2
Tecnologias Relacionadas

2.1
Kerberos

Kerberos é um serviço de autenticação que foi desenvolvido com base no trabalho de Needham e Schroeder, fazendo uso da criptografia com chaves e de um centro de distribuição de chaves (KDC – *Key Distribution Center*). Além da autenticação ele também dá suporte à integridade e sigilo da comunicação entre o cliente e os serviços [27, 18].

O Kerberos foi criado para trabalhar em um ambiente onde o usuário se identifica através de um nome e uma senha, a qual é utilizada para derivar a chave mestra. Os serviços também possuem chaves mestra, que ajudam a evitar o problema da personificação indevida. O KDC armazena uma cópia de todas as chaves mestra, e as utiliza no processo de autenticação.

O protocolo de autenticação dá suporte à identificação de usuários entre diferentes domínios, contudo, o Kerberos é comumente utilizado para segurança de sistemas contidos dentro de um único domínio.

As versões atualmente usadas são a 4 e 5. Apesar da versão 5 ter sido lançada para corrigir e aperfeiçoar alguns detalhes técnicos da anterior, não houve uma migração geral para a nova versão, o que acabou gerando uma concorrência entre elas. O funcionamento geral de ambas versões é apresentado a seguir.

2.1.1
Kerberos 4

O protocolo do Kerberos emprega bilhetes criptografados como forma de garantir a autenticidade do cliente. Para que o cliente obtenha esse bilhete, ele deve interagir com dois componentes do Kerberos localizados no KDC, o *Authentication Service* (AS) e o *Ticket-Granting Service* (TGS) – figura 2.1. O AS é incumbido apenas de realizar a autenticação do usuário,

enquanto o TGS emite os bilhetes que possibilitam ao cliente acessar os serviços da rede.

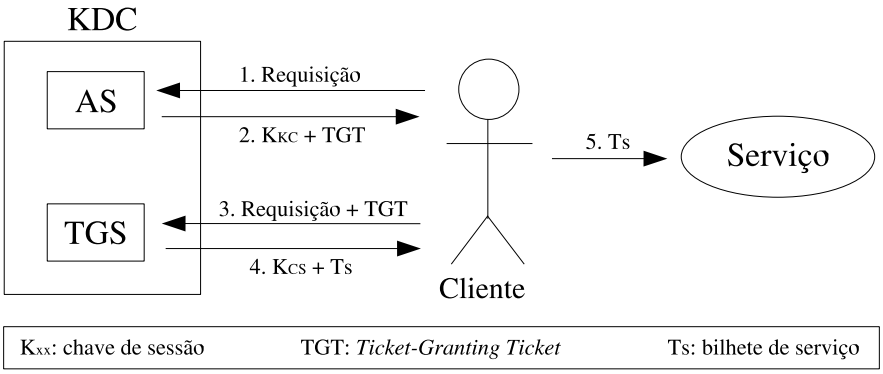

Figura 2.1: Autenticação via Kerberos.

O processo de autenticação se inicia com o cliente entrando em contato com o AS e solicitando uma chave de sessão K_{KC}, que será usada para criptografar os dados nas próximas interações com o KDC. O AS gera essa chave e a criptografa utilizando a chave mestra do usuário, a qual é derivada da senha pessoal. Assim, se a senha utilizada na autenticação estiver correta, o cliente será capaz de extrair a K_{KC}. Além da chave de sessão também é enviado um *ticket-granting ticket* (TGT) que é utilizado para solicitar, junto ao TGS, bilhetes de acesso aos serviços.

Quando o cliente deseja acessar um serviço, ele apresenta ao TGS o TGT junto com o pedido de acesso. O TGS verifica o TGT e, caso este seja válido, retorna uma chave de sessão K_{CS} e um bilhete, que contém K_{CS} criptografada com a chave mestra do serviço. O cliente apresenta o bilhete ao serviço, que extrai a K_{CS} e a usa na comunicação. Com isso, é possível ao serviço e ao cliente efetuarem uma autenticação mútua segura.

O Kerberos permite a divisão da rede em domínios, cada um sob a responsabilidade de um KDC. Se o cliente de um domínio A deseja acessar um serviço em um domínio B, ele necessita entrar em contato com o TGS do domínio B (TGS_B) a fim de retirar um bilhete de acesso ao serviço. No entanto, o cliente não pode usar o TGT de seu domínio na solicitação; ele deve obter com o TGS do seu domínio (TGS_A) um bilhete que será apresentado ao TGS_B no momento do pedido de acesso ao serviço. De posse desse bilhete, o cliente entra em contato com o TGS_B e requisita um bilhete para o serviço desejado. O novo bilhete recebido como reposta é, então, usado no acesso ao serviço. Porém, deve-se destacar que o Kerberos (versões 4 e 5) não provê nenhum mecanismo de controle de permissões, sendo os serviços da rede responsáveis pelo controle de acesso às informações [3].

De um modo geral, há um compartilhamento de chaves para cada par de domínios que se comunicam, ou seja, para que a troca de bilhetes entre os domínios A e B seja possível, é necessário que o TGS_A e TGS_B compartilhem uma chave e que um esteja cadastrado no KDC do outro.

2.1.2 Kerberos 5

A versão 5 do protocolo Kerberos segue os princípios básicos da versão 4, sendo que as principais mudanças foram alterações na codificação das mensagens e extensões em funcionalidades que permitem uma maior flexibilização no uso do protocolo. Por exemplo, as mensagens são definidas usando o padrão ASN.1 e *Basic Encoding Rule* (BER), tornando fácil a especificação e identificação de campos opcionais ou de tamanhos variados [18].

Uma das funcionalidades introduzidas nesta versão foi a delegação de direitos que é feita através da transferência do TGT ou do bilhete. Caso o cliente A forneça um TGT com seu identificador para um cliente B, este terá o direito de retirar junto ao KDC bilhetes de acesso para qualquer serviço. Contudo, todos esses bilhetes possuirão a identificação de A.

Esta abordagem de repassar o TGT para outro cliente só é justificada quando não se sabe ao certo quais recursos precisam ser acessados. No entanto, se o conjunto de serviços for previamente conhecido, A pode fornecer os bilhetes de acesso ao invés do TGT, assim, B poderá apenas usar os serviços aos quais os bilhetes se referem.

O mecanismo de autenticação entre domínios também sofreu algumas mudanças. Primeiro, os domínios foram organizados em forma hierárquica, semelhante ao que acontece na Internet, para reduzir o número de chaves compartilhadas – um nó da hierarquia mantém uma relação de confiança com os seus filhos e seu pai. E, para a autenticação entre os domínios, permitiu-se o encadeamento de requisições.

Para ilustrar, suponha três domínios, A, e seus filhos, B e C. Enquanto que na versão 4 a autenticação entre domínios só era possível se um KDC estivesse cadastrado no outro, na versão 5, um cliente no domínio B que deseja acessar um serviço em C (sem que exista uma ligação de confiança entre os dois) solicita ao domínio A que repasse o pedido de acesso ao KDC de C. Devido à confiança em A, a requisição é aceita por C e o cliente pode acessar o serviço.

Nesse processo de encadeamento, a solicitação pode atravessar diversos domínios até chegar ao destino. O bilhete usado nas solicitações carrega o nome dos domínios por onde ele caminhou, assim, tanto os KDCs quanto os serviços podem verificar a legitimidade do bilhete de acordo com o caminho que ele percorreu.

2.2 Shibboleth

Shibboleth é um projeto desenvolvido pelo grupo de trabalho MACE (*Middleware Architecture Committee for Education*) da Internet2 em conjunto com a IBM para a criação de uma arquitetura para transferência segura de informações entre domínios na web, com o propósito de autenticação [4].

Padrões como HTTP, HTML e SSL/TLS tornaram a web acessível e amplamente usada. Após a adoção da web, uma nova necessidade está surgindo: a intercomunicação segura entre os domínios.

De acordo com o MACE, os elementos para garantia de segurança e integridade dentro de um único domínio são bem compreendidos quando comparados com essa necessidade de interligação de domínios. O Shibboleth provê padrões e protocolos, criando um modelo seguro para troca de informações, onde a base desse modelo é a relação de confiança entre as instituições envolvidas. Um ponto importante a destacar é que os recursos que ele visa proteger são aplicações web sendo acessadas via navegador.

Na visão geral do modelo, o usuário é cadastrado apenas na sua *organização de origem*, ou seja, na organização ao qual ele está afiliado, e acessa *recursos* de outras instituições. A intenção é evitar a centralização da segurança em apenas um elemento, permitindo que cada instituição gerencie seus próprios recursos e usuários, e dando maior escalabilidade ao modelo.

Quando um usuário tenta acessar algum serviço localizado em outra organização, através dos protocolos e dos componentes da arquitetura, o Shibboleth faz com que o mesmo seja autenticado na sua organização de origem, e que os seus dados relevantes sejam transferidos para o serviço que está sendo acessado. Esses dados, também conhecidos como *atributos*, são extraídos de alguma base de informações da organização de origem e são enviados via conexão segura, garantindo seu sigilo e sua integridade.

Por esse motivo, a relação de confiança é fundamental; o serviço que está sendo acessado confia que a instituição de origem fez as devidas verificações para garantir a identidade do usuário, e que os dados recebidos

são válidos. As organizações que mantêm uma relação de confiança são agrupadas em *federações*.

Perceba que o usuário é autenticado na instituição onde ele está afiliado, ou seja, suas credenciais não são manipuladas fora do seu domínio. Caso o recurso tivesse que intermediar o processo de validação dos usuários, estaria sendo adicionado um possível ponto de falha de segurança, pois as credenciais desses usuários seriam manipuladas diretamente pelo recurso, que ficaria responsável por entrar em contato com a organização de origem a fim de realizar a autenticação.

Além da segurança na transmissão dos atributos, outro objetivo do Shibboleth é fornecer meios para que os usuários selecionem quais desses atributos podem ser enviados para outros domínio. Isso é motivado pelo fato de alguns países possuírem leis que regulam a disseminação de dados pessoais, e até mesmo por requisitos dos usuários dos sistemas, que cada vez mais estão preocupados de como seus dados pessoais estão sendo usados.

Num maior detalhamento da arquitetura, o Shibboleth é composto de duas grandes partes [32, 4]. A primeira, instalada nas instituições de origem, tem por finalidade realizar a autenticação e fornecer os dados dos usuários. A segunda, instalada no domínio destino, controla o acesso de usuários remotos, recuperando seus atributos nas instituições de origem e repassando-os às aplicações. Um domínio não precisa necessariamente ter as duas partes do Shibboleth instaladas, a menos que ele proveja recursos e possua usuários que acessam outros domínios. Os componentes de cada uma das partes são mostrados na figura 2.2, onde as setas numeradas indicam a ordem de interação.

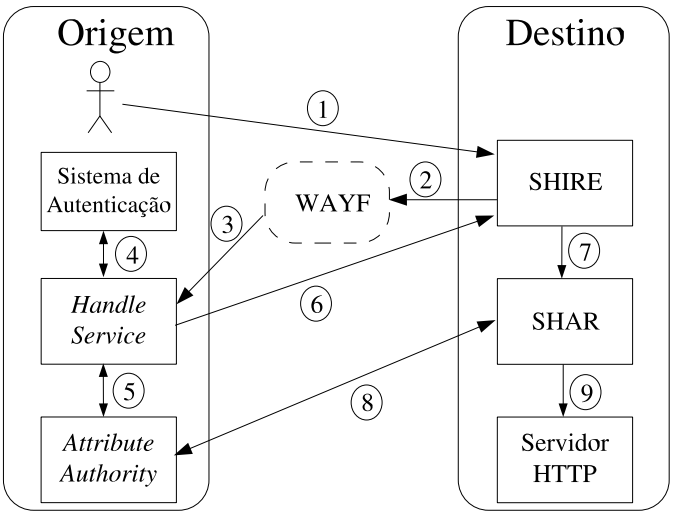

Figura 2.2: Componentes da arquitetura do Shibboleth.

O SHIRE (*Shibboleth Indexical Reference Establisher*) é responsável por garantir que apenas usuários devidamente identificados na arquitetura possam acessar o recurso. Toda vez que uma requisição é feita, (1) o SHIRE a intercepta e verifica se o usuário que a gerou já foi identificado por alguma das instituições da federação.

Caso o usuário não tenha sido identificado, (2) ele é redirecionado para o serviço WAYF (*Where Are You From?*). Este possui a listagem de todas organizações participantes da federação e seus respectivos serviços de identificação de usuário (*Handle Service*). Quando um usuário informa ao WAYF a organização à qual ele é afiliado, (3) ele é redirecionado, através do navegador web, para o serviço de identificação apropriado. Como o WAYF pode estar sendo executado em qualquer um dos servidores da federação, na figura 2.2 ele foi destacado como um componente a parte.

O *Handle Service* (HS) é responsável por validar o usuário e criar um identificador que será usado na obtenção dos atributos junto à instituição de origem. (4) O HS deverá interagir com o usuário, recolher as suas credenciais e verificá-las junto ao sistema de autenticação.

Após a verificação, (5, 6) o HS cria um pacote contendo informações de *status*, o endereço da *Attribute Authority* (AA), um identificador usado pela AA e informações que serão verificadas pelo SHIRE. O pacote gerado não deve possuir dados que de alguma forma forneçam meios para que o receptor descubra a identidade do usuário; apenas os componentes da organização de origem deverão ser capazes de realizar tal identificação.

O SHAR (*Shibboleth Attribute Requester*) é encarregado de solicitar os atributos junto à instituição de origem do usuário. (7, 8) Ele utiliza as informações contidas no pacote de retorno do HS (endereço da AA e o identificador), que são repassadas pelo SHIRE, para contactar a AA e enviar a requisição.

Uma *Attribute Authority* tem acesso a fontes de informações (arquivos, serviços de diretórios, banco de dados, etc.) que armazenam os atributos dos usuário. Dependendo do tamanho do domínio, várias AAs podem ser adotadas, cada uma sendo responsável por um conjunto de usuários. Quando a AA recebe uma requisição do SHAR solicitando informações sobre um determinado usuário, os dados são extraídos e uma resposta contendo os atributos é montada e devolvida ao requisitante. Porém, como visto anteriormente, o Shibboleth permite ao usuário criar regras que definem quais atributos podem ser enviados para outros domínios, assim, a *Attribute Authority*, após resgatar os atributos das fontes de informações, realiza uma filtragem de acordo com o que foi definido como sendo permitido ou não.

Após receber as informações vindas da AA, (9) o SHAR as repassa para o recurso que está sendo acessado.

A interação completa, como mostrado na figura, ocorre apenas no primeiro acesso. Nos acessos subseqüentes, o SHIRE irá reconhecer o usuário e somente os passos 1, 7, 8 e 9 ocorrerão. O passo 8 pode não vir a ser executado com freqüência caso o SHAR adote um esquema de *cache* dos atributos.

2.3 Globus

O conceito de grade computacional foi introduzido nos anos 90 e diz respeito a uma infra-estrutura para computação distribuída envolvendo o compartilhamento de recurso, podendo alcançar múltiplas instituições. Esses recursos podem ser, dentre outros, poder computacional, armazenamento ou dados [7].

As aplicações desenvolvidas para esse ambiente multi-institucional e geograficamente distribuído têm que lidar com problemas que não são comumente encontrados dentro de um único domínio. O dinamismo e a descoberta dos recursos disponíveis, heterogeneidade de equipamentos e sistemas, bem como diferentes políticas organizacionais de uso dos recursos são exemplos das adversidades encontradas.

O Globus é um projeto de implementação da arquitetura OGSA (*Open Grid Services Architecture*), que provê um conjunto de bibliotecas para o desenvolvimento de serviços, que auxilia na criação de aplicações para grade, fornecendo elementos para, por exemplo, a monitoração, disponibilização, descoberta e segurança dos recursos [6, 7, 8]. Ele permite que as instituições envolvidas mantenham o controle sobre os seus recursos, distribuindo a gerência ao longo da grade computacional.

Pelo fato da grade poder envolver diversas organizações, a primeira dificuldade que surge no controle da segurança é identificar de forma única e global um usuário. A solução adotada pelo GSI (*Grid Security Infrastructure*), parte do Globus responsável pela segurança, usa certificados digitais X.509 como identificadores [13], que garantem uma autenticação forte. Isso implica em manter uma (ou várias) Autoridade Certificadora responsável por atribuir certificados aos usuários, o que foi apontado como uma vantagem pelos desenvolvedores do Globus, pois retira das instituições a manutenção dos certificados, ficando essa a cargo de uma terceira parte [33].

Para controlar o acesso aos serviços disponibilizados pela grade, o Globus oferece, dentre outros, um mecanismo chamado *gridmap*, que é uma lista contendo os identificadores de todos os usuários que podem fazer uso dos serviços. Além do controle de acesso, essa lista também tem o propósito de mapear os identificadores globais em contas locais da plataforma de execução. Toda vez que um acesso é solicitado a um serviço protegido por esse mecanismo, o Globus verifica se o identificador do usuário requisitante consta na lista, e se o identificador for encontrado, o serviço é executado usando o identificador local indicado pelo *gridmap*.

Esse mapeamento traz vantagens para a gerência do ambiente de execução. O administrador da plataforma não precisa conhecer os identificadores dos usuários que fazem uso da grade pois todos eles serão relacionados com identificadores locais, então, todas as configurações podem ser feitas, como permissões ao sistema de arquivos ou quota de disco, baseados nos identificadores locais.

Pode-se ainda mapear mais de um identificador global para uma conta local, o que possibilita o agrupamento dos usuários em classes ou grupos. Assim, uma mudança na configuração afeta imediatamente todo conjunto, simplificando o processo.

Assim como o Kerberos, o Globus também oferece a possibilidade de delegar direitos de acesso. Dentro do conceito de grade computacional, um serviço, agindo em nome do usuário, pode decompor uma tarefa em subtarefas menores e distribuí-las para outros serviços. Imagine o seguinte cenário: um usuário solicita uma tarefa a um serviço que está sendo executando na organização A, e a conclusão dessa tarefa depende de que uma subtarefa seja realizada por outro serviço na organização B.

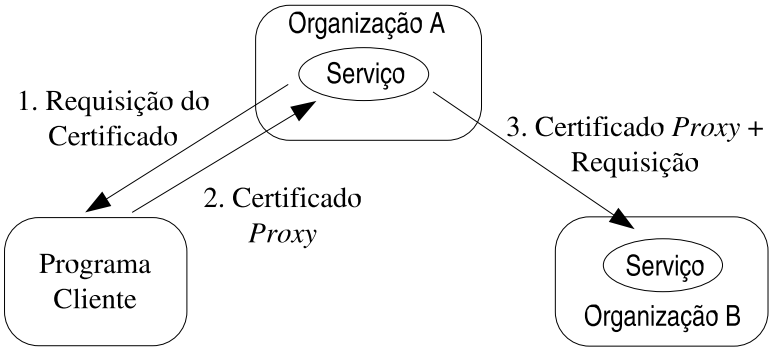

Figura 2.3: Uso do certificado *proxy* na solicitação de serviço.

Caso exista uma relação de confiança entre as duas organizações, B poderia aceitar e executar a subtarefa. Se essa relação não existir, não há,

em princípio, motivos para a organização B realizar o serviço. Entretanto, quando tanto a organização A como B confiam no usuário que solicitou a execução da tarefa, o GSI fornece um mecanismo de delegação que permite A requisitar serviços a B em nome do usuário. Esse mecanismo emprega o certificado *proxy* [13, 33], que é um certificado digital assinado pelo próprio usuário, atestando a confiança em A. Então, como B confia no usuário, ele reconhecerá o certificado *proxy* como válido e aceitará a requisição de serviço.

Para exemplificar melhor o funcionamento desse mecanismo, tomemos o exemplo da figura 2.3. O serviço sendo executado em A (S_A) necessita obter dados de um serviço que está sendo executado em B (S_B) para poder completar a execução de uma tarefa solicitada pelo usuário (entenda usuário como o programa que coordena a execução da tarefa). (1) Através do GSI, S_A cria uma requisição para um certificado *proxy*, que será usado para estabelecer comunicação com S_B, e a envia para o usuário. (2) Este gera o certificado, assinando-o com sua chave privada, e o envia de volta ao serviço. Agora, (3) S_A entra em contato com S_B e apresenta o certificado *proxy*, que é validado através do certificado digital do usuário. Estabelecida a confiança entre os serviços, S_A solicita os dados.

2.4 Controle de Acesso Baseado em Papéis

Uma vez que um usuário é devidamente autenticado e identificado em um sistema, entra em cena o controle das atividades que poderá realizar. Existem dois modelos de controle de acesso bem conhecidos no ambiente de segurança: o discricionário e o compulsório [26].

No modelo discricionário, as permissões (leitura, alteração, execução, etc.) são dadas para cada par identidade do usuário e objeto (algum recurso fornecido e protegido pelo sistema). Isso permite que o controle dos direitos seja feito com uma granularidade muito pequena. Já no modelo compulsório, a gerência das permissões é dada através da atribuição de níveis de segurança para usuários e objetos. Políticas de acesso são montadas levando em consideração a comparação entre o nível do usuário e do objeto.

Ambos os modelos possuem deficiências e diversas alternativas foram propostas para suprí-las. Dentre elas, destaca-se o controle baseado em papéis. O controle de acesso baseado em papéis (CABP) visa prover facilidade na administração da segurança, facilitando a atribuição e manutenção de direitos de acesso nos sistemas. O modelo se baseia na criação de papéis

que representam entidades compostas por um conjunto de responsabilidades como "gerente de projeto" ou "clínico geral" [26, 25, 19].

Estudos realizados pelo NIST (*National Institute of Standards and Technology*) [21] em diversas instituições mostraram que as permissões atribuídas a um papel se alteram pouco com o passar do tempo, diferentemente das atribuídas aos usuários, e que a utilização de papéis para identificar os direitos de acesso era mais fácil do que atribuir os direitos aos usuários.

A partir desse estudo, o NIST propôs uma arquitetura que vem sendo utilizada como base para a implementação do CABP. Para facilitar o entendimento desse modelo proposto, geralmente ele é apresentado em quatro modelos separados que serão chamados de $CABP_0$, $CABP_1$, $CABP_2$ e $CABP_3$. O $CABP_0$ é o modelo base de onde $CABP_1$ e $CABP_2$ se estendem, adicionando novas funcionalidades. $CABP_3$ é o modelo final, que é formado pelo $CABP_1$ e $CABP_2$, e, logo, por $CABP_0$.

O $CABP_0$ é o modelo base da arquitetura e define quatro entidades: usuários, papéis, permissões e sessões. Os usuários representam pessoas ou sistemas para os quais os direitos de acesso devem ser restringidos através dos direitos de acesso descritos pelas permissões. Os papéis representam cargos funcionais ou responsabilidades identificadas na organização que devem ser desempenhadas para que os objetivos sejam alcançados. A figura 2.4 mostra as entidades e o relacionamento entre elas.

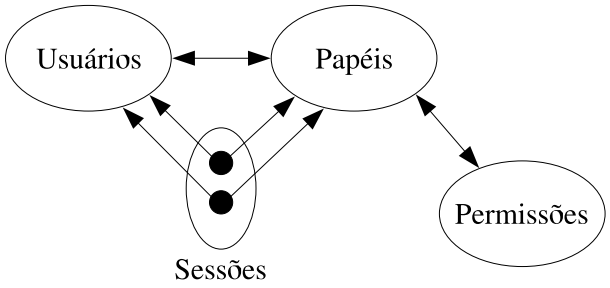

Figura 2.4: Modelo $CABP_0$

Em contraste com o modelo de acesso discricionário, onde os direitos são atribuídos diretamente ao usuário, no $CABP_0$ as permissões são dadas aos papéis, os quais são associados aos usuários de acordo com as tarefas ou cargos que estes possuem. Quando um usuário é associado a um papel ele passa automaticamente a possuir todos os direitos atribuídos a esse papel, tornando a manutenção da segurança simples, pois para mudar o acesso de um usuário basta associá-lo a um novo papel.

Mesmo que um determinado usuário possa assumir diversos papéis, é boa política a cada momento usar apenas aqueles papéis estritamente

necessários para a execução da tarefa em andamento. Esse princípio de segurança tenta evitar "acidentes" e mau uso dos privilégios. Para apoiá-lo, foi introduzido o conceito de *sessão*, na qual o usuário seleciona quais papéis irá usar (ativando esses papéis): a combinação dos direitos de cada um dos papéis ativos resulta no conjunto efetivo de permissões que será atribuído ao usuário. Com isso, pode-se selecionar apenas o que for necessário para a conclusão de uma determinada atividade.

O modelo $CABP_1$ introduz no modelo base o conceito de hierarquia de papéis. Uma hierarquia reflete o conceito de generalização e especialização, o que permite que papéis herdem as permissões e as redefinam num contexto mais específico.

Quando a quantidade de papéis em um esquema de segurança cresce muito, é comum ocorrer a sobreposição de permissões em diversos papéis, tornando a manutenção difícil e susceptível a erros. A criação de papéis genéricos que servem de base para a especificação de outros papéis, insere um mecanismo poderoso para a redefinição coletiva de acesso. Nesse modelo, direitos herdados também podem ser redefinidos.

Como ilustração, suponha por exemplo a existência de um papel genérico *Programador*, onde são dados direitos de leitura sobre a documentação de um sistema e resultados de teste. Os papéis mais específicos *Programador Java* e *Programador C++*, criados a partir de *Programador*, herdam as permissões e ainda são atribuídos novos direitos sobre os arquivos fontes. Caso deseja-se retirar o acesso de ambos ao resultado de teste, basta alterar a permissão no papel *Programador*, caso deseja-se restringir o acesso apenas do *Programador Java*, basta revogar a permissão nesse papel, mantendo os direitos do *Programador C++* inalterados.

$CABP_2$ adiciona a possibilidade de definir restrições no $CABP_0$, que podem impedir, por exemplo, que para um usuário com o papel de gerar ordens de compra seja atribuído o papel de aprovação de compra, introduzindo naturalmente a idéia de supervisão; ou garantir que um papel terá um número máximo de usuários associados. Esse mecanismo de restrições é muito importante para manter a política de segurança de uma instituição consistente, principalmente quando o controle de acesso é gerenciado de forma descentralizada.

A junção do modelo $CABP_1$ com o $CABP_2$ dá origem ao $CABP_3$, adicionando restrições na hierarquia de papéis, como por exemplo, limitar a profundidade máxima de herança de um papel, proibir que um papel tenha descendentes ou impedir que um papel tenha em seu ramo de herança papéis que sejam mutuamente excludentes.

O NIST deixou em aberta a forma concreta de implementação do seu modelo, permitindo que cada organização o adapte à sua realidade e necessidade.