ICS 35.040
L 80
备案号:44633—2014

GM

中华人民共和国密码行业标准

GM/T 0032—2014

基于角色的授权与访问控制技术规范

Specifications for role based privilege management and access control

2014-02-13 发布　　2014-02-13 实施

国家密码管理局　　发布

中华人民共和国密码
行业标准
基于角色的授权与访问控制技术规范
GM/T 0032—2014
*
中国标准出版社出版发行
北京市朝阳区和平里西街甲2号(100029)
北京市西城区三里河北街16号(100045)
网址 www.spc.net.cn
总编室:(010)64275323 发行中心:(010)51780235
读者服务部:(010)68523946
中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销
*
开本 880×1230 1/16 印张 1.75 字数 46 千字
2014年4月第一版 2014年4月第一次印刷
*
书号:155066·2-27010 定价 32.00 元

GM/T 0032-2014

目　　次

前　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本标准由密码行业标准化技术委员会提出并归口。

本标准起草单位：长春吉大正元信息技术股份有限公司、无锡江南信息安全工程技术中心、成都卫士通信息产业股份有限公司、山东得安信息技术有限公司、上海格尔软件股份有限公司、北京数字证书认证中心有限公司、上海市数字证书认证中心有限公司、万达信息股份有限公司、兴唐通信科技有限公司。

本标准起草人：刘平、李伟平、赵丽丽、何长龙、徐强、李元正、高志权、谭武征、李述胜、崔久强、周栋、王妮娜。

基于角色的授权与访问控制技术规范

1 范围

本标准规定了基于角色的授权与访问控制框架结构及框架内各组成部分的逻辑关系，定义了各组成部分的功能、操作流程及操作协议，定义了访问控制策略描述语言、授权策略描述语言的统一格式和访问控制协议的标准接口。

本标准适用于公钥密码技术体系下基于角色的授权与访问控制系统的研制，并可指导对该类系统的检测及相关应用的开发。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 20519 信息安全技术 公钥基础设施 特定权限管理中心技术规范

GM/T 0019 通用密码服务接口规范

3 术语和定义

以下术语和定义仅适用于本文件。

3.1

访问控制判定 access control decision

访问控制决策部件对访问请求的评估结果。

3.2

访问控制决策部件 access control decision function

负责对访问请求做出判定功能的部件。

3.3

访问控制执行部件 access control enforcement function

执行访问控制策略功能的部件。

3.4

访问控制策略 access control policy

由应用确定的角色与资源的绑定关系。

3.5

访问控制策略证书 access control policy certificate

承载应用访问控制策略的属性证书。

3.6

上下文信息 contextual information

访问发生时与访问判定结果相关的环境信息。

3.7

授权管理 privilege management

对主体与角色间的分配关系的管理。

3.8

授权信息　privilege information

标识主体与角色间分配关系的信息。

3.9

授权证书　privilege certificate

承载授权信息的属性证书。

4　缩略语

下列缩略语适用于本文件。

ADF　访问控制决策部件(Access Control Decision Function)

AEF　访问控制执行部件(Access Control Enforcement Function)

5　授权与访问控制框架

5.1　授权与访问控制在公钥密码基础设施应用技术体系框架中的位置

本标准依托于 GM/T 0019,在公钥密码基础设施应用技术体系框架中的位置如图 1 所示。

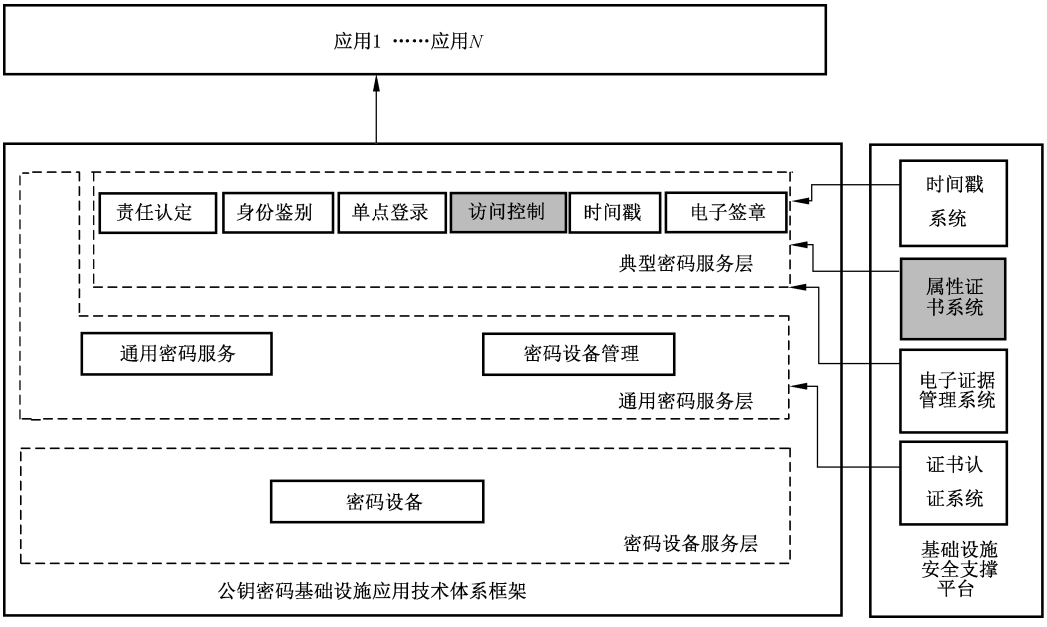

图 1　授权与访问控制在公钥密码基础设施应用技术体系框架中的位置

5.2　授权与访问控制框架概述

基于角色的授权与访问控制框架定义了保障授权信息、访问控制策略的安全性、完整性和有效性的方法,以及与具体应用无关的授权与访问控制的机制。

本框架主要由属性管理系统、访问控制执行部件(AEF)和访问控制决策部件(ADF)组成,其中属性管理系统管理授权信息和访问控制策略;AEF 接收发起者发起的访问请求,将请求按照访问控制协议封装后发送给 ADF;ADF 根据访问控制策略、授权信息等对访问请求作出判定,并将该判定结果返回给 AEF,AEF 根据判定结果决定是否允许发起者对资源进行访问。

对主体的授权信息应按授权策略描述语言进行描述，并采用符合国家密码管理主管部门规定的有效的密码技术保证其完整性、机密性。ADF 可依据发起者身份标识取得对该发起者的授权信息。ADF 使用授权信息前应验证其完整性、真实性。

每一个应用都有一个特定的访问控制策略，该策略采用访问控制策略描述语言进行描述，并采用符合国家密码管理主管部门规定的有效的密码技术保证其完整性、机密性。ADF 使用访问控制策略前应验证其完整性、真实性。

授权与访问控制框架如图 2 所示。

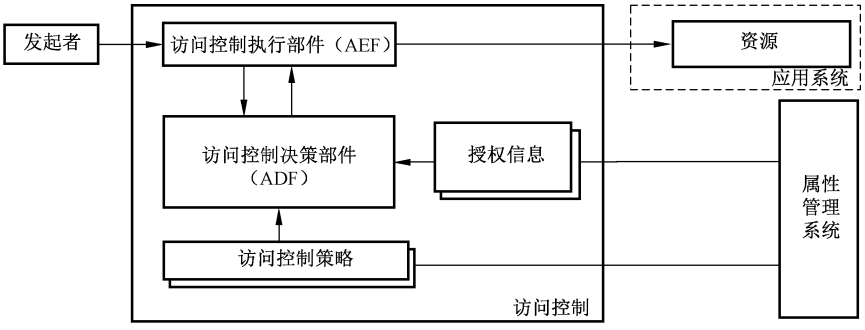

图 2　授权与访问控制框架

5.3　属性管理系统

属性管理系统可以采用多种实现方式完成主体与角色、角色与资源的绑定，如采用授权证书和访问控制策略证书的方式进行。当采用属性证书方式承载绑定关系时，系统应遵循 GB/T 20519 的要求。

5.4　访问控制执行部件（AEF）

AEF 接收访问请求，按照访问控制协议对访问请求进行封装，并依据判定结果控制对资源的访问。

判定结果为允许时，AEF 许可发起者对资源进行访问；判定结果为拒绝时，AEF 应阻止发起者对资源的访问。

AEF 的使用方式包括共享和非共享两种。共享方式下多个应用共同使用一个 AEF，非共享方式下每个应用各自使用各自的 AEF。

5.5　访问控制决策部件（ADF）

ADF 依据授权信息、访问控制策略等信息对访问请求作出判定。

ADF 的输入包括访问请求、授权信息、访问控制策略和 ADF 保留信息。ADF 的输出是访问控制判定结果。

ADF 判定结果包括允许和拒绝。允许意味访问请求符合资源的访问控制策略约束，拒绝意味着访问请求不符合访问控制策略的约束。

ADF 的输入及输出信息如图 3 所示。

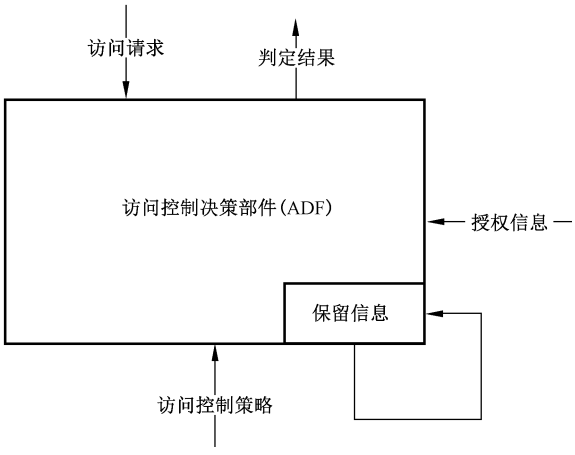

图 3　ADF 的输入、输出信息

ADF 依据访问控制策略进行访问请求判定的过程如下：

1）　ADF 根据请求中的应用及资源信息选择合适的访问控制策略证书；

2）　ADF 根据发起者的身份标识及应用信息从属性管理系统中取得授权证书；

3）　ADF 验证访问控制策略证书和授权证书；

4）　ADF 取得应用的访问控制策略和发起者的授权信息；

5）　ADF 根据请求的资源、操作、发起者的角色及上下文环境信息，在访问控制策略中选择合适的规则进行评估；

6）　如果存在多条规则需要评估，则分别评估每一条规则，并将多个规则评估结果按照合并方法合并成单一评估结果；

7）　ADF 将评估结果按照访问控制协议封装后发送给 AEF。

5.5.1　访问请求

访问请求中主要包括以下四类信息：

1）　发起者身份标识：

发起者身份标识包括三种类型，分别是：

——简单字符串；

——签名证书序列号＋签名证书颁发者主题；

——签名公钥。

2）　资源信息：

资源信息是访问请求中携带的资源标识字符串。

3）　上下文信息：

上下文信息是与访问请求相关且能够标识访问请求环境特征的信息。在进行访问请求判定时可能需要这些信息。它们分别是：

——时间：访问请求发起的时间；

——位置：发出访问请求的源地址；

——发起者身份标识类型；

——自定义信息：应用定义的参与访问判定的信息。

4）　角色标识：

角色标识是访问发生时，主体在当前场景下访问资源应使用的角色信息。

详细的访问请求格式见第 8 章。

5.5.2 授权信息

基于角色的授权与访问控制模型中的授权信息是指主体与角色间的绑定关系。

详细的授权信息描述见第 7 章。

发起者的授权信息可使用授权证书承载，也可使用其他方法承载，但应保证授权信息的真实性、完整性。

5.5.3 访问控制策略

在基于角色的授权与访问控制框架中，每一个应用都有特定的访问控制策略，访问控制策略定义了角色与资源的绑定关系，及对资源进行操作应遵守的约束。

详细的访问控制策略描述见第 6 章。

资源的访问控制策略可使用策略证书承载，对于承载访问控制策略的机制应保证访问控制策略的真实性、完整性。

6 访问控制策略描述语言

6.1 模型

6.1.1 概述

访问控制策略描述语言通过定义访问资源的规则来描述访问控制要求。规则由角色、资源、操作、条件组成。

访问控制策略描述语言模型如图 4 所示。

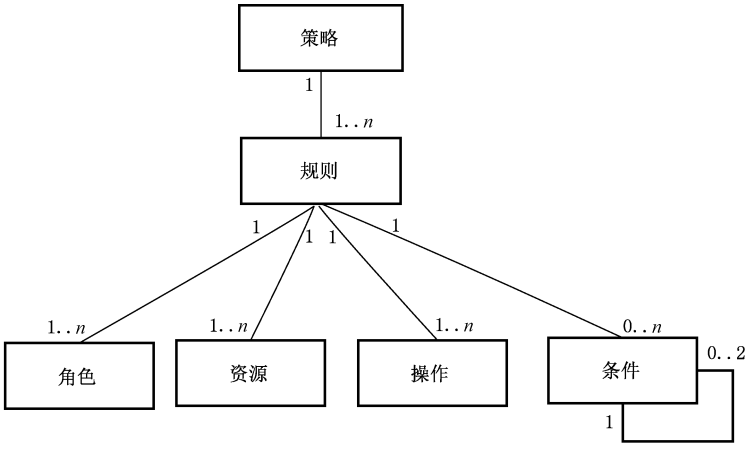

图 4 访问控制策略描述语言模型

图 4 中，策略是多条规则的集合；角色是应用中访问资源的抽象主体；资源是角色访问的对象；操作是角色在资源上执行的动作，其具体内容请参见访问请求承载协议定义。如 HTTP 协议定义了 GET、POST 等操作。

6.1.2 规则

规则是对资源访问进行控制的准则，包括角色、资源、操作和条件等四个要素，其中条件为可选要素。其评估结果可能是允许或拒绝。对于同一资源多个规则评估结果应使用合并算法将其合并为单一

评估结果。

ADF 根据访问请求发起者的角色以及请求的资源、操作和相关的环境因素来选择相应的规则。

对于无条件的规则，访问请求的发起者只要是规则中规定的角色就可以对资源进行相应的操作。对于有条件的规则，访问请求的发起者应是规则中规定的角色并满足条件要求时，才能对资源进行相应的操作。

对于有条件的规则，规则的评估结果依赖条件的评估结果。ADF 评估每一个条件，并将多个条件评估结果通过逻辑组合算法组合成逻辑表达式，最终形成全部条件的评估结果。

6.1.3 条件

条件是在资源上执行指定操作时应满足的上下文限制(如操作的时间限制要求)，其评估结果是 TRUE 或者 FALSE。

条件是由时间、位置、发起者身份标识类型和自定义信息 4 种上下文信息中的一种构成的关系表达式，多个条件可使用逻辑运算符连接并形成逻辑表达式。

本标准定义了条件关系运算和逻辑运算。表 1 是关系运算符及其含义的列表，给出了结果为 TRUE、FALSE 的运算规则。表 2 是逻辑运算符及其含义的列表，并给出了逻辑运算规则。

表 1 关系运算符及其含义

关系运算符	意　义	结果为 TRUE，如果	结果为 FALSE，如果
<	小于	表达式 1 < 表达式 2	表达式 1 >= 表达式 2
<=	小于或等于	表达式 1 <= 表达式 2	表达式 1 > 表达式 2
>	大于	表达式 1 > 表达式 2	表达式 1 <= 表达式 2
>=	大于或等于	表达式 1 >= 表达式 2	表达式 1 < 表达式 2
=	等于	表达式 1 = 表达式 2	表达式 1 ! = 表达式 2
! =	不等于	表达式 1 ! = 表达式 2	表达式 1 = 表达式 2

在 XML 中一些字符具有特定含义，不应直接使用，应对其进行转义。

表 2 逻辑运算符及其含义

逻辑运算符	意义	运算规则
AND	逻辑与	对两个表达式进行逻辑与运算。当且仅当两个表达式均为 TRUE，则结果为 TRUE。如果任一表达式为 FALSE，则结果为 FALSE
OR	逻辑或	对两个表达式进行逻辑或运算。如果两个表达式中至少有一个为 TRUE，则结果为 TRUE
NOT	逻辑非	对表达式执行逻辑非运算。逻辑运算非(NOT)的结果表示布尔值的相反状态：NOT TRUE= FALSE 且 NOT FALSE = TRUE

6.1.4 合并方法

合并方法是将策略中多个规则评估结果组合成一个评估结果的方法。合并多个规则评估结果时可使用下列合并方法中的一种：

——拒绝优先：只要有一个规则的评估结果为拒绝，则最终决策也是拒绝；

——允许优先：只要有一个规则的评估结果为允许，则最终决策也是允许；

——最先应用：以第一个规则的评估结果作为最终决策的评估结果。

本标准使用 DENY-OVERRIDE、PERMIT-OVERRIDE 和 FIRST-APPLICABLE 标识拒绝优先、允许优先、最先应用三种合并方法。

6.2 语法

6.2.1 概述

本标准采用 XML 语法定义访问控制策略描述语言，其语法定义如下。

```
<? xml version = "1.0" encoding = "UTF-8"? >
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">
  <xs:element name = "Policy" type = "PolicyType"/>
  <xs:complexType name = "PolicyType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" default = "1"/>
      <xs:element name = "RuleCombiningAlgId" type = "xs:string"/>
      <xs:element ref = "Rules" maxOccurs = "unbounded"/>
    </xs:sequence>
    <xs: attribute name = "DomainCode" type = "xs:string" minOccurs = "1"/>
    <xs: attribute name = "DomainName" type = "xs:string" minOccurs = "0"/>
  </xs:complexType>
  <xs:element name = "Rules" type = "RulesType"/>
  <xs:complexType name = "RulesType">
    <xs:sequence>
      <xs:element ref = "Roles"/>
      <xs:element ref = "Resources"/>
      <xs:element ref = "Actions"/>
      <xs:element ref = "Condition"/>
    </xs:sequence>
    <xs:attribute name = "RuleId" type = "xs:String" use = "required"/>
  </xs:complexType>
  <xs:element name = "Roles" type = "RolesType"/>
  <xs:complexType name = "RolesType">
    <xs:sequence>
      <xs:element name = "Role" type = "xs:string" maxOccurs = "unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Resources" type = "ResourcesType"/>
  <xs:complexType name = "ResourcesType">
    <xs:sequence>
      <xs:element name = "Resource" type = "xs:string" maxOccurs = "unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Actions" type = "ActionsType"/>
  <xs:complexType name = "ActionsType">
    <xs:sequence>
      <xs:element name = "ActionID" type = "xs:string" maxOccurs = "unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Condition" type = "ConditionType"/>
```

```
  <xs:complexType name = "ConditionType">
    <xs:sequence>
      <xs:choice minOccurs = "0" maxOccurs = "unbounded">
        <xs:element ref = "Condition" minOccurs = "0" maxOccurs = "2"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name = "LogicCombiningAlgId" type = "xs:string" use = "optional"/>
  </xs:complexType>
</xs:schema>
```

6.2.2 策略 Policy

```
<xs:element name = "Policy" type = "PolicyType"/>
<xs:complexType name = "PolicyType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" default = "1"/>
      <xs:element name = "RuleCombiningAlgId" type = "xs:string"/>
      <xs:element ref = "Rules" maxOccurs = "unbounded"/>
     </xs:sequence>
    <xs: attribute name = "DomainCode" type = "xs:string" minOccurs = "1"/>
    <xs: attribute name = "DomainName" type = "xs:string" minOccurs = "0"/>
</xs:complexType>
```

访问控制策略(Policy)由 Version、RuleCombiningAlgId、Rules 组成,分别是协议的版本、合并方法和规则。

版本(Version)是访问控制策略语法的版本标识,当前默认为 1。

合并方法(RuleCombiningAlgId)是多个规则评估结果合并方法的标识。可能的合并方法标识包括 DENY-OVERRIDE、PERMIT-OVERRIDE 和 FIRST-APPLICABLE,分别表示拒绝优先、允许优先和最先应用。

属性 DomainCode 和 DomainName 分别为应用的代码和名称,应用的代码属性指明访问控制策略是为哪一个应用定义的。DomainName 用于人机交互,实际应用过程中可省略。

6.2.3 规则 Rules

```
<xs:element name = "Rules" type = "RulesType"/>
<xs:complexType name = "RulesType">
    <xs:sequence>
      <xs:element ref = "Roles"/>
      <xs:element ref = "Resources"/>
      <xs:element ref = "Actions"/>
      <xs:element ref = "Condition"/>
    </xs:sequence>
    <xs:attribute name = "RuleId" type = "xs: String" use = "required"/>
</xs:complexType>
```

规则(Rules)包括 Roles、Resources、Actions 和 Condition 四个元素,分别是角色、资源、操作和条件。

规则(Rules)的 RuleId 属性是规则的标识。

条件(Condition)是指具有特定角色的访问发起者在资源(Resources)上执行操作(Actions)应满足的上下文信息的约束,详细的语法定义见 6.2.7。

6.2.4 角色 Roles

```
<xs:element name = "Roles" type = "RolesType"/>
<xs:complexType name = "RolesType">
  <xs:sequence>
    <xs:element name = "Role" type = "xs:string" maxOccurs = "unbounded"/>
  </xs:sequence>
</xs:complexType>
```

角色(Roles)指定在资源上执行特定操作应具备的角色标识。

6.2.5 资源 Resources

```
<xs:element name = "Resources" type = "ResourcesType"/>
<xs:complexType name = "ResourcesType">
  <xs:sequence>
    <xs:element name = "Resource" type = "xs:string" maxOccurs = "unbounded"/>
  </xs:sequence>
</xs:complexType>
```

资源(Resources)是资源标识。本标准不定义资源标识的格式,其具体形式参见承载访问请求的协议。

6.2.6 操作 Actions

```
<xs:element name = "Actions" type = "ActionsType"/>
<xs:complexType name = "ActionsType">
  <xs:sequence>
    <xs:element name = "ActionID" type = "xs:string" maxOccurs = "unbounded"/>
  </xs:sequence>
</xs:complexType>
```

操作(Actions)是可在资源上执行的操作标识。本标准不定义操作标识的格式,其具体形式参见承载访问请求的协议。

6.2.7 条件 Condition

```
<xs:element name = "Condition" type = "ConditionType"/>
<xs:complexType name = "ConditionType">
  <xs:sequence>
    <xs:choice minOccurs = "0" maxOccurs = "unbounded">
      <xs:element ref = "Condition" minOccurs = "0" maxOccurs = "2"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name = "LogicCombiningAlgId" type = "xs:string" use = "optional"/>
</xs:complexType>
```

条件(Condition)可由 0 个到 2 个条件项顺序组成并构成表达式二叉树,通过遍历该二叉树能够形成由上下文环境信息构成的资源访问控制条件表达式。条件项的值可以是条件或上下文环境约束。上下文环境约束是由上下文信息标识、关系运算符及上下文信息限定值构成的关系表达式。

当条件项的值仍然是条件时,属性 LogicCombiningAlgId 的值应该是逻辑运算符或括号[可以是“(”和“)”],当条件项的值是上下文环境约束时,属性 LogicCombiningAlgId 的值可以是空或单目逻辑运算符“NOT”。

本标准定义了 E_TIME、E_LOCATION、E_IDTYPE、E_EXTENDTYPE 四个上下文标识，它们分别代表时间、位置、发起者身份标识类型和自定义信息。关于上下文信息见 5.5.1 中 3)。

下面以时间限制为大于 2013 年 9 月 10 日 0 时且小于 2013 年 9 月 10 日 12 时为例，说明条件的使用方法。

```
<Condition LogicCombiningAlgId = "AND">
    <Condition>E_TIME&gt;20130910000000Z</Condition>
    <Condition>E_TIME&lt;20130910120000Z</Condition>
</Condition>
```

根据上面条件形成的判定表达式为：

```
(E_TIME > 20130910000000Z)AND(E_TIME < 20130910120000Z)
```

7 授权策略描述语言

7.1 模型

授权信息是指主体与角色的分配关系。图 5 说明了主体、角色和应用间的关系。在一个应用内可以定义多个角色，一个角色可以分配给多个主体，一个主体可被分配多个角色，获得角色的主体可以访问与该角色相关的特定应用资源。

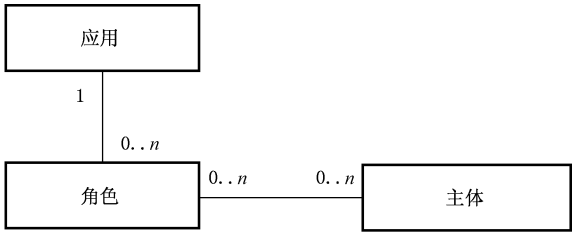

图 5 授权策略描述语言模型

本标准定义了主体与角色分配关系的描述语法，称为授权策略描述语言。

主体的授权信息采用授权策略描述语言描述，授权策略可为强制授权策略或自动授权策略。强制授权策略为单一的授权主体强制性匹配一个角色，自动授权策略定义一个与角色对应的规则群组，符合该规则群组的主体被自动匹配该角色。授权信息可放置在授权证书中，ADF 根据发起者标识和应用标识从属性管理系统中取得该授权证书。

7.2 授权策略描述语言语法

7.2.1 概述

本标准采用 XML 语法定义授权策略描述语言，其语法定义如下。

```
<? xml version = "1.0" encoding = "UTF-8"? >
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">
  <xs:element name = "Policy" type = "PolicyType"/>
  <xs:complexType name = "PolicyType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" minOccurs = "1"/>
      <xs:element ref = "Subject" minOccurs = "1"/>
      <xs:element ref = "Role" minOccurs = "1"/>
    </xs:sequence>
```

```
    </xs:complexType>
    <xs:element name = "Subject" type = "SubjectType"/>
    <xs:complexType name = "SubjectType">
      <xs:sequence>
        <xs:element ref = "singleSubject"/>
        <xs:element ref = "ruleGroupSubject"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name = "singleSubject" type = "singleSubjectType"/>
    <xs:complexType name = "singleSubjectType">
      <xs:sequence>
        <xs:element ref = "entityNameType"/>
        <xs:element ref = "baseCertificateIDType"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name = "entityNameType" type = "xs:string"/>
    <xs:element name = "baseCertificateIDType">
      <xs:complexType>
        <xs:sequence>
          <xs:element name = "issuer" type = "xs:string"/>
          <xs:element name = " serialNumber" type = "xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name = "ruleGroupSubject" type = "ruleGroupSubjectType"/>
    <xs:complexType name = "ruleGroupSubjectType">
      <xs:sequence>
        <xs:element ref = "ruleGroupSubject" minOccurs = "0" maxOccurs = "2"/>
      </xs:sequence>
      <xs:attribute name = "LogicCombiningAlgId" type = "xs:string" use = "optional"/>
    </xs:complexType>
    <xs:element name = "Role" type = "RoleType"/>
    <xs:complexType name = "RoleType">
      <xs:sequence>
        <xs:element name = "RoleCode" type = "xs:string" minOccurs = "1"/>
        <xs:element name = "RoleName" type = "xs:string" minOccurs = "0"/>
        <xs:element name = "DomainCode" type = "xs:string" minOccurs = "1"/>
        <xs:element name = "DomainName" type = "xs:string" minOccurs = "0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
```

7.2.2 授权策略 Policy

```
    <xs:element name = "Policy" type = "PolicyType"/>
    <xs:complexType name = "PolicyType">
      <xs:sequence>
        <xs:element name = "Version" type = "xs:integer" default = "1"/>
        <xs:element ref = "Subject" minOccurs = "1"/>
```

```
    <xs:element ref = "Role" minOccurs = "1"/>
  </xs:sequence>
</xs:complexType>
```

授权策略由 Version、Subject 和 Role 三个元素构成，分别为协议版本、授权主体和角色。

本标准规定版本(Version)默认为整数 1。

7.2.3 授权主体 Subject

```
<xs:element name = "Subject" type = "SubjectType"/>
<xs:complexType name = "SubjectType">
  <xs:sequence>
    <xs:element ref = "singleSubject"/>
    <xs:element ref = "ruleGroupSubject"/>
  </xs:sequence>
</xs:complexType>
```

授权主体(Subject)的类型是授权主体类型(SubjectType)。

授权主体类型包括单一授权主体(singleSubject)和规则群组授权主体(ruleGroupSubject)两种。单一授权主体适用于强制授权策略，规则群组授权主体适用于自动授权策略。

7.2.4 单一授权主体 singleSubject

```
<xs:element name = "singleSubject" type = "singleSubjectType"/>
<xs:complexType name = "singleSubjectType">
  <xs:sequence>
    <xs:element ref = "entityNameType"/>
    <xs:element ref = "baseCertificateIDType"/>
  </xs:sequence>
</xs:complexType>
```

单一授权主体(singleSubject)的类型是单一授权主体类型(singleSubjectType)。

单一授权主体类型定义了两种授权主体标识类型，分别为实体名称类型(entityNameType)和基本证书绑定类型(baseCertificateIDType)，可任选其中一种作为单一授权主体标识类型。

7.2.5 实体名称类型 entityNameType

```
<xs:element name = "entityNameType" type = "xs:string"/>
```

实体名称类型(entityNameType)是单一授权主体的一种标识方法，它采用简单的字符串来表示发起者身份标识，例如“实体 A”或“cn=实体名称，o=组织名称，c=cn”等形式。

7.2.6 基本证书绑定类型 baseCertificateIDType

```
<xs:element name = "baseCertificateIDType">
  <xs:complexType>
    <xs:sequence>
      <xs:element name = "issuer" type = "xs:string"/>
      <xs:element name = "serialNumber" type = "xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

基本证书绑定类型(baseCertificateIDType)是单一授权主体的另一种标识方法。基本证书绑定类型由 issuer 和 serialNumber 两个元素组成，分别对应于授权主体公钥证书的签发者(issuer)和序列号

(serialNumber)。

7.2.7 规则群组授权主体 ruleGroupSubject

```
<xs:element name = "ruleGroupSubject" type = "ruleGroupSubjectType"/>
<xs:complexType name = "ruleGroupSubjectType">
  <xs:sequence>
    <xs:element ref = "ruleGroupSubject" minOccurs = "0" maxOccurs = "2"/>
  </xs:sequence>
  <xs:attribute name = "LogicCombiningAlgId" type = "xs:string" use = "optional"/>
</xs:complexType>
```

规则群组授权主体(ruleGroupSubject)是由属性 LogicCombiningAlgId 连接的逻辑表达式。当属性 LogicCombiningAlgId 的值为逻辑运算符或括号时,所连接的两项中至少有一项是规则群组授权主体类型;当属性 LogicCombiningAlgId 的值为空时,规则群组授权主体为由主体属性标识、关系运算符及主体属性限定值构成的关系表达式。主体属性可包括年龄、职务、部门等信息。

根据自动授权策略,当访问请求发起者的属性符合规则群组,即访问请求发起者的属性使规则群组授权主体的值为“真(TRUE)”时,自动匹配角色(Role)。

下面的规则群组授权主体的例子描述了年龄小于 35 岁,职务为“manager”的一组主体。

```
<ruleGroupSubject LogicCombiningAlgId = "AND">
  <ruleGroupSubject>S_AGE&lt;35</ruleGroupSubject>
  <ruleGroupSubject>S_JOB = ”manager”</ruleGroupSubject>
</ruleGroupSubject>
```

7.2.8 角色 Role

```
<xs:element name = "Role" type = "RoleType"/>
<xs:complexType name = "RoleType">
  <xs:sequence>
    <xs:element name = "RoleCode" type = "xs:string" minOccurs = "1"/>
    <xs:element name = "RoleName" type = "xs:string" minOccurs = "0"/>
    <xs:element name = "DomainCode" type = "xs:string" minOccurs = "1"/>
    <xs:element name = "DomainName" type = "xs:string" minOccurs = "0"/>
  </xs:sequence>
</xs:complexType>
```

角色(Role)的数据类型是角色类型(RoleType)。该类型由 RoleCode、RoleName、DomainCode、DomainName 四项组成,分别为角色标识、角色名称、应用标识和应用名称。

DomainCode 指明了 RoleCode 所属的应用。RoleName 和 DomainName 用于人机交互,实际应用过程中可省略。

8 访问控制协议

8.1 概述

如图 6 所示,访问控制协议是 AEF 与 ADF 间交互的协议。访问控制协议由访问控制请求消息和访问控制响应消息两部分构成。

访问控制请求消息是 AEF 发送给 ADF 的消息。AEF 按照访问控制请求消息的要求,将格式化后的访问请求转发给 ADF。ADF 接收到 AEF 发来的访问控制请求消息后,执行访问控制判定,并以访问控制响应消息回答 AEF。AEF 根据访问控制响应消息中的访问控制判定结果,允许或拒绝访问

请求。

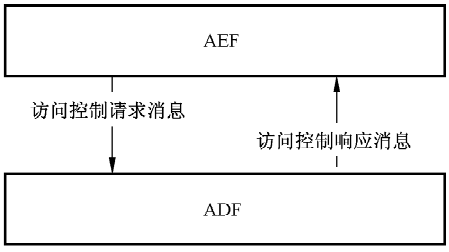

图 6　访问控制协议

8.2　访问控制请求消息

8.2.1　概述

本标准采用 XML 语法定义访问控制请求协议，其语法定义如下。

```
<? xml version = "1.0" encoding = "UTF-8"? >
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">
  <xs:element name = "Request" type = "RequestType"/>
  <xs:complexType name = "RequestType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" minOccurs = "1"/>
      <xs:element ref = "Subject" maxOccurs = "unbounded"/>
      <xs:element ref = "Resources" maxOccurs = "unbounded"/>
      <xs:element ref = "Actions"/>
      <xs:element ref = "Environment"/>
      <xs:element ref = "Role" type = ”xs:String"/>
    </xs:sequence>
    <xs: attribute name = "DomainCode" type = "xs:string" minOccurs = "1"/>
  </xs:complexType>
<xs:element name = "Subject" type = "SubjectType"/>
  <xs:complexType name = "SubjectType">
    <xs:sequence>
      <xs:element ref = "entityNameType"/>
      <xs:element ref = "baseCertificateIDType"/>
    </xs:sequence>
  </xs:complexType>
<xs:element name = "entityNameType" type = "xs:string"/>
  <xs:element name = "baseCertificateIDType">
    <xs:complexType>
      <xs:sequence>
        <xs:element name = "issuer" type = "xs:string"/>
        <xs:element name = "serial" type = "xs:string"/>
      </xs:sequence>
```

```
    </xs:complexType>
  </xs:element>
  <xs:element name = "Resources" type = "ResourcesType"/>
  <xs:complexType name = "ResourcesType">
    <xs:sequence>
      <xs:element name = "Resource" type = "xs:string" maxOccurs = "unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Actions" type = "ActionsType"/>
  <xs:complexType name = "ActionsType">
    <xs:sequence>
      <xs:element name = "ActionID" type = "xs:string" maxOccurs = "unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Environment" type = "EnvironmentType"/>
  <xs:complexType name = "EnvironmentType">
    <xs:sequence>
      <xs:element name  = "E_TIME" type = "xs:string"/>
      <xs:element name  = "E_LOCATION" type = "xs:string"/>
      <xs:element name  = "E_IDTYPE" type = "xs:string"/>
      <xs:element name  = "E_EXTENDTYPE" type = "xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

8.2.2 访问控制请求 Request

```
  <xs:element name = "Request" type = "RequestType"/>
  <xs:complexType name = "RequestType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" minOccurs = "1"/>
      <xs:element ref = "Subject" maxOccurs = "unbounded"/>
      <xs:element ref = "Resources" maxOccurs = "unbounded"/>
      <xs:element ref = "Actions"/>
      <xs:element ref = "Environment"/>
      <xs:element ref = "Role" type = ”xs:String"/>
    </xs:sequence>
    <xs: attribute name = "DomainCode" type = "xs:string" minOccurs = "1"/>
  </xs:complexType>
```

访问控制请求消息(Request)是由 AEF 发送到 ADF,请求 ADF 做出访问控制判定的消息。ADF 根据访问控制请求消息、授权信息、访问控制策略等进行访问控制判定,并将访问控制判定结果以访问控制响应消息(Response)的形式返回给 AEF。

访问控制请求由 Version、Subject、Resources、Actions、Environment 和 Role 六部分组成,分别代表协议的版本号、访问请求的发起者、访问目标(资源)、操作、访问请求的上下文环境信息以及角色标识。这些信息来源于访问请求消息和 AEF 的环境。

本标准规定版本(Version)默认为整数 1。

属性 DomainCode 为应用的代码,用于标识发起者将要访问的应用。

关于 Subject、Resources、Actions 和 Environment 几部分的定义请参阅 8.2.3,8.2.6,8.2.7 和 8.2.8。

8.2.3 发起者 Subject

```
<xs:element name = "Subject" type = "SubjectType"/>
  <xs:complexType name = "SubjectType">
    <xs:sequence>
      <xs:element ref = "entityNameType"/>
      <xs:element ref = "baseCertificateIDType"/>
    </xs:sequence>
</xs:complexType>
```

发起者携带访问发起者身份标识信息。本标准支持两种发起者表示形式,分别为实体名称类型(entityNameType)和基本证书绑定类型(baseCertificateIDType)。但访问控制请求消息中的发起者标识不应同时出现一种以上的表示形式,AEF 构造访问控制请求消息时应决定选择哪一种发起者标识表示形式。

8.2.4 实体名称类型 entityNameType

```
<xs:element name = "entityNameType" type = "xs:string"/>
```

实体名称类型(entityNameType)是一种发起者身份标识的表示形式。实体名称类型采用简单的字符串来表示发起者身份标识,例如"实体 A"或"cn=实体名称,o=组织名称,c=cn"等形式。ADF 应将实体名称类型值的整体作为发起者身份标识。本标准不规定实体名称类型值的内部语法,身份鉴别服务可根据需要自由定义。

基于公钥证书认证的情况下,该元素应与公钥证书的主体(subject)域相同,或者与公钥证书 subjectAltName 域扩展(如果有)的一个值相同。

8.2.5 基本证书绑定类型 baseCertificateIDType

```
<xs:element name = "baseCertificateIDType">
  <xs:complexType>
    <xs:sequence>
      <xs:element name = "issuer" type = "xs:string"/>
      <xs:element name = "serial" type = "xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

基本证书绑定类型(baseCertificateIDType)是另外一种发起者身份标识的表示形式。基本证书绑定类型采用签名数字证书序列号和数字证书颁发者主题来表示发起者身份标识。

8.2.6 资源 Resources

```
<xs:element name = "Resources" type = "ResourcesType"/>
<xs:complexType name = "ResourcesType">
  <xs:sequence>
    <xs:element name = "Resource" type = "xs:string" maxOccurs = "unbounded"/>
  </xs:sequence>
</xs:complexType>
```

资源(Resources)的详细语法定义见 6.2.5。

8.2.7 操作 Actions

```
<xs:element name = "Actions" type = "ActionsType"/>
<xs:complexType name = "ActionsType">
  <xs:sequence>
    <xs:element name = "ActionID" type = "xs:string" maxOccurs = "unbounded"/>
  </xs:sequence>
</xs:complexType>
```

操作(Actions)的详细语法定义见 6.2.6。

8.2.8 上下文环境 Environment

```
<xs:element name = "Environment" type = "EnvironmentType"/>
<xs:complexType name = "EnvironmentType">
  <xs:sequence>
    <xs:element name  = "E_TIME" type = "xs:string"/>
    <xs:element name  = "E_LOCATION" type = "xs:string"/>
    <xs:element name  = "E_IDTYPE" type = "xs:string"/>
    <xs:element name  = "E_EXTENDTYPE" type = "xs:string"/>
  </xs:sequence>
</xs:complexType>
```

本标准定义了 E_TIME、E_LOCATION、E_IDTYPE、E_EXTENDTYPE 4 个访问控制上下文环境参数,分别对应访问请求的发起时间、位置、身份标识类型和自定义信息。

时间指 AEF 接收到访问请求的时间,采用格林尼治标准时间格式(即,YYYYMMDDhhmmssZ 格式)。

位置是访问请求发出的地址,本标准使用发出访问请求的 IP 地址表示。

身份标识类型的值为 EntityNameType、baseCertificateIDType 中的一种,分别表示采用实体名称和数字证书来标识发起者的身份。

自定义信息可用于表达其他未在标准中定义的环境信息,在访问控制请求中可有多个自定义信息项,其值格式须为 KEY=VALUE 字符串形式。其中 KEY 为自己定义的环境信息标识,VALUE 为该标识的值,KEY 和 VALUE 中不允许出现“=”字符,如出现则应转义。

8.3 访问控制响应消息

8.3.1 概述

本标准采用 XML 语法定义访问控制响应协议,其语法定义如下。

```
<? xml version = "1.0" encoding = "UTF-8"? >
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">
  <xs:element name = "Response" type = "ResponseType"/>
  <xs:complexType name = "ResponseType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" minOccurs = "1"/>
      <xs:element ref = "Result"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Result" type = "ResultType"/>
```

```
  <xs:complexType name = "ResultType">
    <xs:sequence>
      <xs:element ref = "Decision"/>
      <xs:element ref = "Status" minOccurs = "0"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name = "Decision" type = "DecisionType"/>
  <xs:simpleType name = "DecisionType">
    <xs:restriction base = "xs:string">
      <xs:enumeration value = "Permit"/>
      <xs:enumeration value = "Deny"/>
      <xs:enumeration value = "Exception"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name = "Status" type = "StatusType"/>
  <xs:complexType name = "StatusType">
    <xs:sequence>
      <xs:element name  = "StatusCode" type = "xs:string"/>
      <xs:element name  = "StatusMessage" minOccurs = "0" type = "xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

8.3.2 访问控制响应 Response

```
  <xs:element name = "Response" type = "ResponseType"/>
  <xs:complexType name = "ResponseType">
    <xs:sequence>
      <xs:element name = "Version" type = "xs:integer" minOccurs = "1"/>
      <xs:element ref = "Result"/>
    </xs:sequence>
  </xs:complexType>
```

访问控制响应消息(Response)由 ADF 发送到 AEF,是 ADF 对 AEF 发出的访问控制请求消息的应答消息。它由 Version 和 Result 两部分组成,分别代表协议的版本号和访问控制判定结果。

本标准规定版本(Version)默认为整数 1,关于 Result 的定义请参阅 8.3.3～8.3.5。

8.3.3 访问控制判定 Result

```
  <xs:element name = "Result" type = "ResultType"/>
  <xs:complexType name = "ResultType">
    <xs:sequence>
      <xs:element ref = "Decision"/>
      <xs:element ref = "Status" minOccurs = "0"/>
    </xs:sequence>
  </xs:complexType>
```

访问控制判定(Result)是 ADF 对访问控制请求的判定。访问控制判定元素的类型是访问控制判定类型(ResultType),它由 Decision 和 Status 两部分组成,分别代表 ADF 的判定结果和判定状态信息。

当 ADF 运行正常时,Decision 表示 ADF 对访问请求的判定结果,其值可能是“Permit”或“Deny”,

此时，判定状态(Status)可以不出现。当 ADF 运行发生异常时，Decision 的值应置为“Exception”，判定状态(Status)应出现并携带异常信息。

8.3.4 判定结果 Decision

```
<xs:element name = "Decision" type = "DecisionType"/>
<xs:simpleType name = "DecisionType">
  <xs:restriction base = "xs:string">
    <xs:enumeration value = "Permit"/>
    <xs:enumeration value = "Deny"/>
    <xs:enumeration value = "Exception"/>
  </xs:restriction>
</xs:simpleType>
```

判定结果(Decision)明确定义了访问控制请求的几种可能判定结果，分别是“Permit”、“Deny”和“Exception”，它们的含义如下：

1) 当访问控制请求符合访问控制策略约束时，其判定结果应是“Permit”，意味着 AEF 应允许该请求对资源的访问；
2) 当访问控制请求不符合访问控制策略约束时，其判定结果应是“Deny”，意味着 AEF 应禁止该请求对资源的访问；
3) 当 ADF 发生异常时，其判定结果应是“Exception”，ADF 的异常信息被放置在判定状态(Status)中。

8.3.5 判定状态 Status

```
<xs:element name = "Status" type = "StatusType"/>
<xs:complexType name = "StatusType">
  <xs:sequence>
    <xs:element name  = "StatusCode" type = "xs:string"/>
    <xs:element name  = "StatusMessage" type = "xs:string"/>
  </xs:sequence>
</xs:complexType>
```

判定状态(Status)的类型是判定状态类型(StatusType)，它由 StatusCode 和 StatusMessage 两部分组成，分别代表判定状态的编码和判定状态详细信息。

判定状态的编码和详细信息见附录 A。

9 对应用系统的要求

9.1 AEF 实现

AEF 除可如图 2 所示在独立的服务中实现外，也可直接由应用系统实现。

9.2 角色的表达

角色的表达包括确定角色名称和角色值。角色名称是为授权管理提供可理解的符号以便于人机交互，角色值是一个角色区别于其他角色的适于计算机处理的唯一代表该角色的代码。

角色的表达确定可供分配角色的数量、角色值的值域及其内部关系。通常，应用功能被确定后，角色表达应相对稳定。

9.3 授权过程

授权者将角色分配给用户，或修改用户与角色的对应关系。属性权威对授权信息进行签名形成属性证书并对外发布。

授权信息发生变化应及时更新以免发生安全风险。

9.4 访问控制策略的描述

应用系统开发者应按第6章的要求来描述应用的访问控制策略。

9.5 身份鉴别

访问系统应在访问控制判定前完成身份鉴别，但本标准不对身份鉴别过程进行规范。

附 录 A
（规范性附录）
访问控制判定状态代码定义和说明

表 A.1 访问控制判定状态代码定义和说明

预定义值	说 明
0	成功
0x71010001	访问控制判定请求消息解析异常
0x71010002	访问控制判定请求消息格式错误
0x71020001	服务发生未知异常
0x71020002	未发现访问发起者授权信息
0x71020003	取访问发起者授权信息异常
0x71020004	解析授权信息异常
0x71020005	未发现访问控制策略
0x71020006	取访问控制策略异常
0x71020007	解析访问控制策略异常

参 考 文 献

[1] The Open Group. 2000. Authorization (AZN) API. Open Group Technical Standard C908.

[2] Pato, J. 1990. DCE Access Control Lists (ACL's). OSF DCE specifications.

[3] ANSI INCITS 359—2004 Role-based Access Control.

[4] GB/T 18794.3—2003 信息技术 开放系统互连 开放系统安全框架 第3部分:访问控制框架.