Capítulo 28

Autenticação

O objetivo da autenticação consiste em identificar as diversas entidades de um sistema computacional. Através da autenticação, o usuário interessado em acessar o sistema comprova que ele/a realmente é quem afirma ser. Para tal podem ser usadas várias técnicas, sendo as mais relevantes apresentadas neste capítulo.

28.1 Introdução

Autenticação é o procedimento de verificar a autenticidade de uma entidade no sistema computacional, ou seja, comprovar que as informações associadas a essa entidade são verdadeiros e correspondem às informações do mundo real que elas representam, como a identidade de um usuário, o construtor de um software, a origem dos dados de uma página Web, etc.

Inicialmente, a autenticação visava apenas identificar usuários, para garantir que somente usuários previamente registrados teriam acesso ao sistema. Atualmente, em muitas circunstâncias também é necessário o oposto, ou seja, identificar o sistema para o usuário, sobretudo no caso de acessos por rede. Por exemplo, quando um usuário acessa um serviço bancário via Internet, deseja ter certeza de que o sistema acessado é realmente aquele do banco desejado, e não um sistema falso, construído para roubar seus dados bancários. Outro exemplo ocorre durante a instalação de componentes de software como *drivers*: o sistema operacional deve assegurar-se que o software a ser instalado provém de uma fonte confiável.

28.2 Usuários e grupos

A autenticação geralmente é o primeiro passo no acesso de um usuário a um sistema computacional. Caso a autenticação do usuário tenha sucesso, são criados processos para representá-lo dentro do sistema. Esses processos interagem com o usuário através da interface e executam as ações desejadas por ele dentro do sistema, ou seja, agem em nome do usuário. A presença de um ou mais processos agindo em nome de um usuário dentro do sistema é denominada uma *sessão de usuário* (*user session* ou *working session*). A sessão de usuário inicia imediatamente após a autenticação do usuário (*login* ou *logon*) e termina quando seu último processo é encerrado, na desconexão (*logout* ou *logoff*). Um sistema operacional servidor ou *desktop* típico suporta várias sessões de usuários simultaneamente.

A fim de permitir a implementação das técnicas de controle de acesso e auditoria, cada processo deve ser associado a seu respectivo usuário através de um *identificador de usuário* (UID - *User IDentifier*), geralmente um número inteiro usado como chave em uma tabela de usuários cadastrados (como o arquivo `/etc/passwd` dos sistemas UNIX). O identificador de usuário é usado pelo sistema operacional para definir o proprietário de cada entidade e recurso conhecido: processo, arquivo, área de memória, semáforo, etc. É habitual também classificar os usuários em grupos, como *professores*, *alunos*, *contabilidade*, *engenharia*, etc. Cada grupo é identificado através de um *identificador de grupo* (GID - *Group IDentifier*). A organização dos grupos de usuários pode ser hierárquica ou arbitrária. O conjunto de informações que relaciona um processo ao seu usuário e grupo é geralmente denominado *credenciais do processo*.

Normalmente, somente usuários devidamente autenticados podem ter acesso aos recursos de um sistema. Todavia, alguns recursos podem estar disponíveis abertamente, como é o caso de diretórios de arquivos compartilhados abertamente na rede ou páginas em um servidor Web público. Nestes casos, assume-se a existência de um usuário fictício "convidado" (*guest*, *nobody*, *anonymous* ou outros), ao qual são associados todos os acessos externos não autenticados e para o qual são definidas políticas de segurança específicas.

28.3 Estratégias de autenticação

As técnicas usadas para a autenticação de um usuário podem ser classificadas em três grandes grupos:

SYK – Something You Know ("algo que você sabe"): estas técnicas de autenticação são baseadas em informações conhecidas pelo usuário, como seu nome de *login* e sua senha. São consideradas técnicas de autenticação fracas, pois a informação necessária para a autenticação pode ser facilmente comunicada a outras pessoas, ou mesmo roubada.

SYH – Something You Have ("algo que você tem"): são técnicas que se baseiam na posse de alguma informação mais complexa, como um certificado digital ou uma chave criptográfica, ou algum dispositivo material, como um *smartcard*, um cartão magnético, um código de barras, etc. Embora sejam mais robustas que as técnicas SYK, estas técnicas também têm seus pontos fracos, pois dispositivos materiais, como cartões, também podem ser roubados ou copiados.

SYA – Something You Are ("algo que você é"): se baseiam em características intrinsecamente associadas ao usuário, como seus dados biométricos: impressão digital, padrão da íris, timbre de voz, etc. São técnicas mais complexas de implementar, mas são potencialmente mais robustas que as anteriores.

Muitos sistemas implementam somente a autenticação por login/senha (SYK). Sistemas mais recentes têm suporte a técnicas SYH através de *smartcards* ou a técnicas SYA usando biometria, como os sensores de impressão digital. Alguns serviços de rede, como HTTP e SSH, também podem usar autenticação pelo endereço IP do cliente (SYA) ou através de certificados digitais (SYH).

Sistemas computacionais com fortes requisitos de segurança geralmente implementam mais de uma técnica de autenticação, o que é chamado de **autenticação**

multifator. Por exemplo, um sistema militar pode exigir senha e reconhecimento de íris para o acesso de seus usuários, enquanto um sistema bancário pode exigir uma senha e o cartão emitido pelo banco. Essas técnicas também podem ser usadas de forma gradativa: uma autenticação básica é solicitada para o usuário acessar o sistema e executar serviços simples (como consultar o saldo de uma conta bancária); se ele solicitar ações consideradas críticas (como fazer transferências de dinheiro para outras contas), o sistema pode exigir mais uma autenticação, usando outra técnica.

28.4 Senhas

A grande maioria dos sistemas operacionais de propósito geral implementam a técnica de autenticação SYK baseada em *login/senha*. Na autenticação por senha, o usuário informa ao sistema seu identificador de usuário (nome de *login*) e sua senha, que normalmente é uma sequência de caracteres memorizada por ele. O sistema então compara a senha informada pelo usuário com a senha previamente registrada para ele: se ambas forem iguais, o acesso é consentido.

A autenticação por senha é simples mas muito frágil, pois implica no armazenamento das senhas "em aberto" no sistema, em um arquivo ou base de dados. Caso o arquivo ou base seja exposto devido a algum erro ou descuido, as senhas dos usuários estarão visíveis. Para evitar o risco de exposição indevida das senhas, são usadas funções unidirecionais para armazená-las, como os resumos criptográficos (Seção 27.8).

A autenticação por senhas usando um resumo criptográfico é bem simples: ao registrar a senha s de um novo usuário, o sistema calcula seu resumo ($r = hash(s)$), e o armazena. Mais tarde, quando esse usuário solicitar sua autenticação, ele informará uma senha s'; o sistema então calculará novamente seu resumo $r' = hash(s')$ e irá compará-lo ao resumo previamente armazenado ($r' = r$). Se ambos forem iguais, a senha informada pelo usuário é considerada autêntica e o acesso do usuário ao sistema é permitido. Com essa estratégia, as senhas não precisam ser armazenadas em aberto no sistema, aumentando sua segurança.

Caso um intruso tenha acesso aos resumos das senhas dos usuários, ele não conseguirá calcular de volta as senhas originais (pois o resumo foi calculado por uma função unidirecional), mas pode tentar obter as senhas indiretamente, através do **ataque do dicionário**. Nesse ataque, o invasor usa o algoritmo de resumo para cifrar palavras conhecidas ou combinações delas, comparando os resumo obtidos com aqueles presentes no arquivo de senhas. Caso detecte algum resumo coincidente, terá encontrado a senha correspondente. O ataque do dicionário permite encontrar senhas consideradas "fracas", por serem muito curtas ou baseadas em palavras conhecidas. Por isso, muitos sistemas operacionais definem políticas rígidas para as senhas, impedindo o registro de senhas óbvias ou muito curtas e restringindo o acesso ao repositório dos resumos de senhas.

Uma técnica muito utilizada em sistemas operacionais para dificultar o ataque do dicionário a *hashes* de senhas consiste em "salgar as senhas". O "sal", neste caso, é um número aleatório (*nonce*) concatenado a cada senha antes do cálculo do respectivo *hash*. Ao cadastrar uma senha, um *nonce* aleatório (o sal) é gerado e concatenado à senha e o *hash* dessa concatenação é calculado. Esse *hash* e o sal são então armazenados juntos no sistema, para uso no processo de autenticação. A verificar a senha informada por um usuário, o sal armazenado é concatenado à senha a ser verificada, o *hash* dessa concatenação é calculado e o resultado é comparado ao *hash* previamente armazenado, para autenticar o usuário. A Figura 28.1 ilustra esses procedimentos.

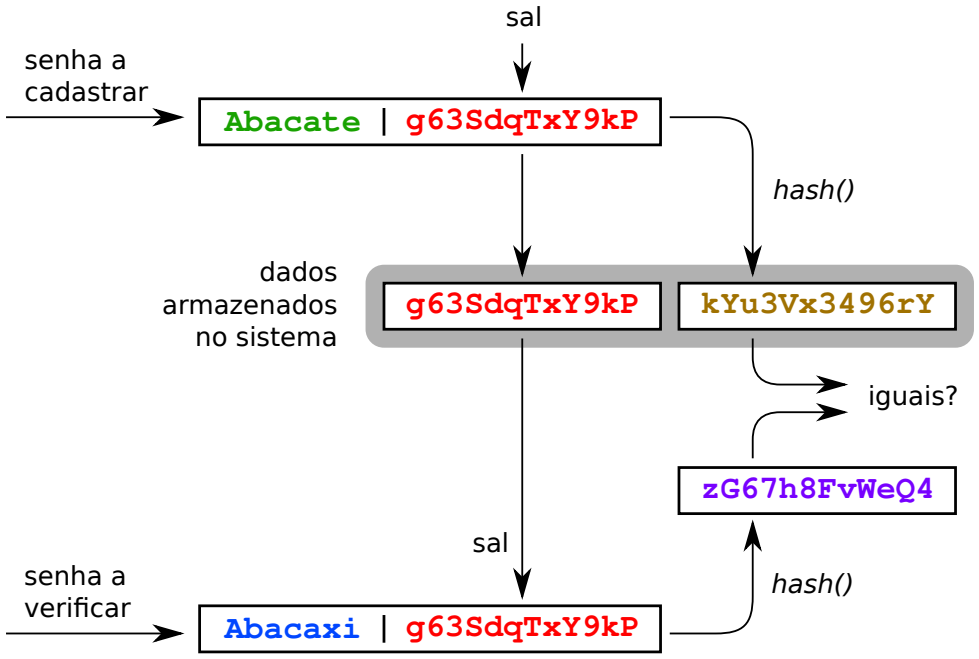

Figura 28.1: Uso de sal na proteção de senhas.

O sal protege os *hashes* das senhas por tornar impraticável o cálculo prévio de tabelas de hashes para o ataque do dicionário. Ao concatenar um sal aleatório com 64 bits de comprimento (8 bytes) a uma senha, essa combinação poderá gerar 2^{64} *hashes* distintos para a mesma senha, o que torna inviável computar e armazenar previamente todos os *hashes* possíveis para cada palavra do dicionário.

28.5 Senhas descartáveis

Um problema importante relacionado à autenticação por senhas reside no risco de roubo da senhas. Por ser uma informação estática, caso uma senha seja roubada, o malfeitor poderá usá-la enquanto o roubo não for percebido e a senha substituída. Para evitar esse problema, são propostas técnicas de senhas descartáveis (OTP - *One-Time Passwords*). Como o nome diz, uma senha descartável só pode ser usada uma única vez, perdendo sua validade após esse uso. O usuário deve então ter em mãos uma lista de senhas predefinidas, ou uma forma de gerá-las quando necessário. Há várias formas de se produzir e usar senhas descartáveis, entre elas:

- Armazenar uma lista sequencial de senhas (ou seus resumos) no sistema e fornecer essa lista ao usuário, em papel ou outro suporte. Quando uma senha for usada com sucesso, o usuário e o sistema a eliminam de suas respectivas listas. A lista de senhas pode ser entregue ao usuário impressa, ou fornecida por outro meio, como mensagens SMS. A tabela a seguir ilustra um exemplo dessas listas de senhas, ainda usadas por alguns bancos:

```
001 342232   002 038234   003 887123   004 545698   005 323241
006 587812   007 232221   008 772633   009 123812   010 661511
011 223287   012 870910   013 865324   014 986323   015 876876
...
```

- Uma variante da lista de senhas é conhecida como *algoritmo OTP de Lamport* [Menezes et al., 1996]. Ele consiste em criar uma sequência de senhas $s_0, s_1, s_2, \cdots, s_{n-1}, s_n$ com s_0 aleatório e $s_i = hash(s_{i-1})\ \forall i > 0$, sendo $hash(x)$ uma função de resumo criptográfico conhecida:

$$\xrightarrow{random} s_0 \xrightarrow{hash} s_1 \xrightarrow{hash} s_2 \xrightarrow{hash} \cdots \xrightarrow{hash} s_{n-1} \xrightarrow{hash} s_n$$

  O valor de s_n é informado ao servidor previamente. Ao acessar o servidor, o cliente informa o valor de s_{n-1}. O servidor pode então comparar $hash(s_{n-1})$ com o valor de s_n previamente informado: se forem iguais, o cliente está autenticado e ambos podem descartar s_n. Para validar a próxima autenticação será usado s_{n-1} e assim sucessivamente. Um intruso que conseguir capturar uma senha s_i não poderá usá-la mais tarde, pois não conseguirá calcular s_{i-1} (a função $hash(x)$ não é inversível).

- Gerar senhas temporárias sob demanda, através de um dispositivo ou software externo usado pelo cliente; as senhas temporárias podem ser geradas por um algoritmo de resumo que combine uma senha predefinida com a data/horário corrente. Dessa forma, cliente e servidor podem calcular a senha temporária de forma independente. Como o tempo é uma informação importante nesta técnica, o dispositivo ou software gerador de senhas do cliente deve estar sincronizado com o relógio do servidor. Dispositivos OTP como o mostrado na Figura 28.2 são frequentemente usados em sistemas de *Internet Banking*.

Figura 28.2: Gerador de senhas descartáveis (fotografia de Mazh3101@Wikipedia).

28.6 Técnicas biométricas

A biometria (*biometrics*) consiste em usar características físicas ou comportamentais de um indivíduo, como suas impressões digitais ou seu timbre de voz, para identificá-lo unicamente perante o sistema. Diversas características podem ser usadas para a autenticação biométrica; no entanto, elas devem obedecer a um conjunto de princípios básicos [Jain et al., 2004]:

Universalidade: a característica biométrica deve estar presente em todos os indivíduos que possam vir a ser autenticados;

Singularidade: (ou unicidade) dois indivíduos quaisquer devem apresentar valores distintos para a característica em questão;

Permanência: a característica não deve mudar ao longo do tempo, ou ao menos não deve mudar de forma abrupta;

Mensurabilidade: a característica em questão deve ser facilmente mensurável em termos quantitativos.

As características biométricas usadas em autenticação podem ser *físicas* ou *comportamentais*. Como características físicas são consideradas, por exemplo, o DNA, a geometria das mãos, do rosto ou das orelhas, impressões digitais, o padrão da íris (padrões na parte colorida do olho) ou da retina (padrões de vasos sanguíneos no fundo do olho). Como características comportamentais são consideradas a assinatura, o padrão de voz e a dinâmica de digitação (intervalos de tempo entre teclas digitadas), por exemplo.

Os sistemas mais populares de autenticação biométrica atualmente são os baseados em impressões digitais e no padrão de íris. Esses sistemas são considerados confiáveis, por apresentarem taxas de erro relativamente baixas, custo de implantação/operação baixo e facilidade de coleta dos dados biométricos. A Figura 28.3 apresenta alguns exemplos de características biométricas empregadas nos sistemas atuais.

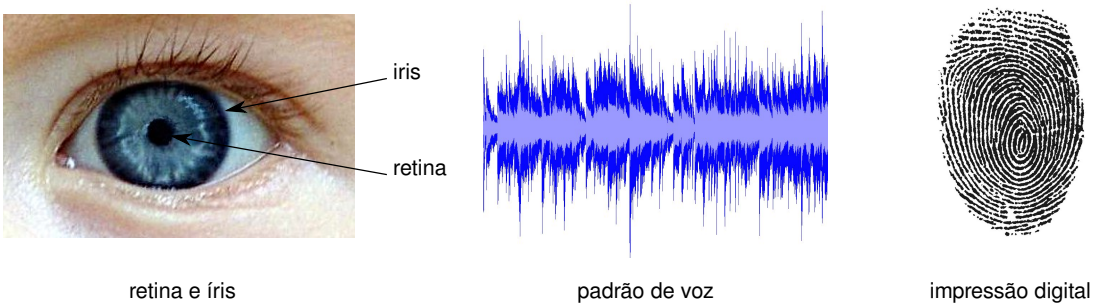

Figura 28.3: Exemplo de características biométricas.

Um sistema biométrico típico é composto de um *sensor*, responsável por capturar dados biométricos de uma pessoa; um *extrator de características*, que processa os dados do sensor para extrair suas características mais relevantes; um *comparador*, cuja função é comparar as características extraídas do indivíduo sob análise com dados previamente armazenados, e um *banco de dados* contendo as características biométricas dos usuários registrados no sistema [Jain et al., 2004].

O sistema biométrico pode funcionar de três modos: no modo de *coleta*, os dados dbiométricos dos usuários são coletados, processados e cadastrados no sistema, junto com a identificação do usuário. No modo de *autenticação*, ele verifica se as características biométricas de um indivíduo (previamente identificado por algum outro método, como login/senha, cartão, etc.) correspondem às suas características biométricas previamente armazenadas. Desta forma, a biometria funciona como uma autenticação complementar. No modo de *identificação*, o sistema biométrico visa identificar o indivíduo a quem correspondem as características biométricas coletadas pelo sensor, dentre todos aqueles presentes no banco de dados. A Figura 28.4 mostra os principais elementos de um sistema biométrico típico.

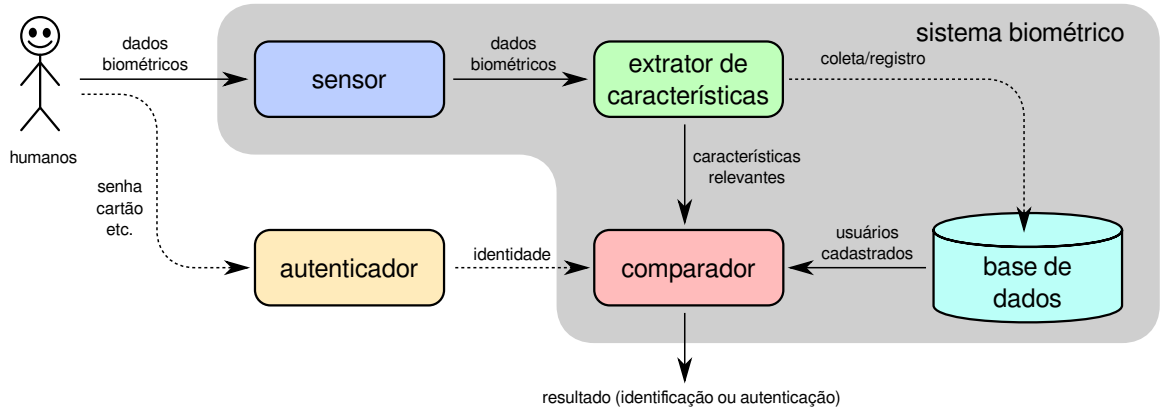

Figura 28.4: Um sistema biométrico típico.

28.7 Desafio/resposta

Em algumas situações o uso de senhas é indesejável, pois sua exposição indevida pode comprometer a segurança do sistema. Um exemplo disso são os serviços via rede: caso o tráfego de rede possa ser capturado por um intruso, este terá acesso às senhas transmitidas entre o cliente e o servidor. Uma técnica interessante para resolver esse problema são os protocolos de *desafio/resposta*.

A técnica de desafio/resposta se baseia sobre um segredo s previamente definido entre o cliente e o servidor (ou o usuário e o sistema), que pode ser uma senha ou uma chave criptográfica, e um algoritmo de cifragem ou resumo $hash(x)$, também previamente definido. No início da autenticação, o servidor escolhe um valor aleatório d e o envia ao cliente, como um *desafio*. O cliente recebe esse desafio, o concatena com seu segredo s, calcula o resumo da concatenação e a devolve ao servidor, como *resposta* ($r = hash(s \parallel d)$). O servidor executa a mesma operação de seu lado, usando o valor do segredo armazenado localmente (s') e compara o resultado obtido $r' = hash(s' \parallel d)$ com a resposta r fornecida pelo cliente. Se ambos os resultados forem iguais, os segredos são iguais ($r = r' \Rightarrow s = s'$) e o cliente é considerado autêntico. A Figura 28.5 apresenta os passos desse algoritmo.

A estratégia de desafio/resposta é robusta, porque o segredo s nunca é exposto fora do cliente nem do servidor; além disso, como o desafio d é aleatório e a resposta é cifrada, intrusos que eventualmente conseguirem capturar d ou r não poderão utilizá-los para se autenticar nem para descobrir s. Variantes dessa técnica são usadas em vários protocolos de rede, como o CHAP (em redes sem fio) e o SSH (para terminais remotos).

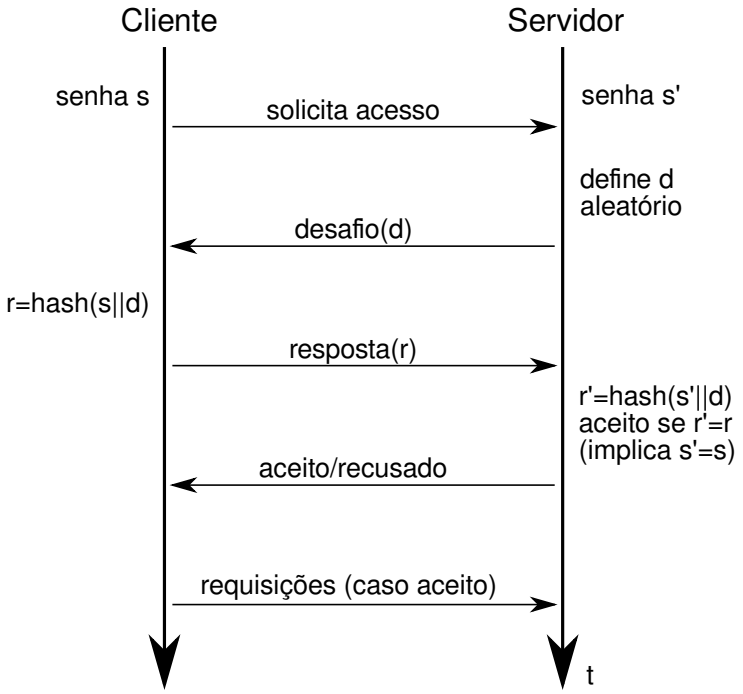

Figura 28.5: Autenticação por desafio/resposta.

28.8 Certificados de autenticação

Uma forma cada vez mais frequente de autenticação envolve o uso de *certificados digitais*. Conforme apresentado na Seção 27.10, um certificado digital é um documento assinado digitalmente, através de técnicas de criptografia assimétrica e resumo criptográfico. Os padrões de certificados PGP e X.509 definem certificados de autenticação (ou de identidade), cujo objetivo é identificar entidades através de suas chaves públicas. Um certificado de autenticação conforme o padrão X.509 contém as seguintes informações [Mollin, 2000]:

- Número de versão do padrão X.509 usado no certificado;
- Chave pública do proprietário do certificado e indicação do algoritmo de criptografia ao qual ela está associada e eventuais parâmetros;
- Número serial único, definido pelo emissor do certificado (quem o assinou);
- Identificação detalhada do proprietário do certificado, definida de acordo com normas do padrão X.509;
- Período de validade do certificado (datas de início e final de validade);
- Identificação da Autoridade Certificadora que emitiu/assinou o certificado;
- Assinatura digital do certificado e indicação do algoritmo usado na assinatura e eventuais parâmetros;

Os certificados digitais são o principal mecanismo usado para verificar a autenticidade de serviços acessíveis através da Internet, como bancos e comércio

eletrônico. Nesse caso, eles são usados para autenticar os sistemas para os usuários. No entanto, é cada vez mais frequente o uso de certificados para autenticar os próprios usuários. Nesse caso, um *smartcard* ou um dispositivo USB contendo o certificado é conectado ao sistema para permitir a autenticação do usuário.

28.9 Infraestruturas de autenticação

A autenticação é um procedimento necessário em vários serviços de um sistema computacional, que vão de simples sessões de terminal em modo texto a serviços de rede, como e-mail, bancos de dados e terminais gráficos remotos. Historicamente, cada forma de acesso ao sistema possuía seus próprios mecanismos de autenticação, com suas próprias regras e informações. Essa situação dificultava a criação de novos serviços, pois estes deveriam também definir seus próprios métodos de autenticação. Além disso, a existência de vários mecanismos de autenticação desconexos prejudicava a experiência do usuário e dificultava a gerência do sistema.

Para resolver esse problema, foram propostas infraestruturas de autenticação (*authentication frameworks*) que unificam as técnicas de autenticação, oferecem uma interface de programação homogênea e usam as mesmas informações (pares *login*/*senha*, dados biométricos, certificados, etc.). Assim, as informações de autenticação são coerentes entre os diversos serviços, novas técnicas de autenticação podem ser automaticamente usadas por todos os serviços e, sobretudo, a criação de novos serviços é simplificada.

A visão genérica de uma infraestrutura de autenticação local é apresentada na Figura 28.6. Nela, os vários mecanismos disponíveis de autenticação são oferecidos às aplicações através de uma interface de programação (API) padronizada. As principais infraestruturas de autenticação em uso nos sistemas operacionais atuais são:

PAM (*Pluggable Authentication Modules*): proposto inicialmente para o sistema Solaris, foi depois adotado em vários outros sistemas UNIX, como FreeBSD, NetBSD, MacOS X e Linux;

XSSO (*X/Open Single Sign-On*): é uma tentativa de extensão e padronização do sistema PAM, ainda pouco utilizada;

BSD Auth: usada no sistema operacional OpenBSD; cada método de autenticação é implementado como um processo separado, respeitando o princípio do privilégio mínimo (vide Seção 29.2);

NSS (*Name Services Switch*): infraestrutura usada em sistemas UNIX para definir as bases de dados a usar para vários serviços do sistema operacional, inclusive a autenticação;

GSSAPI (*Generic Security Services* API): padrão de API para acesso a serviços de segurança, como autenticação, confidencialidade e integridade de dados;

SSPI (*Security Support Provider Interface*): variante proprietária da GSSAPI, específica para plataformas Windows.

Além das infraestruturas de autenticação local, existem também padrões e protocolos para implementar ações de autenticação em redes de computadores e

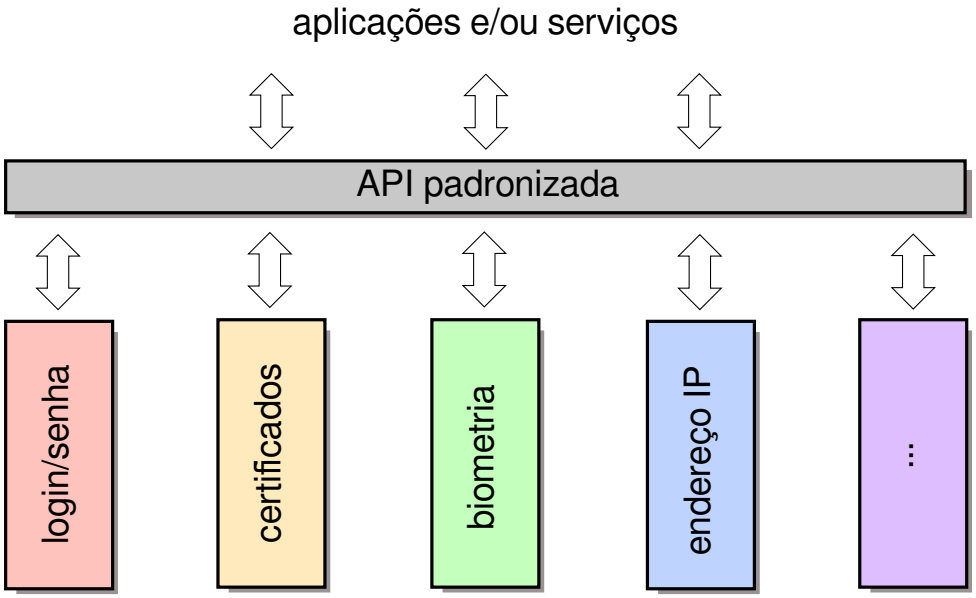

Figura 28.6: Estrutura genérica de uma infraestrutura de autenticação.

sistemas distribuídos, como a Internet. Procolos de autenticação em redes locais incluem o Kerberos (Seção 28.10), Windows NTML, CHAP, Radius/Diameter e LDAP, entre outros. Na Internet, os protocolos de autenticação OpenID e Shibboleth são muito utilizados.

28.10 Kerberos

O sistema de autenticação *Kerberos* foi proposto pelo MIT nos anos 80 [Neuman and Ts'o, 1994]. Hoje, esse sistema é utilizado para centralizar a autenticação de rede em vários sistemas operacionais, como Windows, Solaris, MacOS X e Linux. O sistema Kerberos se baseia na noção de *tickets*, que são obtidos pelos clientes junto a um serviço de autenticação e podem ser usados para acessar os demais serviços da rede. Os tickets são cifrados usando criptografia simétrica DES e têm validade limitada, para aumentar sua segurança.

Os principais componentes de um sistema Kerberos são o Serviço de Autenticação (AS - *Authentication Service*), o Serviço de Concessão de Tickets (TGS - *Ticket Granting Service*), a base de chaves, os clientes e os serviços de rede que os clientes podem acessar. Juntos, o AS e o TGS constituem o *Centro de Distribuição de Chaves* (KDC - *Key Distribution Center*). O funcionamento básico do sistema Kerberos, ilustrado na Figura 28.7, é relativamente simples: o cliente se autentica junto ao AS (passo 1) e obtém um ticket de acesso ao serviço de tickets TGS (passo 2). A seguir, solicita ao TGS um ticket de acesso ao servidor desejado (passos 3 e 4). Com esse novo ticket, ele pode se autenticar junto ao servidor desejado e solicitar serviços (passos 5 e 6).

No Kerberos, cada cliente c possui uma chave secreta k_c registrada no servidor de autenticação AS. Da mesma forma, cada servidor s também tem sua chave k_s registrada no AS. As chaves são simétricas, usando cifragem DES, e somente são conhecidas por seus respectivos proprietários e pelo AS. Os seguintes passos detalham o funcionamento do Kerberos versão 5 [Neuman and Ts'o, 1994]:

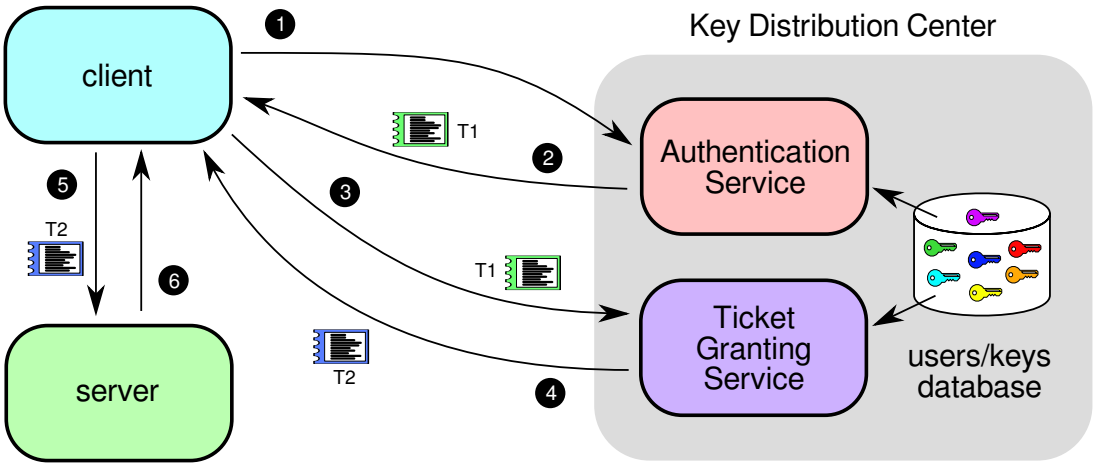

Figura 28.7: Visão geral do serviço Kerberos.

1. Uma máquina cliente c desejando acessar um determinado servidor s envia uma solicitação de autenticação ao serviço de autenticação (AS); essa mensagem m_1 contém sua identidade (c), a identidade do serviço desejado (tgs), um prazo de validade solicitado (ts) e um número aleatório (n_1) que será usado para verificar se a resposta do AS corresponde ao pedido efetuado:

$$m_1 = [c \;\; tgs \;\; ts \;\; n_1]$$

2. A resposta do AS (mensagem m_2) contém duas partes: a primeira parte contém a chave de sessão a ser usada na comunicação com o TGS (k_{c-tgs}) e o número aleatório n_1, ambos cifrados com a chave do cliente k_c registrada no AS; a segunda parte é um ticket cifrado com a chave do TGS (k_{tgs}), contendo a identidade do cliente (c), o prazo de validade do ticket concedido pelo AS (tv) e uma chave de sessão k_{c-tgs}, a ser usada na interação com o TGS:

$$m_2 = [\{k_{c-tgs} \;\; n_1\}_{k_c} \;\; T_{c-tgs}] \quad \text{onde } T_{c-tgs} = \{c \;\; tv \;\; k_{c-tgs}\}_{k_{tgs}}$$

   O ticket T_{c-tgs} fornecido pelo AS para permitir o acesso ao TGS é chamado TGT (*Ticket Granting Ticket*), e possui um prazo de validade limitado (geralmente de algumas horas). Ao receber m_2, o cliente tem acesso à chave de sessão k_{c-tgs} e ao ticket TGT. Todavia, esse ticket é cifrado com a chave k_{tgs} e portanto somente o TGS poderá abrí-lo.

3. A seguir, o cliente envia uma solicitação ao TGS (mensagem m_3) para obter um ticket de acesso ao servidor desejado s. Essa solicitação contém a identidade do cliente (c) e a data atual (t), ambos cifrados com a chave de sessão k_{c-tgs}, o ticket TGT recebido em m_2, a identidade do servidor s e um número aleatório n_2:

$$m_3 = [\{c \;\; t\}_{k_{c-tgs}} \;\; T_{c-tgs} \;\; s \;\; n_2]$$

4. Após verificar a validade do ticket TGT, o TGS devolve ao cliente uma mensagem m_4 contendo a chave de sessão k_{c-s} a ser usada no acesso ao servidor s e o número

aleatório n_2 informado em m_3, ambos cifrados com a chave de sessão k_{c-tgs}, e um ticket T_{c-s} cifrado, que deve ser apresentado ao servidor s:

$$m_4 = [\{k_{c-s} \;\; n\}_{k_{c-tgs}} \;\; T_{c-s}] \quad \text{onde } T_{c-s} = \{c \;\; tv \;\; k_{c-s}\}_{k_s}$$

5. O cliente usa a chave de sessão k_{c-s} e o ticket T_{c-s} para se autenticar junto ao servidor s através da mensagem m_5. Essa mensagem contém a identidade do cliente (c) e a data atual (t), ambos cifrados com a chave de sessão k_{c-s}, o ticket T_{c-s} recebido em m_4 e o pedido de serviço ao servidor (*request*), que é dependente da aplicação:

$$m_5 = [\{c \;\; t\}_{k_{c-s}} \;\; T_{c-s} \;\; request]$$

6. Ao receber m_5, o servidor s decifra o ticket T_{c-s} para obter a chave de sessão k_{c-s} e a usa para decifrar a primeira parte da mensagem e confirmar a identidade do cliente. Feito isso, o servidor pode atender a solicitação e responder ao cliente, cifrando sua resposta com a chave de sessão k_{c-s}:

$$m_6 = [\{reply\}_{k_{c-s}}]$$

Enquanto o ticket de serviço T_{c-s} for válido, o cliente pode enviar solicitações ao servidor sem a necessidade de se reautenticar. Da mesma forma, enquanto o ticket T_{c-tgs} for válido, o cliente pode solicitar tickets de acesso a outros servidores sem precisar se reautenticar. Pode-se observar que em nenhum momento as chaves de sessão k_{c-tgs} e k_{c-s} circularam em aberto através da rede. Além disso, a presença de prazos de validade para as chaves permite minimizar os riscos de uma eventual captura da chave. Informações mais detalhadas sobre o funcionamento do protocolo Kerberos 5 podem ser encontradas em [Neuman et al., 2005].

Exercícios

1. Sobre as afirmações a seguir, relativas às técnicas de autenticação, indique quais são **incorretas**, justificando sua resposta:

   (a) Nas estratégias de autenticação SYK, o sistema autentica o usuário com base em informações fornecidas pelo mesmo.

   (b) Nas estratégias de autenticação SYH, o sistema usa dados coletados do usuário para fazer sua autenticação.

   (c) Nas estratégias de autenticação SYA, o usuário é autenticado com base em suas características físicas.

   (d) Para estar devidamente protegidas, as senhas armazenadas no sistema devem ser cifradas com criptografia simétrica.

   (e) A autenticação multi-fator consiste em autenticar o usuário usando duas senhas simultaneamente.

(f) A autenticação por técnicas biométricas deve usar características físicas *universais*, *singulares*, *permanentes* e *mensuráveis* dos usuários.

(g) Os *tokens* de segurança usados no acesso a serviços bancários pela Internet implementam um esquema de senhas baseado em desafio-resposta.

(h) PAM e SSPI são infraestruturas de autenticação modulares usadas em sistemas operacionais de mercado.

2. Qual a função do "sal" usado em sistemas de autenticação por senhas? Explique como o "sal" é usado; sua explicação deve conter um diagrama.

Referências

A. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1), Apr. 2004.

A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

R. A. Mollin. *An Introduction to Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 2000. ISBN 1584881275.

B. C. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications Magazine*, 32(9):33–38, September 1994.

C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard), July 2005. URL `http://www.ietf.org/rfc/rfc4120.txt`. Updated by RFCs 4537, 5021.