Aspectos Criptográficos no Windows NT

Marcus Cunha Granado
Instituto de Computação, UNICAMP
marcus.granado@dcc.unicamp.br
- Gustavo Maciel Dias Vieira
Instituto de Computação, UNICAMP
gustavo.vieira@dcc.unicamp.br
- Paulo Licio de Geus
Instituto de Computação, UNICAMP
paulo@dcc.unicamp.br

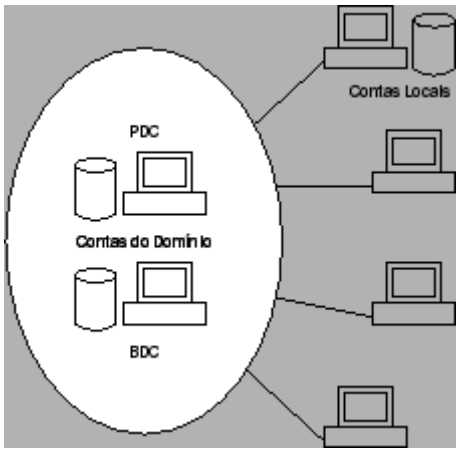
Resumo

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 (*Microsoft CHAP*), na implementação do protocolo PPTP 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).

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 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 falhos de autenticação utilizados no Windows NT 4.0

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) para cada recurso do sistema. Cada elemento dessa lista (ACE) possui um conjunto de descritores de segurança (SIDs), 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 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 rodando 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), deve poder copiar essas informações e utilizá-las para personificar outros usuários. O Windows NT se utiliza de mecanismos de proteção proprietários que pecam por:

- tentam oferecer segurança através do ocultamento do algoritmo 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.
- baseiam-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.

Algoritmos Criptográficos Utilizados no Armazenamento das Senhas

DES (*Data Encryption Standard*)

O DES é um algoritmo de ciframento 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. 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 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.

Table: Tempo para vasculhar um espaço de 2^{56} chaves

Máquina	Op./Sec.	Tempo
	(Approx.)	
Pentium 90Mhz	260000	8800 anos
Pentium 133Mhz	770000	2900 anos
Pentium Pro 200Mhz	970000	2300 anos
EFF DES-cracker	$92 \cdot 10^9$	9 dias

MD4 *Message Digest*

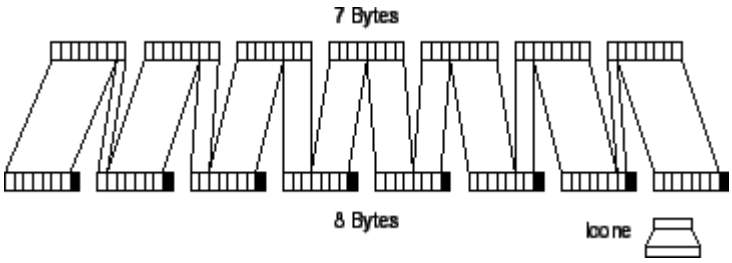
O MD4 é um algoritmo para criar *hashes* criptográficos. Ele recebe uma mensagem de tamanho arbitrário e a transforma num vetor (*hash*) de 16 bytes ou 128 bits. Não é possível inverter o processo. É computacionalmente muito difícil achar outra mensagem que resulte num mesmo *hash*. Portanto, esse *hash* 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, que evita alguns problemas de colisão, onde duas mensagens diferentes poderiam resultar num mesmo *hash*.

Armazenamento de Senhas

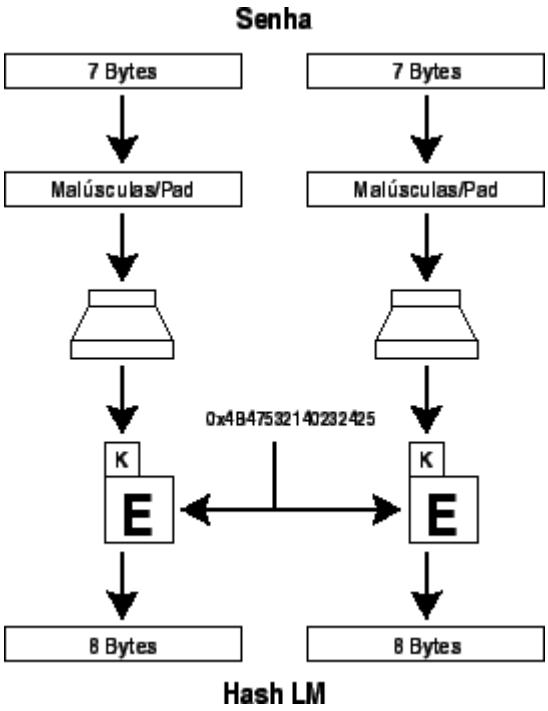
No Windows NT, as senhas e outras informações do usuário são armazenadas na base de dados do SAM (*Security Account Manager*). Por default, apenas o sistema e o administrador têm poderes para acessar seu conteúdo através de chamadas a funções específicas do sistema para tal fim (APIs). Enquanto o sistema está rodando, ele solicita acesso exclusivo ao arquivo onde a base de dados está (*%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 o arquivo SAM._ compactado, que pode ser lido por qualquer um. 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 suscetível a ataques, o que o faz o alvo preferido de ataque quando se deseja quebrar alguma senha.

Hash LM

Usa DES para cifrar uma senha de no máximo 14 caracteres. O *hash* é formado pela cifragem de um ``número mágico'' usando os dois blocos de 7 caracteres como chave. O procedimento da figura abaixo é usado para gerar a chave DES de 56 bits a partir de cada parcela da senha. O blocos em preto de cada um dos 8 bytes indica o bit de paridade 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.



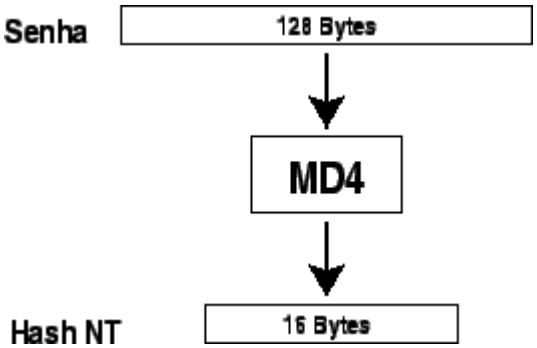
O código ASCII de cada byte da senha é convertido para o correspondente em letras maiúsculas, se for o caso; bytes restantes não utilizados pela senha são preenchidos com conteúdo 0 (*padding*). Os sete bits são transformados em oito; os bytes resultantes do processo são usados como senha do DES. Os dois textos cifrados de 8 bytes formam o *hash* LM.



O número mágico não é guardado na máquina. Ao invés 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.

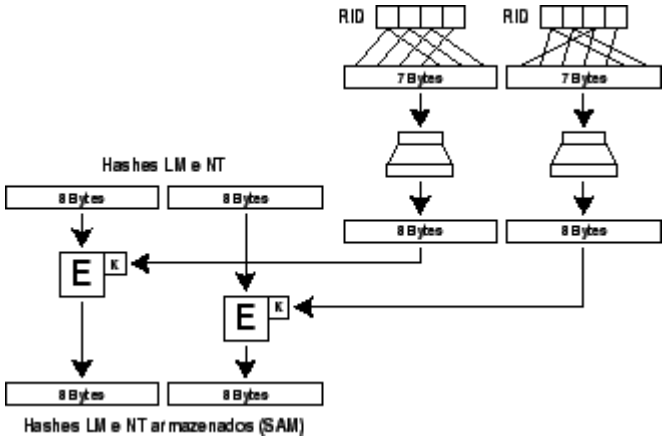
Hash NT

O *hash* NT (que pode ter até 128 caracteres) é construído a partir do *hash* MD4 da senha. Porém o programa de interface com o usuário permite no máximo 14(!).



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 recuperar os hashes NT e LM caso alguém consiga o arquivo SAM. No entanto, a segurança deste método está baseado no segredo do processo, que é um clássico caso de má abordagem do problema de segurança. O algoritmo pode ser quebrado verificando-se o código de máquina do sistema. Atualmente, o algoritmo já é conhecido, o que torna esse processo inútil. 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. O utilitário PWDUMP/PWDUMP2, amplamente disponível na internet, e também utilizado no melhor quebrador de senhas para o Windows NT (L0phtCrack), utiliza o processo da figura abaixo para obter os *hashes* desobscurecidos. Lembrar que a dupla cifragem via DES pela mesma chave corresponde à decifragem.



Deficiências dos *hashes*

Sendo possível conseguir o banco de dados de usuários e senhas, que tipos de ataque podem ser feitos, sabendo-se o processo de ciframento?

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.

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. Mais ainda, 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. A tabela 2 mostra o tempo para realizar um ataque de força bruta em cima do espaço de chaves de um *hash* LM. Numa rápida comparação com o tempo necessário para quebrar o DES com a chave de 56 bits, nota-se que no caso do LM esse ataque é prático 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 caso se considere apenas letras maiúsculas, números e pontuação.

Table: Tempo para vasculhar um espaço de 69^7 chaves

Máquina	Op./Sec.	Tempo
	(Approx.)	
Pentium 90Mhz	260000	331 dias
Pentium 133Mhz	770000	112 dias
Pentium Pro 200Mhz	970000	89 dias
EFF DES-cracker	$92 \cdot 10^9$	1,34 min.

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.

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 já criou um mecanismo fadado ao fracasso. Juntamente com o fato do arquivo SAM poder ser copiado, 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. 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 mas 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.

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 a 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.

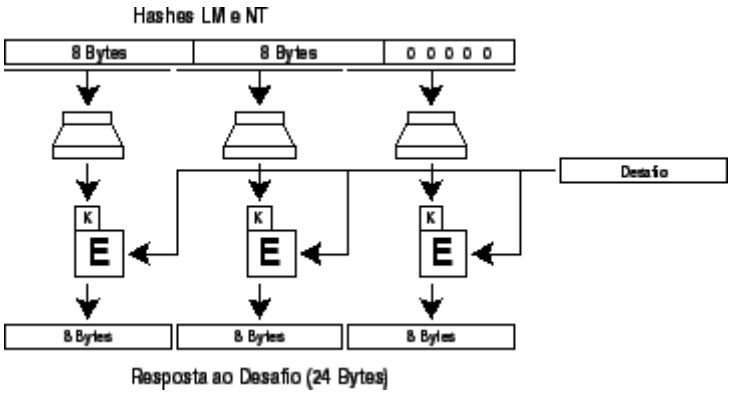
Evolução do Protocolo SMB

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 se autentica respondendo com o *nonce* cifrado com o *hash* LM.

Dialeto NTLM:
Comporta-se como o dialeto LM, exceto que o cliente responde além do *nonce* cifrado com o *hash* LM o *nonce* cifrado com o *hash* NT.



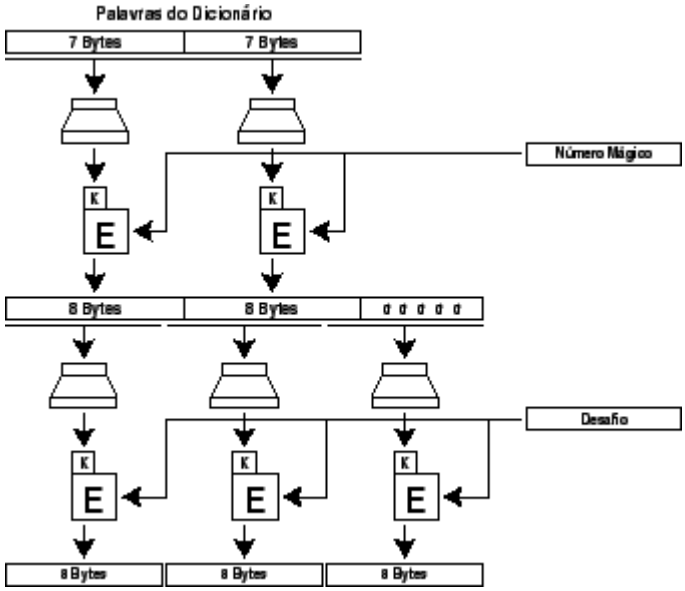
O SMB possui várias deficiências. Primeiro, não há autenticação do servidor. Portanto, o cliente não tem como saber se o servidor é confiável e, mesmo assim, deve responder ao desafio, efetivamente transmitindo informação sobre a senha do usuário. É possível, por exemplo, alterar uma página HTML de modo 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 quem determina o ``nível'' criptográfico a ser utilizado. Esse último problema foi resolvido a partir do Service Pack 3, onde há a opção no cliente e no 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.



A eficácia deste ataque depende de como os pares desafio/resposta foram conseguidos. Caso o invasor esteja escutando o canal de comunicação, o invasor 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: com menos de 8 caracteres na senha, o *hash* resultante H possui a segunda metade de 8 bytes constante. O terceiro terço da resposta R pode ser calculado diretamente: é o resultado da cifragem dos dois últimos bytes desta segunda metade H com os 5 bytes nulos adicionados utilizando-se o desafio como senha. Agora, os seis bytes da direita do segundo terço da resposta R podem ser obtidos dos seis bytes da esquerda do hash H. Fica faltando o primeiro byte do segundo terço de R, que pode ser calculado varrendo-se as 256 possíveis combinações para esse byte. Agora, calcula-se o primeiro terço de R. Sabe-se agora o último byte que compunha a primeira metade H. Aplica-se, então, um ataque de dicionário ou força bruta para este primeiro terço.

Deficiências do dialeto NTLM

A resposta ao desafio inclui os 24 bytes obtidos com o *hash* LM além dos 24 bytes gerados com o *hash* NT. É tão vulnerável quanto o dialeto LM. Os dois dialetos permitem ataques de *replay* e integridade.

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 NT. 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 fácil de quebrar. 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

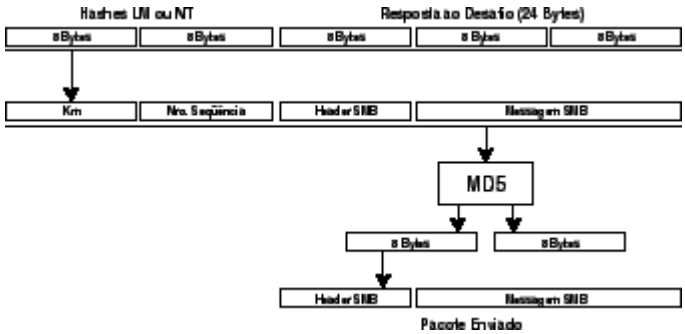
integridade da mensagem, de modo a evitar que terceiros alterem ou repitam o conteúdo das mensagens. Isto foi introduzido no Service Pack 3 com o SMB *signing*, usando o algoritmo MD5. Embora a curto prazo não seja possível substituir o atual esquema de autenticação, a próxima versão do Windows NT contará com o sistema de autenticação Kerberos.

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*[3]]; $S \rightarrow C$ [userinfo, logonscript, UID, SIDs, etc...] (em claro!) Após o cliente dizer ao servidor quais os dialetos que ele suporta, o servidor responde informando qual o dialeto que ele prefere entre os sugeridos e 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. 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 tinha sido alterado por terceiros. Com o SMB Signing, criou-se uma assinatura MD5 baseada:

- na resposta do cliente (ou seja, no identificador de sessão Ds, de modo a associar uma assinatura diferente a cada sessão diferente do mesmo usuário (= resposta do desafio - ver figura abaixo);
- 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. Portanto, uma avaliação mais profunda do SMB Signing permite concluir que pouco se obteve de segurança extra. 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 abaixo é um esquema do funcionamento do SMB *signing*. Os *hashes* LM ou NT (de acordo com o dialeto) são combinados com a resposta ao desafio obtida, formando uma chave Km. A resposta ao desafio é uma função do número de sessão Ds (o desafio), e também uma função do *hash* enviado. No entanto, a resposta passa em claro pela rede, e é necessário incorporar à chave Km 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.



Redes Privadas Virtuais (*Virtual Private Networks*)

O protocolo PPTP (*Point-to-Point Tunneling Protocol*) é usado para assegurar privacidade e segurança em conexões PPP 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 (MMPE *Microsoft Point-to-Point Encryption Protocol*[PZ99]) é fraco e suscetível 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:

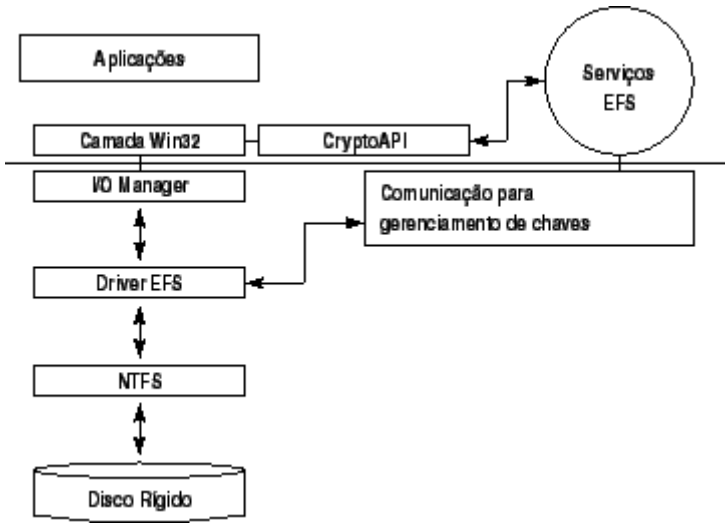
Senha em claro:
a senha é enviada em claro
***Hash* em claro:**
o *hash* da senha é enviado em claro
Desafio/resposta:
usa o protocolo de desafio/resposta da Microsoft.

A última opção é chamada ``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 MMPE apresenta muitos problemas. 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 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 para cifrar o canal byte a byte. Ou seja, não importa a posição em que o trecho procurado esteja, no início ou fim da mensagem, a sua representação cifrada será sempre a mesma, independentemente dos bytes cifrados que apareceram anteriormente. Como a chave é a mesma para cada usuário, bytes de textos claros iguais sempre vão dar bytes de textos cifrados iguais. 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*''.

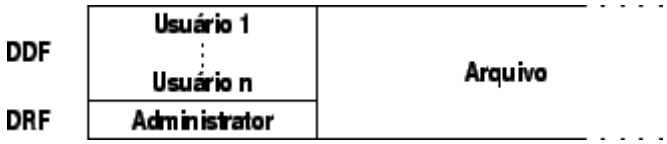
Tendências Futuras

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. 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. 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.



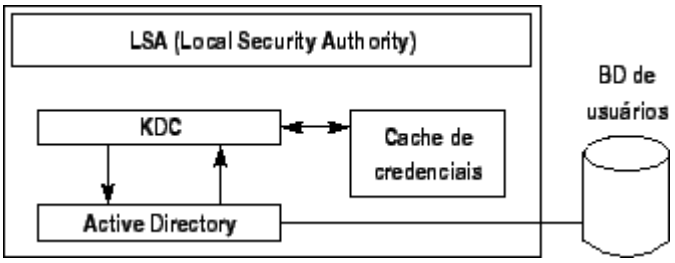
O EFS funciona codificando os arquivos usando um algoritmo de ciframento simétrico, com uma chave FEK *aleatória* associada ao arquivo. A FEK é cifrada usando uma ou mais chaves públicas de ciframento para gerar uma lista de FEKs cifradas (DDF - *Data Decription Field*). A FEK também é cifrada usando uma ou mais chaves de recuperação (DRF - *Data Recovery Field*). O usuário, usando sua chave privada guardada em algum lugar (*smart card* ou algum outro dispositivo seguro) pode obter a chave FEK e portanto desencriptar o arquivo para uso pessoal.



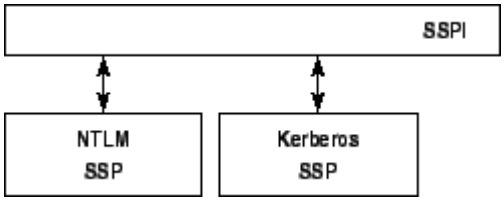
O mecanismo é de concepção robusta, usando chave aleatórias para cifrar, ao invés de derivações de senhas, e utilizando chaves públicas/privadas.

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. Cada PDC e BDC também será um *Key Distribution Center* (KDC) definido no protocolo Kerberos. 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). 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 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 máquina à máquina.

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 de L2TP 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.

Bibliography

Ash98
P.Ashton, *Gaining Domain Admins access on LAN*, NTBugTraq, Jan 1998. http://www.ntbugtraq.com/

Hob97
Hobbit, *CIFS: Common Insecurities Fail Scrutiny,* Avian Research, Jan 1997. http://www.avian.org

Ken97
L.Kenneth, C.Leighton, P.Ashton, D.Stansfield, *NT Domain Authentication*, NT Domain Authentication Protocol, Samba Project, http://www.cb1.com/ lkcl/ntdom/ntdomain.html ftp://ftp.microsoft.com/ developr/drg/CIFS/CIFS-Auth-Spec.txt

L97a L0pht Heavy Industries Inc, *L0phtcrack,* 1997. http://www.L0pht.com/L0phtcrack/

L97b L0pht Heavy Industries Inc, *A L0phtCrack Technical Rant,* Jul 1997.

http://www.l0pht.com/l0phtcrack/rant.html

Mic97a
P.J.Leach, *CIFS Authentication Protocol*, Microsoft, ftp://ftp.microsoft.com/ developr/drg/CIFS/CIFS-Auth.txt

Mic97b
P.J.Leach, *CIFS Authentication Protocols Specification*, Microsoft, ftp://ftp.microsoft.com/ developr/drg/CIFS/CIFS-Auth-Spec.txt

Mic99a
Article ID Q143475, *Windows NT System Key Permits Strong Encryption of the SAM*, Microsoft, http:// support.microsoft.com/support/kb/articles/ q143/4/75.asp

Mic99b
Microsoft Windows 2000 Server, *Windows 2000 Server Authentication - Kerberos Protocol*, Microsoft, http:// www.microsoft.com/ntserver/security/techdetails/ default.asp

Mic99c
Microsoft Windows 2000 Server, *Encrypting File System for Windows 2000*, Microsoft, http:// www.microsoft.com/ntserver/security/techdetails/ default.asp

NIST93
National Institute of Standards and Technology, *Secure Hash Standard,* U.S. Department of Commerce, Maio 1993

NBS77
National Bureau of Standards, NBS FIPS PUB 46, *Data Encryption Standard,* National Bureau of Standards, U.S. Department of Commerce, Janeiro 1977.

PZ99 G.S.Pall and G.Zorn, *Microsoft Point-to-Point Encryption (MPPE) Protocol,* The Internet Engineering Task Force, Internet Draft, IETF, Maio 1999. http://www.ietf.org/internet-drafts/draft-ietf-pppext-mppe-03.txt

Riv91
R.L.Rivest, *The MD4 Message Digest Algorithm,* Advances in Cryptology - CRYPTO'90 Proceedings, Springer-Verlag, 1991, pp.303-311.

Sch98
B.Schneier, Mudge *Cryptanalysis of Microsoft's Point-to-Point Tunneling Protocol (PPTP),* Counterpane Systems, Jun 1998. http://www.counterpane.com/pptp.html

Z99a G.Zorn, *Microsoft PPP CHAP Extensions, Version 2,* The Internet Engineering Task Force, Abril 1999. http://www.ietf.org/internet-drafts/draft-ietf-pppext-mschap-v2-03.txt

Z99b G.Zorn, *Deriving MPPE Keys From MS-CHAP V2 Credentials,* The Internet Engineering Task Force, Novembro 1998. http://www.ietf.org/internet-drafts/draft-ietf-pppext-mschapv2-keys-02.txt

Z99c G.Zorn, *MPPE Key Derivation,* The Internet Engineering Task Force , Fevereiro 1999. http://www.ietf.org/internet-drafts/draft-ietf-pppext-mppe-keys-00.txt

About this document ...

Aspectos Criptográficos no Windows NT

This document was generated using the **LaTeX**2HTML translator Version 98.1p1 release (March 2nd, 1998)

Copyright © 1993, 1994, 1995, 1996, 1997, Nikos Drakos, Computer Based Learning Unit, University of Leeds.

The command line arguments were:

latex2html `-split 0 ntcryptoa.tex`.

The translation was initiated by Marcus Cunha Granado on 1999-06-25

Footnotes

... conhecem[1]
Na realidade, nem o sistema conhece tal senha; ele apenas conhece o valor *hashed* da mesma!

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

...*Hash*[3]
O *hash* da senha de acordo com o dialeto escolhido, cifrado com o DES usando chave Ds (o desafio enviado).

---

next up previous

Marcus Cunha Granado
1999-06-25