CAPÍTULO V

SEGURANÇA DE TERMINAIS E CAIXAS AUTOMÁTICAS

1. Identificação de usuários

O método usual de identificar um usuário perante um sistema operacional consiste em fornecer-lhe um número público de identificação id (em geral o nome do usuário ou o número de uma conta) e uma senha secreta s ("password"). A identidade do usuário é então verificada a cada conexão inicial ("login") ao sistema: a senha fornecida pelo usuário é comparada com a senha correspondente armazenada no arquivo de senhas.

Por outro lado, métodos bancários mais modernos oferecem ao cliente a possibilidade de utilizar terminais para resolver seus problemas. Tipicamente, o usuário pode determinar seu saldo em conta corrente, verificar lançamentos e mesmo depositar ou retirar dinheiro. A identificação do usuário fica então a cargo do terminal: quando uma transação é iniciada, o cliente fornece uma senha secreta s (PIN - "personal idenfication number") juntamente com o número da conta id.

2. Tamanho da senha e busca exaustiva

Na maioria dos sistemas operacionais e de caixas automáticas, a senha usualmente tem apenas 4 a 6 caracteres, a

fim de permitir ao usuário sua memorização. O comprimento curto da senha expõe o sistema, uma vez que busca exaustiva pode ser utilizada, particularmente no caso de sistemas operacionais. Mesmo no caso de senhas grandes, muitas vezes os usuários escolhem nomes de parentes, números de telefones, datas de nascimento, etc., facilitando a busca exaustiva. Num estudo de segurança de senhas executado no UNIX da Bell Laboratories, descobriu-se que 86% das senhas eram "fracas" [MOTH]: das 3289 senhas, coletadas ao longo do tempo,

.15 eram um único caráter ASCII

.72 eram cadeias de dois caracteres ASCII

.464 eram cadeias de três caractéres ASCII

.477 eram cadeias de quatro caracteres

.706 eram cadeias de cinco letras, todas maiúsculas ou todas minúsculas

.605 eram cadeias de seis letras minúsculas

.492 eram nomes próprios ou palavras do dicionário.

Uma maneira de resolver o problema no caso de usuários de sistemas computacionais consiste em aumentar o tamanho do espaço de senhas, permitindo ao usuário qualquer senha com, digamos, 64 bits. Novamente, o problema de memorização inviabiliza esta solução, na prática. Mas a criptografia pode ajudar neste caso, simplesmente fornecendo um método de *compressão de senha*, que nada mais é do que um espalhamento ("hashing") das senhas.

O usuário escolhe senhas longas (16 bits ou mais), de fácil memorização; por exemplo, o usuário pode escolher ao aca

so e independentemente cinco ou mais palavras de um dicionário, ou selecionar uma frase longa. Uma boa função de espalhamento consiste em cifrar a senha com encadeamento e tomar os últimos, digamos, 64 bits do texto cifrado como sendo a senha do usuário para fins internos. A redundância encontrada na senha original (e por isso mesmo, memorizável) é eliminada por esta operação de compressão: espera-se que todas as cadeias de 64 bits tenham aproximadamente a mesma probabilidade de ocorrer.

No caso de terminais bancários não se pode exigir que o cliente tenha a boa vontade de datilografar senhas longas; ademais, dada a não especialização do cliente, senhas longas degradariam provavelmente a qualidade do serviço; finalmente, muitos terminais têm um repertório reduzido de caracteres, o que exige senhas ainda mais longas.

Neste caso, pode-se melhorar a situação exigindo do cliente mais uma informação secreta, não memorizável, armazenada num cartão com fita magnética. A senha de fato seria então uma concatenação da informação secreta do cartão com a senha. Isto exige que o terminal tenha equipamento para a leitura de cartões magnéticos; exige também medidas precautórias no caso de perda do cartão.

Ainda assim, no caso de se permitir consulta automatizada por telefone, o cartão obviamente não pode ser utilizado.

3. *Escuta na Linha*

Um ponto mais interessante para ataque do mau caráter

é na linha de conexão entre terminal e o equipamento central ("host"). Suponha que um usuário faz a conexão inicial com o sistema e fornece a sua senha; se esta senha s (ou uma função dela, como descrito na seção anterior) for enviada, do terminal do usuário para o sistema, às claras, a segurança pode ficar comprometida, pela simples escuta da linha. Pior ainda, um programa pode se fazer passar pelo procedimento de conexão inicial e convencer o usuário a fornecer sua identidade e senha.

Para impedir esses ataques, a única solução é exigir que o terminal tenha inteligência suficiente para pelo menos executar ciframentos e deciframentos. Apresentaremos a seguir um protocolo que não só não expõe a senha do usuário, como também permite ao usuário e ao sistema autenticarem-se mutuamente ("handshaking"). Supõe-se que cada usuário A tenha uma chave secreta CA (armazenada num cartão magnético ou digitada pelo usuário, com compressão); supõe-se também que uma cópia da chave está armazenada no sistema, de maneira segura.

O protocolo é o seguinte (figura 1):

1. O terminal envia ao sistema a identidade iA de A, juntamente com CA(r), onde r é gerado ao acaso pelo terminal.

2. O sistema, de posse de iA, determina CA, decifra CA(r) obtendo r, gera um outro valor aleatório r', envia ao terminal a mensagem CA(r+1,r').

3. O terminal decifra CA(r+1,r') obtendo r+1 (e também r'), e verifica que o resultado r+1 é um mais do que o aleatório r por ele gerado. Em seguida, o usuário envia ao sistema a mensagem CA(r'+sA) onde sA é a senha de A.

4. O sistema decifra CA(r'+sA), obtendo r'+sA. De posse de r', o sistema determina a senha sA dada por A. Finalmente, o sistema verifica, no arquivo de senhas, que de fato sA é a senha de A, completando o procedimento de conexão inicial, e enviando a aprovação ao terminal, na forma da mensagem CA(r'+1).

5. O terminal decifra CA(r'+1), verifica que r'+1 é um mais do que r' e aceita o usuário como se fosse A.

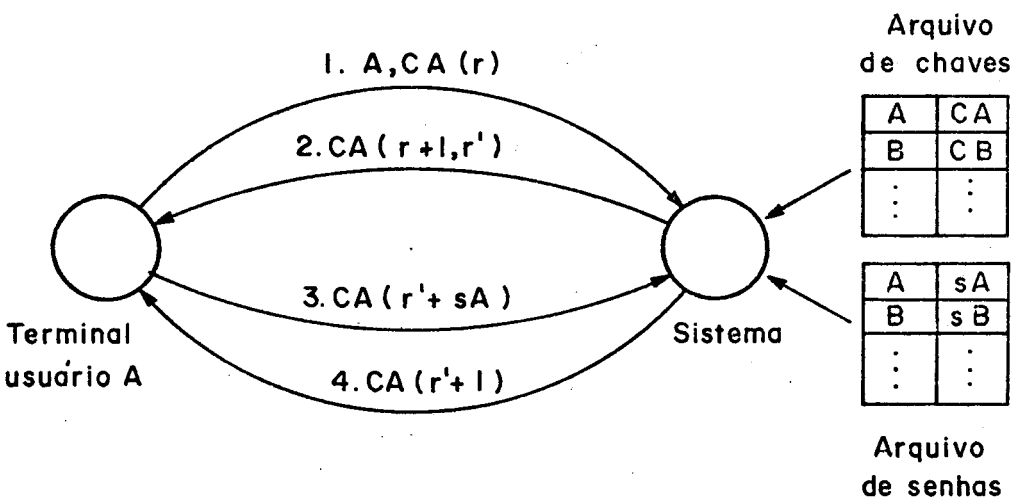

Figura 1. Um protocolo de autenticação de usuário.

A eliminação de r nas mensagens permite ao mau caráter se fazer passar pelo sistema (exercício 1). A eliminação de r' nas mensagens permite ao mau caráter se fazer passar pelo usuário perante o sistema (exercício 2).

A adaptação deste protocolo ao caso de chave pública é elementar e fica a cargo do leitor (exercício 3). Um último comentário sobre este protocolo deve ser feito: se o espião "desmontar" o terminal terá sucesso em sua empreitada; supõe-se que a verificação da mensagem 2 também é feita sem interferência do usuário.

Resta ainda o problema de proteção da senha e da chave do usuário, no sistema central. A proteção da senha será vista a seguir. A proteção de chaves será vista em capítulo posterior.

4. Acesso ao arquivo de senhas

Evidentemente, um oponente mais sofisticado (um funcionário insatisfeito do centro de computação, por exemplo) pode ter acesso ao arquivo de senhas.

Se, por um lado, a criptografia não elimina a necessidade de confiança em certas pessoas em última análise, por outro lado ela permite reduzir o conhecimento privilegiado de informações e o universo de pessoas com esse conhecimento privilegiado. Em outras palavras, ela permite confiar menos informações a menos pessoas. Este é o caso aqui.

Suponhamos que o oponente tenha então acesso ao arquivo de senhas. Nesse caso, podemos proteger o sistema lançando mão de uma função unidirecional f, que pode ser até mesmo pública. Ao invés de armazenar a senha s, armazena-se f(s). Para autenticar uma senha s fornecida pelo usuário, calcula-se f(s) e compara-se o resultado com o correspondente valor armazenado no arquivo. De posse de f(s), o oponente nada pode fazer para determinar s, uma vez que f é unidirecional.

O oponente pode ser ainda mais sofisticado e conseguir alterar o arquivo de senhas cifradas. O oponente escolhe uma senha s', que pode ser a sua própria senha, e altera o arquivo de senhas cifradas, mudando o campo f(sA) correspondente ao usuário A para f(s'). Em seguida, o oponente faz-se passar por A fornecendo ao sistema a identidade de A e a senha s'. Mais tarde, o oponente restaura o valor original f(sA) no campo apropriado.

Para evitar este tipo de ataque, pode-se lançar mão de uma técnica conhecida como *autenticação de dados invariantes com o tempo*, que permite ao sistema determinar se uma informação x foi ou não alterada. No caso, a informação x é f(sA). Para poder implementar este tipo de autenticação, o sistema deve dispor de um *dispositivo criptográfico*, inviolável, que computa duas funções criptográficas, AF e AR, inversas uma da outra: AF(y,AR(y,z))=z; ademais, uma delas, AR no caso, somente é executada quando um funcionário de alto nível aciona uma chave física no dispositivo.

Além da informação x a ser autenticada, o arquivo contém também uma *informação autenticadora* t. Dada a localização

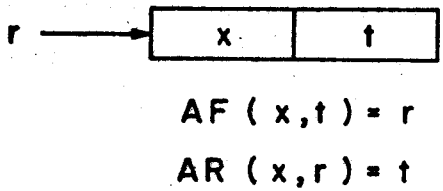

Figura 2. Autenticação de informação invariante.

r de x, a relação entre x, t e r é a seguinte (figura 2):

$$AF(x,t) = r.$$

Para autenticar x armazenada na posição r, o sistema computa AF(x,t), compara com r e aceita x em caso de igualdade. O oponente que deseja alterar x para x' precisa também alterar t para t' tal que AF(x',t') = r.

Para inicializar a localização r com x e t, o funcionário aciona a chave e determina

$$t := AR(x,r).$$

Assim,

$$AF(x,t) = AF(x,AR(x,r)) = r.$$

O espião pode até mesmo ser capaz de inserir um novo registro e nesse caso o esquema acima deixa de ser seguro. Suponha que o espião deseja adicionar uma nova informação x; o espião escolhe um t qualquer, determina

$$r := AF(x,t)$$

e armazena o par x,t na posição r. Este tipo de ataque pode ser impedido pela utilização de uma função unidirecional pública f, eventualmente a mesma que foi utilizada para proteger as senhas. O teste de validade feito pelo sistema muda para

$$AF(x,t) = f(r).$$

De posse de r, x e t o sistema facilmente executa o teste. Para inicializar a localização r com x e t,o funcionário, de posse de x e r, determina, após acionar a chave,

$$t := AR(x,f(r)).$$

O ataque do oponente não tem mais sucesso. Escolhido x e t, AF(x,t) pode ser calculado mas o resultado obtido é f(r), não mais r: a determinação de r é inviável a partir de f(r).

Evidentemente, cada troca de senha exige a reinicialização de x e t, o que pode ser inconveniente. Por outro lado, nem mesmo o funcionário que efetuar as inicializações necessita conhecer a senha s do usuário, pois este pode fornecer-lhe simplesmente f(s).

Finalmente, a implementação de AF e AR pode ser feita facilmente. Por exemplo, o dispositivo criptográfico pode utilizar uma chave secreta constante C armazenada somente dentro dele (uma cópia no cofre do diretor da instituição). As funções AF e AR são então definidas como:

$$AF(x,t) = (C(x))(t)$$
$$AR(x,r) = (C(x))^{-1}(r).$$

Ou seja, o dispositivo, de posse de x, cifra-o sob C, obtendo C(x). No caso de AF, C(x) é então usada como chave de ciframento; no caso de AR, como chave de deciframento.

De fato,

$$AF(x,AR(x,r)) = (C(x))((C(x))^{-1}(r)) = r.$$

5. *Outro protocolo de comunicação terminal-sistema central*

Em muitos casos, o terminal não dispõe de uma leitora de cartões magnéticos e não se pretende pedir ao usuário que forneça a sua chave, além da senha. Ou, por outro lado, o sistema não pretende ter que proteger chaves individuais de usuá-

rios. O sistema prefere armazenar apenas uma chave para cada terminal (e uma chave para cada sistema de computação a que está conectado).

Nesse caso, cada terminal tem um dispositivo criptográfico inviolável onde foi arnazenada previamente uma *chave mestra* secreta de terminal CT, específica daquele terminal, no momento de sua instalação. Quando for iniciada a conexão com um novo usuário A, o sistema gera uma *chave da sessão* CS aleatória, que será utilizada pelo terminal e sistema para comunicação somente durante aquela sessão. Essa chave é enviada cifrada ao terminal, de sorte que o deciframento pode ser efetuado com a chave mestra do terminal, recuperando-se CS. O dispositivo criptográfico a partir daí cifra todas as mensagens originadas do terminal com a chave CS.

O protocolo então é o seguinte (figura 3):

1. O terminal envia a sua identidade T ao sistema.
2. O sistema gera uma chave aleatória da sessão CS e envia ao terminal a mensagem CT(CS).
3. O terminal decifra a mensagem recebida e obtém CS. A chave CS permanece no dispositivo criptográfico, como chave de ciframento e deciframento. O terminal gera outro valor aleatório r, que independe de CS e de CT, e envia ao sistema a mensagem CS(r).
4. O sistema decifra CS(r) e envia a mensagem CS(r+1) ao terminal.
5. O terminal decifra CS(r+1) e verifica que de fato o resultado é um mais do que o aleatório r gerado. Em seguida solicita ao usuário sua identidade iA e sua senha sA e envia ao sistema a mensagem CS(iA,sA)

(ou mesmo CS(iA,f(sA)).

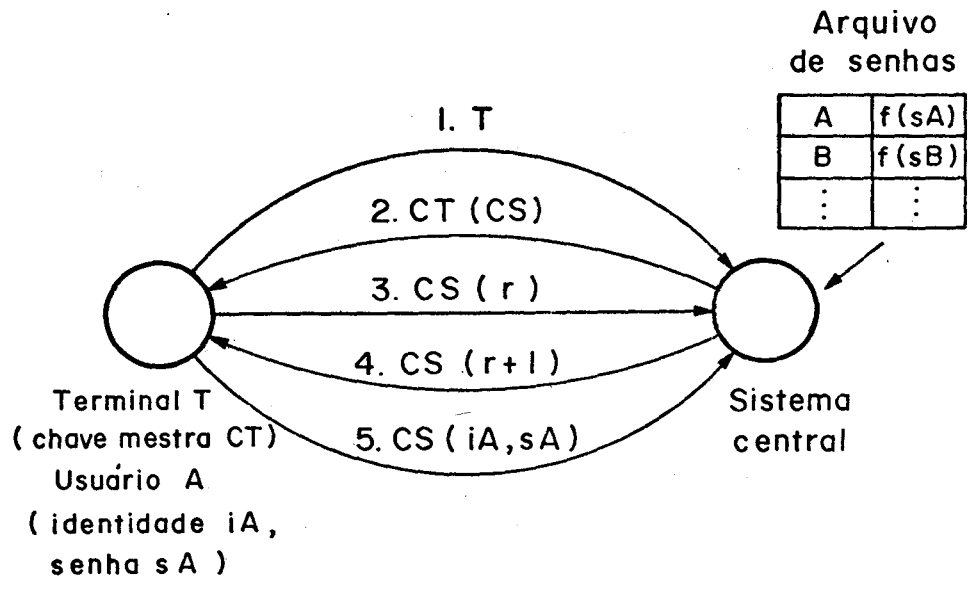

Figura 3. Outro protocolo de autenticação de usuário.

Sem a utilização do valor r, tem-se novamente o perigo do ataque da meia noite. Mais ainda, r não deve ser fornecido pelo usuário, mas gerado pelo terminal. Suponha o contrário. O oponente grava toda a troca de mensagens de uma sessão de um usuário A. Mais tarde, a gravação é enviada ao terminal; com o oponente se fazendo passar pelo sistema. Ao mesmo tempo, o oponente digita r' no terminal e grava o resultado CS(r'). Analogamente, digitando r'+1, grava CS(r'+1). O oponente agora modifica a gravação original, substituindo CS(r) por CS(r') e CS(r+1) por CS(r'+1), conseguindo assim fazer-se passar pelo sistema perante o terminal.

Para evitar que o terminal tenha que gerar r ao acaso