de compatibilidade NetBIOS para o TCP/IP é chamada *NetBIOS sobre TCP/IP*, *NetBT* ou simplesmente *NBT*. A camada

de compatibilidade NetBIOS para NWLink é chamada *NetBIOS sobre IPX*, ou *NBIPX*.

O Windows NT inclui um Emulador NetBIOS para mapear comandos NetBIOS a comandos e eventos da Interface TDI. Componentes internos de rede do Windows NT usam diretamente comandos e eventos TDI, ao invés de NetBIOS, para se comunicarem com a camada de transporte [Q128233].

O protocolo TCP/IP

Este protocolo é implementado um pouco diferentemente dos protocolos anteriores. Ao invés de ser um único *driver* ligado (*bound*) diretamente ao *driver* NDIS, o TCP/IP reside dentro de uma capa (*wrapper*). Esta capa é chamada *Streams*. Chamadas ao TCP/IP devem primeiro atravessar o *driver* de Streams, cujo objetivo é facilitar o porte de pilhas de protocolos para o Windows NT.

NWLink (IPX/SPX)

O protocolo *Internetwork Packet Exchange/Sequenced Packet Exchange (IPX/SPX)* é o cerne do sistema NetWare da Novell. A implementação do protocolo IPX/SPX da Microsoft é chamado NWLink. O protocolo IPX/SPX é baseado no protocolo *Xerox Network System (XNS)* da Xerox. O XNS tinha dois componentes: o IDP (*Internet Datagram Protocol*) e SPP (*Sequenced Packet Protocol*). O IPX é baseado no IDP. O SPX é baseado no SPP. O IPX/SPX é bastante eficiente em redes locais, é mais fácil de implementar que o TCP/IP, e é roteável, permitindo a criação de subredes interconectadas. Devido ao sucesso comercial do NetWare, era o principal protocolo para redes locais no início da década de 1990.

AppleTalk

O AppleTalk é o protocolo de rede principal usado por computadores Macintosh. Permite aos usuários Macintosh também compartilharem impressoras e arquivos armazenados em Servidores Windows NT.

Remote Access Services (RAS)

O RAS não é um protocolo, mas um serviço do Windows NT que habilita conexões temporárias a sistemas que não estão em redes locais, tipicamente conexões de modem (*dial-up*) via uma linha telefônica, através dos protocolos SLIP (*Serial Line Internet Protocol*) ou PPP (*Point-to-Point Protocol*). Quando a conexão é feita, a comunicação até servidores pode ser feita através de NetBEUI, TCP/IP ou IPX/SPX, com ou sem PPTP

(*Point-to-Point Tunneling Protocol*, usado para criar um "túnel" seguro de transmissão de dados através da Internet).

3.1.3 TDI

A Interface de Transporte (TDI, de *Transport Driver Interface*) fornece uma interface comum para sistemas de arquivos e processos de gerenciamento de E/S comunicarem com os vários transportes de rede. É uma camada bastante fina.

3.2 A Arquitetura do Sistema de Rede LAN Manager

A história da Microsoft em redes de computadores começou no MS-DOS 3.1, em 1984. Esta versão do DOS adicionou à FAT extensões para *locking* de arquivos e registros, necessário para permitir a mais de um usuário acessar arquivos no DOS. Com esta versão do DOS, a Microsoft também liberou um produto chamado *Microsoft Networks*, informalmente chamado MS-NET. O MS-NET estabeleceu algumas tradições, implementadas no produto *LAN Manager* para Windows e OS/2, e utilizadas também no Windows NT.

3.2.1 O Protocolo SMB/CIFS e componentes LAN Manager

Uma destas tradições foi o protocolo *Server Message Block* (SMB). O protocolo SMB é um protocolo de nível de aplicação usado para formatar mensagens enviadas através da rede. O SMB é usado por componentes no nível de aplicação LAN Manager chamados Cliente LM e Servidor LM. A interface de sessão NetBIOS é usada para passar requisições SMB através da rede para uma máquina remota. Tanto o protocolo SMB quando a API NetBIOS são adotados em vários produtos de rede, como *LAN Manager 1.0* da IBM, *XENIX CORE* (Xenix era o antigo Unix da Microsoft) da Xenix e LANMAN 0.12 do Windows NT, dando origem a vários "dialetos SMB".

O sistema de rede do Windows NT inclui um cliente LAN Manager, ou Cliente LM (serviço *Workstation*), e um servidor LAN Manager, ou Servidor LM (serviço *Server*) que implementam uma variante do protocolo SMB totalmente compatível com os antigos produtos MS-NET e LAN Manager.

Larry Osterman, o desenvolvedor que transformou o redirecionador MS-NET no redirecionador LAN Manager 1.0 do MS-DOS e depois LAN Manager 2.0, também projetou o redirecionador do Windows NT originalmente. Para o Windows NT, ele acrescentou suporte a operações do sistema como controle de acesso.

O protocolo SMB não é completamente documentado, embora um grupo de pessoas do projeto SAMBA [SMB01] esteja conseguindo realizar avanços em sua compreensão

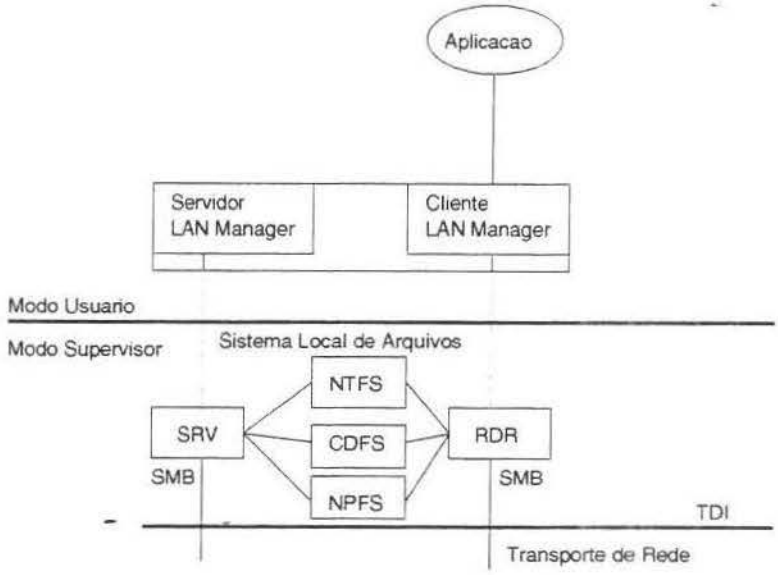


Figura 3.2: O Módulo de rede LAN Manager

utilizando engenharia reversa. Em 1997, a Microsoft emitiu uma proposta de um protocolo chamado *Common Internet File System* (CIFS) [LP96], que era basicamente um subconjunto do SMB que era adequado à Internet.

Em adição aos serviços de Cliente e Servidor do LAN Manager, um número de outros serviços LAN Manager são implementados no Windows NT: o Alertador (*Alerter*), o Mensageiro (*Messenger*), o Navegador (*Browser*), e o Replicador (*Replicator*). O Alertador é utilizado para enviar alertas gerados na máquina local para computadores ou usuários remotos. O Mensageiro recebe mensagens e alertas, e os mostra na tela na forma de um diálogo. O Navegador é usado para coletar informação sobre as máquinas LAN Manager na rede local, e fornecer estas informações quando requisitado. O Replicador permite a cópia automática de um diretório de uma máquina para outra. A fonte de dados é dita estar numa máquina de exportação, enquanto o alvo é uma máquina de importação.

Cliente LM e o Redirecionador

O módulo Cliente LM possui na verdade duas partes (ver figura 3.2). O componente Cliente LM oferece a interface em modo usuário. O outro componente é o RDR, ou redirecionador (*Redirector*), que fica na executiva. Este componente é um Módulo de Sistema de Arquivos (FSD, de *File System Driver*), que realmente faz a interação com as camadas inferiores da pilha de protocolos. Entre enviar uma requisição e receber uma resposta, o redirecionador tem uma tarefa primária: oferecer um "sistema de arquivos" que se comporta como um sistema de arquivos local embora opere sobre uma rede. O

redirecionador envia e recebe pacotes SMBs para realizar seu trabalho.

Os pacotes SMB enviados pelo redirecionador chegam eventualmente à interface TDI, que permite utilizar arbitrariamente qualquer protocolo de transporte de rede para enviar os pacotes SMB.

Servidor LM

O módulo Servidor LM é conectado por Clientes LM de outras máquinas. É composto de duas partes: o componente Servidor LM (modo usuário) e o SRV (modo supervisor). O SRV gerencia a interação com as camadas inferiores e também interage diretamente com outros sistemas de arquivos para satisfazer requisições tais como leitura e escrita de arquivos.

3.2.2 Sessões de *Logon*

O conceito de sessão de logon será usado como base para explicar sessões LAN Manager na próxima seção.

Uma *sessão de logon* é usada no Windows NT para evitar a necessidade de autenticar o usuário toda vez que este acessa um recurso. A autenticação é uma atividade que consome muitos recursos, pode demorar, e normalmente envolve comunicação via rede.

Neste modelo, quando o usuário inicialmente tenta acessar o sistema, o Windows NT usa as credenciais fornecidas e as autentica usando a autoridade de autenticação local (LSA)[1] e se encarrega de criar um ambiente (a sessão de logon) onde deposita um *token de acesso*, contendo o identificador único do usuário (SID), grupo(s) em que está contido e privilégios locais. Este *token* só pode ser manipulado diretamente pelo sistema, para evitar que o usuário o corrompa (por exemplo, adicionando o grupo Administrador a si próprio).

A partir daí, todo processo criado dentro desta sessão de logon herda uma cópia desse *token* de acesso. Quando algum processo tenta acessar um recurso, o sistema cruza as informações de identidade do *token* de acesso com as informações de permissão (ou negacão) de acesso da Lista de Controle de Acesso (ACL) do recurso, terminando por permitir (ou não) o acesso ao recurso.

A primeira sessão de logon de cada máquina é a *sessão de logon do sistema*. A partir desta sessão de logon são criadas todas as outras.

Além do *token*, o sistema pode também acrescentar *credenciais de rede* à sessão de logon. Isto dá origem a dois tipos principais de sessão de logon:

[1] Isto pode exigir a necessidade de comunicação via rede com um Controlador de Domínio NT que possui a conta do usuário armazenada.

Sessão de *Logon* Interativa Criada quando um usuário interativamente "loga" no sistema local. A característica principal deste tipo de sessão de logon é *possuir um conjunto de credenciais de rede.* As credenciais de rede incluem o nome do usuário e a senha de acesso (ou equivalente) associada. Com as credenciais de rede, é possível à sessão de logon local autenticar o usuário transparentemente em máquinas NT diferentes no Domínio[2], criando *sessões de logon de rede remotas.* Desta maneira, o usuário não precisa digitar suas credenciais cada vez que um acesso a um recurso remoto é realizado.

Sessão de *Logon de* Rede Quando uma máquina NT recebe, via rede, um pedido para criar uma sessão de logon, ela cria uma *sessão de logon de rede.* A característica principal deste tipo de sessão de logon é *não possuir um conjunto de credenciais de rede.* Neste tipo de sessão, as credenciais não ficam disponíveis de modo a evitar que servidores mal-intencionados utilizem as credenciais (que no último caso são a prova de identidade de um usuário) para se autenticarem a outras máquinas do Domínio e agirem com má fé.

3.2.3 Sessões Lan Manager

Keith, em [KB00], faz uma boa descrição do comportamento das sessões Lan Manager. Ele define três expressões, que serão utilizadas aqui para descrever o modelo:

servidor LM O servidor de arquivos LAN Manager que recebe requisições SMB

cliente LM O processo cliente LAN Manager que envia mensagens SMB para o servidor LM

sessão LM Uma sessão SMB, para diferenciar de uma *Sessão de Logon.*

O servidor LM é orientado a sessão, ou seja, um cliente deve primeiro estabelecer uma sessão[3] antes de ter acesso a um arquivo, *named pipe* ou outro recurso do sistema de arquivos[4]. Para abrir uma sessão, uma autenticação deve ser realizada primeiro. E enquanto a sessão permanecer aberta, *nenhuma outra autenticação é realizada*, a não ser que o dialeto *SMB Signing* seja negociado durante a autenticação. Como a autenticação é realizada apenas no início da sessão, o sistema fica mais rápido para o cliente, elimina a necessidade de enviar pacotes para a Autoridade de Autenticação (o Controlador de

[2]Um Domínio é um conjunto de máquinas NT numa rede que compartilha o mesmo conjunto de credenciais e de segurança - ver seção 4.1.

[3]Uma sessão pode ser estabelecida implicitamente pelo sistema para o usuário.

[4]Esses recursos do sistema de arquivos serão aqui chamados coletivamente de *arquivos.*

Domínio) a cada acesso a recursos do sistema de arquivos, e pode armazenar algum estado, como o estado do arquivo (aberto, fechado etc) e cursores para o arquivo.

Em cada máquina rodando o serviço *Server*, o servidor LM mantém uma lista de sessões LM abertas. É possível a qualquer usuário obter esta lista de qualquer servidor, e dependendo das informações requisitadas, não há necessidade de possuir qualquer tipo especial de acesso ao sistema. Durante a autenticação inicial, quando uma requisição é realizada por um cliente, o provedor de segurança (SSP, ver seção 5.5) do servidor estabelece uma *Sessão de Logon de Rede* e cria um *token* de acesso do cliente no servidor. O servidor LM pode então obter este *token* de modo a criar uma *thread de resposta que personifica o cliente de modo a lidar com a requisição*. Como o servidor LM deve ser capaz de personificar o cliente em cada requisição, mas a autenticação ocorre apenas quando a sessão LM é inicialmente estabelecida, a sessão LM deve possuir uma referência para a Sessão de Logon de Rede do cliente (ou seja, o contexto de segurança do cliente).

Além de gastar recursos, uma sessão aberta não permite que novas informações de acesso do usuário sejam utilizadas (pois a autenticação e criação do *token* de acesso ocorre apenas no início da criação da sessão). Não é bem documentado qual o tempo que uma sessão permanece aberta[5]: ocasionalmente, o servidor LM varre as sessões LM abertas e fecha aquelas que não possuem arquivos abertos e que estão inativas durante um certo tempo. Isto se torna um problema se uma particular sessão LM fica raramente inativa, porque o servidor LM não será capaz de fechar automaticamente esta sessão. É possível fechar uma sessão LM explicitamente através da função NetSessionDel, mas neste caso quaisquer arquivos abertos na sessão serão fechados e dados podem ser perdidos.

Clientes e Sessões

Múltiplas sessões LM podem existir entre uma máquina cliente (cliente LM) e uma servidora (servidor LM). Mais especificamente,

> Uma *sessão LM* é uma ligação entre uma *sessão de logon* de um cliente e um servidor LM. Uma sessão LM é estabelecida a primeira vez que um processo na *sessão de logon* do cliente faz uma conexão a um recurso compartilhado no servidor LM. Até a sessão LM terminar, todas as outras conexões entre a *sessão de logon* do cliente e o servidor LM são parte da mesma sessão LM.

Enfatiza-se bastante a expressão '*sessão de logon*', porque a sutileza da definição está encerrada nela. Várias sessões de logon (por exemplo, a sessão de logon interativa do

[5]O artigo Q138365 da Microsoft descreve alguns parâmetros que podem ser usados nesse sentido. Sessões LM *explícitas* (criadas pelo usuário) normalmente duram alguns minutos, e *implícitas* (criadas automaticamente pelo sistema) alguns segundos.

	Cliente		Servidor	
TCP/IP	Endereco	Porta	Endereco	Porta
Lan Manager (LM)	Endereco	Logon LUID	Endereco	

Figura 3.3: Comparação entre associações TCP/IP e associações Lan Manager

usuário que está no console, a sessão de logon do sistema, a sessão de logon de um serviço NT local etc) podem coexistir na mesma máquina (cliente LM). E cada sessão de logon pode possuir até *uma* sessão LM com cada servidor LM.

Uma comparação entre as associações TCP/IP com as associações LM permite estabelecer melhor a definição anterior. Cada associação TCP é uma tupla de 4 elementos (ver figura3.3). O servidor ouve em um endereço e uma porta, e muitos clientes podem se conectar de vários endereços simultaneamente. Um único cliente de um endereço pode se conectar várias vezes simultaneamente, pois possui várias portas. As sessões LM são semelhantes, mas têm 3 elementos. No cliente LM, a associação consiste do endereço do cliente e de seu número único da sessão de logon (LUID, de *Local Unique ID*). No servidor LM, a associação consiste apenas do endereço do servidor porque há apenas um único servidor ouvindo (ao contrário do TCP, onde vários servidores podem ouvir em portas diferentes).

Registro de Credenciais numa Sessão LM Explícita

Quando uma sessão LM é estabelecida entre uma *sessão de logon* do usuário cliente e um servidor LM, por default as credenciais utilizadas na autenticação (e que portanto o servidor LM irá usar para personificar sua *thread* de execução) são aquelas que existem na sessão de logon do usuário (ou seja, as credenciais do usuário). Este é o comportamento *default*. Outras credenciais podem ser usadas, se a sessão LM for *explicitamente* estabelecida pelo usuário. Por exemplo, os comandos

```
net use \\MeuServidorLM /u:usuario@dominio.com *   (UPN[6]/Windows 2000)
net use \\MeuServidorLM /u:DOMINIO\usuario *       (pré-Windows 2000)
```

[6]UPN significa *User Principal Name*, e é a convenção de nome do usuário do Windows 2000.

podem ser usados para criar uma sessão LM explícita e fornecer um conjunto de credenciais alternativo. O usuário é fornecido através da opção /u e o parâmetro * indica que uma senha diferente será fornecida. O nome DNS/NetBIOS do ServidorLM pode ser também um número IP caso NetBIOS sobre TCP/IP esteja sendo usado.

Embora, a princípio, várias sessões LM explícitas possam ser estabelecidas ao mesmo servidor LM, cada uma indicando um compartilhamento (*share*), *pipe* ou impressora, apenas uma única sessão LM pode ser estabelecida entre uma sessão de logon e o servidor LM remoto em um dado momento[7]. Assim, é redundante escrever:

```
net use \\MeuServidorLM\CompartilhamentoUm /u:usuario@dominio.com *
net use \\MeuServidorLM\CompartilhamentoDois /u:usuario@dominio.com *
```

E escrever

```
net use \\MeuServidorLM\CompartilhamentoUm /u:mgranado@dominio.com *
net use \\MeuServidorLM\CompartilhamentoDois /u:stef@dominio.com *
```

faz a segunda linha falhar com o erro *'The credentials supplied conflict with an existing set of credentials'*. O cliente LM está indicando ser possível estabelecer ao servidor LM apenas uma única sessão LM com um único conjunto de credenciais, a partir da sessão de logon do usuário.

Ao estabelecer explicitamente uma sessão LM, o cliente LM armazena as credenciais do usuário cliente durante toda a sessão de logon deste usuário, criando o *registro de credenciais* daquela específica sessão LM. Caso a sessão LM fique inativa por diversos minutos e seja fechada automaticamente, o cliente LM usará estas credenciais armazenadas para automaticamente iniciar e autenticar outra sessão LM ao mesmo servidor LM no caso de algum recurso associado à sessão ser novamente utilizado.

Para utilizar um conjunto de credenciais diferente, é necessário *explicitamente* apagar qualquer registro de credenciais da sessão LM associado ao servidor LM. Isto pode ser feito através do comando

```
net use \\MeuServidorLM /d
```

[7]Uma interessante maneira de burlar parcialmente isto é usar um tipo de endereço diferente para o mesmo servidor LM em cada conexão, por exemplo o endereço IP numérico e o endereço DNS/NetBIOS textual!

Sessões NULAS

As sessões nulas (*NULL sessions*) são estabelecidas através de registros de credenciais vazios (ou seja, com usuário vazio e senha vazia)[8]. Elas desabilitam a autenticação, o que pode ser útil em ambientes onde não se deseja a utilização de uma autoridade de autenticação central (por exemplo, redes NT sem Controladores de Domínio, onde não se deseja sincronizar manualmente as contas locais das máquinas). Uma sessão nula pode ser estabelecida pelo comando

```
net use \\MeuServidorLM /u: *
```

onde o parâmetro /u: indica um usuário vazio, e indicando uma senha vazia quando for solicitado. O nome DNS/NetBIOS do ServidorLM pode ser também um número IP caso NetBIOS sobre TCP/IP esteja sendo usado.

De agora em diante, qualquer tráfego LM da sessão de logon do usuário e o servidor LM usará esta sessão nula. Como cada recurso LM no Windows NT é protegido por uma ACL para controle de acesso, o acesso aos recursos do servidor LM somente será permitido se a ACL associada a este recurso permitir acesso a Todos (*Everyone*).

A partir da versão 3.5 do Windows NT, sessões nulas (com exceção do compartilhamento IPC$ para início da comunicação IPC) são proibidas. Elas podem ser permitidas caso a variável *RestrictNullSessionAccess* seja acrescentada na chave *HKLM|SYSTEM|CurrentControlSet|Services|LanmanServer|Parameters* e esteja presente com o valor FALSE. Alternativamente, as variáveis *NullSessionPipes* e *NullSessionShares* indicam quais compartilhamentos poder ser acessados por sessões nulas. No entanto, certas informações importantes, como o nome dos pontos de compartilhamento de arquivos da máquina, continuam acessíveis via sessão nula[9].

Mapeamento de Letras de Discos

Um outro ponto sutil envolvido é a criação de mapeamentos de letras de discos na sessão do usuário. Quando uma letra de disco associada a um recurso LM é criada, ela é *vista por toda a máquina*, mesmo por sessões de logon diferentes. O que é importante frisar, é que embora quaisquer sessões de logon subsequentes (por exemplo, usuários com sessões de logon interativas posteriores) vejam o mesmo conjunto de letras de discos associados aos recursos remotos LM, cada sessão de logon possui seu próprio conjunto de sessões

[8] Uma sessão nula é uma sessão de logon anônimo associada a um usuário não-autenticado (anônimo).

[9] Isto permite a qualquer um, remotamente, listar todos os diretórios compartilhados de uma máquina Windows NT ligada à Internet via *dial-up*, por exemplo. É necessário saber apenas o IP atual da máquina, criar uma sessão nula com este IP, e usar o comando *net view* *IP.da.maquina*.

LM, cada sessão LM com sua credencial de acesso corretamente derivada das credenciais do usuário autenticado no momento.

3.3 Mecanismos de IPC no Windows NT

No Windows NT, há seis principais mecanismos de comunicação entre processos (IPC), utilizados para trocar informações entre um processo cliente e um servidor: *named pipes*, *mailslots*, *NetBIOS*, *Windows Sockets*, *Chamadas de Procedimento Remotos* (RPC, de *Remote Procedure Calls*) e *NetDDE* (*Network Dynamic Data Exchange*). *Named pipes*, *mailslots* e NetBIOS oferecem compatibilidade com sistemas e aplicações legadas baseadas em LAN Manager.

3.3.1 Named Pipes

Named pipes são entidades LAN Manager/SMB que oferecem um conduíte de dados persistente entre processos, com envio garantido de mensagens, baseada na API do OS/2, útil para serviços distribuídos. Pode ou não incluir comunicação via rede.

O serviço oferecido por um servidor de *named pipes* é visível como se fosse simplesmente um arquivo. Como *named pipes* são implementadas como um sistema de arquivos padrão (NPFS, de *Named Pipes File System*), elas podem tirar vantagem do gerenciador de *cache* e do controle de acesso nativo do Windows NT. Toda comunicação passa através do cliente LM e servidor LM usando sessões LM (tipicamente autenticadas).

No Windows NT, quando um cliente inicia explicitamente uma sessão LM, o conjunto de credenciais do cliente é usado para iniciar a sessão LM entre a sessão de logon do cliente e o servidor LM. Todos os arquivos, impressoras e *pipes* deste servidor LM são acessados usando estas mesmas credenciais. As credenciais podem se tornar difíceis de gerenciar, pois são recursos globais na sessão de logon do cliente, e processos executando na sessão de logon podem começar a gerar conflitos na hora de criar, modificar e apagar as credenciais.

Uma outra característica da implementação do Windows NT é a *personificação* (*impersonation*). Na personificação, o servidor pode mudar sua identidade de segurança para aquela do cliente requisitando o serviço, de modo que mesmo que o programa servidor tenha autoridade para realizar a função, o cliente pode não ter, e a requisição é negada.

O SQL Server, por exemplo, usa *named pipes* para transportar *streams* de comandos e dados SQL.

3.3.2 Mailslot

O mecanismo *mailslot* do LAN Manager fornece um serviço de mensagens sem conexão, basicamente *broadcast* de mensagens. O envio da mensagem não é garantido. É útil para descobrir outras máquinas e serviços na rede, ou para notificação em grande escala de um serviço. O uso de *mailslots* deve ser contido tanto quanto possível. Cada *mailslot* transmitido é recebido por cada máquina na rede local e processado até o grau que determina se a mensagem deve ser recebida ou não. Isto pode provocar problemas de performance em todas as máquinas da rede local. Em adição, aplicações projetadas para usar *mailslots* são normalmente limitadas a implementações para a rede local, pois a maioria dos roteadores não transmitem mensagens de *broadcast*.

3.3.3 NetBIOS

A interface NetBIOS é usada como um mecanismo IPC desde a introdução da interface, no início da década de 1980. Como uma nota para programação, no entanto, interfaces de nível mais alto tais como *named pipes* ou RPC são superiores em flexibilidade e portabilidade. A NetBIOS define um ponto de interface comum para múltiplos possíveis provedores de protocolos de transporte.

3.3.4 Windows Sockets

O *Windows Sockets* é uma implementação da amplamente usada interface de programação *Sockets* criada pela *University of California at Berkeley*.

Este mecanismo habilita uma aplicação distribuída a acessar protocolos, tais como TCP/IP ou IPX. O Windows Sockets (WinSock) pode ser usado para construir um canal de comunicação confiável bidirecional entre um cliente e um servidor.

3.3.5 NetDDE

O protocolo *Dynamic Data Exchange* (DDE) é um mecanismo IPC cliente/servidor que usa filas de mensagens das janelas do sistema. O NetworkDDE (NetDDE) permite o compartilhamento de informação entre aplicações distribuídas, usando APIs do NetBIOS para se comunicar com os componentes de rede.

3.3.6 MSRPC

RPC significa Chamada de Procedimento Remoto (*Remote Procedure Call*). Muito do trabalho de RPC original foi iniciado pela empresa de computadores SUN e levado adiante

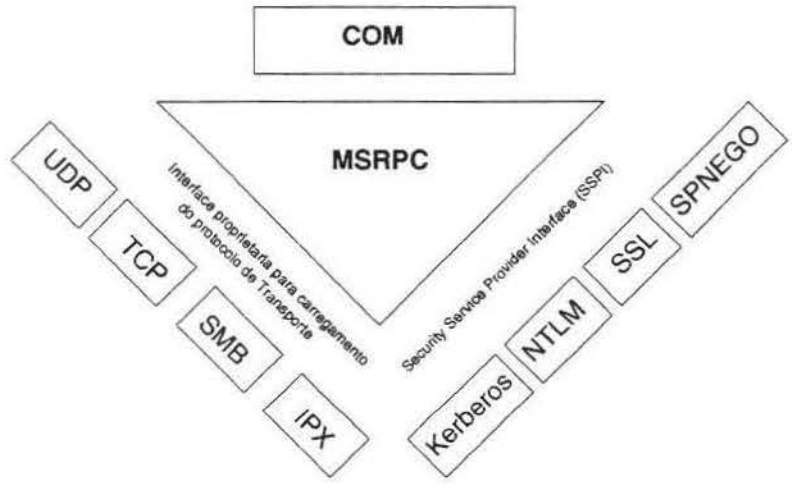


Figura 3.4: Protocolos de transporte e autenticação que o MSRPC pode utilizar

pela *Open Software Foundation* (*OSF*) como parte de seu *Ambiente de Computação Distribuída* (DCE, de *Distributed Computing Environment*). A implementação original da Microsoft (MSRPC) é baseada na especificação da OSF (DCE/RPC), o que faz sentido, já que Paul Leach, um dos co-fundadores do DCE/RPC, foi trabalhar para a Microsoft.

O mecanismo RPC faz chamadas em um computador local ou remoto como se a chamada fosse local; gerencia e agrupa um conjunto de funções semelhantes, permitindo controle de versões; autentica o usuário antes do uso do conjunto de funções; e criptografa os parâmetros da função transparentemente.

Ele é único devido ao fato de usar outros mecanismos IPC para estabelecer comunicações entre o cliente e o servidor. O RPC pode usar *named pipes,* NetBIOS, ou *Windows Sockets* para se comunicar com sistemas remotos. Se o cliente e o servidor estão na mesma máquina, o RPC pode usar o sistema de *Chamada de Procedimentos Locais* (LPC, de *Local Procedure Call*) para transferir informação entre processos e sistemas. Isto faz do RPC o mais flexível e portável mecanismo entre os IPCs disponíveis.

No Windows NT, os mecanismos de RPC beneficiam-se do fato de estarem disassociados de qualquer protocolo de transporte e de autenticação (embora RPC sobre o protocolo SMB seja muito utilizado em ferramentas administrativas do Windows NT - ver seção 3.3.7). Assim, o cliente e o servidor simplesmente concordam em utilizar algum protocolo em particular, e bibliotecas dinâmicas (DLLs) do protocolo de transporte escolhido (TCP, UDP, IPX, SPX, SMB, AppleTalk etc) e do protocolo de autenticação desejado (NTLM, Kerberos, SSL, SPNEGO - ver capítulo 5) serão automaticamente carregadas e utilizadas (figura 3.4).

Como uma nota, a tecnologia DCOM (*Distributed COM*) [Mic00b], disponível a partir do Windows NT 4.0, utiliza RPC para comunicação distribuída entre objetos COM

(*Common Object Model*) da Microsoft na rede, muito utilizada para desenvolvimento de programas no Windows NT. Uma descrição de arquitetura e segurança aprofundada envolvendo RPC e COM pode ser encontrada em [KB00].

3.3.7 MSRPC sobre SMB

O protocolo SMB do Windows NT não é bem documentado pela Microsoft. Um outro protocolo não documentado pela Microsoft é a comunicação MSRPC sobre o protocolo SMB, utilizada pelo Windows NT para autenticação e gerenciamento de segurança de suas máquinas, e também por várias ferramentas administrativas que acessam recursos NT remotos (Monitor de Performance, Gerenciador de Servidores para Domínios, *Event Log* etc). Certas pessoas vêm realizando um interessante trabalho de engenharia reversa deste protocolo (ver, por exemplo, o projeto *Domínios NT para UNIX* [KL00]). A comunicação RPC sobre o protocolo SMB é realizada através de *named pipes* SMB especializadas em autenticar usuários (usando NTLM, Kerberos etc) e gerenciar componentes específicos do sistema (sistema de autenticação LSA, base de dados de usuários SAM, Serviços NT etc). A falta de documentação esconde falhas de segurança do RPC sobre SMB. As principais são aqui citadas.

Várias outras *named pipes* existem para realizar MSRPC sobre SMB: por exemplo, \PIPE\EVENTLOG (para acesso remoto ao serviço de eventos (*logging*) do sistema), \PIPE\svcctl (gerenciamento remoto de Serviços NT), \PIPE\winreg (para acesso remoto ao *registry*) e \PIPE\srvsvc (para gerenciamento remoto do servidor LAN Manager das máquinas NT).

\PIPE\lsarpc

Realiza o acesso ao módulo de autenticação LSA. Oferece: resolução de nomes de domínios para Identificadores únicos de Segurança (SIDs); transformações de SIDs para nomes de usuários e membros do domínio; SID do domínio onde o servidor é um membro; SID da base de dados SAM que o servidor é responsável.

Os riscos de segurança incluem a possibilidade de acesso anônimo (sessões nulas), permitindo acesso a informações demais, como:

- SID e papel do servidor, permitindo a um usuário anônimo descobrir se o sevidor é um Controlador de Domínio, Membro do Domínio, ou uma máquina cliente (*Workstation*). Obter o SID correspondente à máquina é o primeiro passo de um ataque utilizando esta *pipe*.

- Enumeração de uma lista de nomes do Domínio e seus tipos. Depois que um usuário anônimo sabe o SID da máquina alvo, ele pode começar a tentar adivinhar SIDs

para contas naquele Domínio. Os SIDs iniciam-se em um número bem conhecido (0x3fe) e aumentam de valor deterministicamente. Deste jeito, respostas podem indicar não apenas se o SID está em uso, mas o nome e seu tipo: Usuário, Grupo ou Alias. Uma máquina cliente ou uma Conta Confiável (*Trusted Account*) pode ser identificada por um $ no fim do nome.

- Uma lista dos Domínios confiados. Um usuário pode anonimamente obter uma lista dos outros Domínios que são alvos potenciais. Isto é feito na tela de login do Windows NT *antes* do usuário ter "logado", o que provavelmente explica por que esta informação está disponível de maneira anônima.

\PIPE\NETLOGON

Oferece, em Controladores de Domínio, serviços de autenticação NT para usuários, e é também utilizada por uma máquina cliente (*Workstation*) para autenticar terceiros que a contactam. Esta última categoria é conhecida como *autenticação pass-through*, onde a máquina cliente passa a responsabilidade da autenticação para o Controlador de Domínio.

A API para acesso a esta *pipe* não é documentada. A razão disto é que esta API é interna, usada pela LSA para autenticação. Se a autenticação puder ser feita localmente, então a SAM é chamada diretamente pela LSA. Se a autenticação necessitar um Controlador de Domínio, então as funções da *pipe* NETLOGON do Controlador são utilizadas.

As informações de autenticação enviadas para esta *pipe* são protegidas através de um segredo compartilhado (uma chave criptográfica) entre o Controlador de Domínio e a máquina cliente. Esta proteção é chamada Cadeia de Credencial (*Credential Chain*), e o segredo é a senha da Conta Confiável da máquina cliente, armazenada tanto na base de dados SAM local quanto no Controlador de Domínio. A tarefa administrativa de apresentar (acrescentar) uma máquina cliente ao Domínio cria uma Conta Confiável no Controlador de Domínio e a Cadeia de Credencial entre estas duas máquinas é inicializada, permitindo a comunicação segura entre elas.

Os riscos de segurança incluem várias falhas no projeto, existentes principalmente associadas ao inseguro protocolo NTLM usado para autenticar as trocas de mensagens SMB:

- Utilização de sessão nula (anônima) para início da comunicação .

- Valor inicial bem conhecido para as senhas iniciais das Contas Confiáveis das máquinas clientes (*Workstation*) do Domínio: *a senha inicial é o nome da máquina cliente em letras minúsculas*. Assim, se alguém estiver observando a rede no nomento em que uma máquina cliente é adicionada ao Domínio, é possível entender e rastrear todas as trocas seguras subsequentes que compõem a Cadeia de Credenciais desta

máquina, roubar a senha (ou equivalente) do usuário e de todas as futuras senhas da Conta Confiável da máquina cliente e até subverter o processo de autenticação entre esta máquina cliente e o Controlador de Domínio.

- A função NetrSamLogon desta *pipe* responde com códigos de erros descritivos demais, como "usuário não existe", "senha errada", "senha expirou" e outros.

- O método de autenticação NTLM é baseado no envio de um desafio aleatório do servidor ao cliente, que é usado para criptografar a senha (ou equivalente) do cliente ao servidor. Há uma variação no método de *logon* (*Sessão de Logon Interativa* - ver seção 3.2.3) que criptografa a senha/*hash* do usuário usando um desafio NTLM derivado da senha da Conta Confiável da máquina cliente, que já foi visto ter um valor inicial bem determinado. Se alguém possui a senha da Conta Confiável da máquina, então é possível facilmente descriptografar o equivalente da senha de todos os usuários que "logam" nesta máquina.

- Até o Windows NT 4.0 com o Service Pack 1, notou-se que na criação de *Sessões de Logon de Rede* (ver seção 3.2.3), os primeiros 8 bytes do equivalente Lan Manager (*hash LM*) da senha do usuário eram devolvidos em claro. Isto era particularmente danoso porque as chaves das sessões MSRPC criptografadas eram baseadas justamente nestes 8 bytes transmitidos em claro (!).

- Se a assinatura de pacotes SMB (*SMB Signing*) não estiver sendo usada, é possível alterar o conteúdo dos pacotes SMB com autenticação NTLM, permitindo, por exemplo, acrescentar o grupo Administrador ao pacote de autenticação que o usuário recebeu ao "logar" numa máquina cliente.

\PIPE\samr

Acessa o gerenciamento da base de dados de usuários SAM. As funções conhecidas podem ser usadas para adicionar, modificar e remover usuários, grupos e alias. O projeto da API permite a manipulação de qualquer base de dados SAM, local ou remota, desde que nomeada por seu SID. Esta API também não é documentada, mas as poucas funções conhecidas possuem falhas de segurança bem conhecidas, associadas ao acesso anônimo inicial usando sessões nulas:

- Verificação dos usuários e grupos válidos numa base de dados SAM. O identificador do usuário inicia num número bem definido e aumenta deterministicamente. É possível obter várias informações cadastrais relacionadas a cada usuário válido.

- Resolução dos nomes dos grupos para identificadores e vice-versa.

- Enumeração de todas as entradas na base de dados SAM (*Red Button bug*). A solução da Microsoft a isto foi acrescentar uma entrada RestrictAnonymous no *registry*. No Windows NT 4.0, esta solução não evita a utilização de métodos equivalentes disponíveis em \PIPE\lsarpc.

Capítulo 4

Modelo de Segurança do Windows NT

O Windows NT oferece um mecanismo de segurança unificado que pode ser usado para proteger os recursos do usuário de acessos não autorizados. Este capítulo introduz os princípios de segurança NT, destacando os elementos chaves do modelo de segurança e algumas armadilhas da implementação.

O *Modelo de Segurança* entre as diversas versões do Windows NT é basicamente o mesmo. Ele é baseado na existência de uma entidade unificada de segurança, chamada *Domínio*, que define a *política* de segurança, e em componentes de segurança internos do Módulo Executivo do sistema, que definem os *mecanismos* de segurança para proteção do acesso e fazem cumprir a política de segurança do Domínio.

Um dos objetivos do Modelo de Segurança do Windows NT é torná-lo pelo menos um sistema que oferece o nível de segurança C2 (Proteção por Acesso Controlado) do Departamento de Defesa dos EUA (ver seção 2.2).

4.1 O Domínio de Segurança

Um Domínio é uma organização lógica que contém todos os *usuários*, *grupos de usuários*, *máquinas*, *recursos* e *políticas de acesso aos recursos do sistema*. Todas as máquinas e usuários cadastrados no Domínio desfrutam de uma credencial de acesso unificada (*single sign-on*) que pode ser utilizada para acessar recursos localizados em qualquer lugar do Domínio. Isto permite a um usuário do Domínio, por exemplo, ser reconhecido em qualquer máquina, desde que ela pertença também ao Domínio. A política, entre outras coisas, indica quais usuários podem acessar quais recursos, os horários permitidos de acesso, e a quais grupos os usuários pertencem (administradores, usuários, convidados etc). Vários Domínios podem ser ligados entre si através de Laços de Confiança (*Trust Relationships*), de forma a permitir o acesso de recursos de um Domínio a partir de outro.

A base de dados do Domínio é armazenada nos Controladores de Domínio (DC, de *Domain Controllers*).

No ambiente de rede do Windows NT 4.0 e versões anteriores, havia um único *Controlador Primário do Domínio* (PDC, de *Primary Domain Controller*) e um ou mais *Controladores Secundários do Domínio* (BDC, de *Backup Domain Controller*). Os BDCs são passivos: permanecem na rede e obtêm replicações de quaisquer modificações feitas ao PDC - a única entidade ativa capaz de efetivar modificações na base de dados do Domínio. Caso o PDC se torne incomunicável, os BDCs podem responder como se fossem um PDC, permitindo operação continuada do Domínio. No Windows NT 4.0, o Domínio é um espaço de nomes plano.

No Windows 2000 foram introduzidas extensões ao modelo de Domínio do Windows NT 4.0. Estas extensões se baseiam fortemente na utilização de uma base de dados de usuários em um novo *Serviço de Diretório* (*ActiveDirectory Service*, no jargão do Windows 2000), baseado no modelo de informação X.500 e no protocolo LDAP (*Lightweight Directory Access Protocol*). Devido à utilização do Serviço de Diretório, agora todos os DCs são ativos (ou seja, podem escrever na base de dados do Domínio), e não existe mais diferença entre PDCs e BDCs. O novo Domínio usa o DNS (*Internet Domain Name System*) como seu servidor de nomes, e permite que Domínios múltiplos sejam conectados a uma estrutura de árvore.

4.1.1 Credenciais do Cliente

A credencial do cliente se refere ao conjunto de informações relacionadas à identificação do usuário e à senha, mantido por um servidor para um cliente. Esta informação é quase sempre necessária, em adição a um *Token* de Personificação, para acessar recursos remotos da rede.

Autoridade de Autenticação Toda vez que um usuário é autenticado, três informações são necessárias. O Domínio ou *Autoridade de Autenticação* (*Authentication Authority*, ou *Granting Authority*), o nome do usuário, e sua senha. Se a Autoridade de Autenticação não é explicitamente declarada, o sistema a considera como sendo a Autoridade Local (máquina local).

Credenciais NTLM O protocolo de autenticação NTLM é usado por clientes Windows NT para se conectarem a servidores com Windows NT 4.0 ou anterior. As credenciais NTLM consistem do nome do Domínio, nome do usuário, e senha criptografada que é entrada durante o logon inicial ao Windows NT (ver capítulo 6).

Credenciais Kerberos O protocolo de autenticação primário para o Windows 2000 é a autenticação Kerberos. Credenciais Kerberos consistem do nome do Domínio

(*Realm*, na nomenclatura Kerberos) e do usuário (que pode estar na forma de nomes amigáveis, tais como endereços de emails), e uma chave criptográfica que depende do tipo de criptografia utilizada. O usual é uma chave derivada de uma senha digitada pelo usuário. Quando o usuário entra no sistema, o Windows NT obtém um ou mais *tickets* Kerberos para se conectar a serviços da rede. Os *tickets* Kerberos representam as credenciais de rede do usuário na autenticação baseada no Kerberos (ver capítulo 8).

Credenciais Chaves Públicas/Privadas As credenciais na forma de chaves públicas/privadas e certificados são gerenciadas pelo usuário. O *Serviço de Diretório* do Windows 2000 é usado para publicar certificados de chave pública para usuários e protocolos de acesso ao diretório são usados para localizá-las. As chaves privadas e certificados emitidos para usuários finais são mantidos protegidos em locais seguros de armazenamento - ou no sistema local ou em *smartcards*. A implementação destes locais seguros de armazenamentos, conhecidos como *Wallets*, é baseada na arquitetura CryptoAPI da Microsoft para o Windows NT. A CryptoAPI fornece funcionalidades criptográficas mínimas e de gerenciamento de chaves para criar um local de armazenamento seguro. A implementação de protocolos seguros baseados em chaves públicas do Windows NT (ver seção 5) usa chaves e certificados armazenados na *Wallet* como credenciais do usuário para acessar servidores Internet.

4.2 O Mecanismo de Segurança

Internamente, o Windows NT representa vários recursos do sistema como objetos. Entre estes objetos encontram-se arquivos locais e remotos em NTFS, *mailslots*, *named pipes*, *buffers* do console, processos, *threads*, *tokens* de acesso, objetos de gerenciamento de janelas, chaves do *registry*, serviços locais e remotos, pontos de compartilhamento de rede, semáforos, eventos, mutexes e temporizadores.

O Windows NT tem duas formas de controlar o acesso aos objetos protegidos (*securable objects*). A primeira forma - *Controle de Acesso Discreto* - é o mecanismo de proteção onde os proprietários dos objetos permitem ou negam acesso a outros usuários ou grupos de usários. Este mecanismo é implementado através do uso de *Listas de Controle de Acesso* (ACLs, de *Access Control Lists*), associadas a cada objeto.

A segunda forma é o *Controle de Acesso Privilegiado*. Este método é usado para garantir que certos indivíduos privilegiados possam acessar os objetos protegidos se o proprietário destes não está disponível. Este mecanismo é implementado através do uso de *Privilégios* (*user rights*), que basicamente são informações de permissão especial de acesso associadas ao usuário privilegiado. Existem vários Privilégios, e alguns permitem

ignorar a existência das ACLs durante o acesso a um objeto.

4.2.1 O Modelo Básico

Para assegurar a unicidade da identidade dos usuários e dos grupos de usuários no sistema, o Windows NT emprega um *Identificador de Segurança* (SID, de *Security ID*) para referenciá-los. O SID é único, formado por uma combinação do identificador do Domínio e do usuário, e nunca é reutilizado, mesmo se o usuário ou o grupo associado é apagado.

O mecanismo de segurança básico do Windows NT é implementado através do seguinte modelo:

> Cada processo ou *thread* possui um *token de acesso* e cada *objeto protegido* (incluindo processos e *threads)* possui um *descritor de segurança.*

***Token* de Acesso** é associado a um usuário. Inclui todas as informações relevantes para identificar o usuário (SID do usuário, grupos a que o usuário pertence, Privilégios - ver seção 4.2.5 - e permissões de controle de acesso *default* para aplicar aos objetos que a *thread* do usuário cria). Há uma distinção entre um *Token Primário*, aquele que identifica o contexto de segurança de uma *thread*, de um *Token de Personificação*, que a *thread* usa para temporariamente adotar um contexto de segurança diferente, usualmente de outro usuário - muito comum em *threads* de servidores que personificam clientes.

Descritor de Segurança especifica quais usuários têm o privilégio de acessar um objeto protegido, e quais ações são permitidas no objeto. Isto é feito através da indicação do SID do proprietário do objeto e de ACLs que permitem (ou negam) o acesso do proprietário e de outros usuários e grupos. A figura 4.1 mostra esquematicamente o conteúdo de um SD. A DACL e SACL são tipos de ACLs. Uma ACE é o elemento básico de uma ACL para permitir ou negar o acesso de um usuário ao objeto. Estas entidades serão discutidas nos Conceitos Avançados (seção 4.2.3).

Quando uma *thread* é criada, o sistema lhe dá um *token* de acesso. Toda criação ou utilização de um objeto do sistema exige a especificação das ações desejadas por parte desta *thread.* O sistema, através dos componentes OM e SRM (ver seção 2.3), verifica o descritor de segurança do objeto acessado para ver se as ações requisitadas são permitidas. Neste caso, um identificador (*handle*) para o objeto é fornecido. Quando a *thread* tenta realizar uma ação no objeto, o sistema verifica através das permissões anteriormente associadas ao identificador se a *thread* realmente possui permissão de realizar essa particular ação (ver figura4.6).

Figura 4.1: Descritor de Segurança de um objeto protegido

4.2.2 Componentes do Mecanismo de Segurança

O Windows NT possui vários componentes e base de dados que implementam a segurança:

Monitor de Referência de Segurança (SRM, de *Security Reference Monitor*) Um componente na executiva do Windows NT (NTOSKRNL.EXE) que é responsável por realizar verificações de acesso de segurança nos objetos, manipulando privilégios e gerando qualquer mensagem de auditoria resultante. Este componente faz cumprir a política de validação de acesso e de geração de auditoria definida pela LSA.

Autoridade de Segurança Local (LSA, de *Local Security Authority*) Um processo não-privilegiado (LSASS.EXE) que é responsável pela política de segurança do sistema local (quais usuários são permitidos "logar" na máquina, políticas de senhas como, por exemplo, tamanho mínimo, a lista dos privilégios fornecidos aos usuários e grupos e as configurações de auditoria de segurança do sistema), autenticação do usuário, envio das mensagens de auditoria de segurança à Lista de Mensagens do Sistema (*System Event Log*) e geração de *tokens* de acesso. A LSA pode ser acessada através da *named pipe* \PIPE\lsarpc (ver seção 3.3.7).

Base de Dados de política da LSA Uma base de dados que contém as configurações da política de segurança do sistema. Esta base de dados está armazenada no *registry*. Inclui informação tais como quais Domínios são confiáveis para autenticar tentativas de *logon*, quem tem permissão de acessar o sistema e como isto é realizado (interativamente, remotamente ou através de serviços), a quem estão associados quais privilégios e que tipo de auditoria de segurança deve ser realizada.

Gerenciador de Contas de Segurança (SAM, de Security Accounts Manager) Um conjunto de rotinas na biblioteca SAMSRV.DLL responsáveis por gerenciar a

base de dados que contém os nomes de usuários e grupos definidos na máquina local ou para um Domínio NT (neste caso, se o sistema local é um Controlador de Domínio NT). O SAM executa no contexto do processo LSASS e é acessado através da *named pipe* \PIPE\samr (ver seção 3.3.7). No Windows 2000, o SAM é uma interface de acesso que pode acessar tanto o SAM de máquinas que não são Controladoras de Domínio, quanto o novo *Serviço de Diretórios X.500 do Windows 2000* (*ActiveDirectory*) em Controladores de Domínio Windows 2000.

Base de Dados SAM Uma base de dados que contém os usuários e grupos definidos, junto com suas senhas e outros atributos, para acesso ao sistema. Se a base de dados SAM está localizada em um Controlador de Domínio, então ela se aplica a todo o Domínio. Se não, ela aplica-se apenas à máquina local. Nos Controladores de Domínio do Windows 2000, o novo local de armazenamento distribuído da base de dados do Domínio é um *Repositório de Diretórios X.500* (*ActiveDirectory*), e em máquinas que não são Controladoras de Domínio, a base de dados SAM continua a existir.

Pacote de Autenticação Default No Windows NT 4.0 e anteriores, uma biblioteca cliente chamada MSV1_0.DLL, executando no contexto do processo LSASS, implementa a autenticação default (protocolo NTLM) no Windows NT, utilizando para isso a *named pipe* \PIPE\NETLOGON (ver seção 3.3.7). Esta biblioteca é responsável por verificar se o nome e senha fornecidos são iguais aos especificados na base de dados SAM, e se são, retornar a informação sobre o usuário. No Windows 2000, a autenticação default utiliza os serviços do *Centro de Distribuição de Chaves* (KDC, de *Key Distribution Center*) do protocolo de autenticação Kerberos, que compara as informações de *logon* fornecidas pelo usuário com as existentes na nova base de dados distribuída (*ActiveDirectory*), baseada no Serviço de Diretórios X.500. Este serviço pode ser acessado através da interface SAM (para compatibilidade), e de protocolos como LDAP (*Lightweigh Directory Access Protocol*) [MS00a] e Microsoft MAPI (*Messaging API*). Outros pacotes de terceiros podem também ser implementados, como o serviço de rede Novell NetWare.

Processo de *Logon* Um processo não-privilegiado (WINLOGON.EXE) que é responsável por capturar o identificador de *logon* único (nome do usuário) e uma senha (usualmente digitada através do teclado), enviá-los à LSA para verificação, e criar o processo inicial na sessão do usuário.

Serviço de *Logon* de Rede Um serviço não-privilegiado dentro do processo SERVICES. EXE que responde às requisições de *logon* de rede. A autenticação é lidada como os *logons* locais, enviando-a ao processo LSASS para verificação.

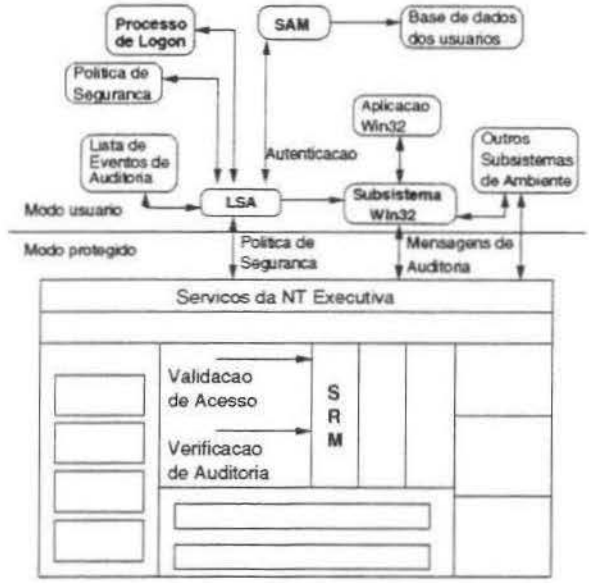


Figura 4.2: Os componentes de Segurança do Windows NT

Um esquema destes componentes pode ser visto na figura 4.2. Juntos, estes componentes são conhecidos como o *Subsistema de Segurança* (*Security Subsystem*). Como ele afeta todo o sistema Windows NT, ele também é considerado um subsistema integral, ao invés de um subsistema do ambiente.

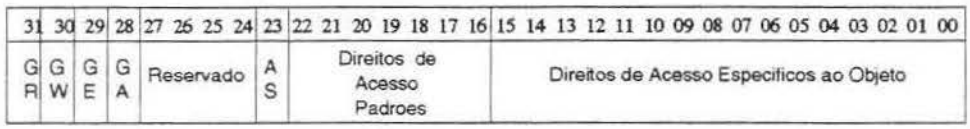
4.2.3 Aspectos Avançados do Mecanismo de Segurança

Máscaras de acesso e direitos de acesso

Um *direito de acesso* (*access right*) é uma *flag* de um bit que corresponde a um particular conjunto de operações que uma *thread* pode realizar em um objeto protegido: ler, escrever etc. Se uma thread tenta realizar uma determinada operação em um objeto mas não tem o necessário direito de acesso ao objeto, o sistema não permite a operação.

Uma *máscara de acesso* (*access mask*) é o veículo para requisitar direitos de acesso. É um valor de 32 bits cujos bits correspondem aos direitos de acesso suportados por um objeto protegido. Ela especifica *Direitos Genéricos* (*Generic Rights*), *Direitos Padrões* (*Standard Rights*), *Direitos Específicos* (*Specific Rights*), *e Direito de acesso* à Lista de Controle de Acesso ao Sistema (SACL, de *System Access Control List*).

Todos os objetos protegidos usam o formato de máscara de acesso do Windows NT mostrado na figura 4.3. Neste formato, a máscara de 32 bits é dividida entre os direitos específicos, padrões e genéricos de acesso aos objetos. Um bit (AS) corresponde ao direito de acessar a SACL do objeto. O sistema permite o direito de acessar a SACL somente se o privilégio SE_SECURITY_NAME (ver seção 4.2.5) está habilitado no *token* de acesso



GR Leitura Generica (Generic Read)
GW Escrita Generica (Generic Write)
GE Execucao Generica (Generic Execute)
GA Generic ALL
AS Direito de acessar o SACL

Figura 4.3: Formato da Máscara de Acesso do Windows NT

da *thread* que está requisitando o pedido.

Os *Direitos Genéricos* (ler, escrever e executar) podem ser aplicados a qualquer objeto do sistema. Internamente, direitos genéricos são mapeados para um conjunto de direitos de acesso padrões e específicos, que dependem do objeto. Por exemplo um arquivo mapeia o bit GENERIC_READ para os direitos padrões READ_CONTROL e SYNCHRONIZE, e para os direitos específicos FILE_READ_DATA, FILE_READ_EA e FILE_READ_ATTRIBUTES.

Cada tipo de objeto protegido tem um conjunto de *Direitos Específicos* de acesso que corresponde a operações específicas àquele tipo de objeto. Eles são únicos a cada tipo de objeto e são definidos e feitos cumprir pelo objeto.

Em adição a estes direitos de acesso específicos, há um conjunto de direitos de acesso padrões que corresponde a operações comuns à maioria dos tipos de objetos protegidos.

Os *Direitos Padrões* geralmente são usados para acessar informações de processo tais como o cabeçalho de segurança na lista de controle de acesso e estado de sincronização; também são usados para permitir ou prevenir o pai do objeto de ser apagado.

Entidades de Controle de Acesso (ACE)

Uma *Entidade de Controle de Acesso* (ACE, de *Access Control Entity*) é a estrutura mais básica no controle de acesso a um objeto. Uma ACE define a ação que pode ou não ser realizada em um objeto, ou se esta ação deve ser guardada na lista de eventos de segurança do sistema (*security log*). Ela consiste de um tipo (*Permitido* ou *Negado*), um SID que especifica o usuário ou grupo tendo a permissão liberada ou negada e uma máscara de acesso.

Listas de Controle de Acesso

Uma ACE é tipicamente um membro de uma *Lista de Controle de Acesso* (ACL, de *Access Control List*), como pode ser visto na figura 4.4.

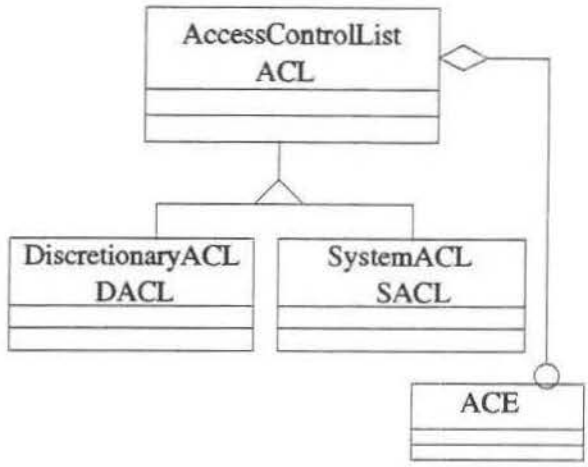


Figura 4.4: O Modelo da Lista de Controle de Acesso (ACL)

Há dois tipos de ACLs: uma *Lista de Controle de Acesso Discretivo* (DACL, de *Discretionary Access Control List*) que governa permissões para ações em objetos, e uma *Lista de Controle de Acesso à Auditoria do Sistema* (SACL, de *System Access Control List*) para governar o armazenamento (*logging*) de atividades de segurança.

DACL especifica uma lista de usuários e grupos e seus direitos associados a um particular objeto. Por exemplo, o usuário X requisita um handle com permissão para operações de leitura e escrita ao arquivo Y. O sistema percorre o DACL pertencente ao arquivo Y, uma entrada (ACE) por vez. Ele procura por qualquer entrada que diga que o usuário X (ou qualquer grupo que o usuário X seja membro) possa realizar as operações requisitadas no objeto. Se ele falha para encontrar tal entrada, ou encontra uma entrada especificando que o usuário X (ou qualquer grupo que X faça parte) não pode realizar as operações requisitadas, ele se recusa a fornecer ao usuário X direitos de acesso e nega o handle requisitado (Figura 4.5).

SACL contém ACEs que especificam os tipos de tentativas de acesso que geram relatórios de auditoria (*audit reports*). Cada ACE identifica um usuário/grupo, um conjunto de direitos de acesso, e um conjunto de *flags* que indica se o sistema gera mensagens de auditoria para tentativas de acesso fracassadas, bem-sucedidas, ou ambas. Para ler ou escrever em um SACL de um objeto, a thread deve primeiro possuir o privilégio SE_SECURITY_NAME (ver seção 4.2.5). O sistema escreve mensagens de auditoria na lista de eventos de segurança do sistema (*security event log*).

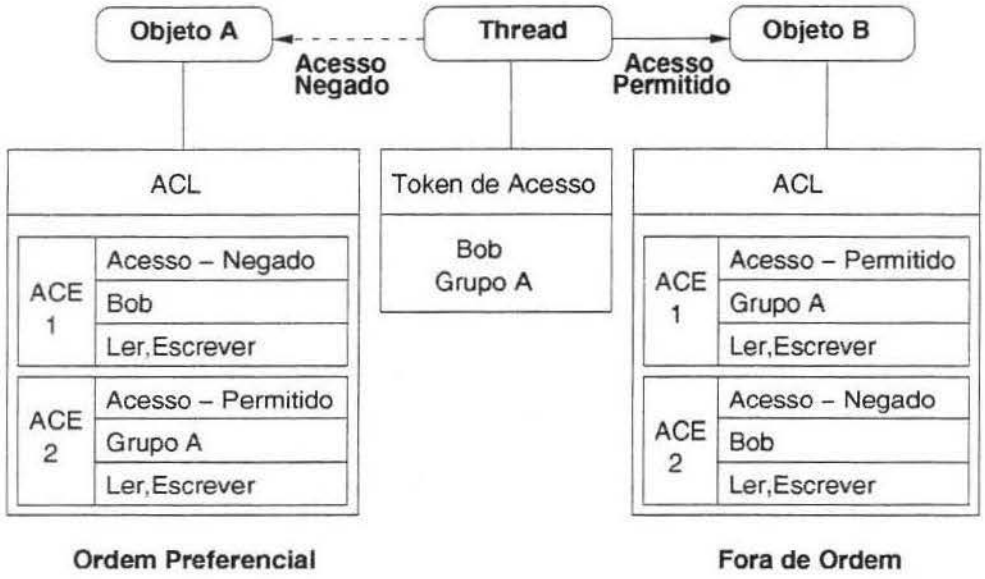


Figura 4.5: Ordem correta das ACEs numa ACL

Ordem correta das ACEs em uma ACL

Os direitos de acesso que uma ACL fornece a um usuário variam dependendo da ordem das ACEs na ACL. Consequentemente, a Microsoft define uma ordem preferencial para as ACEs nas ACLs de um objeto protegido. Esta ordem provê um arcabouço simples que assegura que uma ACE que nega acesso realmente nega o acesso.

Para o Windows NT 4.0 e anteriores, a ordem preferencial das ACEs é simples: em uma ACL, todas as ACEs que negam acesso devem preceder quaisquer ACEs que permitem acesso.

A figura 4.5 ilustra como as mesmas ACEs podem permitir diferentes direitos de acesso, dependendo de sua ordem na ACL. O sistema nega acesso ao objeto A quando ele lê a ACE que nega acesso; mas a ACL fora de ordem do objeto B faz o sistema permitir o acesso.

Para o Windows 2000 e posteriores, a ordem preferencial de ACEs é mais complicada devido à introdução de *ACEs específicas a objetos* e de *herança automática* das ACEs a partir dos objetos pais:

- Para assegurar que ACEs não-herdadas tenham precedência sobre ACEs herdadas, as ACEs não-herdadas devem ser agrupadas no início antes de quaisquer ACEs herdadas. Esta ordenação assegura, por exemplo, que uma ACE não-herdada que nega acesso seja considerada, quaisquer que sejam as ACEs herdadas que permitam acesso.

- Dentro dos grupos de ACEs não-herdadas e herdadas, as ACEs, de acordo com o tipo da ACE, devem ser ordenadas da seguinte maneira:

  1. ACEs que negam acesso que se aplicam ao próprio objeto;
  2. ACEs que negam acesso que se aplicam a um sub-objeto do objeto;
  3. ACEs que permitem acesso que se aplicam ao próprio objeto;
  4. ACEs que permitem acesso que se aplicam a um sub-objeto do objeto.

Naturalmente, nem todos os tipos de ACEs são necessárias em uma ACL.

Verificando um acesso de uma thread a um objeto

Quando uma thread tenta acessar um objeto protegido, o sistema ou permite ou nega o acesso. *Se o objeto não tem uma DACL, o sistema permite o acesso requisitado.* Caso possua uma DACL, o sistema procura por ACEs que se aplicam à *thread.*

O sistema compara o SID em cada ACE com os SIDs identificados no token de acesso da thread. Um token contém também um *SID do logon*, que identifica a sessão de logon corrente. Durante uma verificação de acesso, o sistema ignora SIDs de grupos desabilitados.

Tipicamente, o sistema usa o token de acesso primário da thread que está requisitando acesso. No entanto, se a thread está personificando outro usuário, o sistema usa o token de personificação .

O sistema examina cada ACE na sequência até que um dos seguintes eventos ocorra:

- Uma ACE de negação de acesso explicitamente nega algum dos *direitos de acesso* (ver seção 4.2.3) requisitados a um dos SIDs listados no token de acesso da thread;
- *Uma ou mais* ACEs de permissão de acesso para SIDs listados no token de acesso da thread explicitamente permite todos os direitos de acesso requisitados;
- Todas as ACEs foram verificadas e ainda há ao menos um direito de acesso requisitado que não foi explicitamente permitido, de modo que o acesso é implicitamente negado.

A figura 4.6 ilustra como uma DACL de um objeto pode permitir acesso a uma thread enquanto nega acesso a outra.

Duas Threads desejam acessar um objeto para leitura e escrita. Para a Thread A, o sistema lê a ACE 1 e imediatamente nega o acesso porque a ACE de negação de acesso se aplica ao usuário no token de acesso da thread. Neste caso, o sistema não verifica as ACEs 2 e 3. Para a Thread B, a ACE 1 não se aplica, assim o sistema procede para a

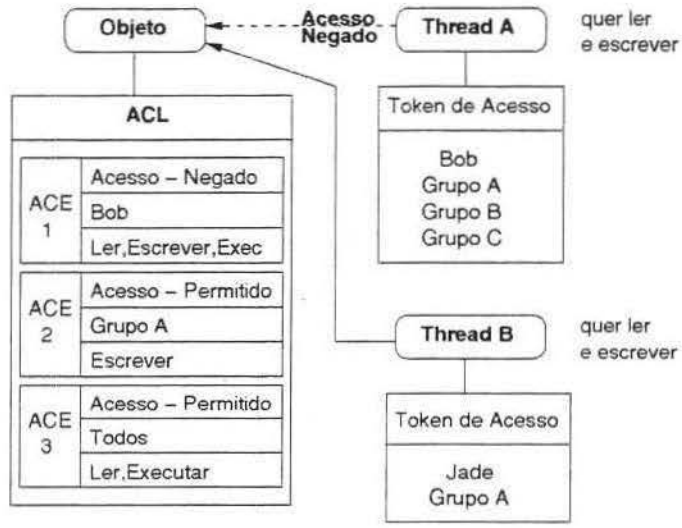


Figura 4.6: Verificando o acesso de uma thread a um objeto

ACE 2, que permite acesso de escrita, e para a ACE 3 que permite acesso de leitura e execução. No caso da Thread B, a ACE 2, que apenas permite escrita para a Thread B, não era suficiente para permitir todas as ações requisitadas durante o acesso. Assim, a ACE 3, que permite leitura para a Thread B, teve que ser encontrada. Se a ACE 3 não existisse, o acesso seria negado para a Thread B.

Armadilhas Comuns no uso de ACLs

Há alguns detalhes importantes na implementação da segurança NT. Por exemplo, um descritor de segurança pode ou não conter um DACL. Embora um descritor com um DACL vazio signifique que não é permitido o acesso de ninguém àquele objeto, um descritor sem DACL indica que nenhuma proteção é desejada para aquele objeto, de modo que qualquer um pode acessá-lo.

Outro detalhe importante é que o sistema pára de percorrer o DACL tão logo um acesso seja ou completamente permitido ou parcialmente/completamente negado. Assim, no Windows NT, a determinação da Microsoft de preencher um ACL iniciando sempre com os ACEs que negam acesso deve ser seguida de modo a assegurar que os algoritmos de controle de acesso do Windows NT funcionem e corretamente neguem acesso nas ocasiões em que deveriam negar.

4.2.4 Personificação

Um dos objetivos do modelo de segurança do Windows NT é assegurar que programas que um usuário usa não tem mais direitos de acesso do que o próprio usuário. A *personificação*

(*impersonation*) provê uma maneira de limitar o grau de acesso para aquele do cliente tentando acessar o sistema. A uma *thread* personificando um usuário são fornecidos apenas os direitos de acesso do usuário, supondo que o usuário deu permissão para a personificação. A vantagem deste procedimento é que quando o servidor cria um processo ou thread para lidar com o pedido do usuário, o servidor pode mandar a *thread* personificar este usuário e então permitir que o sistema NT providencie as proteções necessárias contra acesso não autorizado.

O NT suporta vários níveis de personificação, especificados por um processo cliente para limitar a ação de personificação dos servidores:

Security Anonymous: proíbe qualquer personificação por um servidor.

Security Identification: permite que o servidor adquira o *token* de acesso do cliente, mas não o personifique. Com isto, um servidor pode descobrir as permissões, identidade, e grupos de um cliente, mas não agir como aquele cliente.

Security Impersonation: permite ao servidor assumir o *token* de acesso do cliente, com quase todos os privilégios e habilidades daquele cliente. O servidor não pode subsequentemente conectar-se a outra máquina remota como o cliente personificado, a não ser que o recurso remoto, como arquivos ou impressora, suporte sessões nulas (*null sessions*) - um tipo de conexão onde a senha não é necessária. A *thread* que personifica também não pode criar um novo processo em nome do cliente.

Security Delegation: o Windows 2000 suporta um novo nível de personificação, que permite a um servidor personificar um cliente em sistemas locais e remotos. O protocolo Kerberos dá suporte a este nível, através de seu mecanismo de *Delegação de Autenticação* (ver capítulo 8).

4.2.5 Privilégios

Um privilégio é o direito de um usuário de realizar várias operações relacionadas com o sistema no computador local, tal como desligar (*shut down*), carregar drivers de dispositivos (*device drivers*), ou modificar a data do sistema. Os privilégios diferem dos direitos de acesso por dois fatores:

- Os privilégios controlam o acesso a recursos do sistema e tarefas relacionadas ao sistema, enquanto direitos de acesso controlam o acesso a objetos protegidos.
- Um administrador do sistema designa privilégios a usuários e grupos, enquanto que o sistema permite ou nega direitos de acessos a um objeto protegido baseado nas ACEs da DACL do objeto.

Quando o usuário "loga", o sistema produz um *token* de acesso que contém uma lista dos privilégios do usuário e dos grupos a que ele pertence. Os privilégios se aplicam somente ao computador local. Uma conta do domínio pode ter privilégios diferentes em computadores diferentes.

Quando o usuário tenta realizar alguma operação privilegiada, o sistema verifica o token de acesso do usuário para determinar se este possui os privilégios necessários. Se assim for, ele verifica se os privilégios estão habilitados. Se algum destes testes falhar para o usuário, o sistema não realiza a operação.

Os privilégios incluem, entre outros:

- Realizar operações de *backup* (SE_BACKUP_NAME);
- Criar *tokens* de acesso (SE_CREATE_TOKEN_NAME);
- Realizar *debug* de um processo (SE_DEBUG_NAME);
- Carregar um *driver* de dispositivo (SE_LOAD_DRIVER_NAME);
- Desligar um computador (SE_SHUTDOWN_NAME);
- Desligar um computador remotamente (SE_REMOTE_SHUTDOWN_NAME);
- Modificar a hora local (SE_SYSTEMTIME_NAME);
- Acessar certos atributos especiais de segurança, como o SACL (SE_SECURITY_NAME).

Uma lista completa dos privilégios pode ser encontrada em [MSDN98b].

4.2.6 Resumo das Extensões do Windows 2000

Há três extensões ao modelo de segurança do NT no Windows 2000[MR98].

A primeira extensão é a habilidade de controlar o acesso a objetos protegidos que outros sistemas operacionais (SOs) criaram. Objetos protegidos de outros SOs têm suas próprias definições para direitos de acesso. O Windows 2000 introduz a idéia de *direitos de acesso independentes do provedor* (*provider-independent access rights*). Quando uma aplicação identifica que está lidando com objetos estrangeiros (de outros SOs), ela pode implementar direitos de acesso independente do provedor através do uso de APIs para mapear os direitos estrangeiros para os correspondentes direitos do NT. Direitos de acesso independentes do provedor eliminam a carga desta tradução da aplicação. Para cada sistema operacional suportado, o Windows 2000 tem um provedor que converte os direitos de acesso independentes do provedor para os correspondentes direitos de acesso específicos do objeto[MSDN98c].

A segunda extensão aumenta a capacidade das ACLs para permitir uma granularidade por propriedade/atributo (sub-objetos) dentro de uma ACL. Esta extensão permite a um único descritor de segurança ter uma DACL ou SACL que referencia diferentes sub-objetos de um objeto protegido. Por exemplo, uma ACE de uma DACL de um arquivo de um processador de texto poderia permitir mudanças no texto, e outra ACE poderia negar a habilidade de modificar o autor do documento (ver seção 4.2.3).

A terceira extensão é uma API que cria um novo *token* a partir de um *token* existente, contendo subconjuntos ativos dos privilégios e grupos contidos no *token* original. Por exemplo, um servidor pode personificar um cliente iniciando um processo pelo cliente, e assegurar que este processo não desligue a máquina, mesmo que para o cliente haja esse privilégio. O servidor, neste caso, cria para si um *token* idêntico ao *token* do cliente, mas sem o privilégio de desligar a máquina. Se um grupo é desabilitado em um *token*, o NT considera o grupo ausente para os propósitos de permitir o acesso, mas presente para negar acesso. Desta forma, remover um grupo de um *token* não permite acesso a objetos que não estavam acessíveis.

Capítulo 5

Protocolos de Autenticação e Segurança

Existe uma miríade de protocolos de autenticação e segurança. A maioria são métodos obsoletos que na verdade não oferecem segurança. Infelizmente, eles englobam a maior parte dos protocolos em uso hoje, tanto na Internet quanto em intranets.

O Windows NT não é um caso a parte. Ele suporta múltiplos protocolos de autenticação e segurança, mas a maior parte é implementada para oferecer compatibilidade a clientes já existentes ou a redes heterogêneas como a Internet. Deste modo, a maior parte desses protocolos de segurança do Windows NT continua sujeita a falhas de segurança amplamente conhecidas. Nele, protocolos de autenticação e segurança modernos ainda são incipientes e, apesar de desde o início da década de 1990 terem surgido várias novas idéias interessantes sobre *protocolos fortes para transporte de senhas e autenticação do usuário* (ver seção), imunes a ataques de dicionário, parece que nenhuma dessas idéias será incluída nas próximas versões do sistema. O futuro da autenticação segura no Windows NT parece direcionar-se para soluções baseadas em chaves públicas e privadas, utilizando variações dos protocolos já existentes (como o Kerberos), ou talvez até adotando outros.

5.1 Terminologia

A partir deste capítulo, vários conceitos de criptografia e segurança serão utilizados. Os mais importantes para o entendimento do texto serão aqui delineados. Àqueles que desejarem aprofundar esses conceitos, recomenda-se ler [Sch96].

5.1.1 Criptografia

Criptografia é a ciência de manter mensagens seguras e protegidas. Funções comuns oferecidas pela criptografia são o sigilo, a autenticação, a integridade e a não-repudiação. O sigilo evita que pessoas não autorizadas consigam ler as mensagens. A autenticação

assegura o ponto de origem da mensagem (o emissor da mensagem). A integridade verifica se a mensagem não foi modificada durante sua transmissão. E a não-repudiação evita que o emissor negue falsamente mais tarde que ele enviou uma mensagem.

Texto Claro e Texto Cifrado

Uma mensagem que pode ser prontamente entendida pelo emissor e receptor é dita estar em *texto claro* (*plaintext*). Após a mensagem ser protegida usando criptografia, ela é dita estar em *texto cifrado* (*ciphertext*).

Algoritmo Criptográfico e Chaves Criptográficas

Um algoritmo criptográfico, também chamado cifrador, é uma função matemática usada para cifrar (transformar texto claro em texto cifrado) e decifrar (transformar texto cifrado em texto claro). Uma chave criptográfica também é fornecida ao algoritmo. Toda a segurança do algoritmo é baseada nessa chave. Isto significa que o algoritmo pode ser publicado e analisado. Para manter secreta a transformação que o algoritmo realiza, basta manter secreta a chave fornecida para a execução do algoritmo. Apenas quem possuir essa mesma chave será capaz de reproduzir a transformação e será capaz de realizar a transformação inversa.

O *espaço de chaves* é determinado pelo tamanho da chave. Uma chave pequena (com poucos bits) permite poucas variações no valor da chave. Uma chave grande (com muitos bits) permite mais variações no valor da chave, tendo associado um maior espaço de chaves. Para alguém que não conhece a chave para descriptografar, quanto maior o espaço de chaves, mais difícil é descobrir a chave correta.

Algoritmos Simétricos e Assimétricos

Existem dois tipos principais de algoritmos criptográficos: *simétricos* e *assimétricos*.

Os *algoritmos simétricos* são aqueles onde a chave de cifragem pode ser calculada facilmente a partir da chave de decifragem, e vice-versa. Na maioria desses algoritmos, a chave de cifragem e decifragem é a mesma. Esses algoritmos são divididos em outras duas categorias. Os *cifradores de fluxo* (*stream ciphers*) operam no texto claro um bit ou byte por vez. Os *cifradores de bloco* (*block ciphers*) operam em grupos de bits chamados blocos, usualmente de 8 bytes ou mais.

Os *algoritmos de chave pública* (também chamados *algoritmos assimétricos*) são projetados de modo que a chave usada para criptografia seja diferente da chave usada para a descriptografia. Mais ainda, a chave de descriptografia não pode (em uma quantidade de tempo razoável) ser calculada a partir da chave de criptografia. Estes algoritmos são chamados de chave pública porque a chave de criptografia pode ser pública. Apenas quem

possui a chave de descriptografia associada (também chamada *chave privada*), mantida em segredo, é capaz de realizar corretamente a operação de descriptografar.

Funções Criptográficas Unidirecionais (*One-way hash functions*)

As *funções criptográficas unidirecionais* (também chamadas *funções de hash criptográfico* e *message-digest*) são aquelas que são relativamente fáceis de ser computadas, mas difíceis de serem revertidas. Tipicamente, elas recebem uma entrada de tamanho variável (chamada *pré-imagem*) e convertem-na em uma saída menor de tamanho fixo (chamada *valor de hash*). Uma boa função de *hash* é *livre de colisão* (*collision-free*): é difícil gerar duas pré-imagens com o mesmo valor de *hash*. A saída de uma boa função de *hash* também não deve ser dependente da entrada de qualquer maneira discernível. A mudança de um único bit da pré-imagem muda, na média, metade dos bits do valor de *hash*.

Um interessante problema estatístico associado a estas funções é o Ataque do Aniversário (*Birthday Attack*). Suponha que uma função criptográfica unidirecional é segura e produz uma saída com m bits. Achar uma pré-imagem que produza um valor de *hash* determinado exigiria computar 2^m pré-imagens aleatórias. No entanto, achar duas pré-imagens que produzem um mesmo valor de *hash* exige apenas $2^{m/2}$ pré-imagens aleatórias. Uma máquina que computa um milhão de valores de hashes por segundo levaria 600.000 anos para achar uma segunda pré-imagem com valor de *hash* de 64 bits igual a um outro valor de hash determinado. A mesma máquina poderia achar um par de pré-imagens que possuíssem um mesmo valor de *hash* em cerca de uma hora.

Uma aplicação destas funções seria o armazenamento mais seguro de senhas de usuários de um sistema. O sistema pode evitar armazenar diretamente as senhas (pré-imagem), armazenando apenas os valores de *hash* associados. Assim, caso o sistema fosse invadido e a lista de valores de *hash* fosse copiada, seria muito difícil recuperar a lista de senhas original. Esses valores de *hash* também podem ser manipulados como se fossem os equivalentes das senhas, permitindo comparações entre os valores armazenados. Valores de *hash* iguais indicam pré-imagens iguais, e valores diferentes indicam pré-imagens diferentes.

5.1.2 Criptoanálise

O principal objetivo da criptografia é tornar ininteligível o texto claro, transformando-o em texto cifrado usando um algoritmo e uma chave. A Criptoanálise é a ciência de recuperar o texto claro de uma mensagem sem acesso à chave. Há vários tipos de ataques para executar uma criptoanálise. Os mais importantes para o entendimento do texto são delineados abaixo, e supõem que o algoritmo criptográfico utilizado é conhecido.

Texto Cifrado Apenas (*Ciphertext-only*)

É conhecido apenas o texto cifrado de uma ou várias mensagens, todas criptografadas usando o mesmo algoritmo de criptografia. O objetivo é obter o texto claro, ou melhor ainda, a chave.

Texto Claro Conhecido (*Known-plaintext*)

É conhecido não apenas o texto cifrado de várias mensagens, mas também o texto claro delas. O objetivo é deduzir a chave usada para criptografar as mensagens.

Texto Claro Escolhido (*Chosen-plaintext*)

É conhecido não apenas o texto cifrado e o texto claro associado para várias mensagens, mas também é possível escolher o texto claro que vai ser criptografado. O objetivo é deduzir a chave usada para criptografar as mensagens.

Ataque de Dicionário (*Dictionary attack*)

Em algumas situações em que é possível utilizar ataques de texto claro escolhido, as chaves utilizadas nos algoritmos de criptografia possuem certos valores que são muito mais prováveis que outras. Por exempo, as senhas inventadas por pessoas. As senhas usualmente são convertidas para chaves numéricas que são usadas em algoritmos criptográficos, e portanto são equivalentes às chaves criptográficas. Para facilitar a memorização, as pessoas tendem a utilizar combinações simples para suas senhas, baseadas em palavras comumente encontradas em dicionários, às vezes com pequenas variações. Um ataque inteligente tentaria utilizar inicialmente essas chaves mais prováveis para criptografar o texto claro conhecido através do mesmo algoritmo criptográfico utilizado originalmente. Para cada uma dessas chaves, é verificado se o resultado do algoritmo é igual ao texto cifrado conhecido. Caso seja igual, essa chave é a chave associada à senha procurada[1].

Enquanto uma senha em particular possa ser bem difícil de adivinhar, na média as senhas derivam de palavras de dicionários ou outros valores óbvios, como data de aniversário, o que torna o ataque de dicionário bastante atraente. Se m for a quantidade de palavras no dicionário, e n for a quantidade de variações por palavra, então a dificuldade de realizar este ataque é $O(m * n)$. No entanto, caso a senha procurada não seja semelhante a alguma palavra do dicionário utilizado, será necessário procurar por todas

[1]Pode acontecer também de a intenção ser realizar a transformação inversa, ou seja, transformar o texto cifrado conhecido em texto claro conhecido. Nesse caso, quando o resultado encontrado for igual ao texto claro conhecido, a senha usada como chave é a senha procurada.

as outras possíveis combinações de senhas menos prováveis. Esse caso corresponde a uma degeneração do ataque de dicionário para um ataque de força bruta.

Ataque de Força Bruta (*Brute-force attack*)

Caso as possíveis chaves sejam equiprováveis, então um possível ataque seria tentar usar cada uma de todas essas possíveis chaves para tentar obter o texto claro original, realizando um ataque de força bruta. Esses ataques não são eficientes, e a dificuldade de realizá-los é exponencialmente proporcional ao tamanho da chave. À taxa de 1.000.000 de tentativas por segundo (tipicamente obtidas por um computador Pentium atualmente), uma chave de 20 bits levaria cerca de 1 segundo para ter todas suas combinações percorridas. Uma chave de 40 bits levaria cerca de 12 dias, e uma chave de 60 bits cerca de 36000 anos. Algoritmos criptográficos com chaves efetivas maiores, com 128 bits ou mais, são na prática muito resistentes a ataques de força bruta[2].

5.1.3 Outros conceitos

Zero-knowledge proofs

Uma prova de conhecimento zero (*zero-knowledge proof*) oferece a oportunidade de provar a posse de uma informação, sem revelar essa informação. É uma prova interativa com um provador e um verificador, em que o provador convence o verificador de uma afirmação (com alta probabilidade). Para mais detalhes, ver [Sch96, p.101].

Ataques de um Intermediário (*Man-in-the-middle*)

Por ocasião de uma troca de mensagem entre duas entidades, pode haver uma tentativa de modificação dessa mensagem por uma terceira entidade, localizada no caminho por onde a mensagem passa. Essa terceira entidade, também chamada *intermediário* (ou *man-in-the-middle*), pode possuir a capacidade de alterar, acrescentar, repetir ou remover o conteúdo da mensagem, ativamente influenciando a conversa e a subvertendo para proveito próprio. A modificação pode passar despercebida, caso o protocolo responsável pela troca de mensagens não seja resistente a esse tipo de ataque.

[2] De acordo com [Sch96, p.234], a quantidade de bits de informação existente em blocos de 8 caracteres de texto em inglês é cerca de 2,3 bits por letra, ou 18,4 bits por bloco. Portanto, chaves criptográficas geradas a partir de senhas textuais, tipicamente memorizadas por usuários de sistemas computacionais, tendem a ter um tamanho efetivo pequeno.

5.2 Métodos Obsoletos de Autenticação de Senhas

Esses métodos constituem o mecanismo da maioria dos protocolos de autenticação amplamente difundidos.

Senha em Claro

Métodos que usam senha em claro ainda são predominantes na Internet atualmente, tal como Telnet, FTP, e qualquer outra sessão não criptografada (como no método de autenticação básico HTTP sem SSL). Eles são vulneráveis a ataques onde se observa pacotes na rede (*sniffing*).

Senha Misturada

Os métodos de senha misturada (*scrambled*) simplesmente obscurecem a senha, usando algum algoritmo bem conhecido ou facilmente descoberto (por exemplo, a notação BASE64). Não são mais seguros que os que usam senha em claro.

Desafio/Resposta-Criptografada (CHRAP)

Uma solução comum para os problemas de segurança do modelo de autenticação através de senha é usar um protocolo de *desafio/resposta-criptografada* (CHRAP, de *Challenge/Hashed-Response Authentication Protocol*) entre o cliente e o servidor.

Nesse modelo, a senha real (em texto claro) nunca é transmitida. Ao invés disso, o servidor envia em texto claro um *desafio numérico* aleatório D_s que pode apenas ser respondido corretamente se o cliente conhece a senha P_s armazenada no servidor.

O cliente vai responder calculando uma *resposta-criptografada* $R_c(D_s, P_c)$, que é uma função criptográfica com dois parâmetros: como texto claro, o desafio original D_s do servidor; e como chave criptográfica, uma senha P_c fornecida pelo cliente. Essa resposta-criptografada é transmitida de volta ao servidor. O servidor possui armazenada a senha P_s, e conhece D_s, criado por ele. Ao receber R_c, o servidor calcula $R_s(D_s, P_s)$. Se $R_c = R_s$, então $P_c = P_s$, e a senha que o cliente enviou é a mesma que o servidor conhece. Portanto, o cliente conhece a senha P_s do servidor e tem o direito de acessá-lo.

Durante o protocolo, apenas D_s e R_c foram transmitidos. Como D_s é um número aleatório, a mesma senha do cliente dificilmente vai dar origem a uma mesma resposta criptografada. No entanto, como o texto claro D_s e o texto cifrado R_c são conhecidos após o protocolo terminar, se a função criptográfica for conhecida, é possível aplicar um ataque de texto claro escolhido usando um dicionário para minimizar o tempo de ataque (ataque de dicionário).

Muitas variações do método de autenticação baseado em desafio/resposta-criptografada para senhas são utilizadas desde o início dos anos 1980. Elas são todas vulneráveis a ataques de dicionário quando a senha do usuário é pequena, ou não é aleatória o suficiente. Apesar de melhor que não usar proteção alguma, estes métodos são um grande problema, pois são amplamente usados. As autenticações dos protocolos como Novell NetWare 3 ([Nov01] e [Gut01]), Banyan Virtual Integrated Network Services (VINES) [Ban93], Windows NT CIFS/SMB ([LP96] e [SMB01]), NT LAN Manager (capítulo 6), HTTP Digest Access Authentication [RFC2617], Point to Point Tunneling Protocol (PPTP)[Sch98], MS-CHAP (Microsoft Point-to-Point Protocol (PPP) Challenge Handshake Authentication Protocol (CHAP) Extensions) [Z99a], Challenge-Response Authentication Mechanism (CRAM)[RFC2195], e Point-to-Point Protocol (PPP) Challenge Handshake Authentication Protocol (CHAP) ([RFC1334] e [Z99a]) são alguns exemplos.

São inúmeros os problemas de segurança já reportados pela utilização do CHRAP. Por exemplo, o Windows NT, quando tentava acessar algum recurso remoto disponível via HTTP na Internet, utilizava automaticamente o CHRAP caso esse recurso remoto estivesse protegido por senha, transmitindo inevitavelmente a senha criptografada do Domínio, passível de ser atacada por um dicionário (*Web Client NTLM Authentication Vulnerability*, ver [MS01]). Outros problemas incluem a criação de redes privadas virtuais (VPNs, de *Virtual Private Networks*) usando a implementação PPTP da Microsoft, onde os problemas de segurança surgem em uma quantidade impressionante ([Sch98] e [Sch99]).

Login Kerberos

No Kerberos Versão 4 e Versão 5, uma senha criptografa um *ticket* inicial. Os dados contidos no ticket permitem um *ataque de texto claro verificável*[3] [Sch96] (ver [BM91], [Gon95], [Jas96] e [KPS95]).

O Windows 2000, especificamente, que usa a Versão 5, possui muitas variações proprietárias na implementação dos métodos criptográficos do protocolo Kerberos (ver capítulo 8), que acabam facilitando ainda mais o ataque de dicionário.[4]

O método da Versão 4 é ainda mais fraco que o CHRAP, já que um ataque de dicionário pode ser realizado por qualquer um que simplesmente solicite um *ticket* inicial (TGT), a qualquer hora. Este ticket inicial é solicitado durante o login inicial do Kerberos versão 4, e esta solicitação pode ser feita a partir de qualquer computador conectado à rede. Mesmo com a adição da pré-autenticação na Versão 5, o protocolo é ainda vulnerável a

[3]Um ataque de texto claro verificável é aqui definido como um ataque de texto claro conhecido em que é fácil verificar a validade do texto claro. No caso do Kerberos versão 5, por exemplo, o texto claro verificável é um conjunto de dígitos numéricos ASCII contendo uma data em formato UTC (*Universal Time*).

[4]Por exemplo, não há o conceito de *salt* nas senhas da implementação proprietária da Microsoft.

um ataque de dicionário feito por alguém que fique observando a rede (*eavesdropping*).

Certos tipos de extensões Kerberos, baseadas em chaves públicas e privadas, são implementadas no Windows 2000, mas não são default e são de difícil gerenciamento. O protocolo Kerberos poderia se aproveitar muito dos novos métodos de protocolos fortes de senhas, imunes a ataques de dicionários.

S/Key

O mecanismo S/Key é um esquema de senhas usadas uma única vez (*one-time passwords*) que se baseia em uma função criptográfica de *hash* unidirecional [Sch96]. Usando esta função, uma lista de senhas é gerada: uma senha S1 é gerada a partir de uma senha S0 fornecida pelo usuário, uma senha S2 é gerada a partir da senha S1, até obter-se o número de senhas desejadas. A utilização das senhas começa a partir da última, até chegar em S1, quando a lista é novamente gerada.

O S/Key enfrenta um problema diferente, pois é usado em conjunto com programas originalmente projetados para aceitar apenas senhas em texto claro. Mas o S/Key sozinho não resolve o problema de ataques de dicionários contra senhas pequenas (isto é, com o parâmetro S0 pequeno).

SecurID

Dispositivos SecurID são mecanismos de autenticação baseados em algo que o usuário conhece (uma senha) e algo que ele tem (um autenticador, como um cartão), mas não resolve o problema da senha pequena e simples. Um ataque de dicionário pode ser usado com um autenticador roubado para tentar obter acesso não autorizado a um sistema.

Métodos Assistidos por Chaves Públicas

No *login* do NetWare 4, por exemplo, a senha é protegida pela criptografia com uma chave pública de longo prazo. A confiança em chaves públicas persistentes pode causar problemas. Quando a chave pública é enviada em claro, como no NetWare 4, é possível utilizar *spoofing* e ataques *man-in-the-middle*, onde um intermediário captura a chave pública do servidor e emite sua própria chave pública para o cliente. O cliente devolve sua senha criptografada na chave pública do intermediário, que a descriptografa usando sua própria chave privada, obtendo a senha em claro do cliente. Em seguida ele criptografa esta senha do cliente com a chave pública do servidor e a envia para o servidor. Nem o cliente nem o servidor perceberam a existência do intermediário, que agora possui a chave (senha em claro) do cliente.

Quando a chave pública é pré-distribuída ou pré-certificada, como em sessões SSL anônimas, há complexidades e problemas adicionais relacionados à necessidade de uma

infraestrutura cara e complexa para certificação e revogação de chaves públicas.

5.3 Métodos Modernos de Autenticação de Senhas

No início dos anos '90, dois grupos de pesquisadores independentes publicaram as primeiras soluções que resolviam o problema de verificação de senhas na rede (ver [Jab01a] e [Jab01b]). Esses métodos mostram de maneira segura o conhecimento da senha, sem revelá-la a qualquer um que já não a conheça. Eles também podem ser utilizados para autenticação mútua e para oferecer uma chave de sessão para comunicação secreta autenticada, sem usar chaves ou certificados adicionais.

Todos esses *protocolos fortes de senhas* (*strong password authentication*, também conhecidos como *key amplifiers*) usam técnicas de chaves públicas, mas ao invés de exigir chaves armazenadas ou uma hierarquia de certificados, eles usam chaves *efêmeras*. Pares efêmeros de chaves públicas e privadas são aleatoriamente criados, usados uma única vez, e então descartados tão logo a autenticação esteja completa.

Steve Bellovin e Michael Merritt, da Bell Labs, desenvolveram e patentearam uma família de métodos chamada *Encrypted Key Exchange* (EKE) (ver [BM92] e [Sch96]). Li Gong, Mark Lomas, Roger Needham e Jerome Saltzer desenvolveram os métodos chamados *Secret Public Key* [GLNS93]. Um método próximo relacionado, chamado SPEKE, foi desenvolvido por David Jablon [Jab96]. Vários métodos parecidos existem atualmente. Como uma classe, todos estes métodos derrotam ataques de dicionário.

Métodos Estendidos

Há uma família de protocolos fortes de senhas estendidos baseados no SPEKE, como B-SPEKE e Augmented-EKE [Jab97], e outros protocolos como o Open Key Exchange (OKE) [Luc97] e Secure Remote Protocol (SRP) [Wu98]. Nesses métodos estendidos, o servidor *não* grava a senha do usuário, ou qualquer coisa que seja equivalente à senha em claro, de forma que se o servidor ficar comprometido, não há nenhuma informação útil para o invasor. Um método recentemente adicionado à lista é o PAK [BMP00].

O importante na autenticação baseada no conhecimento de algo, como uma senha - *proofs of human knowledge* - é que ela verifica a presença de alguém vivo. Isto é distinto de uma autenticação biométrica (algo que você é) onde características físicas não podem ser modificadas ou escolhidas, e é distinto de uma autenticação baseada no armazenamento de uma chave (algo que você tem) que pode somente verificar a presença da máquina ou dispositivo.

Uma chave memorizada tem tantas vantagens quanto desvantagens sobre uma chave armazenada. O melhor é combinar esses dois fatores, mas é importante tratar cada classe

de chave com o método mais forte apropriado. O ideal, como os métodos estendidos proporcionam, é não ter que armazenar o fator memorizado.

5.4 Protocolos de Autenticação e Segurança no Windows NT

A infraestrutura de segurança do Windows NT suporta os seguintes protocolos de autenticação e segurança. Com exceção dos protocolos baseados em chaves pública e privada, de natureza diferente, todos os outros protocolos são suscetíveis a ataques de dicionário.

5.4.1 Windows NT Lan Manager (NTLM)

Usado pelo Windows NT 4.0 e versões anteriores. Essencialmente um método CHRAP, susceptível a ataques de dicionário, o protocolo NTLM continua a ser suportado e usado para autenticação, acesso a arquivos remotos e conexões RPC autenticadas com versões anteriores do Windows NT. Uma exposição detalhada será vista no capítulo 6.

5.4.2 Kerberos versão 5

Substitui o protocolo NTLM, no Windows 2000, como o protocolo primário de segurança para acesso a recursos dentro ou através de domínios Windows NT. Alguns dos benefícios do Kerberos são: autenticação mútua entre cliente e servidor, carga do servidor reduzida durante estabelecimento de conexões e suporte para delegação de autenticação de clientes para servidores através do uso de mecanimos de procuração (*proxy*). No entanto, a implementação do Windows 2000 é susceptível a *ataque de texto claro verificável*. Uma exposição detalhada será vista no capítulo 8.

5.4.3 Autenticação Internet Proprietária

É encontrada nos protocolos *Microsoft Network (MSN) Authentication Protocol*, *Distributed Password Authentication* (DPA) e *Remote Passphrase Authentication* (RPA) [Com00]. São protocolos de autenticação por segredo compartilhado, do tipo CHRAP, usado por algumas organizações, como MSN e CompuServe.

5.4.4 Protocolos baseados em chave pública

Tentam oferecer privacidade e confiabilidade em redes públicas como a Internet. O *Security Sockets Layer* (SSL) da Netscape [FKK96] é um padrão *de facto* para conexões

entre navegadores clientes e servidores Web. Outros protocolos, como o *Transport Layer Security* (TLS)[RFC2246] podem ser usados. Esses protocolos usam certificados baseados em chaves públicas para autenticar clientes e servidores e *dependem de uma infraestrutura de chaves públicas para funcionarem efetivamente.*

O Windows 2000, com a CryptoAPI 2.0, tem um suporte criptográfico desenvolvido para protocolos de chave pública, oferecendo:

- Suporte para certificados X.509v3 através de funções centralizadas para codificar, decodificar, ler (*parse*) e avaliar certificados.
- Suporte para requisições de certificados no formato PKCS #10 e dados assinados no formato PKCS #7.
- Assinatura digital, verificação e criptografia de dados através de funções criptográficas de alto nível disponível para aplicações.

Estes mecanismos são oferecidos por Provedores de Serviços Criptográficos (CSP, de *Cryptographic Service Provider*) [Mic00g], que são bibliotecas criptográficas facilmente adicionadas ao sistema (*add-ons*). Cada biblioteca oferece características criptográficas específicas, como tamanho da chave (40, 56, 128, 512, 1024 bits etc) e algoritmos utilizado (RSA, Secure Hash Algorithmo (SHA), Digital Signature Standard (DSS), etc).

O grande problema destes protocolos é que embora atualmente haja uma infraestrutura razoável para a certificação das chaves públicas dos servidores, poucos são os *clientes* que possuem uma chave pública certificada para oferecer ao servidor e com isso permitir sua autenticação. Para mais detalhes, ver [Tho00].

5.4.5 SPNEGO

O protocolo SPNEGO (*Secure Protected Negotiation Protocol*) [RFC2478] é usado entre um cliente e servidor para negociar o protocolo de autenticação mais forte que ambos os lados possuem. Ele também evita ataques de rebaixamento (*downgrade attacks*).

Este protocolo verifica no servidor e no cliente quais os protocolos de autenticação disponíveis, dando preferência ao Kerberos e depois ao NTLM. No fim deste protocolo, tanto o cliente quanto o servidor sabem qual o protocolo e dialeto devem usar para a comunicação.

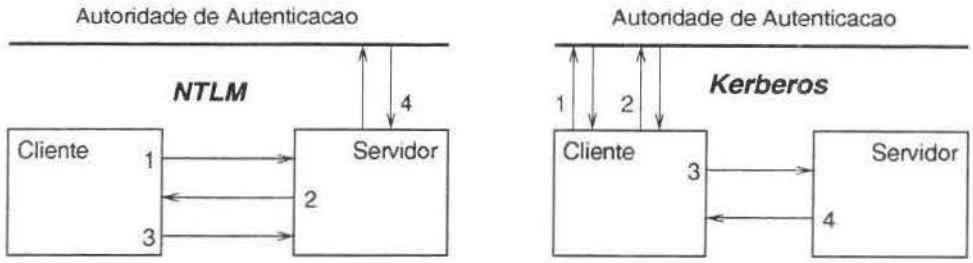


Figura 5.1: Diferenças entre os protocolos NTLM e Kerberos

5.5 A Arquitetura SSPI para Múltiplos Protocolos de Autenticação

Cada um dos diferentes protocolos de autenticação e segurança é implementado como um *Provedor de Serviço de Segurança* (SSP, de *Security Support Provider*). Um SSP cria e processa estruturas de autenticação que são opacas para os protocolos de transporte. Quando um protocolo de transporte vai iniciar a comunicação com o outro lado, o SSP escolhido se encarrega de iniciar uma autenticação segura e retornar uma chave de sessão para ambos os lados, que pode ser usada para criar um canal seguro de comunicação para o protocolo de transporte.

A figura 5.1 mostra as diferenças entre os protocolos dos SSPs NTLM e Kerberos, os mais usados no Windows 2000. Os números representam a ordem das mensagens trocadas. No NTLM, inicialmente há comunicação entre o cliente e servidor, seguido da comunicação entre o servidor e sua autoridade de autenticação. No Kerberos, por outro lado, há comunicação entre o cliente e sua autoridade, seguido da comunicação entre o cliente e servidor. No NTLM, não há *caching* no cliente ou servidor, de modo que cada requisição de autenticação do cliente exige que o servidor se comunique com sua autoridade. No Kerberos, se um *ticket* válido existe no *cache*, o cliente pode nem precisar se comunicar com sua autoridade.

O Windows NT oferece suporte para separar o *protocolo de autenticação* do *protocolo de transporte de dados* através de sua *Interface de Provedor de Suporte de Segurança* (SSPI, de *Security Support Provider Interface*). Ela é baseada na *Interface de Programação de Aplicação de Serviço de Segurança Genérico* (GSS-API, de *Generic Security Service Application Program Interface*), documentada em [RFC1508] e [RFC1509], e fornece abstração de interface similar para gerenciamento de contextos de segurança. A SSPI é uma maneira conveniente de lidar de modo uniforme com as diferenças fundamentais entre diversos protocolos de autenticação e segurança. O SSPI permite, por exemplo, abstrair as diferenças fundamentais mostradas acima entre o Kerberos e o NTLM.

O SSP SPNEGO é o provedor default em muitos subsistemas de comunicação, para

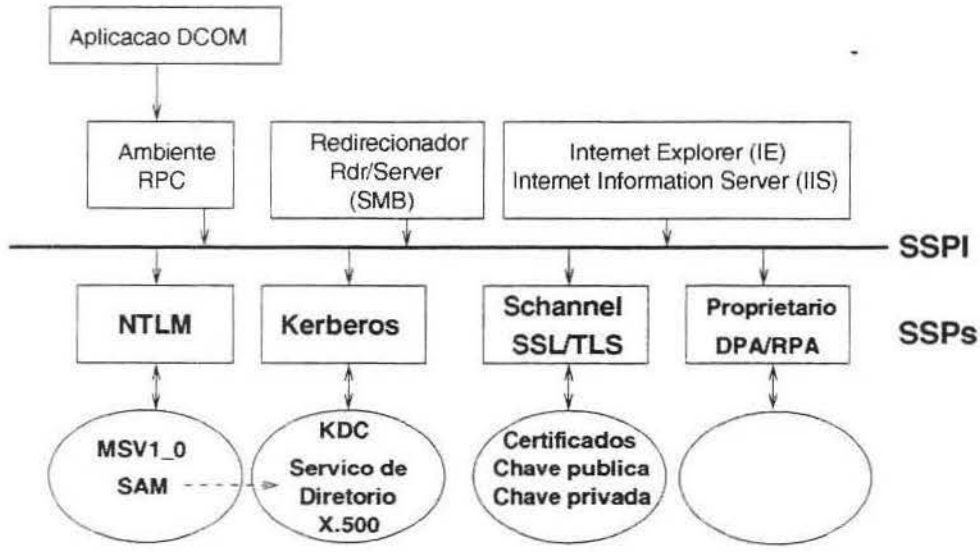


Figura 5.2: A arquitetura para suportar múltiplos protocolos de segurança (SSPI)

garantir a máxima interoperabilidade. Protocolos de alto nível para gerenciar conexões de rede, oferecidos por mecanismo RPC autenticado e DCOM, utilizam indiretamente a interface SSPI e também SPNEGO.

A figura 5.2 mostra a arquitetura para suportar múltiplos protocolos implementados no Windows NT usando SSPI.

Provedores de segurança diferentes usam diferentes credenciais (ver seção 4.1.1) para autenticar o usuário. Os protocolos de segurança interagem com diferentes serviços de autenticação e armazenadores de informação.

- O provedor de segurança NTLM usa o serviço de autenticação MSV1_0 cliente e o serviço NETLOGON num Controlador de Domínio para autenticação de um cliente e intercâmbio de informação de autorização. No Windows 2000, as chamadas para a base de dados SAM do Domínio são redirecionadas para o novo Serviço de Diretórios X.500 (*ActiveDirectory*).

- O provedor de segurança Kerberos usa o serviço do Centro de Distribuição de Chaves Kerberos (KDC, de *Key Distribution Center*) e a base de dados do Serviço de Diretórios X.500 para criar os *tickets* de autorização. Ele suporta formatação dos dados de segurança de acordo com os mecanismos GSS-API para o Kerberos 5 definidos em [RFC1964].

- Os provedores MSN, DPA e RPA usam os serviços de segurança proprietários de provedores de acesso Internet para autenticação e acesso a informações de autorização.

- Os provedores de canais seguros (*Secure Channel (Schannel) security providers*), como o SSL e o TLS, são baseados em certificados de chave pública emitidos por Autoridades de Certificados (CAs, de *Certificate Authorities*) confiáveis.

Autenticação para usuários externos

O suporte para a autenticação baseada em certificados e chaves públicas no Windows NT permite aplicações clientes conectarem-se a serviços seguros no papel de usuários que não têm uma conta no Domínio. Os usuários que podem ser autenticados através de um certificado de chave pública emitido por uma Autoridade de Certificados (CA) podem acessar recursos do Windows NT. O Serviço de Diretório pode associar um ou mais usuários externos a uma conta Windows NT já existente, para controle de acesso. O campo *subject name* no certificado X.509v3 é usado para identificar o usuário externo que está associado com a conta.

5.6 Autenticação HTTP no Windows NT

A autenticação sobre protocolo HTTP no Windows NT é muito utilizada atualmente, e é interessante descrever como ela se relaciona com os protocolos de autenticação nativos do HTTP e com os existentes no sistema ([KB00] e [Mic00e]).

Cada *thread* servidora do *Internet Information Server* (IIS) - o servidor Web da Microsoft - que executa uma requisição HTTP de um cliente tem que personificar uma conta no Domínio ou na máquina local. Há várias maneiras do IIS determinar a identidade do cliente associada a esta conta. Elas são, em ordem de preferência: autenticação anônima (sem autenticação), autenticação baseada em certificados (chaves públicas e privadas), autenticação integrada ao Windows, autenticação *Digest*, e autenticação Básica.

Anônima O IIS prefere não autenticar os clientes. A autenticação consome recursos da CPU e muitos sistemas (*web sites*) não se baseiam na autenticação do sistema operacional. Em vez disso, ela é feita no nível de aplicação. O IIS mapeia um cliente não autenticado para uma conta especial (IUSR_MACHINE).

Baseada em Certificados Antes que o IIS aceite um certificado do cliente, ele deve verificar pela data e pelas assinaturas, se este é válido. Através de um sistema de validação e revogação de certificados, o IIS permite estabelecer quais certificados são confiáveis e quais não são. Há duas maneiras de mapear certificados para contas de usuários: um certificado para uma conta, ou muitos certificados para uma conta.

Autenticação Integrada O IIS possui uma opção de tentar fazer o cliente *web* do usuário inicialmente tentar uma autenticação usando algum protocolo nativo do Windows NT, NTLM ou Kerberos, em cima de HTTP, via SPNEGO. Estas extensões não são suportadas por clientes que não sejam o Internet Explorer, restringindo o alcance desta solução. Outro problema é que estes protocolos não funcionam bem com *firewalls*: o Kerberos, por exemplo, exige que a porta 88 TCP ou UDP esteja livre. Isto não é algo que um administrador deveria fazer em uma *firewall*. [5]. Dentro do perímetro de uma intranet, no entanto, esta é uma opção de autenticação conveniente. Não há necessidade de certificados, apenas de utilizar o Internet Explorer.

Digest Este mecanismo de autenticação nativo foi introduzido no HTTP/1.1 e implementado no IIS 5.0 e Internet Explorer 5.0. Sendo um método de autenticação CHRAP, ele é similar ao NTLM e possui todos os problemas associados: não oferece autenticação mútua, nem uma chave de sessão para criptografia da comunicação após o processo de autenticação. Além disso, exige que as senhas sejam guardadas usando um método criptográfico reversível, para extração da senha em claro. Portanto, é um mecanismo que não deve ser usado. Outros mecanismos deste tipo são o MSN, DPA e RPA, que são proprietários.

Básica Este é o mecanismo nativo de autenticação do HTTP/1.0. O cliente envia uma requisição HTTP para o servidor, e o servidor envia de volta um erro, exigindo que o cliente prove sua identidade através de um nome de usuário mais uma senha codificada no formato base64, equivalente a mandar a senha em claro. Assim, a autenticação básica só faz sentido em linhas criptografadas com SSL. Desde que o servidor seja confiável e desde que seja certo que ninguém naquele servidor irá ver a senha, a autenticação em claro com SSL é razoável. Com este mecanismo, o usuário autenticado pode ganhar credenciais de rede NT (usando a opção IIS de *logon* interativo). Com essas credenciais, a *thread* servidora possui o direito de acessar outras máquinas na rede NT em que o IIS se encontra. Isto é necessário caso scripts CGI/ASP tenham que acessar componentes DCOM que se encontram em máquinas NT diferentes do servidor IIS. Um fato importante ao usar esta autenticação é que, mesmo após o IIS autenticar o usuário internamente, ele deixa a senha em claro disponível através da variável AUTH_PASSWORD, que pode ser lida via CGI/ASP.

Portanto, com exceção do mecanismo de certificados - relativamente complexos e caros de gerenciar - nenhuma das soluções de autenticação via HTTP oferece suporte para segu-

[5] A Microsoft e a Cisco propuseram extensões ao Kerberos para solucionar este problema do *firewall* (ver http://www.ietf.org/internet-drafts/draft-ietf-cat-iakerb-03.txt)

rança forte. A autenticação integrada é baseada no NTLM ou Kerberos, via SPNEGO, mas estes protocolos são passíveis de ataque de dicionário.

Capítulo 6

Autenticação NTLM/LAN Manager

Este capítulo engloba um artigo apresentado na 2a Conferência de Redes de Computadores em Évora, Portugal, em outubro de 1999 (CRC'99) [Gra99]. Ele explora as fraquezas associadas à implementação da autenticação dos protocolos NTLM e LAN Manager do Windows NT, que são basicamente protocolos de autenticação obsoletos de desafio-e-resposta-criptografada (CHRAP).

Aspectos Criptográficos no Windows NT

O Windows NT implementa vários mecanismos criptográficos para proteger dados sensíveis. No entanto, por razões de projeto e de compatibilidade com versões anteriores (principalmente do Windows), vários desses mecanismos possuem falhas. Este artigo explora as falhas no armazenamento de senhas na base de dados do sistema, na transmissão destas utilizando o protocolo de autenticação de desafio/resposta da Microsoft (MS-CHAP - *Microsoft Challenge-Handshake Authentication Protocol*), na implementação do protocolo PPTP (*Point-to-Point Tunneling Protocol*) da Microsoft para criar Canais Seguros, e aponta novas tendências do Windows NT relevantes ao assunto, como a migração para o sistema de autenticação Kerberos e utilização de um sistema de arquivos cifrado (EFS - *Encrypted File System*).

6.1 Introdução

Atualmente os mecanismos de armazenamento e transmissão de senhas do Windows NT são criptograficamente fracos e, por motivos de compatibilidade com sistemas legados, mantêm informações duplicadas, armazenadas com graus de segurança distintos, limitando a segurança do sistema ao menor denominador comum de segurança usado.

Alguns desses mecanismos são de concepção ingênua e têm sua segurança baseada no segredo do processo e não em alguma informação secreta.

As informações de interesse criptográfico consideradas relevantes aqui são as senhas, os canais de comunicação e os dados no disco. A partir disso, este trabalho se propõe a explicar e analisar criticamente os aspectos criptográficos envolvidos no armazenamento, transmissão de senhas e no protocolo de autenticação utilizados no Windows NT até a versão 4.0. Ele explica o que são os *hashes* LM (Lan Manager) e NT, como são obtidos a partir da senha, onde ficam guardados, e como são utilizados no processo de autenticação para o acesso a recursos remotos de outras máquinas rodando Windows NT.

Inicialmente, uma breve introdução mostrará o funcionamento básico de uma rede Windows NT e como o conceito de segurança é generalizado para um conjunto de máquinas da rede. A seção seguinte explicará como as senhas são armazenadas e as deficiências desse processo. Em seguida, será explicado como ocorre o processo de autenticação via rede para acesso a recursos remotos. Na sequência, serão mostradas as fragilidades da concepção e da implementação do protocolo PPTP da Microsoft, utilizado para a construção de redes privadas virtuais sobre redes não seguras, como a Internet. Finalizando, comentaremos as tendências relevantes ao tema, como o novo Sistema de Arquivos Cifrado (EFS), que irá proteger os dados guardados no disco, e o sistema de autenticação Kerberos, do Windows NT 5.0/2000, que irá substituir os atuais protocolos de autenticação utilizados no Windows NT 4.0

6.2 Organização de uma Rede Windows NT

O Windows NT é o sistema operacional de rede da Microsoft que oferece um ambiente seguro para os usuários. Esta segurança é implementada no sistema através da associação de uma lista de controle de acesso (ACL - *Access-Control List*) para cada recurso do sistema. Cada elemento dessa lista (ACE - *Access-Control Entry*) possui um conjunto de descritores de segurança (SID - *Security Identifier*), descrevendo as permissões de acesso associadas a cada usuário/grupo que tem acesso ao recurso. Um recurso pode ser, por exemplo, um arquivo, uma impressora, ou um processo.

O usuário, para acessar esses recursos, deve autenticar-se no sistema, identificando-se com um *login* e uma senha de acesso que somente ele e o sistema conhecem[1]. A autenticação é baseada no conhecimento mútuo dessa senha. Após a autenticação, uma ficha de acesso é construída, contendo os SIDs do usuário, identificando-o unicamente perante o sistema e indicando quais privilégios especiais e quais grupos de usuários pertencem a ele. Essa ficha de acesso é mantida com o usuário (que não pode alterá-la), durante todo

[1]Na realidade, nem o sistema conhece tal senha; ele apenas conhece o valor criptografado da senha através de uma função criptográfica unidirecional (função *one-way* ou *hash* criptográfico)!

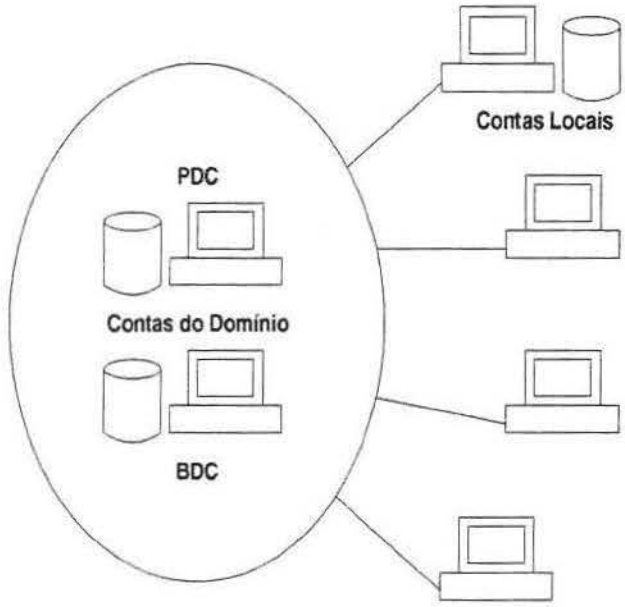


Figura 6.1: Uma típica rede Windows NT

o tempo que durar a sessão de *logon*. Quando um recurso é aberto para acesso, os SIDs dessa ficha são comparados com os do ACL do recurso, e se o ACL permite o acesso, um identificador é devolvido ao usuário para acessos posteriores ao recurso.

Para guardar essas informações, o sistema oferece uma base de dados de usuários e de política de segurança, que fica guardada em disco. Toda máquina baseada em Windows NT possui uma base de dados e de política de segurança local. Caso esteja configurada para fazer parte de um Domínio[2] , essa máquina poderá também utilizar a base de dados e de política de segurança do Domínio a que pertence.

Para uniformizar o acesso a recursos em um Domínio, a mesma base de dados de usuários do PDC, replicada em cada BDC, é usada para autenticar todos os serviços em todas as máquinas compondo esse domínio (*login*, acesso a arquivos etc).

É crítico para a segurança do sistema que o armazenamento e a transmissão dessas senhas pela rede seja feita de forma segura; ninguém que tenha acesso a esses meios (disco e rede) deveria ser capaz de copiar essas informações e utilizá-las para personificar outros usuários.

O Windows NT se utiliza de mecanismos de proteção proprietários que pecam por:

- tentarem oferecer segurança através de técnicas baseadas no ocultamento do algo-

[2]Um Domínio é um conjunto de máquinas executando Windows NT que compartilham uma política de segurança comum. Um servidor Windows NT especial, o Controlador de Domínio Primário (PDC - *Primary Domain Controller*), e um ou mais Controladores de Domínio Secundários (BDC - *Backup Domain Controller*), são responsáveis por manter uma base de dados de usuários e política de segurança do Domínio

Máquina	Op./Sec. (Approx.)	Tempo
Pentium II 450MHz	$4 \cdot 10^6$	571 anos
EFF DES-cracker	$92 \cdot 10^9$	9 dias

Tabela 6.1: Tempo para vasculhar um espaço de 2^{56} chaves. Os tempos foram extrapolados a partir dos dados da seção 6.4.4

ritmo de segurança. Tais técnicas fornecem uma falsa segurança porque alguém sempre pode vasculhar o código de máquina do sistema operacional em busca do algoritmo, que possivelmente é criptograficamente fraco;

- basearem-se em protocolos antigos que sugerem uma fraca concepção de um ambiente de segurança. Protocolos mais novos são oferecidos de tempos em tempos, mas todos tentam oferecer compatibilidade com essas versões anteriores, e portanto herdam várias fraquezas.

6.3 Algoritmos Criptográficos Utilizados no Armazenamento das Senhas

6.3.1 DES (*Data Encryption Standard*)

O DES [NBS77] é um algoritmo de criptografia que cifra blocos de texto claro de 64 bits (8 caracteres) em blocos de texto cifrado de mesmo tamanho, utilizando para isso uma chave de 56 bits. Esse tipo de algoritmo é inversível. Para obter o texto claro original, basta pegar o texto cifrado e cifrá-lo novamente com a mesma chave, informando ao algoritmo que se deseja decifrar.

O DES é um algoritmo de difícil criptoanálise. No entanto, possui um pequeno espaço de chaves (*key space*) em comparação com outros algoritmos, sendo vulnerável a ataques de força bruta. Apesar disso, ainda fornece uma segurança razoável e é amplamente utilizado.

A tabela 6.1 mostra o tempo máximo necessário para fazer um ataque de força bruta no DES em computadores domésticos, e numa máquina de US$200.000,00 especialmente projetada.

É interessante notar que atualmente um ataque ingênuo de força bruta no DES utilizando computadores domésticos é inviável. No entanto, isso não é verdade para máquinas especializadas.

6.3.2 MD4 *Message Digest*

O MD4 [RFC1320] é um algoritmo para criar *hashes* criptográficos. Ele recebe uma mensagem de tamanho arbitrário e a transforma num vetor (*hash*) de 16 bytes ou 128 bits. Não é possível inverter o processo. É computacionalmente muito difícil achar outra mensagem que resulte num mesmo *hash*. Portanto, esse *hash* de 128 bits pode ser usado para associar uma assinatura a uma mensagem. Esse imenso espaço de 2^{128} possibilidades para valores de *hash* torna inviável qualquer tentativa de ataque de força bruta.

O MD4 foi sucedido pelo MD5 [RFC1321], que evita alguns problemas de colisão, onde duas mensagens diferentes poderiam resultar num mesmo *hash*.

6.4 Armazenamento de Senhas

No Windows NT, as senhas e outras informações do usuário são armazenadas na base de dados do SAM (*Security Account Manager*). Por default, apenas o sistema e o administrador têm poderes para acessar seu conteúdo através do uso de funções específicas do sistema (APIs). Enquanto o sistema está executando, ele solicita acesso exclusivo ao arquivo no qual está a base de dados (*%SYSTEMROOT%/System32/config/SAM*), evitando qualquer tentativa de cópia do mesmo.

No entanto, existem métodos para se conseguir ler seu conteúdo sem precisar recorrer às APIs. Fitas de *backup* contêm o arquivo SAM, e o uso do utilitário RDISK para criar discos de reparo cria um diretório *%SYSTEMROOT%/Repair*, contendo um arquivo compactado de nome "SAM._" , que pode ser lido por qualquer um em uma instalação padrão do Windows NT.

Como no UNIX, as senhas não são armazenadas em texto claro. São armazenados apenas *hashes* das senhas. Para cada usuário são usados dois *hashes* chamados NT e LM, sendo este último apenas para garantir compatibilidade no processo de autenticação com versões de Windows anteriores ao Windows NT. Os dois *hashes* são guardados um ao lado do outro na SAM. Como será visto, o *hash* LM é muito mais susceptível a ataques, o que faz dele o alvo preferido de ataque quando se deseja quebrar alguma senha.

6.4.1 *Hash* LM

Usa DES para cifrar uma senha de no máximo 14 caracteres, dividida em dois blocos independentes de 7 caracteres. O *hash* resultante, um valor composto por 16 bytes, é formado pela cifragem de um "número mágico" usando os dois blocos de 7 caracteres como chave.

O procedimento da figura 6.2 é usado para gerar a chave DES de 56 bits a partir de cada parcela da senha. O bloco em preto de cada um dos 8 bytes indica o bit de paridade

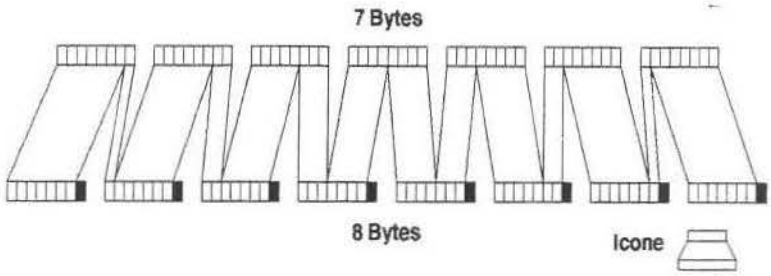


Figura 6.2: Criação dos bits de paridade da chave DES

que deveria haver a cada 7 bits obtidos da senha. Este bit é ignorado pelo DES. O ícone observado embaixo da figura será usado daqui em diante toda vez que tal procedimento for utilizado, por simplicidade.

O código ASCII de cada caracter da senha que seja uma letra minúscula é convertido para o código correspondente em letra maiúscula; bytes restantes não utilizados pela senha são preenchidos com conteúdo 0 (*padding*). Os sete bits são transformados em oito; os bytes resultantes do processo são usados como senha do DES. Os dois textos cifrados de 8 bytes formam o *hash* LM (ver figura 6.3).

O número mágico 0x4B47532140232425 não é guardado na máquina. Em vez disso, ele é guardado como uma constante resultante da cifragem desse número mágico usando DES com a chave composta apenas de bits 0. Para obter o número mágico, o sistema aplica o DES com chave 0x00000000000000 nessa constante.

6.4.2 *Hash* NT

O *hash* NT, um valor também composto por 16 bytes, é obtido a partir da aplicação do *hash* MD4 aos caracteres da senha, como pode ser visto na figura 6.4. Enquanto o *hash* LM usa na sua construção apenas os primeiros 14 caracteres da senha do usuário, o *hash* NT usa todos os caracteres da senha, até um máximo de 128 caracteres[3]. Nas primeiras versões do Windows NT a interface gráfica permitia ao usuário inserir no máximo 14 caracteres, efetivamente desperdiçando os possíveis 114 caracteres restantes.

6.4.3 O processo de Obscurecimento

Antes de serem armazenadas no banco de dados de usuários as senhas passam por um processo de obscurecimento (*obfuscation*), que visa tornar difícil a recuperação dos hashes

[3]Uma senha no Windows NT não pode ser maior do que 128 caracteres. Isto é uma limitação interna do Windows NT, e não do algoritmo MD4. O arquivo *smbencrypt.c* possui detalhes da implementação e desta limitação. Este arquivo pode ser encontrado no pacote Samba [SMB01], um pacote independente que replica o protocolo SMB do Windows NT e dialetos LM e NTLM.

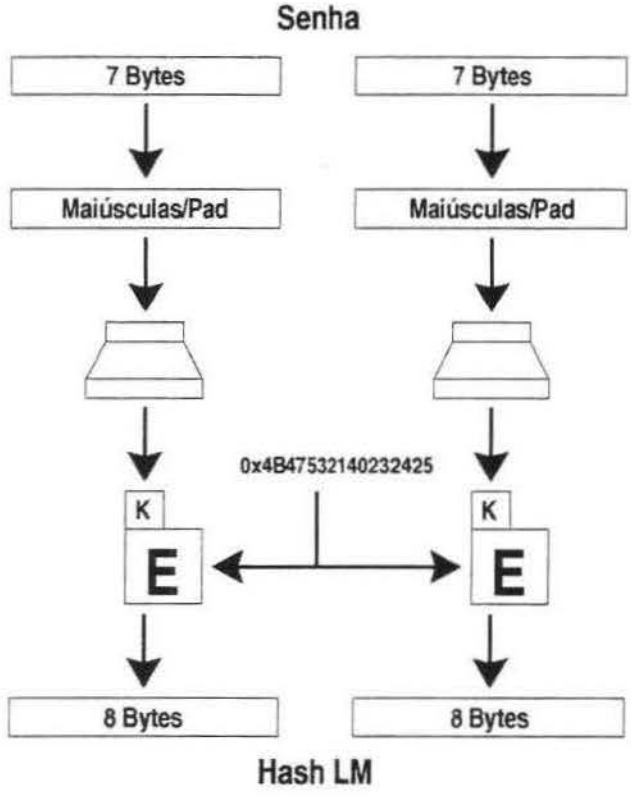


Figura 6.3: Criação do *hash* LM a partir da senha

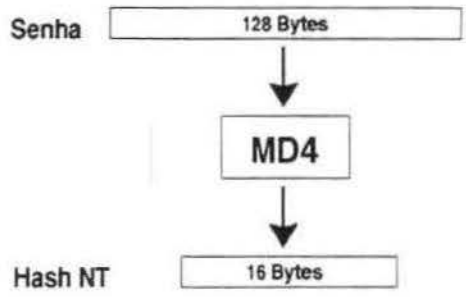


Figura 6.4: Criação do *hash* NT a partir da senha

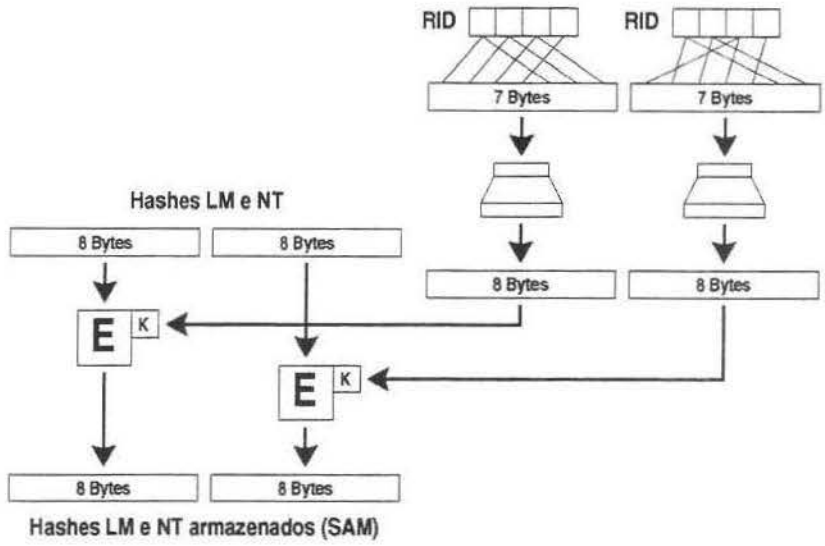


Figura 6.5: O processo de obscurecimento e armazenamento dos *hashes*

NT e LM caso alguém consiga o arquivo SAM. No entanto, a segurança deste método está baseada no segredo do processo, que é um clássico caso de má solução a um problema de segurança. O algoritmo pode ser quebrado analisando-se o código de máquina do sistema. Atualmente, o algoritmo já é conhecido, o que torna inútil esse processo de obscurecimento.

O procedimento de obscurecimento é simples: pega-se o RID do usuário, permuta-se alguns bits, e usa-se o resultado como chave do DES para cifrar o *hash*. O RID é um número que faz parte do SID do usuário, e é facilmente obtido com uma chamada de sistema.

Os utilitários PWDUMP e PWDUMP2, disponíveis na Internet, são usados no melhor quebrador de senhas (*cracker*) para o Windows NT (L0phtCrack [L97]). Eles utilizam o processo da figura 6.5 para obter os *hashes* desobscurecidos.

6.4.4 Deficiências dos *hashes*

Conseguindo o banco de dados de usuários e senhas, e sabendo o processo de ciframento, que tipos de ataques podem ser feitos?

Força Bruta: O invasor obtem o *hash* de todas as senhas possíveis e verifica quais ele encontra no banco de dados.

Dicionário: Semelhante ao método de força bruta. O invasor testa somente algumas senhas comuns e permutações das mesmas.

Máquina	Op./Sec. (Approx.)	Espaço de Senhas	Tempo
Pentium II 450Mhz	$4 \cdot 10^6$	43^7 (LM: maiúsculas + algarismos+pontuação)	19 horas
Pentium II 450Mhz	$4 \cdot 10^6$	69^7 (NT: maísculas + minúsculas + algarismos + pontuação)	22 dias
EFF DES-cracker	$92 \cdot 10^9$	69^7	1,34 min.

Tabela 6.2: Tempo para vasculhar vários espaços de senhas de 7 letras (ver texto)

Deficiências dos *hashes* LM

As duas metades que formam o *hash* são cifradas de modo independente, diminuindo substancialmente o espaço de chaves. Senhas de menos que 8 dígitos têm a mesma terminação: 0xAAD3B435B51404EE. Além disso, a senha é convertida para maiúsculas antes de ser cifrada, diminuindo mais ainda o espaço de chaves. Finalmente, não há o conceito de sal (*salt*), como em senhas no Unix. Portanto, no Windows NT, se duas pessoas tiverem a mesma senha os *hashes* gerados serão idênticos.

Em [L97], um Pentium II/450MHz vasculhou em 5,5 horas todo o espaço alfanumérico de senhas (letras maiúsculas e algarismos) compostas por até 7 letras - ou 36^7 combinações - a aproximadamente $4 \cdot 10^6$ de operações[4]/seg. A tabela 6.2 utiliza esta velocidade de operações para extrapolar os tempos necessários para realizar um ataque de força bruta em senhas de 7 letras compostas de caracteres comumente utilizados na construção dos *hashes* LM e NT. Ela também mostra o tempo em que a máquina DES-cracker [EFF98] realizaria a o mesmo ataque. Numa rápida comparação com o tempo necessário para quebrar o DES com a chave de 56 bits, nota-se que neste caso esse ataque é viável mesmo com computadores domésticos.

A razão disso é que o espaço de chave é reduzido de 256^7 no caso do DES para apenas 69^7, no caso do maior espaço de senhas considerado na tabela.

Deficiência dos *hashes* NT

Como o *hash* LM, o *hash* NT não possui sal. Portanto, se duas pessoas tiverem a mesma senha os *hashes* gerados serão idênticos. Por razões de compatibilidade com protocolos legados (Windows), os *hashes* são armazenados lado a lado.

[4]Uma operação é aqui considerada como sendo a cifragem de um bloco de texto claro usando a implementação de DES de [L97]

6.4.5 Soluções para o Armazenamento

O armazenamento de senhas é uma parte crítica da segurança do sistema. O fato do obscurecimento estar baseado no segredo do processo e o arquivo SAM poder ser copiado criou um mecanismo fadado ao fracasso. Dois procedimentos são sugeridos para aumentar a segurança do sistema.

O primeiro é a implementação de políticas rígidas de segurança para evitar o vazamento do SAM, atribuindo acesso mais restrito aos usuários do sistema, principalmente aos arquivos de sistema e ao(s) arquivo(s) SAM. Fitas e outros dispositivos de backup do sistema deveriam ser protegidos fisicamente contra manuseio não autorizado.

O segundo é a possibilidade de se acrescentar mais uma camada de obscurecimento. Dessa vez, no entanto, o segredo deveria estar baseado no conhecimento de uma chave secreta que poderia ficar guardada com o administrador. É exatamente o que a Chave de Sistema (*System Key*), introduzida no Service Pack 3, faz.

System Key

A partir do Service Pack 3 do Windows NT 4.0, a Microsoft passou a oferecer a possibilidade de usar uma Chave do Sistema (*System Key*) para proteger as senhas armazenadas. Essa Chave do Sistema cria uma nova camada de obscurecimento na base de dados de usuários, cifrando apenas o *hash* das senhas [Mic99a]. As outras informações na base de dados permanecem intocadas. O procedimento usa um algoritmo simétrico com chave de 128 bits. (Este algoritmo pode ser exportado para fora dos EUA porque cifra apenas senhas).

Ao contrário do processo anterior de obscurecimento, onde a segurança estava no segredo do processo (algoritmo), agora a segurança reside no conhecimento dessa Chave do Sistema.

Uma chave aleatória (PEK - *Password Encryption Key*) é criada para cada computador. A PEK é cifrada com a Chave do Sistema. A Chave do Sistema pode ser gerada:

- Aleatoriamente pela máquina e escondida no computador;
- Aleatoriamente e guardada num disquete;
- Por uma senha introduzida pelo administrador.

A primeira alternativa é mais cômoda; no entanto, por usar um algoritmo determinístico, permite que alguém analise o código de máquina do sistema e descubra esse algoritmo, conseguindo então reconstruir a mesma chave, seguindo os mesmos passos e condições da máquina. A segunda e terceira opções são mais fortes do ponto de vista da segurança, mas exigem a inserção de um disquete e da presença de um administrador na inicialização da máquina, respectivamente.

6.5 Autenticação via Rede

Caso o usuário esteja num Domínio, muito provavelmente ele necessitará acessar recursos que não estão na sua máquina local. Portanto, será necessária uma autenticação remota em algum servidor antes de acessar esses recursos.

No Windows NT essa autenticação remota é baseada no conhecimento comum da senha do usuário. O cliente envia para o servidor do recurso o nome de usuário e a senha. A senha pode estar obscurecida de alguma forma, de acordo com o dialeto SMB utilizado (como será visto na próxima seção), de modo a evitar que alguém mal-intencionado, ouvindo a rede de comunicação, possa roubar o par usuário/senha e usar essa informação roubada para se autenticar no sistema fingindo ser quem não é.

No servidor, o pedido de autenticação é recebido. Supondo que esse servidor seja um Controlador de Domínio (PDC, BDC), ele acessa sua base de dados do Domínio indexada pelo nome do usuário, recupera a senha e a processa usando o mesmo algoritmo do cliente. Caso o resultado final seja igual ao que o cliente enviou, então o cliente realmente sabia a senha e o processo de autenticação é finalizado com o envio do recurso requisitado. Caso o servidor não seja um Controlador de Domínio, então o servidor deve autenticar o pedido com algum Controlador de Domínio, redirecionando o pedido recebido do cliente. O servidor espera o Controlador de Domínio autenticar o pedido, e retorna o recurso requisitado ao cliente. Essa autenticação é feita logo após qualquer pedido de conexão usando-se o protocolo SMB.

O protocolo SMB (*"Server Message Block"*), desenvolvido originalmente com a IBM para o OS/2 nos anos 80, é usado pelo Windows NT como protocolo nativo para acessar os recursos da rede Windows [Hob97] [Ken97] [Mic97a] [Mic97b].

6.5.1 Evolução do Protocolo SMB

Os três principais dialetos SMB são:

Dialeto SMBcore: O cliente manda a senha em claro ao fazer um pedido de serviço.

Dialeto LM: O cliente faz o pedido de serviço e negocia com o servidor o dialeto a ser usado. O servidor responde com um desafio (*nonce*) que é uma cadeia aleatória de 64 bits. O cliente autentica-se com uma resposta construída a partir da concatenação de três cifragens do *nonce*. Cada cifragem utiliza um terço diferente de uma chave de 21 bytes construída a partir da concatenação do *hash* LM com 5 bytes nulos (ver figura 6.6).

Dialeto NTLM: Comporta-se como o dialeto LM. A diferença é que o cliente responde também o *nonce* cifrado com o *hash* NT (ver figura 6.6).

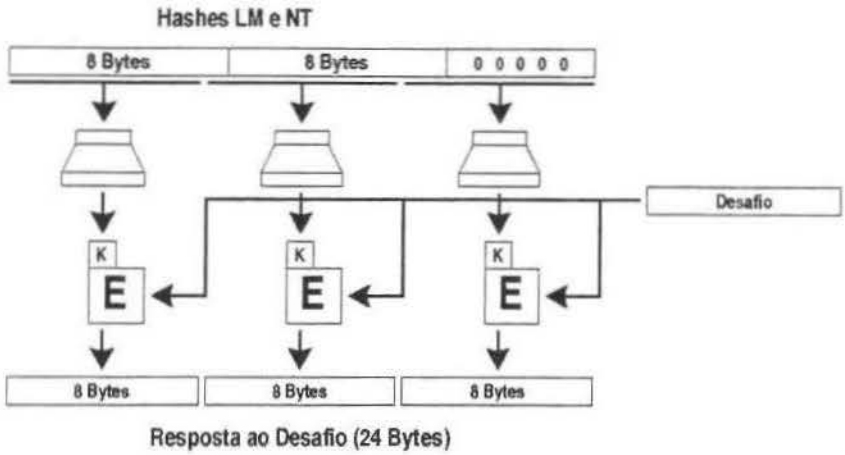


Figura 6.6: O processo de obscurecimento do *hash* na autenticação via rede

O SMB possui várias deficiências. Primeiro, não há autenticação do servidor. Portanto, o cliente não tem como saber se o servidor é confiável e, mesmo assim, deve responder ao desafio, efetivamente transmitindo informação sobre a senha do usuário. É possível, por exemplo, alterar uma página HTML de modo que uma URL do tipo *"file:////computador/sharename/mensagem.html"* aponte para algum servidor Windows NT falso. Quando alguém clicar no link, estará mandando o *hash* da senha para autenticação. Não há autenticação prévia. Um cliente como o Internet Explorer sempre responde.

Segundo, há o problema de equivalência de *hash*/senha. Apenas os *hashes* das senhas são usados para autenticação. Portanto, os *hashes* são equivalentes às senhas.

Por último, há o problema do ataque de *downgrade*. Por questões de compatibilidade com sistemas legados, o dialeto menos seguro é que determina o "nível" criptográfico a ser utilizado. Esse último problema foi resolvido a partir do Service Pack 3, onde há a opção para o cliente e para o servidor de escolher o "nível" mínimo do protocolo que deve ser aceito. No entanto, isso é algo que deve ser manualmente configurado pelo administrador.

Deficiências do dialeto SMBcore

A senha é enviada em claro, e está sujeita à captura ou a servidores falsos.

Deficiências do dialeto LM

A cifragem é feita em blocos independentes. Desta forma o espaço de chaves é reduzido substancialmente. Um par desafio/resposta está sujeito a ataques de força bruta ou dicionário em um tempo não muito superior àquele usado para quebrar uma senha no SAM.

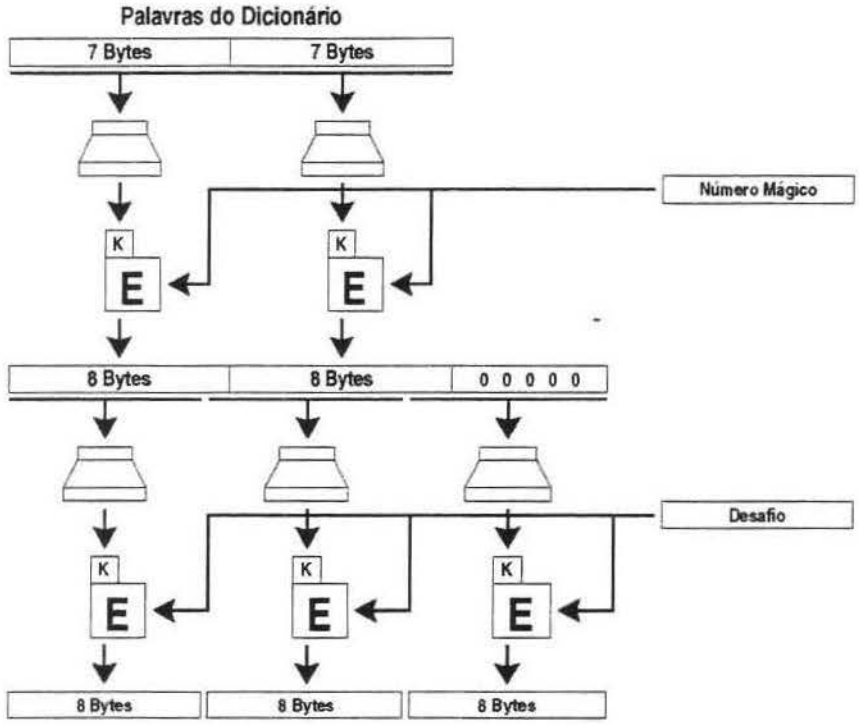


Figura 6.7: Um procedimento de ataque aos *hashes* obscurecidos

A eficácia deste ataque depende de como os pares desafio/resposta foram conseguidos. Caso o invasor esteja escutando o canal de comunicação, ele conhece pares distintos de desafio/resposta. Deve esgotar a busca para cada par. Ineficiente, mas factível. É semelhante ao ataque às senhas de uma máquina UNIX, sendo o desafio equivalente ao sal.

Caso o invasor seja um servidor falso, ele conhece pares desafio/resposta com o mesmo desafio. Muito mais eficiente que o primeiro caso. Ligeiramente mais lento que um ataque a base de dados SAM. Como o desafio é constante, permite a construção de uma tabela que leva um par desafio/resposta diretamente a uma senha em claro que conste no dicionário utilizado.

Caso a senha que deu origem ao *hash* tenha menos que 8 caracteres, um ataque otimizado pode ser feito da seguinte maneira, de modo a obter um *hash* a partir da resposta e do desafio (ver figura 6.7): com menos de 8 caracteres na senha, a metade de 8 bytes à direita no *hash* resultante H é constante. Assim, o terço da direita da resposta R é calculado diretamente a partir da cifragem do desafio utilizando-se como chave a concatenação dos dois bytes constantes à direita desta metade direita de H e de 5 bytes nulos.

O terço do meio da resposta R é calculado cifrando-se o desafio com uma chave com-

posta dos seis bytes constantes à esquerda da metade direita de H e do byte à direita da metade esquerda de H. Este byte à direita não é conhecido, pois faz parte da metade de H que não é constante para senhas com menos de 8 caracteres, mas há apenas 256 possíveis valores para ele. Todos estes possíveis valores são experimentados, até o resultado corresponder ao terço do meio da resposta R. Deste modo, o byte à direita da metade esquerda do hash H fica determinado.

O terço da esquerda da resposta R é calculado a partir do desafio e dos sete bytes da esquerda da metade esquerda do *hash* H. Estes sete bytes não são conhecidos e devem ter todas as combinações possíveis experimentadas até o resultado corresponder ao terço da esquerda da resposta R. No entanto, as combinações possíveis não são muitas para o *hash* H quando este *hash* é do tipo LM, como já foi visto. Isso permite que um ataque de dicionário ou força bruta seja aplicado de modo a fornecer as diversas possíveis combinações de valores para esta metade da esquerda. Apenas senhas com menos de 8 caracteres precisam ser consideradas para este ataque. Uma outra otimização se aplica: os sete bytes à esquerda da metade esquerda do *hash* LM calculado somente devem ser considerados para calcular o resultado quando o byte à direita for igual ao já determinado durante o ataque ao terço do meio da resposta R.

Deficiências do dialeto NTLM

A resposta ao desafio inclui tanto os 24 bytes obtidos com o *hash* LM quanto os 24 bytes gerados com o *hash* NT.

Como foi visto, o *hash* NT utiliza procedimentos mais seguros para sua criação: considera caracteres maiúsculos/minúsculos de maneira distinta e suporta senhas maiores, dificultando ataques de dicionário; e a senha é transformada em um *hash* em uma única operação criptográfica utilizando o algoritmo especializado MD4, dificultando ataques otimizados. Assim, a resposta gerada com o *hash* NT é mais segura.

No entanto, como as duas variedades de respostas LM e NT são enviadas na resposta à autenticação NTLM, o menor denominador comum de segurança - a resposta gerada a partir do *hash* LM - pode ser usada para realizar um ataque. Portanto, o dialeto NTLM é tão vulnerável quanto o dialeto LM.

Os dois dialetos permitem ataques de *replay* e integridade.

6.5.2 Soluções para Autenticação

A solução ideal seria a substituição completa do atual do sistema de autenticação. Como isso não é possível a curto prazo devido a problemas de compatibilidade, alguns procedimentos podem aumentar a segurança do protocolo.

O primeiro seria poder estabelecer o grau mínimo de segurança aceitável. Por incrível que pareça, isso não estava disponível até o Service Pack 3. A partir do Service Pack 3, há a opção do cliente não responder a servidores que somente aceitam senhas em claro, e de servidores recusarem clientes que não entendem o dialeto NTLM. A partir do Service Pack 3 também há a possibilidade de o cliente enviar somente o *hash* NT. Isso evita que alguém que esteja escutando obtenha o *hash* LM, mais vulnerável. No entanto, quando esta opção foi lançada, ainda como uma correção pós Service Pack 2, a utilização dessa opção evitava que alguns serviços do Windows NT funcionassem.

Outra medida útil seria estabelecer algum tipo de verificação na integridade da mensagem, de modo a evitar que terceiros alterem ou repitam o conteúdo das mensagens (como foi feito em [Ash98]). Isto foi introduzido no Service Pack 3 com o SMB *signing*, usando o algoritmo MD5.

Embora a curto prazo não seja possível substituir o atual esquema de autenticação, a próxima versão do Windows NT contará com o sistema de autenticação Kerberos.

6.5.3 SMB *Signing*

Abaixo está um resumo do protocolo SMB ($C \rightarrow S$ indica Cliente envia ao Servidor):

$C \rightarrow S$ [Mensagem negociação de dialeto];

$S \rightarrow C$ [Dialeto escolhido, Ds];

$C \rightarrow S$ [Usuário, $\{Ds\}Hash$[5]];

$S \rightarrow C$ [userinfo, logonscript, UID, SIDs, etc...] (em claro!)

Após o cliente comunicar ao servidor quais os dialetos que ele suporta, o servidor responde informando qual o dialeto que ele prefere entre os sugeridos e enviando um desafio aleatório Ds. O cliente cifra esse desafio aleatório Ds usando o hash da senha como chave para o algoritmo DES e envia o resultado pela rede. O servidor compara o resultado recebido com o que ele calculou independentemente e se os dois resultados forem iguais, então o cliente sabia a senha. Por último, o servidor envia de volta as informações do recurso requisitado. Por exemplo, no processo de logon do usuário, o caminho para o script de logon, os SIDs do usuário, e outras informações do usuário. Essas últimas informações são todas enviadas em claro!

Antes do SMB Signing, era possível utilizar algum ataque do tipo *man in the middle* para alterar, repetir e estragar os pacotes SMB enviados entre cliente e servidor. Por exemplo, era possível alterar os SIDs [Ash98] enviados de volta ao cliente de modo que ao invés do usuário pertencer a um certo grupo de usuários, pertencesse ao grupo dos administradores! Não havia como nenhuma das duas partes saber se o pacote recebido

[5] O desafio enviado (Ds), cifrado com o DES usando como chave o enghash da senha de acordo com o dialeto escolhido.

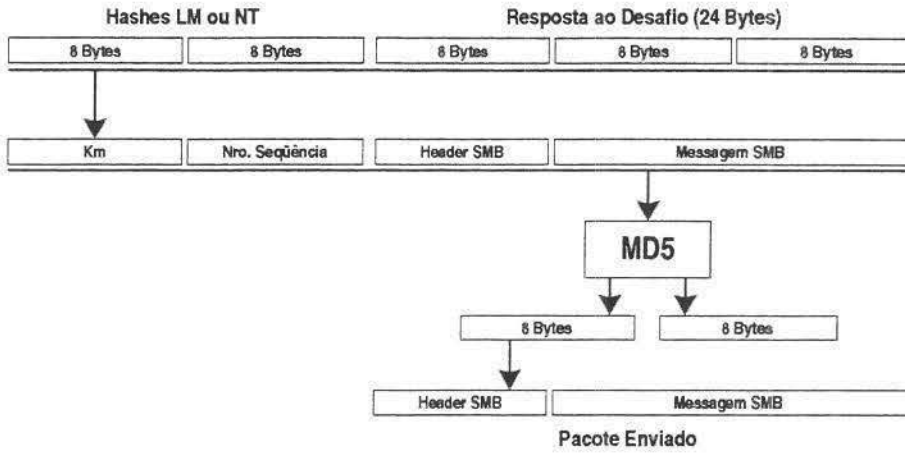


Figura 6.8: O procedimento de assinatura do SMB Signing

tinha sido alterado por terceiros.

Com o SMB Signing, criou-se uma assinatura MD5 baseada:

- na resposta do cliente (ou seja, na resposta do desafio, que é baseada no identificador de sessão Ds, de modo a associar uma assinatura diferente a cada sessão diferente do mesmo usuário (ver figura 6.8);
- no *hash* do usuário;
- na mensagem enviada;
- e num número de sequência.

Essa assinatura permitiu criar um mecanismo para garantir a integridade e a autenticidade do pacote. Porém, embora ela evite que os dados sejam adulterados, ela não garante sigilo dos dados, que ainda trafegam em claro.

Quebrar o SMB Signing pode ser feito sem muito esforço: a assinatura depende da resposta do cliente, do *hash*, da mensagem e do número de sequência. A mensagem e a resposta do cliente são enviados em claro pela rede, e portanto são conhecidos. A implementação da Microsoft inicia o número de sequência sempre a partir do zero, e portanto ele pode ser deduzido observando-se o número de pacotes utilizado naquela sessão. O único fator não conhecido é o *hash* do usuário. Retorna-se, portanto, ao problema de se descobrir qual o hash que deu origem à resposta do cliente. Isto pode ser tentado através de um ataque de dicionário ou um ataque de força bruta no espaço de chaves, como já visto anteriormente.

A figura 6.8 é um esquema do funcionamento do SMB *signing*. Os 16 bytes do *hash* LM ou NT (de acordo com o dialeto usado) são combinados com os 24 bytes da resposta ao desafio obtida, formando uma chave K_m.

Por que não usar apenas a resposta ao desafio, que já é uma função do *hash*? A resposta ao desafio é uma função tanto do número de sessão D_s (o desafio original) quanto do *hash* LM ou NT enviado. No entanto, a resposta passa em claro pela rede. Portanto, para que a chave K_m seja secreta, é necessário incorporar uma informação secreta que somente os dois lados conhecem. Neste caso, a informação secreta é o *hash* da senha.

Um número de sequência conhecido pelos dois lados também é acrescentado, para evitar o *replay* de mensagens.

Essas informações, junto com o *header* SMB e a mensagem, são usadas como entrada para formar uma assinatura baseada no resultado de uma função de *hash* MD5. Os primeiros 8 bytes desse resultado são então incorporados ao *header* SMB e o pacote é enviado. Do outro lado, um procedimento semelhante é executado para verificar a integridade do pacote.

6.6 Redes Privadas Virtuais (*Virtual Private Networks*)

O protocolo PPTP (*Point-to-Point Tunneling Protocol*) é usado para assegurar privacidade e segurança em conexões PPP estabelecidas ("tuneladas") sobre conexões TCP/IP de redes não confiáveis, como a Internet.

Isso é conseguido através de criptografia. A especificação PPTP permite que a segurança seja estabelecida por qualque algoritmo criptográfico.

A implementação PPTP da Microsoft é parte do Windows NT Server e pode ser facilmente instalada. Devido a isto, é amplamente utilizada. Infelizmente, o protocolo de cifragem dessa implementação (MPPE *Microsoft Point-to-Point Encryption Protocol*[PZ99]) é fraco e susceptível a um ataque de dicionário. Ataques contra a cifragem e vários ataques de negação de serviço podem ser feitos.

Há três tipos de autenticação suportados nessa implementação:

Senha em claro: a senha é enviada em claro

***Hash* em claro:** o *hash* da senha é enviado em claro

Desafio/resposta: usa o protocolo de desafio/resposta da Microsoft.

O último tipo é chamado "Autenticação Microsoft" na documentação do usuário, e deve estar habilitada para ser possível cifrar pacotes PPTP.

O MPPE tem uma metodologia para cifrar os pacotes PPTP. Supõe-se a existência de uma chave secreta conhecida por ambos os lados da conexão, e usa-se o cifrador de fluxo RC4 com uma chave de 40 ou 128 bits, conforme as restrições de exportação.

O MPPE apresenta muitos problemas [Sch98]. A chave de cifragem do fluxo de dados é derivada da senha do usuário, tornando-a suscetível a um ataque de dicionário. No modo

40 bits, a chave é obtida aplicando-se a função de hash SHA (*Secure Hash Algorithm*) [NIST93] no *hash* LM do usuário. Os primeiros 24 bits do resultado de 64 bits são sempre forçados para 0xD1269E. Como a chave deriva da senha do usuário, para cada usuário *a mesma chave* é usada para o RC4 no modo 40 bits, ou seja, o fluxo de chaves que o RC4 gera para ser combinado com a mensagem é sempre o mesmo! Isso permite criar um ataque baseado numa tabela contendo os trechos mais comuns encontradas numa transmissão PPTP (*headers* SMB, sequências de zeros, etc) cifradas com a chave derivada de palavras de dicionário comumente usadas como senha. Caso algum desses trechos cifrados da tabela seja encontrado na mensagem escutada, é provável que a chave RC4 seja aquela usada para cifrar a linha da tabela contendo o texto cifrado.

No modo 128 bits, os 64 bits retornados pelo SHA usando o *hash* NT são combinados com 64 bits de um número aleatório (*nonce*), que passa em claro pela rede. Embora seja possível determinar na hora qual a senha, o método 128 bits evita que se pré-compute *a priori* uma tabela de conversão para trechos bem escolhidos, baseada num dicionário de palavras comuns para senhas, mas não evita que um ataque semelhante seja aplicado *on-the-fly*.

Como se isso não bastasse, o RC4 é usado no modo OFB (*Output-feedback mode*) para cifrar o canal byte a byte. Como a chave é a mesma para cada usuário, isto significa que não importa a posição em que um trecho procurado esteja, no início ou fim da mensagem; desde que esta posição não mude, a sua representação cifrada também será sempre a mesma, independentemente dos bytes cifrados que apareceram anteriormente na mensagem.

Finalmente, a cada 256 pacotes a chave muda deterministicamente, aplicando-se novamente a função SHA para a chave atual. Mesmo apesar de ser possível determinar essa nova chave sabendo-se a anterior, isso não é necessário: o protocolo determina que um pacote com um *header* inválido deve reiniciar a sequência de pacotes para zero, *sem mudar a chave*. Portanto, um ataque completo deveria enviar um pacote com *header* inválido antes do contador de sequência de pacotes atingir 256, evitando a troca de senha do RC4.

Uma segunda versão atualizada do protocolo [Z99a] deriva a chave K do *hash* NT. Cria duas chaves mestras KM, uma para transmissão e outra para recepção. Estas chaves usam SHA para criar as chaves usadas no RC4.

Muitas outras informações podem ser obtidas em claro e sem opção de cifragem. É possível obter o IP das duas partes, o nome NetBIOS da máquina cliente, o *username* do cliente, *hashes* da senha do cliente etc.

Além do projeto de cifragem falho, a implementação do canal de controle do protocolo é falha. De acordo com [Sch98], "é trivial fazer um Windows NT travar com uma tela azul de tipos variados, testando valores inválidos em campos aleatórios do *header*".

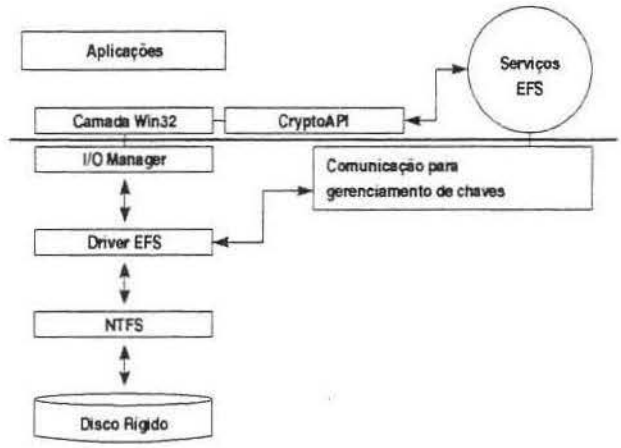


Figura 6.9: Esquema dos módulos envolvidos no funcionamento do EFS

6.7 Tendências Futuras

6.7.1 *Encrypted File System*

O NTFS (*NT File System*) oferece acesso seletivo e controlado aos arquivos do disco. No entanto, já existem *drivers* para Linux e DOS que permitem ler partições NTFS acessando todos os arquivos inescrupulosamente. Isto pode ocorrer caso a máquina possua outros sistemas operacionais (Linux, DOS etc) em outras partições do disco, ou então a segurança física não seja suficiente, permitindo carregar um sistema operacional diferente na máquina a partir de um disquete de *boot*. A única maneira de controlar efetivamente o acesso aos dados é cifrando a informação.

O *Encrypted File System* (EFS) é o sistema de arquivos cifrados que irá ser lançado junto com o Windows NT 5.0/2000 [Mic99c]. Ele permitirá usar qualquer algoritmo de ciframento. A primeira versão usará o DES com chave de 56 bits nos EUA e de 40 bits para os outros países.

Com o EFS, será possível a qualquer usuário cifrar/decifrar qualquer arquivo que ele tenha acesso de forma transparente, sem se preocupar com o processo. Os arquivos são guardados no disco cifrados. Há um recurso de recuperação, no entanto, que permite que o administrador tenha acesso a todos os arquivos.

Um *driver* EFS interceptará os acessos ao sistema de arquivos NTFS de forma transparente (ver figura 6.9). Um serviço EFS gerenciará as chaves do usuário utilizando-se de funções da *Crypto API* do Windows NT e transferirá essas informações ao *driver* EFS.

O EFS funciona codificando os arquivos usando um algoritmo de cifragem simétrico e uma chave *aleatória* de cifragem associada ao arquivo (FEK, de *File Encryption Key*), como pode ser visto na figura 6.10. A FEK, por sua vez, é cifrada usando uma ou mais

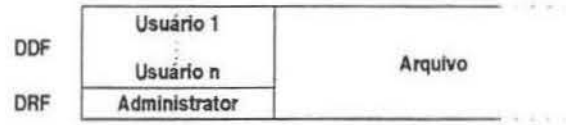


Figura 6.10: As duas listas de chaves de deciframento do arquivo

chaves públicas de cifragem para gerar uma lista de FEKs cifradas (DDF - *Data Decription Field*). A FEK também é cifrada usando uma ou mais chaves de recuperação (DRF - *Data Recovery Field*).

O usuário, usando sua chave privada guardada em algum lugar seguro (*smart card* ou algum outro dispositivo seguro) pode obter a chave FEK e portanto decifrar o arquivo para uso pessoal.

O mecanismo é de concepção robusta, usando chave aleatórias para cifrar, ao invés de derivações de senhas, e utilizando chaves públicas/privadas.

6.7.2 Kerberos no NT

Como foi visto neste artigo, a autenticação no Windows NT é fraca do ponto de vista criptográfico, enviando informações desnecessárias na hora da autenticação, não protegendo a integridade dos dados enviados, e permitindo ataques otimizados de *hashes* cifrados enviados pela rede.

No Windows NT 5.0/2000, o Kerberos [Mic99b] será usado como um novo sistema de autenticação centralizado. Todo o acesso a recursos do Domínio e entre Domínios será autenticado pelo Kerberos.

No Windows NT, cada PDC e BDC também terá um *Key Distribution Center* (KDC) definido no protocolo Kerberos (ver figura 6.11). O KDC rodará no mesmo espaço de endereçamento protegido do *Local Security Authority* (LSA) e do serviço de diretório do Windows NT (*Active Directory*) e será carregado pelo LSA no momento do *boot* do sistema. Seu *cache* de credenciais e chaves será apenas em memória volátil (RAM), evitando que credenciais sejam escritas no disco. O acesso ao banco de dados local de usuários será feito através do serviço de diretório utilizando-se de ACLs para proteger o acesso às informações. Todos os serviços necessários à autenticação (LSA, KDC e SAM) estão locais no PDC/BDC e rodam num espaço protegido, evitando o acesso indevido aos dados dos mesmos.

O Kerberos será implementado como um *Security Support Provider* (SSP), acessível através de uma interface SSP (SSPI, ver figura 6.12). O protocolo NTLM do Windows NT 4.0 também será acessível por motivos de compatibilidade, mas o protocolo de autenticação para Domínios contendo apenas Windows NT5.0/2000 é o Kerberos.

Os parâmetros Kerberos que serão utilizados incluem tempo de vida máximo do ticket

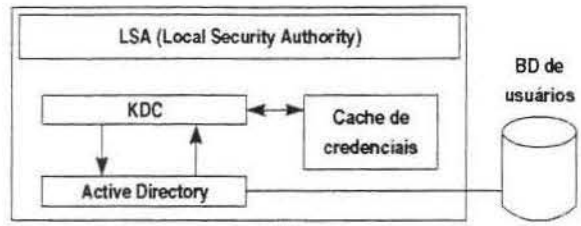


Figura 6.11: Esquema geral dos módulos envolvidos no funcionamento do Kerberos

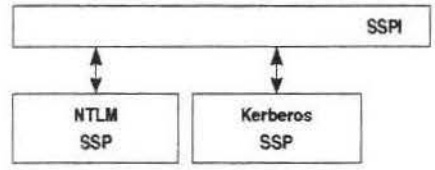


Figura 6.12: As interfaces para acesso aos sistemas de autenticação do Windows 2000

de 10 horas, tempo de renovação máximo do ticket de 7 dias, tolerância máxima na diferença da sincronização de relógios de 5 minutos. A permissão para delegar autenticação usando *forwardable tickets* poderá ser ajustada para qualquer combinação de usuário e máquina.

Entre os serviços autenticados via Kerberos incluem-se o PrintSpooler, o acesso a arquivos utilizando-se SMB, o DFS (*Distributed File System*), o serviço de RPC, e autenticação IPSec[6] máquina à máquina.

6.8 Conclusão

Depreende-se, após a análise destes mecanismos criptográficos no Windows NT, que a Microsoft foi muito ingênua na construção dos mesmos, projetando mecanismos de proteção facilmente quebráveis. Talvez isto tenha ocorrido por não prever que algum dia sistemas com Windows NT, projetados inicialmente para rodar em pequenas redes locais, pudessem estar ligados a uma rede mundial como a Internet, contendo informações que compensassem o estudo da quebra desses mecanismos.

A implementação do protocolo PPTP pela Microsoft é uma outra mostra de ingenuidade e descuido criptográfico. Infelizmente, não parece haver muita saída nesse caso a não ser refazer o MPPE. De acordo com os últimos *drafts* do MPPE disponíveis na Internet, este não parece ser o caso. Entretanto, a padronização do protocolo de segurança L2TP[7]

[6] O IPSec (*IP Security Protocol*) é um protocolo que oferece mecanismos de segurança para o protocolo IP. Mais informações podem ser encontradas em http://www.ietf.org/html.charters/ipsec-charter.html

[7] O L2TP (*Layer 2 Tunneling Protocol*), definido no RFC2261, é um protocolo para estabelecer ("tunelar") sessões PPP em cima de vários tipos de rede. Mais informações podem ser obtidas em

e o uso de IPSec poderão trazer segurança a redes Windows NT.

Felizmente, a utilização do Kerberos, um sistema de autenticação devidamente escrutinizado, no Windows NT5.0/2000, como um sistema para centralizar a autenticação de todos os serviços e acesso a recursos, e a presença de um Sistema de Arquivos Cifrados são boas tendências na busca de um sistema mais robusto e seguro.

http://www.ietf.org/html.charters/l2tpext-charter.html

Capítulo 7

Análise de Segurança do SNTP

Este capítulo apresenta o protocolo de sincronização de relógios *Simple Network Time Protocol* (SNTP) utilizado no Windows 2000 e características particulares da implementação deste protocolo neste sistema que poderiam ser utilizadas para alterar, em certas condições, o relógio local de máquinas na rede. O relógios das máquinas dão suporte a vários serviços e protocolos (por exemplo, o protocolo Kerberos).

7.1 Descrição do protocolo SNTP

O *Simple Network Time Protocol* (SNTP), definido em [RFC1769], é o protocolo padrão utilizado para a sincronização do relógio das máquinas rodando Windows 2000. A sincronização de relógio é crítico no Windows 2000 porque o protocolo de autenticação padrão (Kerberos) usa o relógio local como parte do processo de geração do *ticket* de acesso. Se os relógios entre máquinas do Domínio não estivessem sincronizados, o protocolo Kerberos iria cessar seu correto funcionamento. O STNP é uma simplificação do protocolo *Network Time Protocol* (NTP)[RFC1305]. No STNP, a maior parte dos campos do NTP é preenchida com dados fixos. Ele utiliza um algoritmo simplificado para calcular a hora correta, possui menos verificações de erros e o campo de autenticação é opcional.

Hierarquia de Sincronização

O SNTP funciona através de uma hierarquia de sincronização que pode ser representada como uma árvore cujas folhas são as máquinas SNTP clientes que nomeiam como seus servidores pais de sincronização de tempo seus respectivos servidores de autenticação/Controladores de Domínio. Neste caso, as máquinas clientes estão no 'modo cliente' do SNTP e os Controladores de Domínio estão no 'modo servidor' do SNTP. Se este Controlador de Domínio se tornar indisponível, o cliente reenvia uma requisição para um

outro Controlador de Domínio.

Todos os Controladores de Domínio em um Domínio nomeiam um Controlador de Domínio Principal como seu servidor de sincronização de tempo. Aqui, os controladores de domínio estão no 'modo cliente' do SNTP, e o Controlador Principal está no 'modo servidor'. O Controlador Principal pode ser um cliente de um outro Domínio, ou então a raiz de toda a hierarquia. Neste caso, ele pode ser manualmente configurado para sincronizar com uma fonte de sincronização externa confiável, como o *United States Naval Observatory*.

Daqui para frente, a máquina no 'modo servidor' do SNTP será chamada apenas 'servidor', e as máquinas no 'modo cliente' serão chamadas apenas 'clientes'.

7.2 SNTP no Windows 2000

Há um serviço novo no Windows 2000, específico para a sincronização do relógio local, chamado 'W32Time Service' , que implementa o protocolo SNTP. Este serviço de sincronização pode ser manualmente ativado através do comando

```
w32tm -v -once
```

Isto provoca o início do processo de sincronização do relógio local no modo *verbose*. Este comando induz artificialmente o processo de sincronização, que normalmente ocorre periodicamente, de acordo com a configuração do serviço de sincronização local.

De acordo com [Q223184], há uma entrada no Registro do Sistema responsável por controlar a periodicidade da sincronização. O valor da entrada deve ser um entre:

- *0* = uma vez por dia
- *65535* ou *"BiDaily"* = uma vez a cada 2 dias
- *65534* ou *"Tridaily"* = uma vez a cada 3 dias
- *65533* ou *"Weekly"* = uma vez a cada semana (7 dias)
- *65532* ou *"SpecialSkew"* [padrão] = uma vez a cada 45 minutos até 3 sincronizações com sucesso, então a cada 8 horas (3 vezes por dia)
- *65531* ou *"DailySpecialSkew"* = uma vez a cada 45 minutos até uma sincronização com sucesso ocorrer, então uma vez por dia
- *freq* = *freq* vezes por dia

Outras entradas são responsáveis por determinar se o serviço deve funcionar como um cliente ou servidor SNTP, se o tempo informado é confiável, e outros parâmetros de configuração.

De acordo com [Q224799], além de realizar a sincronização periodicamente, o cliente também sincroniza seu relógio toda vez que é iniciado (*boot*) . O algoritmo de ajuste do relógio local determina que se a hora alvo está à frente da hora local, a hora local é imediatamente ajustada para a hora alvo. Se a hora alvo está atrás da hora local, o relógio local é atrasado durante os próximos 20 minutos para alinhar as duas horas, a não ser que a hora local esteja mais que 2 minutos defasada, onde a hora local é imediatamente ajustada.

7.3 Análise de segurança do SNTP

O protocolo SNTP foi projetado para permitir a um cliente obter a hora correta de um servidor de tempo. Esta informação (a hora correta) é pública, e não deveria ser escondida de possíveis observadores no caminho, mas deveria ser protegida contra modificações de terceiros.

O SNTP permite a inclusão de um campo opcional de autenticação. Este campo de autenticação permite ao cliente verificar se o conteúdo do pacote foi modificado após ter sido criado pelo servidor.

Há uma entrada no Registro do Sistema do cliente[1], que determina a presença ou não deste campo de autenticação. Ela pode ter três valores:

Nt5DS Sincronizar com a hierarquia do Domínio, ou fonte configurada manualmente (padrão)

NTP Sincronizar a uma fonte configurada manualmente.

NoSync Não sincronizar o relógio local

Apenas o valor 'Nt5DS' determina a presença do campo de autenticação, e deve ser usado apenas quando o servidor é uma máquina executando Windows 2000 no papel de Controladora de Domínio.

Caso o servidor seja uma máquina executando Windows NT 4.0 ou anterior, ou ainda um outro sistema operacional, o valor no cliente deveria ser 'NTP', por motivos de compatibilidade. Este valor configura o serviço de sincronização no cliente para não enviar o campo de autenticação.

[1] HKLM\SYSTEM\CurrentControlSet\Services\W32Time\Parameters\Type

Evitando a criação e envio do campo de autenticação

Ambientes heterogêneos (Windows 2000 e Windows NT 4.0/anteriores) deveriam ser evitados: máquinas Windows 2000 deveriam sempre possuir um servidor de sincronização de tempo também Windows 2000. Um valor 'Nt5DS' em um cliente rodando Windows 2000, mas pertencente a um domínio NT 4.0, retorna um erro '*Member of an NT4 domain. Cannot synchronize.*' Um valor 'NTP' em um cliente rodando Windows 2000, pertencente a um domínio Windows 2000, força a criação de pacotes sem campo de autenticação. Assim, é possível burlar a criação dos campos de autenticação dos pacotes SNTP trocando o valor 'Nt5DS' por 'NTP' no cliente. Isto permite o uso de ataques *man-in-the-middle* para alterar as mensagens de sincronização de relógio.

Traço de pacotes

O campo de autenticação é construído através do conhecimento mútuo de um conjunto de chaves secretas de 64 bits[RFC1305, Apendice C] entre servidor e cliente. Esta chave secreta é identificada por um índice (*key_id*), enviado nos primeiros 32 bits do campo de autenticação. Os bits restantes são formados pela aplicação de um algoritmo de *checksum* criptográfico. De acordo com a especificação do SNTP, este *checksum* deveria ter 64 bits e ser gerado com a utilização de DES, mas em pacotes reais foi observado um *checksum* de 128 bits.

O seguinte traço de pacotes foi observado entre uma máquina *Windows 2000 Professional* agindo como cliente SNTP e uma máquina *Windows 2000 Advanced Server* agindo como servidor SNTP do tipo nt5DS. Somente 2 pacotes são mostrados, mas o traço completo compreende 14 pacotes, ou 7 interações cliente-servidor, usadas para diminuir a incerteza de tempo associada a cada interação.

Pacote 1: Cliente → *Servidor*

```
0000: xx xx xx xx xx xx yy yy yy yy yy yy 08 00 45 00
0010: 00 60 01 dd 00 00 80 11 ec 4c zz zz zz zz ww ww
0020: ww ww 04 4f 00 7b 00 4c 9c a8 13 00 0b 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 bc e5 fd 65 de 76 c8 b3 55 04 00 00 5f 93
0060: d5 77 a8 fb 06 00 2a 00 00 00 28 fe 06 00
```

Neste pacote pode ser observado:

- 0x0022- 0x0023, o valor da porta UDP de origem do cliente (0x044f = 1103)

- 0x0024-0x0025, o valor da porta UDP de destino do servidor(0x007b = 123)
- a partir de 0x002a, sublinhado, o pacote SNTP
- 0x002a, 'SNTP versão 2' , Modo Cliente
- 0x0052-0x0059, a hora do cliente (=3169189221,3732326579 segundos desde 1 jan 1900)
- 0x005a-0x006d, em ***itálico,*** o campo de autenticação. Os primeiros 4 bytes correspondem ao índice da chave secreta, e os 16 bytes restantes correspondem ao *checksum.*

Pacote 2: Servidor → *Cliente*

```
0000: xx xx xx xx xx xx yy yy yy yy yy yy 08 00 45 00
0010: 00 60 1c 84 00 00 80 11 d1 a5 zz zz zz zz ww ww
0020: ww ww 00 7b 04 4f 00 4c d1 76 14 02 0b f9 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 bc e5 fd 65 de 76 c8 b3 bc e5 fd 65 f2 2d
0050: 0e 55 bc e5 fd 65 f2 2d 0e 55 55 04 00 00 37 99
0060: 0e ed 92 f0 27 ef a8 16 33 4e b2 87 01 4c
```

Neste pacote pode ser observado:

- 0x002a, 'SNTP versão 2' , Modo Servidor
- 0x002b, valor 2 (= Servidor Secundário)
- 0x0042-0x0049, a hora do cliente enviada no pacote 1
- a hora do servidor, repetida em 0x004a-0x0051 e 0x0052-0x0059 (=3169189221,4063039061 segundos desde 1 jan 1900)
- 0x005a-0x006d, em ***itálico,*** o campo de autenticação. Os primeiros 4 bytes correspondem ao identificador da chave secreta, e os 16 bytes restantes correspondem ao *checksum.*

7.3.1 Ataques ao protocolo

O SNTP é uma simplificação do NTP, e possui um algoritmo simples para obter a hora correta a partir das mensagens trocadas entre o cliente e o servidor. Devido a isso, é possível provocar mensagens com a hora defasada, através da inundação da rede por pacotes, de modo a provocar atrasos na comunicação entre o cliente e o servidor.

Nas configurações de máquinas onde a inclusão do campo opcional de autenticação não está habilitada, vários diferentes tipos de ataques são possíveis: primeiro, alguém pode fazer uma máquina personificar um servidor de tempo falso; também é possível alguém modificar algumas ou todas as mensagens enviadas por um servidor de tempo; e alguém pode reenviar alguma mensagem do servidor de tempo coletada anteriormente.

No caso do campo opcional de autenticação estar habilitado (o padrão), o ataque consistiria em descobrir o valor da chave secreta entre o cliente Windows 2000 e o Controlador de Domínio Windows 2000 para forjar autenticadores válidos. Esta chave secreta está associada ao índice (bytes 0x005a-0x005d dos pacotes) que é enviado junto com o autenticador.

Capítulo 8

Kerberos no Windows 2000

O protocolo de autenticação Kerberos define as interações entre um cliente e o Serviço de Autenticação da rede conhecido como o Centro de Distribuição de Chaves (KDC, de *Key Distribution Center*). O Windows NT implementa um KDC como um serviço de autenticação em cada Controlador de Domínio. O Domínio do Windows NT é equivalente a um *realm* do Kerberos, mas é ainda referenciado como um Domínio. A implementação do Kerberos no Windows NT é baseada no RFC 1510 do protocolo Kerberos [NK93]. O ambiente do cliente Kerberos é implementado como um provedor de segurança (SSP, de *Security Support Provider*) do Windows NT, acessado através da Interface de SSPs (ver também seção 5).

A autenticação inicial Kerberos é integrada com a arquitetura WinLogon de autenticação única (*single sign-on*). O servidor Kerberos, ou KDC, é integrado com serviços de segurança existentes no Windows NT executando no Controlador de Domínio e usa o Serviço de Diretório do Windows NT (*ActiveDirectory*) como base de dados para os usuários (*principals*, no jargão Kerberos) e grupos.

Este capítulo apresenta uma introdução ao Kerberos e logo em seguida o conteúdo apresentado na 10a Conferência de Administração de Sistemas, Redes e Segurança de 2001 (SANS2001) em Baltimore, EUA. Esse conteúdo apresenta técnicas para explorar eficientemente a implementação do protocolo Kerberos do Windows 2000 de forma a obter senhas e recursos dos usuários de um Domínio. Essas técnicas abrem os sistemas Windows 2000 ao mesmo tipo de ataques de dicionário disponíveis no protocolo NTLM do Windows NT 4.0. Algumas possíveis soluções são sugeridas, baseadas no acoplamento do Kerberos com novos *protocolos fortes de senhas*.

8.1 O Protocolo de Autenticação Kerberos

Na mitologia grega, o Kerberos (Cerberos) era um cão com várias cabeças, normalmente três, às vezes com rabo de serpente, o guardião da entrada da terra dos mortos (Hades). Da mesma forma que o Kerberos grego tinha três cabeças, o Kerberos moderno teria três componentes para guardar a entrada da rede: autenticação, contabilidade e auditoria. As duas últimas cabeças nunca foram implementadas.

O Kerberos foi desenvolvido em 1988 no MIT como parte do projeto Athenas. Há várias descrições interessantes deste protocolo ([Sta98], [Mic00d], [Sch96] e [Bry88]).

Duas versões do Kerberos são comumente usadas: a Versão 4 ([Mil88] e [SNS88]), de 1988, é ainda bastante usada, mas não no Windows NT; a Versão 5 [KNT94], de 1993, corrige algumas das deficiências de segurança da versão 4, acrescenta outras características, e foi editada como um padrão RFC [NK93]. As versões de 1 a 3 foram versões de desenvolvimento interno.

O protocolo foi bastante escrutinizado quanto a segurança durante os anos que se seguiram à sua criação ([BM91] e [NS93]), e várias deficiências técnicas apontadas na versão 4 foram tratadas na versão 5.

8.1.1 Kerberos Versão 5

A versão 5 possui vários melhoramentos técnicos que permitem obter uma série de novas características:

Independência do sistema criptográfico permitindo o uso de quaisquer algoritmos criptográficos, através de um identificador de tipo criptográfico (etype, de *encryption type*). A versão 4 somente utilizava o algoritmo criptográfico DES.

Independência de protocolo de rede permitindo o uso de endereços do Protocolo Internet (IP), redes ISO e outros. A versão 4 somente permitia o uso de IP.

Independência de ordenação de bytes utilizando convenções bem estabelecidas. Na versão 5, as estruturas são definidas usando o padrão ASN.1 (*Abstract Syntax Notation One*) e BER (*Basic Encoding Rules*) para obter ordenação de bytes sem ambiguidades. A versão 4 utilizava um *flag* com aplicação limitada.

Tempo de vida do Ticket O tempo de vida para o ticket de acesso do Kerberos na versão 4 era codificado em uma quantidade expressa em 8 bits, permitindo um máximo de cerca de 21 horas de validade. Na versão 5, os tickets incluem um horário específico de início e fim usando UTC (*Universal Time*).

Repasse de Autenticação (Authentication Forwarding) A versão 4 não permite que credenciais de acesso, emitidas para um cliente acessar um servidor específico, sejam usadas por algum outro cliente. Isto permitiria que um cliente conectasse um servidor e este servidor conectasse ainda outro servidor no papel do cliente. A versão 5 oferece esta capacidade.

Autenticação entre Realms (Domínios) Na versão 4, a interoperabilidade entre N realms exigia $O(N^2)$ relacionamentos Kerberos. A versão 5 suporta um método hierárquico que exige menos relacionamentos.

8.1.2 Mecanismos

Imagine duas pessoas com a necessidade de trocar informações através de um meio não confiável. Elas desejam que essas informações sejam protegidas contra terceiros que poderiam ler e adulterar o conteúdo sendo trocado. Uma maneira de proteger essas informações é através de criptografia. As duas pessoas possuiriam uma chave criptográfica secreta em comum, e usando essa chave elas poderiam utilizar algoritmos criptográficos de maneira a tornar as informações sendo trocadas ininteligíveis para terceiros. Mas como obter essa chave em comum? Se o procedimento ingênuo de transportar a chave em claro através do meio não confiável fosse realizado, outros poderiam obtê-la, e portanto compreender as informações criptografadas com ela. Também, se ao invés de duas pessoas houvesse N pessoas, todos teriam que ter a chave de todos para permitir a comunicação entre o grupo, e caso a segurança de apenas um fosse comprometida, todas as chaves secretas do grupo estariam disponíveis para o invasor. Seria interessante se houvesse um protocolo pronto, já bastante analisado quanto a segurança, para distribuir essas chaves de maneira segura para as pessoas corretas. O objetivo do Kerberos é fazer isso.

O Kerberos baseia-se fortemente numa técnica de autenticação que envolve um segredo compartilhado apenas pelas partes envolvidas. No Kerberos, esse segredo é de longa duração, utilizado para provar ao sistema a identidade de quem está tentando utilizar os serviços do Kerberos. Ele tipicamente permanece válido durante meses, e a chave criptográfica secreta associada ao segredo é chamada *chave de longa duração* (*long-term key*). Normalmente, ela é derivada diretamente de uma senha através de uma função de *hash* criptográfico.

Através dessa chave de longa duração, o Kerberos provê mecanismos para autenticação mútua entre um cliente e um servidor, ou entre servidores, antes que uma conexão seja estabelecida entre eles. Ao final da autenticação, ambas as partes possuem uma chave criptográfica efêmera aleatória em comum, utilizada para proteger a troca de informações. Essa chave criptográfia aleatória é chamada *chave de sessão* (*short-term key*) no jargão

Kerberos, por ser uma chave de curta duração[1], válida apenas durante a "sessão" em que as informações vão ser trocadas. O protocolo supõe que transações iniciais entre clientes e servidores acontecem numa rede aberta onde a maioria dos computadores não está fisicamente segura, e onde pacotes viajando através dos cabos podem ser monitorados e modificados à vontade.

Um centro de distribuição destas chaves (KDC, de *Key Distribution Center*), que executa num servidor fisicamente seguro (em cada Controlador de Domínio), funciona como um intermediário entre as partes, chamadas *principals*. Um principal pode ser um usuário, um serviço, um computador, ou qualquer outra entidade que deseja autenticar-se, cliente ou servidor.

O KDC possui todas as chaves de longa duração de cada principal do sistema, de forma a poder autenticá-los na etapa inicial do protocolo *(login)*. Além disto, é responsável por gerar *tickets de acesso* e *chaves de sessão*, através de seus serviços AS (*Authentication Service*) e TGS (*Ticket-Granting Service*). O serviço AS é responsável pelo *login* inicial e devolve um TGT (*Ticket-Granting Ticket*), um ticket de acesso especial para ser apresentado ao TGS para obter outros tickets de acesso a serviços genéricos. Esses outros tickets são apresentados a servidores de aplicação responsáveis por oferecer tipos diferentes de serviços e recursos como arquivos e impressoras (ver figura 8.1). No fim, esses servidores de aplicação, por intermédio do Kerberos e do ticket, são informados de maneira segura sobre qual cliente está tentando usar seus serviços e quais permissões de acesso deveriam ser fornecidas a ele.

Qualquer acesso a qualquer servidor de aplicação e recursos S exige a apresentação de um ticket. Um ticket é uma estrutura que contém a chave de sessão, um tempo de validade, *capabilities* do cliente que quer acessar o servidor e outras informações. Essa estrutura é criptografada com a chave do servidor S que queremos acessar, de forma que somente S pode ler esta estrutura, e, particularmente, a chave de sessão associada. Para cada ticket existe uma única chave de sessão associada. Ao obter o ticket e a chave de sessão, o cliente os guarda e não precisa mais acessar o KDC para abrir conexões com S enquanto o ticket mantiver sua validade, tipicamente da ordem de algumas horas.

Quando um cliente C apresenta um ticket a S, também apresenta um *autenticador*. O autenticador possui um identificador (por exemplo, o nome) do cliente C que está tentando acessar S, e a hora em que o autenticador foi criado, para evitar *replay* do autenticador (qualquer autenticador de C com a hora igual ou anterior ao último autenticador de C é rejeitado por S). Para evitar que alguém altere o conteúdo do autenticador, este é criptografado por C, utilizando a chave de sessão, antes de enviá-lo junto com o ticket

[1]Sendo efêmera, uma criptoanálise bem sucedida que a descubra tem valor apenas enquanto essa chave for válida. Dificultando a criptoanálise através do uso de chaves maiores e reduzindo a validade para apenas algumas horas inviabiliza qualquer criptoanálise usando força bruta.

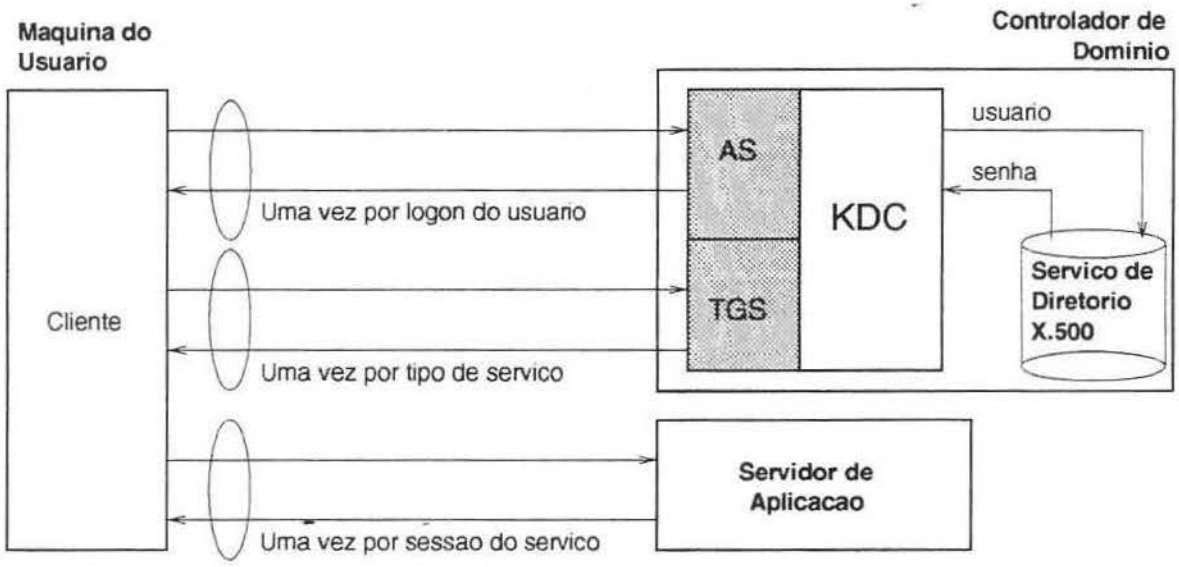


Figura 8.1: KDC e subsistemas AS e TGS do Kerberos

para S.

O servidor S é capaz de decifrar o ticket, obter a chave de sessão que está dentro, e utilizá-la para descriptografar o autenticador enviado pelo cliente C. Se o nome e hora dentro do autenticador estão corretos, S tem certeza de que foi C quem realmente enviou o ticket. Se a autenticação mútua é desejada, o servidor S envia um novo autenticador para C.

8.1.3 O Kerberos no Windows 2000

O protocolo de autenticação Kerberos melhora as características básicas de segurança do Windows NT e fornece:

Maior performance para autenticação no servidor durante o estabelecimento inicial de conexão. O servidor de aplicação/recursos não tem que se conectar ao controlador de domínio para autenticar o cliente, como acontecia no antigo protocolo de autenticação NTLM do Windows NT 4.0. Isto permite uma melhor escalabilidade.

Delegação de Autenticação para arquiteturas de aplicação cliente/servidor com múltiplas camadas. Quando um cliente se conecta a um servidor, o servidor personifica o cliente naquele sistema. Mas se este servidor necessita fazer uma conexão de rede a outro servidor para completar a transação do cliente, o protocolo Kerberos permite a delegação de autenticação para o primeiro servidor se conectar ao outro servidor como se fosse o cliente. A delegação permite ao segundo servidor também personificar o cliente. Obviamente, esta opção, apesar de útil, é perigosa, pois permite que

servidores mal-intencionados tenham a total capacidade de se comportarem como o usuário, acessando todos os recursos remotos que estes têm acesso (ver também a seção 4.2.4).

Relações de Confiança Transitivas para autenticação entre domínios. O usuário pode se autenticar em domínios em qualquer lugar na árvore de domínios porque os servidores de autenticação (KDCs) em cada domínio confiam em tickets emitidos por outros KDCs na árvore. A confiança transitiva simplifica o gerenciamento de domínios para redes grandes com múltiplos domínios. No Windows NT 4.0, a confiança entre domínios tinha que ser explicitamente estabelecida entre cada par de domínios, elevando a quantidade de ligações explícitas para $O(n^2)$.

Usualmente, o usuário prova ao KDC sua identidade utilizando senhas secretas compartilhadas apenas entre o usuário e o KDC. No entanto, o Windows NT também possui extensões ao protocolo Kerberos baseadas em pares de chaves públicas/privadas (PK-INIT [Tun00]). Essas extensões permitem a clientes requisitar uma autenticação inicial utilizando uma chave privada, enquanto o KDC confirma a autenticação tentando achar a chave pública correspondente dentro do repositório de certificados X.509. Este repositório localiza-se dentro do objeto representando o usuário no Serviço de Diretórios (*Active Directory*) do Windows NT. O certificado do usuário pode ser emitido por alguma Autoridade de Certificados (CA), ou através do Serviço de Certificados do Windows NT. Depois desta autenticação inicial, protocolos Kerberos padrões são utilizados para a conexão a recursos. Essas extensões fornecem uma fundação para autenticação na rede utilizando tecnologia baseada em *smartcards*.

8.2 Atacando o Protocolo de Autenticação Kerberos do Windows 2000

A versão 5 revisão 6 do Kerberos ([NKT00] e [Mic00c]) é utilizada amplamente no Microsoft Windows 2000 como um substituto ao protocolo de autenticação NT LAN Manager (NTLM) das versões anteriores deste sistema [Mic00d]. As falhas de segurança do NTLM eram notórias, como os famosos ataques de dicionários aplicados aos *hashes* NTLM. Estas falhas têm sido apontadas desde 1997 ([Hob97],[Vis97] e [L97]).

O protocolo NTLM é implementado no Windows 2000 apenas por motivos de compatibilidade com sistemas legados que não entendem o protocolo Kerberos (Windows 3.11, Windows 95, Windows 98, e Windows NT 4.0). O NTLM também é utilizado com sistemas Windows 2000 no modo *standalone*, e quando o usuário "loga-se" localmente na máquina sem utilizar a rede. No Windows 2000, o KDC guarda os *principals* e chaves

respectivas no *Active Directory*, que é um serviço de diretório estilo X.500 que funciona como repositório de dados do sistema.

O Windows 2000 também acrescenta algumas extensões Microsoft ao Kerberos:

- Utilização de novos tipos de criptografia em pacotes Kerberos ([SB00] e [Mic00c]) (*ETYPE - Kerberos Encryption Type*). Verificou-se nestes pacotes dois tipos principais baseados no algoritmo criptográfico RC4 [Riv] e usando uma verificação (*checksum*) do tipo HMAC-MD5 ([RFC1321] e [RFC2104]):

  1. KERB_ETYPE_RC4_HMAC (tipo 23)
  2. KERB_ETYPE_RC4_HMAC_EXP (tipo 24, semelhante ao anterior, para exportação para fora dos EUA)

- Utilização do campo AUTHORIZATION-DATA do ticket Kerberos para transportar SIDs (*Windows unique Security Identifier*) do usuário e dos grupos a que ele pertence [Mic00d].

A Microsoft vem apresentando o Kerberos como a resposta para todo tipo de problema relacionado à autenticação segura no sistema. Em sua implementação, um mecanismo Kerberos para dificultar ataques, chamado *pré-autenticação*, é usualmente utilizado. No entanto, o tipo de pré-autenticação escolhido pela Microsoft para ser usado, baseado em um *timestamp* (horário) criptografado com uma chave derivada diretamente da senha do usuário (PA_ENC_TIMESTAMP, no jargão Kerberos), é suscetível a um ataque de dicionário, como será visto.

Não há uma pesquisa extensiva nesta área, aplicada aos sistemas operacionais da família Windows. Hobbit, em 1997, apresentou um artigo famoso onde o compartilhamento de arquivos e protocolos de autenticação associados são explorados [Hob97]. Em 1999, Thomas Wu delineou um interessante ataque a um grande ambiente UNIX na Universidade de Stanford que usava Kerberos 4 e etypes baseados no DES [Wu99]. Kenneth e Leighton também descreveram um trabalho muito interessante baseado na engenharia reversa de aspectos fundamentais da segurança do protocolo *Server Message Block* (SMB) para Domínios NT e o associado protocolo de autenticação NTLM [KL00].

8.2.1 Logon através do Kerberos

O usuário começa realizando o *logon* na rede. Ele digita seu nome (*username*) e sua senha. O cliente Kerberos na máquina do usuário converte sua senha para uma *chave criptográfica de longo prazo* (*long-term key*) e salva o resultado no seu *cache* de credenciais. Esta chave é usada como uma *chave criptográfica simétrica* [Sch96] para criptografar e descriptografar partes importantes da informação que irá atravessar a rede.

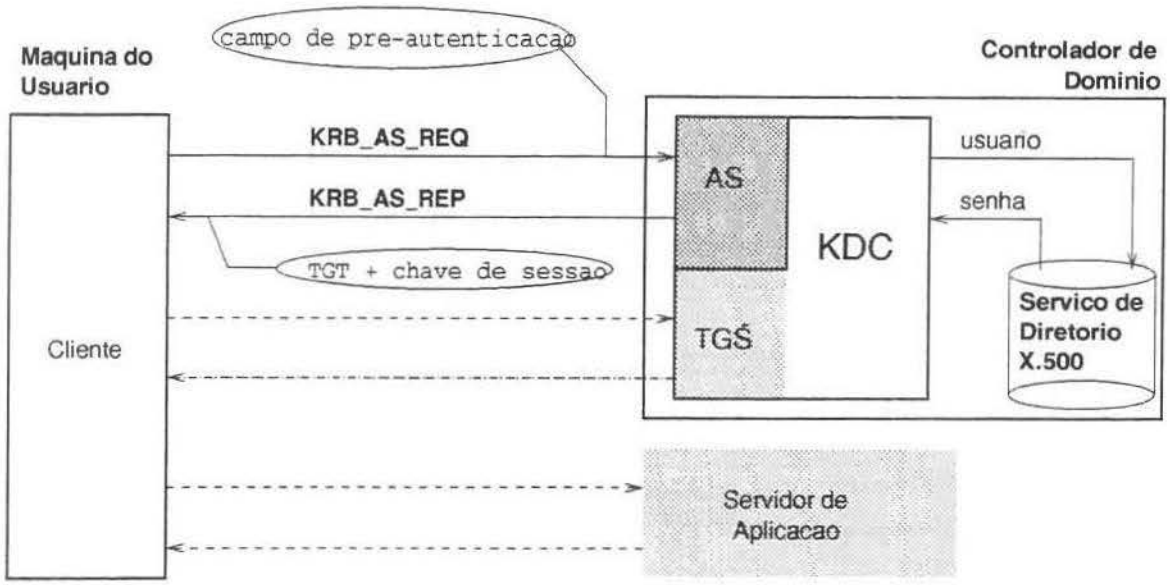


Figura 8.2: O processo de Logon através do Kerberos

Como pode ser visto na figura 8.2, o cliente então envia uma mensagem de requisição de serviço para o *Serviço de Autenticação do Kerberos* (AS, de *Kerberos Authentication Service*), através da emissão do pacote KRB_AS_REQ. Este serviço é oferecido pelo Centro de Distribuição de Chaves Kerberos (KDC, de *Key Distribution Center*), existente em cada Controlador de Domínio NT. A primeira parte desta mensagem identifica o usuário e o nome do serviço para o qual ele está requisitando credenciais (neste caso, o TGS, *Ticket-Granting Service*, que está no KDC). A outra parte da mensagem contém *dados de pré-autenticação* que provam que o usuário conhece a senha. Isto é usualmente um *timestamp* criptografado com a chave de longo prazo do usuário, embora o protocolo permita outras formas para pré-autenticação dos dados.

Após o recebimento do pacote KRB_AS_REQ, o KDC procura o nome do usuário em sua base de dados dentro do Serviço de Diretório X.500 (*ActiveDirectory*), pega a sua chave de longo prazo, descriptografa os dados de pré-autenticação, e avalia o *timestamp* lá contido. Se este *timestamp* for igual ao *timestamp* local da máquina (dentro de uma faixa de erro - 5 minutos, normalmente[Mic99b, Kerberos Policy]), o KDC pode estar seguro que os dados de pré-autenticação foram criptografados utilizando a chave de longo prazo do usuário.

Depois que confirmou a identidade do usuário, o KDC cria credenciais para o cliente apresentar ao TGS, incluindo uma chave de sessão e o TGT (*Ticket Granting Ticket*). O TGT é usado para quaisquer acessos posteriores a recursos do sistema. O KDC criptografa o TGT com sua própria chave de longo prazo[2], e a chave de sessão com a chave de longo

[2]Como o TGT somente pode ser enviado ao TGS - e este é uma parte do KDC - então neste caso o

prazo do usuário, e envia isto tudo de volta ao usuário em um pacote KRB_AS_REP de resposta do Serviço de Autenticação Kerberos.

De acordo com [SB00], a chave de cada principal para o ETYPE RC4_HMAC guardada no KDC do Windows 2000, é, por default, gerada utilizando-se a seguinte função MD4 [RFC1320]:

```
String2Key(password) = MD4(UNICODE(password))[3]
```

Esta é exatamente a mesma função utilizada para criar os *hashes* NTLM do Windows NT, e foi mantida por motivos de compatibilidade, e para reaproveitamento do material criptográfico das senhas já existentes quando um *upgrade* do Windows NT para o Windows 2000 fosse feito. No entanto, outros tipos de chave, por exemplo, chaves DES [NBS77], podem ser criadas, para utilização de *ETYPES* que usam o algoritmo criptográfico DES.

8.3 Análise de um pacote KRB_AS_REQ do Kerberos

O pacote abaixo foi obtido após a execução do comando *'runas administrator'* (equivalente ao *'su'* do UNIX) em uma máquina cliente do Windows 2000. Este pacote Kerberos, do tipo KRB_AS_REQ, o primeiro enviado ao KDC, porta UDP 88, indica que deseja-se acessar o serviço de autenticação AS do KDC para realizar o login e obter o TGT para acessos posteriores ao sistema. O conteúdo UDP, com 341 bytes, pode ser obtido com qualquer programa que faz um *sniffing* de pacotes da rede. Ele está codificado na sintaxe ASN.1 e foi abaixo decodificado para uma estrutura de texto ASCII utilizando o utilitário *'trval'* que vem com a distribuição krb5 do Linux.

Verifica-se facilmente a estrutura ASN.1 correspondente ao pacote KRB_AS_REQ, como está definida na seção 5.4.1 (*KRB_KDC_REQ Definition*) do RFC1510. Os textos em itálico correspondem à descrição dos tipos e dos valores dos campos mais importantes. Atenção especial deve ser dada ao campo marcado como *PA-DATA.padata-value*, do tipo *2 = PA_ENC_TIMESTAP*, ou seja, contendo um *timestamp* ASN.1 criptografado com a chave de longo prazo derivada da senha do usuário (no caso, a senha do administrador informada ao programa *'runas'*.

De acordo com a codificação utilizada pelo programa *'trval'*:

- cada linha representa um campo da estrutura ASN.1;

KDC é servidor e seu próprio futuro cliente.

[3] Cada caracter UNICODE do Windows é codificado em formato *little-endian* de 2 octetos cada. Então uma operação de *hash* criptográfico MD4 é realizada em apenas os caracteres UNICODE da senha (não incluindo os octetos zero finais).

- as expressões [XXX] contendo UNIV, APPL ou CONT indicam o tipo ASN.1 do campo (inteiros, strings, estruturas e outros). A semântica de um tipo UNIV é universal para todo pacote ASN.1, enquanto a semântica dos outros tipos depende de sua posição no pacote e está definida no RFC1510;
- os campos que possuem a string 'constr' são estruturas que contêm subcampos; eles provocam o aumento do espaço de identação, indicando que os campos subsequentes são subcampos;
- as expressões <XXX> indicam o tamanho dos campos, em bytes. Este tamanho inclui o tamanho dos subcampos, quando existem.

```
<341>
6a 82 01 51 [APPL 10] constr <337>
.  30 82 01 4d [UNIV 16] constr <333>

KDC-REQ.pvno
.  .  a1 03 [CONT 1] constr <3>
.  .  .  02 01 [UNIV 2] 5                    5 = VERSÃO 5 DO KERBEROS

KDC-REQ.msg-type
.  .  a2 03 [CONT 2] constr <3>
.  .  .  02 01 [UNIV 2] 10                   10 = KRB_AS_REQ

KDC-REQ.PA_DATA (PRE-AUTHENTICATION DATA)
.  .  a3 5f [CONT 3] constr <95>
.  .  .  30 5d [UNIV 16] constr <93>
.  .  .  .  30 48 [UNIV 16] constr <72>

                PA-DATA.padata-type
.  .  .  .  .  a1 03 [CONT 1] constr <3>
.  .  .  .  .  .  02 01 [UNIV 2] 2          2 = PA_ENC_TIMESTAP

                PA-DATA.padata-value
.  .  .  .  .  a2 41 [CONT 2] constr <65>
.  .  .  .  .  .  04 3f [UNIV 4] <63>
                   30 3d a0 03 02 01 17 a2 36 04 34 97
                   dc 07 f7 a6 ee c6 17 0e 18 c5 87 4c
                   2c 60 9b 6d 2d 93 ff 8b f6 fc 72 e1
```

```
                        f5 77 45 6e 32 91 a7 d6 2b 15 a5 fd
                        3e d2 50 fd 28 4a 9b 07 29 96 12 4f
                        cc bb b3
. . . . 30 11 [UNIV 16] constr <17>
. . . . . a1 04 [CONT 1] constr <4>
. . . . . . 02 02 [UNIV 2] 128
. . . . . a2 09 [CONT 2] constr <9>
. . . . . . 04 07 [UNIV 4] <7>
                        30 05 a0 03 01 01 ff
```

KDC-REQ.KDC_REQ_BODY

```
. . a4 81 df [CONT 4] constr <223>
. . . 30 81 dc [UNIV 16] constr <220>
. . . . a0 07 [CONT 0] constr <7>
. . . . . 03 05 [UNIV 3] 0x40810010          KDC_OPTIONS
. . . . a1 23 [CONT 1] constr <35>           CNAME
. . . . . 30 21 [UNIV 16] constr <33>        (PRINCIPAL_NAME CLIENT)
. . . . . a0 03 [CONT 0] constr <3>
. . . . . 02 01 [UNIV 2] 10
. . . . . a1 1a [CONT 1] constr <26>
. . . . . 30 18 [UNIV 16] constr <24>
. . . . . 1b 16 [UNIV 27] "administratOr@coredump"
. . . . a2 10 [CONT 2] constr <16>
. . . . . 1b 0e [UNIV 27] "COREDUMP.LOCAL"   REALM (DOMÍNIO)
. . . . a3 23 [CONT 3] constr <35>           SNAME
. . . . . 30 21 [UNIV 16] constr <33>        (PRINCIPAL_NAME SERVER)
. . . . . a0 03 [CONT 0] constr <3>
. . . . . 02 01 [UNIV 2] 2
. . . . . a1 1a [CONT 1] constr <26>
. . . . . 30 18 [UNIV 16] constr <24>
. . . . . 1b 06 [UNIV 27] "krbtgt"
. . . . . 1b 0e [UNIV 27] "COREDUMP.LOCAL"
. . . . a5 11 [CONT 5] constr <17>           TILL
. . . . . 18 0f [UNIV 24] "20370913024805Z"  (tempo máximo de vida)
. . . . a6 11 [CONT 6] constr <17>           RENEW_TILL
. . . . . 18 0f [UNIV 24] "20370913024805Z"  (renovar até)
. . . . a7 06 [CONT 7] constr <6>            NONCE
. . . . . 02 04 [UNIV 2] 469025590           (devolvido criptografado
```

```
                                                    pelo servidor)
. . . . a8 19 [CONT 8] constr <25>
. . . . . 30 17 [UNIV 16] constr <23>   ENCRYPTION_TYPE (ETYPE)
                                        (ordem de preferência)
. . . . . 02 01 [UNIV 2] 23             KERB_ETYPE_RC4_HMAC [SB00]
. . . . . 02 02 [UNIV 2] -133           .
. . . . . 02 01 [UNIV 2] -128           KERB_ETYPE_RC4_MD4 [Mic00c]
. . . . . 02 01 [UNIV 2] 3              KERB_ETYPE_DES_CBC_MD5 [NK93]
. . . . . 02 01 [UNIV 2] 1              KERB_ETYPE_DES_CBC_CRC [NK93]
. . . . . 02 01 [UNIV 2] 24             KERB_ETYPE_RC4_HMAC_EXP [SB00]
. . . . . 02 02 [UNIV 2] -135           .

. . . . a9 2c [CONT 9] constr <44>       ADDRESSES (onde o ticket é
. . . . . 30 2a [UNIV 16] constr <42>              válido)
. . . . . 30 19 [UNIV 16] constr <25>
. . . . . a0 03 [CONT 0] constr <3>
. . . . . 02 01 [UNIV 2] 20
. . . . . a1 12 [CONT 1] constr <18>   NETBIOS NAME
. . . . . 04 10 [UNIV 4] "LAMBERT-WIN2K   "
. . . . . 30 0d [UNIV 16] constr <13>
. . . . . a0 03 [CONT 0] constr <3>
. . . . . 02 01 [UNIV 2] 2
. . . . . a1 06 [CONT 1] constr <6>
. . . . . 04 04 [UNIV 4] <4>          IP NAME
                  c0 a8 01 01
```

8.3.1 Atacando o Campo de Pré-Autenticação do KRB_AS_REQ

O campo *PA-DATA.padata-value* (uma estrutura do tipo *EncryptedData*, conforme a seção 6.1 do RFC1510) possui dois subcampos. O primeiro deles é um campo *etype*, contendo o tipo de algoritmo criptográfico utilizado (no caso, 0x17 = 23 = KERB_ETYPE_RC4_HMAC). O segundo é um campo *cipher* que contém o *timestamp* criptografado.

```
PA-DATA.padata-value
a2 41 [CONT 2] constr <65>
   04 3f [UNIV 4] <63>
      30 3d
         a0 03 02 01 17  etype  = KERB_ETYPE_RC4_HMAC
         a2 36
```

```
04 34        cipher = <texto cifrado segue abaixo>
   97 dc 07 f7 a6 ee c6 17 . (checksum)
   0e 18 c5 87 4c 2c 60 9b .     "
   6d 2d 93 ff 8b f6 fc 72 . (confounder)
   e1 f5 77 45 6e 32 91 a7 . (conteúdo)
   d6 2b 15 a5 fd 3e d2 50 .     "
   fd 28 4a 9b 07 29 96 12 .     "
   4f cc bb b3             .     "
```

Um ataque de dicionário *offline* com texto claro conhecido (*known-plaintext attack)* [Sch96] pode então ser aplicado. Neste ataque, várias senhas são utilizadas como candidatas para a senha correta. Caso a senha correta seja utilizada para descriptografar o texto cifrado do campo *padata-value*, um texto claro ASN.1 com a estrutura *timestamp* será encontrado.

Como exemplo, o campo de texto cifrado observado acima resulta no texto claro abaixo caso o algoritmo KERB_ETYPE_RC4_HMAC seja usado com a senha *'vmonlo;'* (sem as aspas simples). Os números entre parênteses correspondem ao tamanho do campo em bytes.

texto claro:

```
97 dc 07 f7 a6 ee c6 17 . checksum (16)
0e 18 c5 87 4c 2c 60 9b .
7d 24 5e 3e 7c 35 57 5f . confounder criptográfico (8)
30 1a a0 11 18 0f 32 30 . conteúdo (28)
30 30 30 35 32 39 32 32 .
35 32 34 34 5a a1 05 02 .
03 0c f6 8a
```

estrutura ASN.1 do conteúdo do texto claro:

```
30 1a                (estrutura ASN.1)
.  a0 11
.  .  18 0f         (timestamp '20000529225244Z')
.  .  .  32 30 30 30 30 35 32 39 32 32 35 32 34 34 5a
.  a1 05
.  .  02 03
.  .  .  0c f6 8a   (microssegundos - campo pausec)
```

O *timestamp* obtido acima significa "29/05/2000, 22:52:44" mais os microssegundos. O caracter Z do *timestamp* indica zona de tempo UTC (*Universal Time*).

8.3.2 Estendendo o Ataque para outros Sistemas

O tipo de pré-autenticação PA_ENC_TIMESTAMP cifra um texto claro conhecido, composto por um timestamp no formato ASN.1, utilizando uma chave criptográfica.

Esta chave e o algoritmo de cifragem estão determinados pelo ETYPE utilizado. Um ETYPE não está necessariamente associado a algum tipo particular de pré-autenticação. Há ETYPES que determinam que a chave deve ser construída a partir da senha do usuário. Outros ETYPES podem determinar métodos diversos de construir a chave.

Por exemplo, o ETYPE RC4_HMAC, preferencialmente utilizado no Kerberos da Microsoft e no Windows 2000, é baseado no algoritmo criptográfico RC4, e utiliza uma chave diretamente derivada da senha do usuário através da função de *hash* MD4. Há outros ETYPES também utilizados no Kerberos da Microsoft, mas não preferencialmente. Por exemplo o ETYPE ĐES_CBC_MD5, que também deriva a chave a partir da senha do usuário, embora através de outras funções. Outros ETYPEs usualmente utilizados podem ser vistos na lista de preferência de ETYPES no pacote KRB_AS_REQ.

Quaisquer outros sistemas que usem um método de pré-autenticação Kerberos cifrando um texto claro conhecido, como acontece no método PA_ENC_TIMESTAMP, e um ETYPE que derive diretamente da senha do usuário a chave que cifra estas informações, está sujeito a este ataque de dicionário.

Assim, não é apenas o Windows 2000 que está potencialmente sujeito a este ataque: o Kerberos "puro" define este método de pré-autenticação como um dos possíveis métodos para serem utilizados, e também define ETYPES que derivam a chave de cifragem a partir da senha do usuário. Assim, o Kerberos "puro" está sujeito a estes ataques.

8.3.3 Ataques na Ausência de Pré-Autenticação

Se a pré-autenticação não estiver sendo usada devido a alguma política do administrador do sistema usando Kerberos 5 e Windows 2000[4], o pacote KRB_AS_REP é sempre emitido pelo KDC ao cliente em uma mensagem de resposta na sequência de login[5]. Portanto, é possível atacar o campo *AS_REP.enc-part* deste pacote, porque o ETYPE RC4_HMAC preferencialmente usado no Windows 2000 determina que este campo também é criptografado com uma chave de longo prazo derivada diretamente da senha do usuário. As informações cifradas possuem campos com texto claro conhecido [NKT00, secao 5.4.2], conforme indicado abaixo:

[4]O Kerberos versão 4 (não usado no Windows), junto com outras deficiências, não possui mecanismos de pré-autenticação e portanto não tem defesas contra este tipo de ataque estendido. Problemas relacionados ao Kerberos 4 podem ser encontrados em [Wu99].

[5]Quando a pré-autenticação é usada, o pacote KRB_AS_REP somente é devolvido se o campo de pré-autenticação demonstrar que o cliente realmente conhece a senha correta.

Estrutura AS_REP.enc-part		
nome do campo	**tipo de dados**	**comentários**
.key	EncryptionKey	repositório da chave de sessão
.last-req	LastReq	
.nonce	INTEGER	
.key-expiration	KerberosTime	OPCIONAL
.flags	TicketFlags	
.authtime	KerberosTime	
.starttime	KerberosTime	OPCIONAL
.endtime	KerberosTime	
.renew-till	KerberosTime	OPCIONAL
.srealm	Realm	
.sname	PrincipalName	'krbtgt'
.caddr	HostAddresses	OPCIONAL

Todos estes campos são estrutura ASN.1 específicas ao Kerberos, compostos de vários bytes de texto claro bem conhecidos, como aqueles existentes no campo de pré-autenticação. Os campos com nomes semelhantes aos campos do pacote KRB_AS_REQ possuem conteúdo semelhante: a string '*krbtgt*' foi observada dentro de *sname*, o nome do *realm* (Domínio NT) em *srealm*, e *timestamps* do tipo KerberosTime contendo horários UTC codificados como dígitos ASCII. Quando algum destes textos claros for encontrado, a chave usada para descriptografar o campo *enc-part* é correta.

Neste caso, o campo *enc-part.key* contém a chave de sessão aleatória usada para criptografar o autenticador. *Com esta chave de sessão, é possível forjar autenticadores válidos* para o ticket (o TGT, neste caso) que também foi enviado junto no KRB_AS_REP. Com o TGT e autenticadores válidos, seria possível ficar acessando o TGS para obter acesso aos recursos do usuário até a expiração do TGT, mesmo depois que o usuário já saiu (*logged out*) do sistema. O sistema não geraria eventos de *login* associado a estes acessos "extras". Um evento de *login* seria uma ajuda extra para auditar acessos indevidos.

No Kerberos sem pré-autenticação, o ataque pode ser feito de fora da rede local, pois não há necessidade de interceptar o pacote que sai do usuário e vai para o KDC. É necessário apenas requisitar um *logon* a um KDC que se encontra em um endereço de rede qualquer e um pacote KRB_AS_REP é enviado de volta ao endereço de rede original que originou a requisição de *logon*.

8.3.4 Generalizando as Oportunidades de Ataque

A chave derivada da senha do usuário é utilizada apenas em 4 locais durante o subprotocolo AS de autenticação Kerberos:

1. Quando o cliente criptografa os dados de pré-autenticação em *AS_REQ.pa-data.padata-value.cipher;*

2. Quando o KDC descriptografa os dados de pré-autenticação *AS_REQ.pa-data.padata-value.cipher;*

3. Quando o KDC criptografa campos importantes como chave de sessão de logon dentro de *AS_REP.enc-part;*

4. Quando o cliente descriptografa o campo *AS_REP.enc-part.*

Um texto cifrado, criptografado com a chave de longo prazo derivada da senha do usuário, atravessa a rede entre as etapas 1-2 (utilizado no Ataque à Pré-Autenticação) e as etapas 3-4 (utilizado no Ataque na Ausência de Pré-Autenticação), podendo ser capturado e explorado: um ataque de dicionário *off-line* de texto claro conhecido pode ser aplicado para a descoberta da senha em um tempo prático, da ordem de horas ou de poucos dias, conforme explicado na próxima seção.

8.3.5 Eficiência do Ataque

A tabela abaixo (retirada de [Sch96]) ilustra a ordem de grandeza do espaço de senhas para diferentes classes de senhas.

Número possível de Senhas para vários espaços de senhas					
	4 bytes	5 bytes	6 bytes	7 bytes	8 bytes
letras minúsculas (26)	460.000	1,2E7	3,1E8	8,0E9	2,1E11
letras minúsculas e números (36)	1.700.000	6,0E7	2,2E9	7,8E10	2,8E12
caracteres alfanuméricos (62)	1,5E7	9,2E8	5,7E10	3,5E12	2,2E14
caracteres imprimíveis (95)	8,1E7	7,7E9	7,4E11	7,0E13	6,6E15
caracteres ASCII (128)	2,7E8	3,4E10	4,4E12	5,6E14	7,2E16
caracteres ASCII 8bits (256)	4,3E9	1,1E12	2,8E14	7,2E16	1,8E19

A tabela abaixo mostra a ordem de tempo necessária para buscar nos espaços de senhas da tabela acima, considerando um milhão de tentativas por segundo. Um milhão de tentativas por segundo é uma ordem de grandeza de velocidade usualmente encontrada em programas que vasculham espaços de senhas nos computadores mais rápidos atualmente disponíveis para os usuários em geral (Pentium II 450MHz e acima). Assim, da tabela abaixo é possível extrair uma idéia do esforço exigido para um ataque que analisasse todas as possíveis senhas, e a praticidade de se realizar tal ataque.

O potencial ataque de força bruta do ataque acima pode ser tremendamente otimizado utilizando primeiro palavras derivadas de dicionários e variações destas (ataque de dicionário). Por exemplo, a palavra "senha" e variações dela como "senha0", "senha123",

Tempo de Busca Exaustiva para vários espaços de senhas (10^6buscas/s)					
	4 bytes	5 bytes	6 bytes	7 bytes	8 bytes
letras minúsculas (26)	0,5 s	12 s	5 min	2,2 h	2,4 d
letras minúsculas e números (36)	1,7 s	1 min	36 min	22 h	33 d
caracteres alfanuméricos (62)	15 s	15 min	16 h	41 d	6,9 a
caracteres imprimíveis (95)	1,4 min	2,1 h	8,5 d	2,2 a	210 a
caracteres ASCII (128)	4,5 min	9,5 h	51 d	18 a	2300 a
caracteres ASCII 8bits (256)	1,2 h	13 d	8,9 a	2300 a	580.000 a

"ahnes", "seNha+", "s3nh4" etc. Embora uma senha em particular possa ser difícil de ser adivinhada, na média as senhas dos usuários tendem a ser fáceis de adivinhar, o que torna o ataque de dicionário extremamente eficiente.

Em um estudo de ataques de dicionários de 1989 [Kle90], para cada usuário de um sistema UNIX vários milhões de variações foram tentadas a partir de um dicionário com cerca de 60.000 palavras. Cerca de 25% de todas as senhas do sistema foram encontradas.

Dicionários maiores e mais variações por palavra aumentam a taxa de acerto. Atualmente, é fácil encontrar dicionários com várias centenas de milhares de palavras, como por exemplo em [Wor01].

Um exemplo recente mostra o poder de um ataque de dicionário: o utilitário *L0phtCrack 2.5* [L97], de 1999, altamente otimizado para Pentium, é capaz de testar *todas* as senhas alfanuméricas em menos de 24 horas em um Pentium II/450Mhz para fazer um ataque aos *hashes* Lan Manager e NT do Windows NT 4.0. O *hash* NT é um valor derivado da senha do usuário[6], utilizado no protocolo de autenticação NTLM do Windows NT 4.0, amplamente adotado antes do advento do Kerberos no Windows 2000. De acordo com a documentação do utilitário, 90% das senhas do Windows NT 4.0 podem ser encontradas em menos de 48 horas.

8.4 O Protótipo

No programa não-otimizado desenvolvido para testar a viabilidade de encontrar o texto claro original do campo de pré-autenticação, cada estrutura *padata-value* pôde ser descriptografada em *0,30 ms* em um Pentium 166MHz (isto equivale a uma taxa de 3333 tentativas por segundo). Esta taxa pode ser bastante aumentada através das otimizações citadas mais à frente. Ele foi desenvolvido baseando-se no RFC 1510, em documentos da Microsoft [SB00], e em códigos fontes de distribuições Kerberos encontradas na Internet,

[6]Na verdade, o valor é igual ao resultado da função *String2Key* mostrado anteriormente.

Notação	Significado
K	chave de longo prazo do usuário (*long-term key*), derivada da senha
T	tipo da mensagem Kerberos (*message-type*)
K1 e K3	*hash* HMAC_MD5 com 16 bytes
chksum	o *checksum* do texto cifrado (primeiros 16 bytes)
edata	os bytes restantes do texto cifrado. os primeiros 8 bytes de edata são o *confounder*[Sch96]
data	os dados descriptografados, em texto claro

Tabela 8.1: Notação utilizada no Algoritmo para decodificar PADATA

como o pacote *Heimdal* [Hei01] e o pacote *Krb5* do Linux. Versões especiais do 'tcpdump 3.5' (para obter os pacotes de *logon* Kerberos da rede) e 'John the Ripper 1.6' (para aplicar o ataque de dicionário com variações de palavras) foram criadas para implementar o ataque. Mais informações sobre o protótipo e testes realizados podem ser encontradas no Apêndice A.

8.4.1 O Algoritmo

O algoritmo mostrado aqui é funcionalmente equivalente ao utilizado no Windows 2000, conforme foi comprovado através do correto funcionamento do protótipo. A tabela 8.1 mostra a notação utilizada nesta seção.

```
; Algoritmo para obter o texto claro
; usando etype=KERB_ETYPE_RC4_HMAC
;
; Escolher uma senha candidata, obtida a partir
; de um dicionário ou gerada aleatoriamente:
;
;   Obter a chave K de longo prazo do usuário
;
```

```
        K=String2Key(senha)          (etapa 1)
;
;   Decodificar_PADATA
;
        K1=HMAC_MD5(K,T)             (etapa 2)
        K3=HMAC_MD5(K1,chksum)       (etapa 3)
        data=RC4(K3,edata)           (etapa 4)
;
; Se em data houver uma estrutura ASN.1 com
; um timestamp, então a senha candidata
; é a senha correta.
```

O tipo da mensagem T depende de qual campo e qual mensagem estão sendo criptografados. Na mensagem analisada (KRB_AS_REQ), T=1. Para KRB_AS_REP, T=8. Outros valores podem ser obtidos em [SB00] e em [NKT00, 6.3.5].

Caso a senha correta seja usada, depois da etapa 4 o campo *data* contém uma estrutura ASN.1 com um *timestamp*.

8.4.2 Otimizações

A decodificação da informação presente no campo criptografado é feita em 4 etapas principais, conforme a tabela abaixo.

Etapas do Algoritmo		
Etapa	Função Executada	Tempo em $\mu s(Pentium166MHz)$
1	Obtenção da chave de longo prazo K	20
2	Cálculo da chave K1	110
3	Cálculo da chave K3	110
4	Execução do algoritmo RC4 nos bytes do texto cifrado	60-70

Várias otimizações podem ser feitas para diminuir o tempo de execução do protótipo:

- Otimização do código fonte, cujas funções criptográficas são genéricas e não otimizadas. Há várias chamadas de funções, criações de contextos para variáveis locais, atribuições e comparações que são completamente dispensáveis.

- As etapas 1 e 2 somente precisam ser feitas uma vez para cada palavra do dicionário. A função *String2Key* para o *etype* analisado do Windows 2000 não utiliza o conceito de '*salt*'[7] das senhas UNIX de forma que, *para cada senha* do dicionário, é possível armazenar temporariamente o resultado do fim da etapa 2 e aplicar *apenas* as etapas 3 e 4 para todos os textos cifrados dos usuários encontrados (as etapas 3 e 4 são dependentes do texto cifrado).

- Calcular antecipadamente as etapas 1 e 2 para cada uma das palavras do dicionário, e *armazenar os resultados em memória/disco* para utilizá-los na hora do ataque.

- Descriptografar apenas os primeiros 9 bytes de *edata*. Os primeiros 8 bytes são o *confounder*, que é composto por uma string aleatória, cujo texto claro não se conhece. O byte seguinte deve ser o valor 0x30 (em hexadecimal), que indica uma estrutura ASN.1. Se não for 0x30, então o resto pode ser ignorado. Se for, então ainda há uma chance em 256 de que o valor foi uma coincidência. Mais alguns bytes devem ser descriptografados e comparados para assegurar que o texto foi descriptografado corretamente.

8.5 Possíveis defesas ao ataque

Algumas defesas podem ser utilizadas para evitar esta vulnerabilidade. Textos claros conhecidos não deveriam ser cifrados com a senha do usuário ou com qualquer função que possua apenas a senha como parâmetro, já que na média os usuários escolhem senhas simples. Além disto, há duas outras possibilidades interessantes.

Existem modernos protocolos de transmissão segura de senhas (ver seção 5.3), como o protocolo PAK (*Provably Secure Password Authentication and Key Exchange using Diffie-Hellman*) de Phil MacKenzie [BMP00], o protocolo SRP (*Secure Remote Password Protocol*) de Thomas Wu [Wu98], e os protocolos seguros de R.Perlman e C.Kaufmann [PK99], onde as chaves derivadas da senha do usuário nunca são utilizadas para transmitir nenhuma informação (*zero-knowledge protocols*), e portanto são imunes a ataques de dicionários. Estes protocolos usam chaves públicas e privadas efêmeras que são aleatoriamente criadas usando Diffie-Hellman, usadas uma vez, e então descartadas. Em alguns

[7] O *salt* é uma string aleatória acrescentada no fim de cada senha do usuário. Quando o *salt* existe, tanto ele quanto a senha são parâmetros da função de cálculo *String2Key*, de modo que dois usuários com a mesma senha dificlmente vão ter a mesma chave de longo prazo. Alguns *etypes* utilizam *salt*.

métodos extendidos, o servidor de autenticação nem mesmo armazena a senha do usuário (ou qualquer equivalente). O acoplamento destes novos protocolos com a autenticação inicial do Kerberos (subprotocolo AS) deverá fornecer muito mais segurança.

Uma solução de curto prazo seria usar extensões Kerberos que permitem chaves públicas/privadas resistentes a ataques de dicionários, como a extensão PKINIT [Tun00]. Há uma implementação PKINIT no Windows 2000. No entanto, esta não é uma configuração usual, provavelmente porque sistemas que usam PKINIT devem estar acoplados a dispositivos *smartcards* de logon e também porque usuários não estão cientes das vulnerabilidades existentes no campo de pré-autenticação, baseado em senha, do protocolo Kerberos do Windows 2000. Um problema relacionado com este tipo de extensão é que ela é baseada em 'alguma coisa que o usuário tem' (*stored-key approach*), que apenas verifica a presença de um dispositivo, e não 'alguma coisa que o usuário sabe' (*knowledge-based approach*), que verifica a presença de vida humana. Além disto, toda uma complicada infraestrutura de chave pública deve ser implementada e gerenciada.

8.6 Conclusão

O ataque descrito ao campo de pré-autenticação do Kerberos do Windows 2000 permite a descoberta *off-line* de senhas do sistema em um período de tempo prático, abrindo o Windows 2000 ao mesmo tipo de ataques de dicinário existentes no protocolo NTLM do Windows NT 4.0. Foi mostrado que o método PA_ENC_TIMESTAMP para a pré-autenticação usada no Windows 2000 foi uma má escolha para um sistema que está se esforçando para ser seguro.

Conforme explicado na seção 8.3.2, estas falhas do Kerberos do Windows 2000 podem também ser aplicadas a outros sistemas que utilizam Kerberos com o método de pré-autenticação PA_ENC_TIMESTAMP ou semelhante, caso o ETYPE utilizado também derive a chave de cifragem a partir da senha do usuário (como acontece no ETYPE RC4_HMAC, o ETYPE padrão do Windows 2000). O Kerberos "puro" está sujeito a estes ataques.

A forma estendida do ataque poderia ser usada para obter tanto a senha do usuário quanto a chave de sessão do TGT, permitindo, até a expiração do TGT, forjar autenticadores válidos que permitiriam acesso a recursos do usuário sem a necessidade da sequência de login.

Uma solução de curto prazo é a extensão PKINIT (chave pública) do Kerberos, implementada no Windows 2000, mas alguns problemas foram apontados na seção 8.5, que não impossibilitam mas dificultam a adoção desta solução. Uma solução de longo prazo mais atraente parece ser o acoplamento da autenticação inicial do Kerberos com algum moderno protocolo forte de transmissão de senhas, que estão se tornando amplamente

disponíveis.

Capítulo 9

Conclusões

O Windows NT da Microsoft é um sistema operacional relativamente novo, com pouco mais de 10 anos de existência. Como sistema operacional comercial para corporações, possui várias características inovadoras. Um exemplo é a arquitetura modular, em camadas, contendo um microkernel e facilmente expansível para acomodar o surgimento de novas tecnologias. Essa facilidade para adicionar novas funcionalidades ao sistema sem interferir nos módulos já existentes, é o principal pilar que permite ao sistema evoluir.

Mais especificamente, a concepção e implementação dos subsistemas de segurança e de rede permite que nestes haja modificações, adições e evoluções, sem exigir modificações substanciais em outras partes do sistema. O grande potencial para a evolução da segurança do sistema advem disto.

Análise dos Protocolos de Comunicação e Segurança

O presente trabalho mostrou a situação atual da segurança destes subsistemas de segurança, de rede, e dos protocolos de comunicação do Windows NT. O Windows NT permite uma enorme quantidade de protocolos de comunicação (SMB, NetBEUI, TCP/IP, IPX/SPX, AppleTalk e muitos outros) e a incorporação de mais protocolos é fácil devido à implementação modular do sistema.

A segurança destes protocolos de comunicação, devido à política de *single sign-on* do sistema (que permite que um usuário dentro de um Domínio NT identifique-se durante o *login* inicial e acesse todos os recursos pertencentes a este Domínio sem necessitar identificar-se novamente), repousa em outros poucos protocolos específicos de segurança. Estes protocolos de segurança contêm os protocolos de autenticação que suportam a estrutura de *single sign-on*.

O Windows NT, até a versão 4.0, era marcado pela utilização de protocolos de segurança obsoletos, derivados do sistema de autenticação LAN Manager do DOS, OS/2 e Windows 3.1. Estes protocolos ou enviavam informações importantíssimas e secretas (co-

mo a senha do usuário) em claro, ou as enviava fracamente criptografadas. Isto permitia ataques amplamente variados, como ataques de força bruta (por exemplo, protocolo de autenticação LAN Manager (LM) do DOS e Windows 3.1), ataques de dicionários (por exemplo, no protocolo LAN Manager do Windows NT (NTLM)), e muitas criptoanálises bem-sucedidas (por exemplo, no protocolo LM e na implementação PPTP da Microsoft). Outros protocolos de subsistemas de rede também muito utilizados, como o Netware 3 da Novell e Banyan VINES, não eram menos vulneráveis que as soluções nativas do Windows NT.

O Windows 2000, a última versão do Windows NT da Microsoft até o momento, aproveitou-se da modularidade e facilidade de adição de novos módulos ao Windows NT e reestruturou o sistema de protocolos de segurança. Uma nova Interface de Provedores de Suporte de Segurança (SSPI, de *Security Support Provider Interface*) foi acrescentada, e dois novos protocolos de segurança acrescentados ao NTLM: o Kerberos e o SPNEGO (*Secure Protected Negotiation Protocol*). Através da SSPI, qualquer um destes três protocolos podem ser escolhidos para comunicação: o obsoleto NTLM (por motivos de compatibilidade com sistemas legados), o novo protocolo Kerberos (com novos recursos para autenticação, delegação de autenticação e segurança), ou o protocolo de negociação automática SPNEGO (que negocia, dificultando ataques de *downgrade* [1], qual a versão mais segura do NTLM ou Kerberos existe em comum entre as máquinas que desejam comunicar-se).

Da maneira como foi projetada, esta reestruturação permite que a autenticação inicial, dados criptografados e outros mecanismos de segurança sejam transportados usando quaisquer protocolos de comunicação disponíveis (como SMB, TCP/IP, NetBEUI etc). Os protocolos de comunicação e transporte, desta forma, ficam desvinculados de qualquer particular protocolo de autenticação e segurança.

Vulnerabilidades

No entanto, o presente trabalho também mostrou que nem o obsoleto NTLM nem a implementação do protocolo Kerberos pela Microsoft estão imunes a ataques. Foi mostrado, por exemplo, que tanto um quanto o outro são suscetíveis a ataques de dicionário que permitem encontrar a senha do usuário. Um ataque deste tipo é bastante eficiente para encontrar uma porcentagem das senhas pertencentes a um grande conjunto de usuários normalmente existentes em Domínios NT. Devido à arquitetura de *single sign-on* da Microsoft, uma vez descoberta a senha de um usuário, é possível acessar qualquer recurso do Domínio NT (arquivos, mais senhas, configurações etc) a que este usuário teria acesso.

[1] Em um ataque de *downgrade*, um intermediário interferindo na rede força as entidades em negociação a escolher o protocolo mais fraco que possuem em comum de modo a facilitar um possível ataque.

Em sistemas corporativos, nos quais segredos valiosos devem ser guardados, as vulnerabilidades apontadas indicam que o Windows 2000 permite que estes segredos sejam roubados com pouco esforço. Um cenário ilustra isto: um funcionário de outra empresa (ou até mesmo da própria empresa) pode usar um computador da empresa ou inserir um *notebook* na rede e roubar pacotes de rede contendo senhas criptografadas durante o *logon* de outros funcionários, inclusive do administrador da rede ou chefe de seção. Após um dia ou dois aplicando um ataque de dicionário, em casa, às senhas criptografadas coletadas, ele poderia personificar esses funcionários perante o sistema, inclusive o administrador e o chefe de seção, que normalmente têm acesso a informações valiosas.

Defesas

Algumas defesas a este ataque de dicionário foram apontadas. A utilização do obsoleto protocolo NTLM não é recomendada, por ser altamente vulnerável a este ataque, devendo ser evitada a todo custo[2]. No caso do Kerberos, uma solução de curto prazo poderia utilizar a extensão PKINIT do Kerberos, que também foi implementada no Windows 2000. Esta extensão envolve a adição de um sistema de chaves públicas e privadas armazenadas (compostas por um grande conjunto de bits aleatórios, com alta entropia), para reforçar a segurança da autenticação inicial e evitar ataques de dicionários[3]. Neste caso, *smartcards* são utilizados pelo usuário durante o *logon* inicial. Esta solução, além dos custos relacionados ao equipamento extra, envolve também todo o custo e esforço de criação, controle, distribuição e gerenciamento destas chaves públicas. Além disto, baseia-se na idéia de autenticar 'algo-que-o-usuário-tem', ou seja, em autenticar o dispositivo *smartcard*, e não o usuário.

Uma solução de longo prazo para o Kerberos seria acoplar a autenticação inicial baseada em senha - que atualmente permite o ataque de dicionário - com novos *protocolos fortes de autenticação de senha* (seção 5.3). Estes protocolos usam técnicas de chaves públicas, mas em vez de exigir chaves armazenadas ou uma hierarquia de certificados, eles usam chaves *efêmeras*. Pares efêmeros de chaves públicas e privadas são aleatoriamente criados, usados uma única vez, e então descartados tão logo a autenticação esteja completa. O importante de manter a autenticação baseada no conhecimento de uma senha - *proofs of human knowledge* - é que ela verifica a presença de alguém vivo. Isto é distinto de uma autenticação biométrica (algo que você é) em que características físicas não podem ser modificadas ou escolhidas, e é distinto de uma autenticação baseada no armazenamento

[2]Em tempos de Internet, muito cuidado deve ser tomado para evitar que a autenticação a servidores Web via protocolo HTTP, utilizando a "autenticação integrada automática" da Microsoft entre o 'Internet Explorer' e o servidor Web 'Internet Information Server', esteja configurada para permitir a utilização de NTLM.

[3]Ataques de dicionários são eficientes somente quando o algoritmo criptográfico utiliza como chave apenas palavras curtas de baixa entropia memorizadas pelo usuário, que são de fácil adivinhação.

de uma chave (algo que você tem) que pode somente verificar a presença da máquina ou dispositivo. Para mais informações sobre esta solução, ver a seção 5.3.

Comentários finais

Atualmente o Windows NT, mesmo na sua última versão (o Windows 2000), não oferece em uma instalação padrão protocolos de comunicação seguros e simples o suficiente para guardar informações altamente sigilosas. Felizmente, devido à modularidade do sistema operacional, é possível que próximas versões incorporem protocolos mais seguros que impossibilitem os ataques mostrados e acrescentem mais segurança ao sistema.

Apêndice A

Ferramentas para Análise de Segurança

Este apêndice visa comparar diversos pacotes de análise de segurança disponíveis para sistemas baseados no Windows NT, e fornecer uma visão do que esses pacotes, no todo, podem contribuir para aumentar a segurança desses sistemas. De forma nenhuma esta análise deve ser vista como endosso à utilização em particular de algum dos pacotes analisados.

Estes pacotes de segurança constituem uma boa base de dados contendo uma ampla lista de vulnerabilidades documentadas do Windows NT, e estão sempre em evolução, usualmente incorporando as últimas vulnerabilidades conhecidas. Além desses pacotes, em [MSK99] pode ser encontrada uma ampla lista de vulnerabilidades documentadas do Windows NT.

Inicialmente, modifiquei dois pacotes de segurança, o *TCPDUMP* e o *John the Ripper*, utilizados para realizar o ataque ao Kerberos discutido no capítulo 8, serão analisados.

A.1 TCPDUMP w2k5

Implementei uma versão especial derivada do TCPDUMP 3.5, chamada TCPDUMP w2k5, para filtrar os pacotes vulneráveis de *logon* Kerberos da rede direcionados para as portas TCP/UDP 88. Se um pacote inicial de *logon* Kerberos (KRB_AS_REQ) é identificado e o tipo de pré-autenticação é baseado no *timestamp* (PA_ENC_TIMESTAMP), essa versão imprime em *stdout* o nome do usuário (*username*) que iniciou o processo de *logon*, e os bytes criptografados de pré-autenticação que podem ser explorados para obter a senha do usuário. Esta funcionalidade foi adicionada em um módulo chamado *print-w2k5.c*, e foi baseada na especificação dos pacotes Kerberos encontrada no [RFC1510].

Esta versão especial é executada com o comando

```
tcpdump port 88
```

Isso aciona o programa, que inicia a filtragem da rede, escrevendo em *stdout* as tentativas de *logon* Kerberos detectadas, conforme o exemplo mostrado nas linhas abaixo :

```
Administrator:$w2k5$8505bc5ffd3fd25eccfc2892e2146bcb0628de11f5ada8c8
e875ebe5585e35cdf7c3bde7f838dcd300b8cce9582d6d71d4769393
administrator:$w2k5$97dc07f7a6eec6170e18c5874c2c609b6d2d93ff8bf6fc72
e1f577456e3291a7d62b15a5fd3ed250fd284a9b072996124fccbbb3
```

Elas indicam que o usuário 'administrador' iniciou o processo de *logon* na rede duas vezes. Qualquer outro usuário do sistema que iniciasse o *logon* seria detectado também. Os caracteres *$w2k5$* que vêm logo em seguida indicam que o programa utilizado para obter as informações foi a versão especial do TCPDUMP. Essa informação é utilizada pelo John The Ripper para ativar o módulo de ataque adequado. Os 52 bytes hexadecimais que se seguem contêm os bytes criptografados de pré-autenticação.

A.2 John The Ripper w2k5

O programa John the Ripper é utilizado para aplicar um ataque de dicionário com variações de palavras. Isto significa que o programa vasculha um grande dicionário, com dezenas de milhares de palavras obtidas de várias línguas. Para cada palavra, ele realiza milhões de variações, como inserção e troca de letras, números e símbolos, de forma a tentar descobrir mesmo senhas que não são palavras de dicionário, mas são derivadas delas. O modo como ele constrói estas variações é totalmente customizável através de um arquivo de configuração, e geralmente inclui as variações mais comuns que as pessoas aplicam às suas senhas. Este ataque é muito mais eficiente que um ataque de força bruta, e consegue descobrir rapidamente senhas que não são suficientemente aleatórias. Enquanto uma senha típica de oito caracteres de oito bits aleatórios cada um possui $(2^8)^8 = 2^{64} \simeq 1,8 * 10^{19}$ possíveis combinações, um ataque deste tipo com, por exemplo, um grande dicionário com 100.000 palavras e 1.000.000 de variações para cada palavra, possui $10^5 * 10^6 = 10^{11}$ possíveis combinações. Se cada tentativa demorar $10\mu s$, aquele ataque levaria cerca de 5,85 milhões de anos, enquanto este levaria cerca de 11 dias. Como o ataque é altamente paralelizável, podendo utilizar n máquinas, cada uma usando um subconjunto disjunto das palavras do dicionário, é possível dividir este período de 11 dias por n e realizar um ataque usando um grande dicionário em questão de horas.

Uma versão especial chamada John The Ripper w2k5, derivada do John The Ripper 1.6, foi construída por mim para implementar os cálculos que descriptografam os bytes de pré-autenticação criptografados obtidos pelo TCPDUMP w2k5. Esses cálculos estão detalhados no capítulo 8. Essa versão especial aproveita a capacidade nativa do John

The Ripper de ler um dicionário e criar variações a partir das palavras. Cada variação gerada pelo John The Ripper é usada como entrada desses cálculos. Caso a variação seja a senha do usuário, no fim dos cálculos é obtido como texto claro um *timestamp* Kerberos. Nesse caso, o John The Ripper w2k5 armazena o nome do usuário e a senha em claro encontrada.

O John The Ripper w2k5 aceita como entrada a saída do TCPDUMP w2k5 (um arquivo texto), e após a execução, escreve em um arquivo de saída "john.pot" os nomes dos usuários e as senhas em texto claro associadas descobertas. O tempo necessário para encontrar as senhas em claro é bastante variável, dependendo muito do dicionário utilizado e de quão parecida a alguma palavra do dicionário a senha era. Usualmente varia de alguns minutos a poucas horas (no caso das senhas muito semelhantes a palavras do dicionário). Caso o ataque de dicionário não seja bem sucedido, um ataque de força bruta pode ser realizado, mas dependendo da quantidade de caracteres da senha e possíveis valores para estes caracteres, o ataque de força bruta pode demorar demais (ver seção 8.3.5).

Caso a saída do TCPDUMP w2k5 mostrada anteriormente fosse usada como entrada para o John The Ripper w2k5, o seguinte resultado dentro do arquivo "john.pot" seria obtido:

```
Administrator:$w2k5$8505bc5ffd3fd25eccfc2892e2146bcb0628de11f5ada8c8
e875ebe5585e35cdf7c3bde7f838dcd300b8cce9582d6d71d4769393:vmonlo;
administrator:$w2k5$97dc07f7a6eec6170e18c5874c2c609b6d2d93ff8bf6fc7
2e1f577456e3291a7d62b15a5fd3ed250fd284a9b072996124fccbbb3:vmonlo;
```

Esse resultado indica que a senha que o administrador usou durante as duas tentativas de *logon* é 'vmonlo;' (sem as aspas simples).

A.3 Outros Pacotes Analisados

A utilização de pacotes de segurança como um meio para analisar o ambiente de rede é um dos melhores meios de assegurar que todas as máquinas do ambiente estão tão seguras quanto possível e se adequam uniformemente à política de segurança escolhida para o Domínio. Eles concentram em um programa todo o conhecimento sobre vulnerabilidades descobertas até o momento em que foram feitos. Alguns ainda possuem um serviço onde a lista de vulnerabilidades pode ser expandida para acomodar as últimas vulnerabilidades descobertas. Esta expansão pode ser manual ou automática.

São muitos os pacotes de segurança disponíveis. Os pacotes profissionais disponíveis são relativamente recentes, desenvolvidos a partir de 1997, e alguns nem são conhecidos amplamente. Bons pontos de referência para encontrar tais pacotes são as URLs:

- *http://www.ntsecurity.net*
- *http://www.winntmag.com*

Destes, alguns são altamente especializados para certos sistemas operacionais. O destaque, aqui, vai ser dado para os *"Security Scanners"*[1], ou pacotes de análise de segurança que realizam *análises* locais e *simulações* de ataques remotos via rede, listando as vulnerabilidades encontradas, sugerindo correções, e às vezes corrigindo automaticamente a falha. Aqueles que lidam com o Windows NT/9x serão o foco principal da pesquisa.

Os pacotes analisados foram:

- HackerShield 2.0 (da *Bind View*)
- InternetScanner 6.0 (da *Internet Security Systems)*
- Security Test & Analysis Tool (STAT) 2.0 (da *Harris Corporation*)
- SecurityAnalyser (da *WebTrends*)
- Kane Security Analyst (KSA) 4.60 (da *Security Dynamics Technologies, Inc.*)
- NetPulse 1.0, (da *Labcal Technologies, Inc.*)
- Security Scanner for SAMBA (ADMsmb) v0.2 (*GNU General Public License*)

As características mais interessantes destes pacotes serão mostradas. Em geral, estes pacotes classificam as vulnerabilidades em 4 níveis:

Alto Vulnerabilidades que podem ser exploradas através da rede para ganhar acesso a uma máquina. Isto também inclui vulnerabilidades que podem ser exploradas por usuários locais para obter acesso administrativo na máquina.

Médio Estas vulnerabilidades permitem a um usuário exceder suas permissões. Se um usuário pode exceder suas permissões, ele também pode ser capaz de explorar outras vulnerabilidades para obter acesso administrativo na máquina. Ataques de Negação de Serviço (" *Denial-of-Service Attacks*") que podem comprometer todo o sistema também são incluídos.

[1]Por exemplo, "*Port Scanners*" - que vasculham portas TCP/UDP em busca de serviços ativos com vulnerabilidades conhecidas - poderiam ter sido analisados, mas estes produtos oferecem apenas um subconjunto das funcionalidades de análise desejadas. Os "*Security Scanners*" são produtos que apresentam funcionalidades complexas e que portanto são interessantes de serem comparados. Muitos "*Security Scanners*" possuem " *Port Scanners*" inclusos.

Baixo Vulnerabilidades que podem ser exploradas via rede para permitir a um usuário não autorizado juntar informações sobre uma máquina específica. Também se incluem ataques de negação de serviço que param um serviço, porem não todo o sistema.

Aviso Prática de boas segurança recomendadas

As análises foram realizadas em uma máquina padrão Windows NT cliente *(Windows NT Workstation 4.0* com *Service Pack 6)* com serviços Internet instalados (Internet Information Server (IIS) 4.0, que vem no pacote *Option Pack*), como WWW e FTP.

A.3.1 Comentários Gerais dos Resultados

Considerando-se como critério a escolha de um *security scanner* que possa analisar o maior número de falhas com a maior profundidade e abrangência possível, o HackerShield, o InternetScanner, o Security Analyser e o STAT parecem ser os mais adequados.

O HackerShield e o InternetScanner têm a desvantagem de não serem específicos para o NT, apesar de possuirem bastante testes para este sistema. A vantagem deles é a existência de muitos testes para sistemas UNIX, roteadores e *switches,* o que pode ser vantajoso numa rede de sistemas heterogêneos.

O Security Analyser parecer ser bastante completo e meticuloso para o Windows NT. Ele foi muito meticuloso na análise do Internet Information Server (IIS) 4.0.

O STAT parece ser altamente específico para o Windows NT, e na versão completa procura por várias centenas de vulnerabilidades. Infelizmente, o STAT disponível para avaliação na Internet procura por apenas algumas vulnerabilidades, o que impede a realização de uma comparação adequada.

O KSA e o NetPulse são principalmente analisadores de *políticas de segurança* do Domínio, ou de configuração do sistema. Possuem um subconjunto reduzido da funcionalidade dos outros pacotes. Dada uma política de segurança desejada, eles verificam se todas as máquinas do Domínio se adequam a ela, e apontam aquelas que não.

A.3.2 HackerShield

O HackerShield 2.0 build 2085 não é um pacote específico para NT. Sua intenção é analisar um ambiente heterogêneo de rede constituído por máquinas rodando Windows NT, Linux, SunOS e outros sabores de Unix. Possui uma base de dados atualizável, que é obtida através de *email* ou *download* a partir do *site* do produto. Fácil de utilizar. Possui facilidades como suporte para exportar para formatos padrões de banco de dados, como o MDB da Microsoft.

Alguns aspectos fracos incluem o fato de não ser específico para Windows NT e ser lento. Muitas falhas relacionadas com o Registro do Sistema, DCOMs, Componentes ActiveX e outras vulnerabilidades altamente especializadas do Windows NT não são verificadas.

Ele examina os seguintes tipos de vulnerabilidades:

Negação De Serviço: falhas de segurança geralmente exploradas a partir do lado de fora através de um ataque via rede, que comprometem o funcionamento adequado de alguma máquina ou serviço.

Compartilhamento De Arquivos: verifica os serviços de compartilhamento de arquivos em uso tais como o serviço de compartilhamento de arquivos do NT e NFS.

Obtenção de informações Confidenciais: descobre que tipo de informação está disponível que poderia ser utilizada em um ataque. Finger, Sendmail, e outros serviços são analisados.

Servidores FTP, Mail, SSH, NNTP e TFTP : descobre vulnerabilidades em várias distribuições destes servidores para NT e UNIX.

Integridade: cria uma lista de atualizações (*Hotfixes*) da Microsoft que podem ou não podem ser instalados na máquina. Confirma que todas as possíveis atualizações foram aplicadas e estão atualizadas, tanto do sistema quanto de aplicativos (por exemplo, Internet Explorer).

Verificação de Vulnerabilidade: verifica configurações que podem comprometer o sistema, tais como tamanho mínimo da senha, e vulnerabilidades associadas ao registro, RPC.

Acesso Remoto: verifica vulnerabilidades como RSH e rexec.

Roteadores e Hubs: verifica potenciais vulnerabilidades contra o hardware da rede local. Muitos hardwares vêm configurados com senhas defaults que podem ser exploradas por intrusos.

Servidores Web: verifica a presença de servidores Web conhecidos e examina-os atrás de vulnerabilidades e versões desatualizadas.

Foram encontradas 29 vulnerabilidades na máquina: 2 consideradas de alto risco, 7 de risco médio, e 20 de baixo risco. A documentação da análise é muito abrangente, incluindo uma explicação sucinta de cada vulnerabilidade encontrada, uma classificação de acordo com a

área do sistema operacional que é afetada, e uma descrição detalhada da vulnerabilidade indicando também como corrigi-la e se a autocorreção pode fazer isto automaticamente.

Item	Vulnerabilidade	Descrição	Risco
1	nt_smb_enumerate	Mostra compartilhamento de arquivos regulares e escondidos	baixo
2	nt_service_list	Serviços X,Y e Z de terceiros estão rodando	baixo
3	smb_domain_info	pessoas não autorizadas podem obter informação demais da máquina	baixo
4	smb_enum_group_members	pessoas não autorizadas podem obter informações demais dos grupos de usuários	médio
5	smb_enum_passwd_policy	pessoas não autorizadas podem obter informações demais da política de senhas	baixo
6	smb_enum_shares	pessoas não autorizadas podem obter informações demais sobre arquivos compartilhados	baixo
7	smb_enum_transports	pessoas anônimas podem obter informações sobre os protocolos de transporte utilizados	baixo
8	smb_enum_users	pessoas não autorizadas podem obter informações demais dos usuários	médio
9	smb_timestamp	pessoas não autorizadas podem obter o horário da máquina	baixo
10	smb_walk_sids_1	pessoas não autorizadas podem obter SIDs de usuários e da máquina	médio
11	tcp_open_ports	certas portas estranhas foram encontradas no estado *listening*	baixo
12	nt_event_log_readable	Os Logs de Segurança do Sistema podem ser lidos por usuários normais	médio
13	nt_startup_promote	Usuários podem adicionar programas que rodam com permissões maiores	ALTO
14	nt_auto_reg_write	É possível executar arquivos *.reg que podem se constituir em Cavalos de Tróia	médio
15	enumerate_users_by_right	É possível a qualquer um descobrir os Privilégios (*User Rights*) de cada usuário	baixo
16	nt_admin	A conta do administrador não está renomeada	baixo
17	nt_screensaver	O screensaver não está instalado	baixo

18	iis_rdl	O módulo RDS DataFactory do IIS permite que qualquer um acesse qualquer BD no sistema	ALTO
19	nt_passwd_filter	Não há filtro de senhas instalado	baixo
20	nt_passwd_length	Não há limite mínimo para o tamanho da senha	baixo
21	nt_passwd_memory	Senhas podem ser reutilizadas por usuários	baixo
22	nt_passwd_minimum	Não há limite máximio de idade para as senhas	médio
23	c2_allocate_drives	Drives não são associados exclusivamente ao usuário do console	baixo
24	c2_display_last_user	O nome do último usuário é mostrado no na tela de *login*	baixo
25	c2_drives_and_printers	Usuários podem alocar letras de drives e impressoras	baixo
26	c2_logon_message	Mensagem de aviso Legal não é mostrada na tela de *login*	baixo
27	c2_security_log_overwritten	O log de eventos pode ser sobreescrito	médio
28	c2_shutdown_button	O usuário pode fazer um *shutdown* no sistema sem antes "logar"	baixo
29	nt_os2_enabled	É recomendável desabilitar o subsistema OS/2	baixo

Executando o programa com a opção de fazer todas as análises, mesmo as agressivas, foi descoberto ainda:

Item	Vulnerabilidade	Descrição	Risco
30	nt_spools_dos	O *print spooler* pode ser explorado para travar a máquina	médio

A.3.3 Internet Scanner

O Internet Scanner 6.0 não é um pacote específico para o Windows NT, analisando também sistemas UNIX, roteadores e *switches*.

No Windows NT, ele procura por vulnerabilidades no Registro do Sistema através da análise das *chaves críticas* onde cavalos de tróia podem ser implantados, faz uma verificação (*checksum*) de bibliotecas (DLLs) do sistema, verifica compatibilidade com

HKLM\Software\Microsoft\Windows\CurrentVersion\Run
HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce
HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx
HKLM\Software\Microsoft\Windows NT\CurrentVersion\AeDebug
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File

Tabela A.2: Entradas do Registro do Sistema vulneráveis que permitem aumento do nível de acesso detectadas pelo Internet Scanner

o nível de segurança C2, analisa privilégios, procura por serviços desconhecidos ou não necessários, e vulnerabilidades bem específicas, como a de objetos DCOM e protocolo Lan Manager sendo usado.

Foram encontradas 54 vulnerabilidades classificadas como de médio risco e 3 como de alto risco, bem como dezenas relacionadas com políticas de segurança não suficientemente estritas. Nestas vulnerabilidades, não havia relato de nada relacionado ao servidor FTP, NNTP ou qualquer coisa relacionada com o Internet Information Server 4.0 que estava rodando na máquina.

Foram consideradas como vulnerabilidades de alto risco o fato dos usuários do grupo "Operadores do Sistema" (*Server Operators*) possuírem privilégios de realizar *Backup* e *Restore*. Estes privilégios são perigosos, porque permitem passar por cima das listas de controle de acesso (ACL) dos arquivos. A outra vulnerabilidade deste grupo foi o fato da biblioteca '*fpnwclnt.dll*' ter falhado na verificação.

As chaves críticas do Windows NT 4.0, mesmo com a última atualização (*Service Pack 6*), permitem a qualquer um (*Everyone*) alterar as informações nelas contidas de modo a conseguir alguma vantagem, como pode ser visto nas tabelas A.2 e A.3. Nestas chaves, o acesso irrestrito (*Everyone*) deveria ter direito apenas de leitura.

A.3.4 Security Test & Analysis Tool (STAT)

O STAT oferece detecção de várias centenas de vulnerabilidades conhecidas do Windows NT na versão completa. Há um recurso de "AutoFix" que permite a aplicação automática da correção sugerida. Há um número limitado de correções automáticas via "AutoFix". Esta função também possui uma função de "desfazer" disponível. Além das vulnerabilidades, ele detecta as versões e atualizações do sistema operacional e de aplicativos, e sugere a aplicação de novas atualizações quando necessário. Cada vulnerabilidade possui uma descrição de como poderia ser utilizada por alguém mal-intencionando, a solução necessária para corrigir a vulnerabilidade, informações extras e apontadores para o assunto, e uma extensa base de conhecimento, apontando artigos da Microsoft.

HKLM\Software\Microsoft\Windows\CurrentVersion\App Paths
HKLM\Software\Microsoft\Windows\CurrentVersion\Controls Folder
HKLM\Software\Microsoft\Windows\CurrentVersion\DeleteFiles
HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer
HKLM\Software\Microsoft\Windows\CurrentVersion\Extensions
HKLM\Software\Microsoft\Windows\CurrentVersion\ExtShellViews
HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings
HKLM\Software\Microsoft\Windows\CurrentVersion\ModuleUsage
HKLM\Software\Microsoft\Windows\CurrentVersion\RenameFiles
HKLM\Software\Microsoft\Windows\CurrentVersion\Setup
HKLM\Software\Microsoft\Windows\CurrentVersion\SharedDLLs
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensions
HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Drivers
HKLM\Software\Microsoft\Windows NT\CurrentVersion\drivers.desc
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Drivers32\0
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Embedding
HKLM\Software\Microsoft\Windows NT\CurrentVersion\MCI
HKLM\Software\Microsoft\Windows NT\CurrentVersion\MCI Extensions
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Ports
HKLM\Software\Microsoft\Windows NT\CurrentVersion\ProfileList
HKLM\Software\Microsoft\Windows NT\CurrentVersion\WOW

Tabela A.3: Entradas do Registro do Sistema vulneráveis que permitem inclusão de Cavalos de Tróia detectadas pelo Internet Scanner

Infelizmente, a versão disponível para avaliação na Internet analisa apenas algumas vulnerabilidades, o que torna difícil a comparação com os outros produtos. No geral, no entanto, parece que o STAT é altamente específico para o Windows NT e eficiente para analisar e encontrar vulnerabilidades.

A.3.5 Security Analyser

O Security Analyser 2.1 possui um conjunto de níveis de análise que determinam a profundidade da análise que vai ser realizada. Entre estas análises destaca-se o *port scan* de portas TCP/IP. Uma das características interessantes é a capacidade de atualizações automáticas da base de dados para detecção de novas vulnerabilidades, quando elas se tornam disponíveis. Cada teste possui uma ampla descrição explicando sua funcionalidade e algumas referências a artigos da Microsoft, e a correção associada. As descrições são autocontidas e bastante elucidativas.

O conjunto final de vulnerabilidades encontradas é bem organizado em 5 listas diferentes:

Máquinas Contendo as vulnerabilidades encontradas em cada máquina

Vulnerabilidades Classificadas em Risco Alto, Médio e Baixo

Serviços Contendo as vulnerabilidades encontradas em cada Serviço

Usuários Contendo as vulnerabilidades associadas a cada Grupo/Usuário

Correções Necessárias As correções associadas a cada vulnerabilidade encontrada

Nível adotado Contém os testes de segurança associados ao nível de análise adotado.

A análise demorou 25 minutos para ser completada. Ela encontrou 376 vulnerabilidades associadas à configuração padrão na instalação do sistema operacional (!). Destas 376 vulnerabilidades, 64 foram classificadas como de alto risco, 128 de risco médio, e 184 como sendo de baixo risco. Cerca de 173 correções foram sugeridas para as 376 vulnerabilidades. Das 64 vulnerabilidades de alto risco, 35 possuíam correções.

Um relatório no formato HTML também foi gerado. Os resultados mais importantes são:

1. Atualização (*patch*) para 'MSHTML.DLL' não aplicado. Esta biblioteca é o interpretador HTML do Internet Explorer (IE). Há vários problemas de segurança associados a ela na versão 4.0 e 5.0 do IE: variações do problema de cruzamento de frames, colar/copiar feito por *scripts* não confiáveis, problema de privacidade envolvendo a *tag* "IMG SRC" .

2. Atualização (*Hotfix*) não aplicada de um módulo chamado ISM.DLL do Internet Information Server (IIS) 4.0 - o servidor web da Microsoft. Sem esta atualização o IIS 4.0 fica sujeito a um *buffer overflow*[2].

3. Atualização para um problema de sincronização e Negação de Serviço (DoS, de *Denail of Service*) do serviço CSRSS.EXE.

4. Atualização para LSA (*Local Security Authority*) não aplicado. Uma requisição mal feita para o serviço LSA pode fazer com que o serviço pare de responder.

5. SysKey não instalado. A Syskey permite usar criptografia forte para proteger as senhas das contas dos usuários.

6. Vulnerabilidade da *sandbox* da Máquina Virtual Java no ambiente de operação Win32. Uma falha na *sandbox* da máquina virtual Java no IE4.0 e IE5.0 permite que um applet Java ganhe controle total como se fosse uma aplicação normal do usuário.

7. *Proxying* de senhas no IIS 4.0 está habilitado. O website tem um diretório apontando para C:\winnt\system32\inetsrv\iisadmpwd. Isto permite a modificação de contas mesmo atrás de um *firewall* através do WebServer.

8. Uma conta foi encontrada com uma senha muito fácil de adivinhar.

9. Conta com senha curta demais.

10. DoS ao fazer telnet para a porta 135 TCP, provocando um problema no módulo de RPC. Um ataque poderia usar telnet para se conectar à porta 135 na máquina, digitar dez ou mais caracteres aleatórios e desconectar. A utilização de CPU no servidor iria para 100% e não diminuiria até o servidor ser reiniciado.

11. Entradas do Registro do Sistema permitem a inoculação de cavalos de Tróia (ver seção A.3.3).

12. Permissões perigosas para chaves do Registro do Sistema relacionadas com *Open Database Connectivity* (ODBC) podem permitir acesso a informações confidenciais.

13. Usuários podem acessar a chave HKLM\Software\Microsoft\Windows NT\ CurrentVersion\Perflib, o que permite verificar performance do sistema e que processos estão rodando.

[2]Este ataque, muito interessante, afetou na época (junho de 1999) 90% da base instalada de IIS, e foi realizado enviando o cabeçalho de requisição HTTP 'GET /[overflow].htr HTTP/1.0', onde [overflow] deveria ter mais ou menos 3000 bytes.

14. Controle *ActiveX Eyedog* detectado. Este controle permite que usuários consigam remotamente informações do Registro do Sistema e de características da máquina.

15. Controle *scriptlet.typelib* detectado. Ele permite que arquivos sejam criados ou modificados remotamente.

16. O IIS pode instanciar o objeto *RDSServer.DataFactory*, o que significa que um cliente web pode usar o objeto RDS.DataFactory para instruir o servidor a acessar dados em um provedor OLE DB, o que inclui a execução de chamadas SQL para base de dados compatíveis com ODBC. Os resultados são retornados via HTTP para o cliente.

17. Atualização para o IIS 4.0 contra cabeçalhos HTTP mal feitos não está presente, o que permite que um ataque específico do tipo DoS seja aplicado.

18. Atualização para o vazamento de memória do processo SPOOLS não está aplicado

19. Atualização para as chamadas IOCTLs não protegidas não está aplicado, o que permite que qualquer usuário desabilite o mouse e o teclado do sistema.

20. O IIS 4.0 está mandando erros ASP muito detalhados, o que inclui o nome do arquivo, mensagem de erro e número da linha onde o erro ocorreu.

21. Permissão de usar caminhos relativos através de ".." em scripts ASP está habilitada, o que pode fazer com que algum script ASP venha a ser explorado para executar algum programa.

22. *Includes* são permitidos na execução de scripts ASP pelo servidor, o que pode permitir páginas executar programas.

23. Uma vulnerabilidade existe que faz com que pacotes PASV FTP enviados para o IIS 4.0 possam criar um ataque do tipo DoS.

24. Uma vulnerabilidade existe que faz com que requisições HTTP GET enviadas para o IIS 4.0 possam criar um ataque do tipo DoS, e até mesmo paralisar o servidor

25. Uma vulnerabilidade existe que permite que requisições FTP LIST causem um ataque do tipo DoS e até mesmo permita a execução de código malicioso no nível de permissão do serviço IIS.

26. Uma vulnerabilidade foi detectada no serviço RPCSS.EXE que permite degradar a performance do sistema usando RPC *spoofing*.

A.3.6 Kane Security Analyst (KSA)

O KSA 4.60 analisa principalmente a *política de segurança* do Domínio, implementada pela Autoridade Local de Segurança (LSA). Isto inclui verificação de políticas padrões de segurança, como:

- contas não usadas há mais de um número definido de dias
- horários do dia permitidos para a utilização da máquina
- idade máxima da senha
- número mínimo de caracteres da senha
- privilégios (*User Rights*) excessivos
- permissão de acesso ao CD-ROM e aos disquetes
- serviço de discagem (RAS, de *Remote Dialup Access Server*)
- bloqueamento da conta caso número determinado de tentativas de autenticação falhe
- auditoramento de logins com sucesso e sem sucesso
- serviços de FTP, SQL Server, DHCP e WWW rodando
- sistema de arquivos NTFS instalados.

Há também um utilitário incluído para tentar descobrir senhas em dicionário. Houve 63 falhas críticas encontradas na política do sistema, distribuída na seguinte proporção:

Política avaliada	Quantidade de erros na política	Classificação
Restrições nas contas dos usuários	78%	falha
Política de Senhas	71%	falha
Controle de Acesso	81%	atenção
Monitoramento do Sistema	42%	falha
Integridade dos Dados	33%	falha
Confidencialidade dos Dados	75%	falha

Os 10 maiores riscos encontrados nas configurações default das máquinas[3]:

[3] *Bloqueamento de conta* indica uma situação onde o usuário tentou se autenticar um determinado número de vezes e falhou em todas essas vezes. É interessante usar esta política para evitar ataques de força bruta para adivinhar senhas das contas.

Item por ordem de risco	Risco
1	Bloqueamento de conta não está ligado
2	Auditoramento não habilitado na máquina
3	31% das contas possuem privilégios administrativos
4	Usuários não são bloqueados por tempo suficiente
5	Usuários não são bloqueados depois de tentativas frustradas de *login*
6	100% dos usuários estão inativos
7	100% das máquinas não mostram Mensagens de Teor Legal antes do *login*
8	*Logins* fracassados não demoram tempo suficiente
9	Muitos *logins* fracassados são permitidos antes do bloqueamento
10	50% dos usuários estão disabilitadas pelo administrador

O pacote concentra-se na análise das políticas implementadas nas máquinas do Domínio, e não cobre vulnerabilidades como gerenciamento de versões e atualizações do sistema e aplicativos, serviços WWW, FTP, NNTP e SMTP.

A.3.7 Netpulse

Duas opções de instalação: Administrator (pode analisar e aplicar as correções) e Auditor (pode apenas analisar). A primeira tarefa do programa é vasculhar a rede local em busca de máquinas potenciais a serem analisadas. Após construir uma lista com as máquinas visíveis, é possível iniciar a análise.

O NetPulse 1.0 é muito similar ao KSA. Ele se limita a apenas analisar a política de segurança do LSA (tamanho mínimo de senha, auditoria para logons, comparação com nível de segurança C2 e outros), sem se aprofundar em áreas como buscas por Cavalos de Tróias, *port scanners*, *checksums* de bibliotecas, análise de permissão de acesso a áreas críticas do sistema, como o Registro do Sistema e certos diretórios que por padrão permitem a escrita por qualquer usuário.

A.3.8 Security Scanner for SAMBA

Este pacote, também chamado ADMsmb, de código fonte aberto, realiza apenas alguns testes de segurança relacionados a acessos a compartilhamentos de arquivos e *named pipes* via LAN Manager/SMB e um utilitário para realizar ataques de dicionário. A principal

adição deste pacote à segurança em geral do Windows NT é conter também o código fonte que realiza estas ações.

Referências Bibliográficas

[Ash98] ASHTON, P. Gaining Domain Admins access on LAN. *NTBugTraq Mailing List Archives*, janeiro 1998. Disponível na Internet: http://www.ntbugtraq.com/ [15 março 2001]

[Ban93] BANYAN. *VINES protocol definition*, junho 1993. Disponível na Internet: http://support.banyan.com/banyan.htm [15 março 2001]

[BM91] BELLOVIN, S.M.; MERRITT, M. Limitations of the kerberos authentication system. In: *Proceedings of the Winter 1991 Usenix Conference*, janeiro 1991. Disponível na Internet: http://web.mit.edu/kerberos/www/papers.html [15 março 2001]

[BM92] BELLOVIN, S. M.; MERRITT, M. Encrypted key exchange: password-based protocols secure against dictionary attacks. *Proceedings of the I.E.E.E. Symposium on Research in Security and Privacy*, Oakland, maio de 1992. Disponível na Internet: http://www.alw.nih.gov/Security/FIRST/papers/crypto/neke.ps e ftp://ftp.research.att.com/dist/smb/ [15 março 2001]

[BMP00] BOYKO, V.; MACKENZIE, P.; PATEL, S. Provably secure password authenticated key exchange using diffie-hellman. In: *Advances in Cryptology, EUROCRYPT 2000*, pp.156-171 (PAK, PAK-X and PPK protocols), maio de 2000. Disponível na Internet: http://www.bell-labs.com/user/philmac/pak.html [15 março 2001]

[Bry88] BRYANT, W. Designing an authentication system: a dialogue in four scenes. *Project Athena Document*, fevereiro 1988. Disponível na internet: http://web.mit.edu/kerberos/www/dialogue.html [15 março 2001]

[Com00] COMPUSERVE. RPA mechanism specifications. Compuserve Computing, 1999. Disponível na Internet: http://www.compuserve.com/rpa/rpadoco.htm [18 dezembro 2000]

[Cus93] CUSTER, H. *Inside Windows NT.* Microsoft Press, 1993.

[DoD85] DEPARTMENT OF DEFENSE STANDARD. *Department of Defense Trusted Computer System Evaluation Criteria* (DoD 5200.28-STD), dezembro 1985. Disponível na Internet: http://www.radium.ncsc.mil/tpep/process/overview.html e http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html [15 março 2001]

[EFF98] ELECTRONIC FRONTIER FOUNDATION. *Cracking DES*: Secrets of Encryption Research, Wiretap Politics & Chip Design. EFF DES Cracker Project, julho 1998. Disponível na Internet: http://www.eff.org/pub/Privacy/Crypto/Crypto_misc/DESCracker/ [15 março 2001]

[FKK96] FREIER, A.O.; KARLTON, P.; KOCHER, P.C. *The SSL Protocol Version 3.0.* Netscape Communications Corporation, março de 1996. Disponível na Internet: http://www.netscape.com/eng/ssl3/ [15 março 2001]

[GLNS93] GONG, L.; LOMAS, M.; NEEDHAM, R.; SALTZER, J. Protecting poorly chosen secrets from guessing attacks. *I.E.E.E. Journal on Selected Areas in Communications*, vol. 11, no. 5, junho de 1993, p.648-656.

[Gon95] GONG, L. Optimal authentication protocols resistant to password guessing attacks. *Proceedings of the 8th IEEE Computer Security Foundations Workshop*, p.24-29, junho 1995.

[Gra99] GRANADO, M.C.; VIEIRA, G.; GEUS, P.L. Aspectos criptográficos no windows NT. *2a Conferência de Redes de Computadores*, Évora, Portugal, 18 e 19 de outubro de 1999. Disponível na Internet: http://www.eventos.uevora.pt/crc99/ [15 março 2001]

[Gra01a] GRANADO, M.C.; GUIMARÃES, C.C. Aplicando ataques de dicionários no protocolo kerberos do windows 2000. *Workshop em Segurança de Sistemas Computacionais (WSeg2001)*, Florianópolis, 5 e 6 de março de 2001. Disponível na Internet: http://www.sctf2001.lcmi.ufsc.br/WS21.html [15 março 2001]

[Gra01b] GRANADO, M.C.; GUIMARÃES, C.C. Attacking windows 2000 kerberos password security. *The Tenth Conference on System Administration, Networking and Security (SANS) 2001.* Baltimore, EUA, 13 a 20 de maio

de 2001. Disponível na Internet: http://www.sans.org/SANS2001.htm [15 março 2001]

[Gut01] GUTMANN, P. Netware 3.x and 4.x authentication. *Godzilla Crypto Tutorial Part 4*, 2001. Disponível na Internet: http://www.cs.auckland.ac.nz/~pgut001/ [15 março 2001]

[Hei01] HEIMDAL: A free Implementation of Kerberos 5. Dezembro de 2000. Disponível na Internet: http://www.pdc.kth.se/heimdal [15 março 2001]

[Hob97] HOBBIT. *CIFS:* Common Insecurities Fail Scrutiny. Avian Research, janeiro 1997. Disponível na Internet: http://www.avian.org [15 março 2001]

[Jab96] JABLON, D. Strong password-only authenticated key exchange. *Computer Communication Review, ACM SIGCOMM*, vol.26, no.5, p.5-26, outubro de 1996. Disponível na Internet: http://www.integritysiences.com/isipubs.html [15 março 2001]

[Jab97] JABLON, D. Extended password key exchange protocols immune to dictionary attacks. *Proceedings of the Sixth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE'97)*, IEEE Computer Society, Cambridge, MA, p.248-255, junho de 1997. Disponível na Internet: http://www.cerc.wvu.edu/wetice/WETICE97/97.html [15 março 2001]

[Jab01a] JABLON, D.P. *Publications on strong password authentication.* Integrity Sciences Inc, 2001. Disponível na Internet: http://www.integritysciences.com/links.html [15 março 2001]

[Jab01b] JABLON, D.P. *History of strong password methods.* Integrity Sciences Inc, 2000. Disponível na Internet: http://world.std.com/~dpj/strong.html [15 março 2001]

[Jas96] JASPAN, B. Dual-workfactor encrypted key exchange: efficiently preventing password chaining and dictionary attacks. *Proceedings of the Sixth Annual USENIX Security Conferences*, pp.43-50, julho 1996. Disponível na Internet: http://www.usenix.org/publications/library/proceedings/sec96/jaspan.html [15 março 2001]

[KB00] BROWN, K. *Programming Windows Security.* Addison-Wesley, 2000.

[Ken97] KENNETH, L.; LEIGHTON, C.; ASHTON, P.; STANSFIELD, D. NT domain authentication protocol. *NT Domains for UNIX Samba Project*, 2000.

Disponível na Internet: http://www.cb1.com/~lkcl/ntdom/ntdomain.html [15 março 2001]

[KL00] KENNETH, L.; LEIGHTON, C. *DCE/RPC over SMB*: SAMBA and windows NT domain internals. Macmillan Technical Publishing, 2000.

[Kle90] KLEIN, D.V. 'Foiling the Cracker': A Survey of, And Implications to, Password Security. In: *Proceedings of the USENIX UNIX Security Workshop.* Aug 1990. pp.5-14.

[KNT94] KOHL, J.; NEUMAN, B.; TS'O T. The evolution of the kerberos authentication service. In: *Distributed Open Systems*, p.78-94, IEEE Computer Society Press, 1994. Disponível na Internet: http://web.mit.edu/kerberos/www/papers.html [15 março 2001]

[KPS95] KAUFMAN, C.; PERLMAN, R.; SPECINER, M. *Network security*: private communication in a public world. Prentice-Hall, 1995.

[L97] L0PHTCRACK HEAVY INDUSTRIES. *L0phtCrack 2.5 Manual.* Security Software Technologies Inc., 1997. Disponível na Internet: http://www.securitysoftwaretech.com/l0phtcrack/documentation/readme.html

[LP96] LEACH, P.; PERRY, D. CIFS: a common internet file system. *Microsoft Interactive Developer*, novembro 1996. Disponível na Internet: http://www.microsoft.com/mind/1196/inthisissue1196.htm [15 março 2001]

[Luc97] LUCKS, S. Open key exchange: how to defeat dictionary attacks without encrypting public keys. *The Security Protocol Workshop '97*, Ecole Normale Superieure, abril de 1997. Disponível na Internet: http://th.informatik.uni-mannheim.de/People/Lucks/papers.html [15 março 2001]

[Mal97] MALMGREN, R. *NT Security Frequently Asked Questions.* 1997. Disponível na Internet: http://www.it.kth.se/~rom/ntsec.html [15 março 2001]

[Mic97a] LEACH, P.J. *CIFS Authentication Protocol.* Microsoft, março 1997. Disponível na Internet: ftp://ftp.microsoft.com/ developr/drg/CIFS/CIFS-Auth.txt

[Mic97b] LEACH, P.J. *CIFS Authentication Protocols Specification.* Microsoft, março 1997. Disponível na Internet: ftp://ftp.microsoft.com/ developr/drg/CIFS/CIFS-Auth-Spec.txt

[Mic99a] MICROSOFT, Windows NT System Key Permits Strong Encryption of the SAM. *Microsoft Knowledge Base*, Article ID Q143475, fevereiro 2001. Disponível na Internet: http:// support.microsoft.com/support/kb/articles/q143/4/75.asp [15 março 2001]

[Mic99b] MICROSOFT. Windows 2000 Server Authentication: Kerberos Protocol. *Microsoft Technical Library*, 1999. Disponível na Internet: http:// www.microsoft.com/windows2000/library/technologies/security/default.asp [15 março 2001]

[Mic99c] MICROSOFT. Encrypting File System for Windows 2000. *Microsoft Windows 2000 Server Technical Library*, 1999. Disponível na Internet: http:// www.microsoft.com/windows2000/library/technologies/security/default.asp [15 março 2001]

[Mic00a] SHELDON, B.; WILANSKY, E. Active directory services. In: MICROSOFT. *MCSE Training Kit for Windows 2000 Server*. Microsoft Press, 2000.

[Mic00b] MICROSOFT. Distributed Component Object Model (DCOM). *Microsoft COM Technologies*, 2000. Disponível na Internet: http://www.microsoft.com/com/tech/dcom.asp [5 dezembro 2000]

[Mic00c] MICROSOFT. *Kerbcon.h*, Microsoft Platform SDK, 2000.

[Mic00d] MICROSOFT. Windows 2000 Kerberos Authentication. *Microsoft Technical Library*, 1999. Disponível na Internet: http://www.microsoft.com/windows2000/library/howitworks/security/kerberos.as [15 março 2001]

[Mic00e] MICROSOFT. Handling authentication. *Microsoft Platform SDK documentation*, Visual Studio 6.0 MSDN CDROM, 1998

[Mic00f] MICROSOFT. Comparison of Windows NT Network Protocols. *Microsoft Knowledge Base*, Article ID Q128233, outubro 2000. Disponível na Internet: http://support.microsoft.com/support/kb/articles/Q128/2/33.ASP [15 março 2001]

[Mic00g] MICROSOFT. Microsoft Cryptographic Service Providers. Microsoft Platform SDK documentation, 2000. Disponível na Internet: http://msdn.microsoft.com/library/psdk/crypto/cryptoref_4dmb.htm [15 março 2001]

[Mil88] MILLER, S.; NEUMAN, B.; SCHILLER, J.; SALTZER, J. Section E.2.1: kerberos authentication and authorization system. *Project Athena Technical Plan*, M.I.T. Project Athena, Cambridge, MA, outubro de 1988. Disponível na Internet: http://web.mit.edu/kerberos/www/papers.html [15 março 2001]

[MR98] RUSSINOVICH, M. Windows NT security. *Windows NT Magazine*, junho 1998. Disponível na Internet: http://www.win2000mag.com/Articles/Index.cfm?ArticleID=3143 [15 março 2001]

[MS00a] MICROSOFT. Common LDAP RFCs. *Microsoft Knowledge Base*, Article ID Q221606, outubro 2000. Disponível na Internet: http://support.microsoft.com/support/kb/articles/Q221/6/06.ASP [15 março 2001]

[MS01] MICROSOFT. Web client NTLM authentication vulnerability. *Microsoft Security Bulletin (MS01-001)*, janeiro 2001. Disponível na Internet: http://www.microsoft.com/technet/security/bulletin/ms01-001.asp [15 março 2001]

[MSDN98a] MICROSOFT. Well-known SIDs. *Microsoft Platform SDK Documentation*, dezembro 2000. Disponível na Internet: http://msdn.microsoft.com/library/psdk/winbase/accctrl_416b.htm [15 março 2001]

[MSDN98b] MICROSOFT. Windows NT privileges. *Microsoft Platform SDK Documentation*, dezembro 2000. Disponível na Internet: http://msdn.microsoft.com/library/psdk/winbase/accctrl_96lv.htm [15 março 2001]

[MSDN98c] MICROSOFT. Provider-independent access mask format. *Microsoft Platform SDK Documentation*, Visual Studio 6.0 MSDN CDROM, 1998.

[MSK99] MCCLURE, S.; SCAMBRAY, J.; KURTZ, G. *Hacking exposed*: network security secrets and solutions. Berkeley: McGraw-Hill, 1999.

[NBS77] NATIONAL BUREAU OF STANDARDS. *Data Encryption Standard (DES)*. NBS FIPS PUB 46, janeiro 1977. NBS FIPS PUB 46-1, janeiro 1988. NBS FIPS PUB 46-2, dezembro 1993. Disponível na Internet: http://www.itl.nist.gov/fipspubs/fip46-2.htm [15 março 2001]

[NIST93] NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. *Secure Hash Standard.* NIST FIPS PUB 180, maio 1993. NIST FIPS PUB 180-1, abril 1995. Disponível na Internet: http://www.itl.nist.gov/fipspubs/fip180-1.htm [15 março 2001]

[NK93] KOHL, J.; NEUMAN, C. The Kerberos Network Authentication Service (V5). *Network Working Group*, RFC1510, setembro 1993. Disponível na Internet: http://rfc.net/rfc1510.html [15 março 2001]

[NKT00] NEUMAN, C.; KOHL, J.; TS'O, T. The Kerberos Network Authentication Service (V5). *Network Working Group*, draft-ietf-cat-kerberos-revisions-06.txt, julho 2000. Disponível na Internet: http://www.ietf.org/ID.html [15 março 2001]

[Nov01] NOVELL. *NetWare 3.2 Documentation.* Disponível na Internet: http://www.novell.com/products/netware3/ [15 março 2001]

[NS93] NEUMAN, B.C.; STUBBLEBINE, S.G. A note on the use of timestamps as nonces. *Operating Systems Review*, 27(2):10-14, abril 1993. Disponível na Internet: http://web.mit.edu/kerberos/www/papers.html [15 março 2001]

[NSA95] NATIONAL SECURITY AGENCY. *'Microsoft Corporation Windows NT Workstation and Windows NT Server Version 3.5 with Service Pack 3' Trusted Computer System Evaluation.* NSA Trusted Product Evaluation Program (TPEP) List, julho 1995. Disponível na Internet: http://www.radium.ncsc.mil/tpep/epl/entries/CSC-EPL-95-003.html [15 março 2001]

[NSA99] NATIONAL SECURITY AGENCY. *'Microsoft Corporation Windows NT Workstation and Windows NT Server Version 4.0 with Service Pack 6a and C2 Update' Trusted Computer System Evaluation.* NSA Trusted Product Evaluation Program (TPEP) List, novembro 1999. Disponível na Internet: http://www.radium.ncsc.mil/tpep/epl/entries/CSC-EPL-95-003.html [15 março 2001]

[PK99] PERLMAN, R.; KAUFMAN, C. Secure Password-Based Protocol for Downloading a Private Key. In: *Proceedings of the 1999 Network and Distributed System Security Symposium.* 1999. Disponível na Internet: http://www.isoc.org/ndss99/proceedings [15 março 2001]

[PZ99] PALL, G.S.; ZORN, G. Microsoft point-to-point encryption (MPPE) protocol. *Network Working Group*, draft-ietf-pppext-mppe-03.txt, maio 1999. Disponível na Internet: http://www.ietf.org/ID.html [15 março 2001]

[Q128233] MICROSOFT. Comparison of Windows NT Network Protocols. Microsoft Knowledge Base, Article Q128233, outubro 2000. Disponível na Internet: http://support.microsoft.com/support/kb/articles/Q128/2/33.ASP [15 março 2001]

[Q223184] MICROSOFT. Registry Entries for the W32Time Service. *Microsoft Knowledge Base*, Article Q223184, abril 2000. Disponível na Internet: http://support.microsoft.com/support/kb/articles/Q223/1/84.ASP [15 março 2001]

[Q224799] MICROSOFT. Basic Operation of the Windows Time Service. *Microsoft Knowledge Base,* Article Q224799, março 2001. Disponível na Internet: http://support.microsoft.com/support/kb/articles/q224/7/99.asp [15 março 2001]

[Q243993] MICROSOFT. Windows Time Service Continues to Monitor Secondary Client Broadcasts. *Microsoft Knowledge Base,* Article Q243993, dezembro 2000. Disponível na Internet: http://support.microsoft.com/support/kb/articles/Q243/9/93.ASP [15 março 2001]

[Q258059] MICROSOFT. How to Synchronize the Time on a Windows 2000-Based Computer in a Windows NT 4.0 Domain. *Microsoft Knowledge Base*, Article Q258059 , outubro 2000. Disponível na Internet: http://support.microsoft.com/support/kb/articles/Q258/0/59.ASP [15 março 2001]

[RFC1305] MILLS, D. Network Time Protocol (Version 3): Specification, Implementation and Analysis. *Network Working Group,* RFC 1305, março 1992. Disponível na Internet: http://rfc.net/rfc1305.html [15 março 2001]

[RFC1320] RIVEST, R. The MD4 Message-Digest Algorithm. *Network Working Group*, RFC-1320, abril 1992. Disponível na Internet: http://rfc.net/rfc1320.html [15 março 2001]

[RFC1321] RIVEST, R. The MD5 Message-Digest Algorithm. *Network Working Group*, RFC-1321, abril 1992. Disponível na Internet: http://rfc.net/rfc1321.html [15 março 2001]

[RFC1334] LLOYD, B.; SIMPSON, W. PPP challenge handshake authentication protocol (CHAP). *Network Working Group*, RFC 1334, agosto de 1996. Disponível na Internet: http://rfc.net/rfc1334.html [15 março 2001]

[RFC1508] LINN, J. Generic security service application program interface (GSS-API). *Network Working Group*, RFC1508, setembro de 1993. Disponível na Internet: http://rfc.net/rfc1508.html [15 março 2001]

[RFC1509] WRAY, J. Generic security service API: C-bindings. *Network Working Group*, RFC1509, setembro de 1993. Disponível na Internet: http://rfc.net/rfc1509.html [15 março 2001]

[RFC1510] Favor ver [NK93].

[RFC1769] MILLS, D. Simple Network Time Protocol (SNTP). *Network Working Group*, RFC 1769, março 1995. Disponível na Internet: http://rfc.net/rfc1769.html [15 março 2001]

[RFC1964] LINN, J. The kerberos version 5 GSS-API mechanism. *Network Working Group*, RFC1964, junho 1996. Disponível na Internet: http://rfc.net/rfc1964.html [15 março 2001]

[RFC2104] KRAWCZYK, H.; BELLARE, M.; CANETTI, R. HMAC: Keyed-Hashing for Message Authentication. *Network Working Group*, RFC 2104, fevereiro 1997. Disponível na Internet: http://rfc.net/rfc2104.html [15 março 2001]

[RFC2195] KLENSIN, J.; CATOE, R.; KRUMVIEDE, P. IMAP/POP AUTHorize extension for simple challenge/response. *Network Working Group*, RFC2195, setembro de 1997. Disponível na Internet: http://rfc.net/rfc2195.html [15 março 2001]

[RFC2246] DIERKS, T.; ALLEN, C. The TLS protocol version 1.0. *Network Working Group*, RFC2246, janeiro de 1999. Disponível na Internet: http://rfc.net/rfc2246.html [15 março 2001]

[RFC2478] BAIZE, E.; PINKAS D. The simple and protected GSS-API negotiation mechanism (SPNEGO). *Network Working Group*, RFC2478, dezembro de 1998. Disponível na Internet: http://rfc.net/rfc2478.html [15 março 2001]

[RFC2617] FRANKS, J. et all. *HTTP authentication*: basic and digest access authentication. *Network Working Group*, RFC 2617, junho de 1999. Disponível na Internet: http://rfc.net/rfc2617.html [15 março 2001]

[Riv] RIVERST, R. RSA DATA SECURITY. Algoritmo Criptográfico Proprietário RC4. Algumas informações disponíveis em http://theory.lcs.mit.edu/~rivest/crypto-security.html [15 março 2001]

[SB00] SWIFT, M.; BREZAK, J. The Windows 2000 RC4-HMAC Kerberos encryption type. *CAT Working Group*, draft-brezak-win2k-krb-rc4-hmac-03.txt, junho 2000. Disponível na Internet: http://wuarchive.wustl.edu/systems/unix/NetBSD/NetBSD-current/src/crypto/dist/heimdal/doc/standardisation/rc4-hmac.txt. Disponível em [Hei01].

[Sch96] SCHNEIER, B. *Applied Cryptography*. New York: Wiley, 1996

[Sch98] SCHNEIER, B.; MUDGE. Cryptanalysis of Microsoft's Point-to-Point Tunneling Protocol (PPTP). In: *Proceedings of the 5th ACM Conference on Communications and Computer Security*, ACM Press, novembro 1998. Disponível na Internet: http://dev.acm.org/pubs/contents/proceedings/commsec/288090/ e http://www.counterpane.com/pptp.html [15 março 2001]

[Sch99] SCHNEIER, B.; MUDGE. Cryptanalysis of Microsoft's PPTP Authentication Extensions (MS-CHAPv2). In: *Secure Networking - CQRE [Secure] '99*, Springer-Verlag, p.192-203, novembro 1999. Disponível na Internet: http://link.springer.de/link/service/series/0558/tocs/t1740.htm e http://www.counterpane.com/pptp.html [15 março 2001]

[SH96] SUTTON, S.; HINRICHS, S. *National Security Agency Multi-Level Information System Security Initiative (MISSI) B-level Windows NT Feasibility Study*. Trusted Systems Services Inc, dezembro 1996. Disponível na Internet: http://www.trustedsystems.com/b-level.htm [15 março 2001]

[SMB01] SAMBA. *The SAMBA Project*. Disponível na Internet: http://www.samba.org [15 março 2001]

[SNS88] STEINER, J.G.; NEUMAN, B.C.; SCHILLER, J.I. Kerberos: An Authentication Service for Open Network Systems. In *Proceedings of the Winter 1988 Usenix Conference*, p.191-202, fevereiro 1988. Disponível na Internet: http://web.mit.edu/kerberos/www/papers.html [15 março 2001]

[Sol98] SOLOMON, D.A. *Inside Windows NT Second Edition*. 2a. ed. Microsoft Press, 1998.

[Sol00] SOLOMON, D.A.; RUSSINOVICH, M.E. *Inside Microsoft Windows 2000 Third Edition*. 3a. ed. Microsoft Press, 2000.

[Sta98] STALLINGS, W. *Cryptography and network security*: principles and practice. Second Edition, Prentice Hall, 1998.

[Tho00] THOMAS, S. *SSL and TLS Essentials*. John Willey & Sons, Inc, 2000.

[TNet00a] BOSCH, P. Windows NT security system basics. *Visual C++ Developer*, janeiro 1997. Disponível em CDROM: *Microsoft Technet*, julho 2000. Disponível na Internet: http://msdn.microsoft.com/library/periodic/period97/vc0197.htm [15 março 2001]

[TNet00b] MICROSOFT. Moving Window Manager and GDI into the Windows NT 4.0 Executive. *Microsoft Technet*, 2000. Disponível na Internet: http://msdn.microsoft.com/library/backgrnd/html/msdn_movuser.htm [15 março 2001]

[Tun00] TUNG, B. et al. Public Key Cryptography for Initial Authentication in Kerberos. *Network Working Group*, draft-ietf-cat-kerberos-pk-init-12.txt, julho 2000. Disponível na Internet: http://www.ietf.org/ID.html [15 março 2001]

[Vis97] VISSER, J. *On NT Password Security*. Open Solution Providers, 1997. Disponível na Internet: http://www.osp.nl/infobase/ntpass.html [15 março 2001]

[Wor01] THE WORDLIST PROJECT. Dezembro de 2000. Disponível na Internet: http://wordlists.security-on.net [15 março 2001]

[Wu98] WU, T. The secure remote password protocol. *Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, San Diego, p.97-111, março de 1998. Disponível na Internet: ftp://srp.stanford.edu/pub/srp/srp.ps e http://www.isoc.org/ndss98 [15 março 2001]

[Wu99] WU, T. A Real-World Analysis of Kerberos Password Security. In: *Proceedings of the 1999 Internet Society Network and Distributed System Security Symposium*. 1999. Disponível na Internet: http://www.isoc.org/ndss99/proceedings [15 março 2001]

[Z99a] ZORN, G. Microsoft PPP CHAP Extensions Version 2. *The Internet Engineering Task Force*, draft-ietf-pppext-mschap-v2-03.txt, abril 1999. Disponível na Internet: http://www.ietf.org/proceedings/99jul/I-D/draft-ietf-pppext-mschap-v2-03.txt [15 março 2001]

[Z99b] ZORN, G. Deriving MPPE keys from MS-CHAP V2 credentials. *Network Working Group*, draft-ietf-pppext-mschapv2-keys-02.txt, novembro 1998. Disponível na Internet: http://www.ietf.org/ID.html [15 março 2001]

[Z99c] ZORN, G. MPPE key derivation. *Network Working Group*, draft-ietf-pppext-mppe-keys-00.txt, fevereiro 1999. Disponível na Internet: http://www.ietf.org/ID.html [15 março 2001]

Índice Remissivo