ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.8, August 2010, pp.2059−2069
doi: 10.3724/SP.J.1001.2010.03646

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种面向 DRM 的责任授权模型及其实施框架*

钟 勇[1,3,4+], 秦小麟[3], 刘凤玉[1,2]

[1](南京理工大学 计算机科学与技术博士后流动站,江苏 南京 210094)

[2](南京理工大学 计算机科学与技术学院,江苏 南京 210094)

[3](南京航空航天大学 信息安全研究所,江苏 南京 210016)

[4](佛山科学技术学院 信息与教育技术中心,广东 佛山 528000)

Obligation Authorization Model and Its Implementation Framework for DRM

ZHONG Yong[1,3,4+], QIN Xiao-Lin[3], LIU Feng-Yu[1,2]

[1](Postdoctoral Station of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China)

[2](School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China)

[3](Institute of Information Security, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

[4](Information and Educational Technology Center, Foshan University, Foshan 528000, China)

+ Corresponding author: E-mail: zhongyong1970@126.com, http://datasecu.fosu.edu.cn

Zhong Y, Qin XL, Liu FY. Obligation authorization model and its implementation framework for DRM. *Journal of Software*, 2010,21(8):2059−2069. http://www.jos.org.cn/1000-9825/3646.htm

Abstract: Due to the absence of actual obligation description and implementation abilities in existing DRM (digital rights management) mechanism, this paper presents an obligation authorization model and its implementation framework that can be applied in DRM. The model is based on distributed temporal logic and Active-U-Datalog rules, which empowers the model to express event-driven, time-driven, obligation compensation and other semantics of obligation descriptions, and also give the model a favorable feasibility of implementation. The semantics and syntax of the model are analyzed and explained. And the implementation mechanism of the model is discussed. Finally, the implementation, application and expressiveness of the model are showed and illustrated. The model improves the flexibility and capability of usage control of data in DRM system.

Key words: obligation model; usage control; digital rights management; Datalog language; access control

摘 要: 针对现有的 DRM(digital rights management)机制缺乏真正的责任描述和实施能力问题,提出一种应用于 DRM 的责任授权模型及其实施框架.该模型基于分布式时态逻辑和 Active-U-Datalog 语法规则,具有表达事件驱动、时间驱动和责任补偿等各类责任授权的语义能力,并具有良好的可实施性.对该模型的语法语义进行了分析和

* Supported by the National Natural Science Foundation of China under Grant No.60673127 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2007AA01Z404 (国家高技术研究发展计划(863)); the China Postdoctoral Science Foundation under Grant No.20070421015 (中国博士后科学基金); the Guangdong Provincial Natural Science Foundation of China under Grant No.8452800001001086 (广东省自然科学基金); the Jiangsu Planned Projects for Postdoctoral Research Funds of China under Grant No.0801045B (江苏省博士后科研资助计划)

Received 2008-10-08; Revised 2009-02-16; Accepted 2009-04-27

说明,描述了责任实施机制,并对该模型的实现、应用和表达力进行了说明和示例.该模型提高了 DRM 系统对数据使用控制的灵活性和能力.

关键词: 责任模型;使用控制;数字权利管理;Datalog 语言;访问控制

中图法分类号: TP309 **文献标识码**: A

传统访问控制模型利用主客体属性进行授权决策,是一种使用前预先授权机制[1],无法满足许多现代应用(如数字权利管理(digital rights management,简称 DRM)、隐私保护和数字图书馆等)的需要.这些应用不仅需要解决允许谁能访问数据的问题,而且需要解决之后数据如何使用的问题.因而,访问控制模型的扩展和泛化研究近年来得到了重视,如 Jajodia 等人[2]提出的多策略灵活授权框架(FAF)中时态、环境语义等属性的引入[3,4]和动态授权[5]等.而使用控制[1]和责任授权(obligation authorization)等[6,7]概念的提出,进一步推动了新型访问控制模型的研究.

责任指的是现在或未来使用数据所必须承担的义务,如数据必须在 1 个月内删除.责任与授权的一个重要区别在于:如果授权规则被违反,则用户将无法获取相应的权限;如果责任被违反,则用户可能需要执行相应的补偿或接受惩罚,但不一定影响权限获取[7].因而,责任机制与访问控制机制相关但应具有独立运行的责任实施机制.另外,责任的表达也应包含如下因素:(1) 时态驱动因素.责任往往具有时间驱动因素,如在特定时间后执行、按某种周期执行等.(2) 事件驱动因素.责任往往具有一定的事件驱动性,如数据在使用、传输和删除等操作期间或之后产生相应责任.(3) 责任补偿因素.责任被违反后,用户需要执行相应的补偿行动或接受一定的惩罚.

完整的责任表达和实施应包括时态驱动、事件驱动和责任补偿等因素,并与访问控制机制分离.在现有的研究中,Bettini 等人[6]通过 Datalog 谓词来表达责任,并通过在授权规则中增加责任谓词作为责任授权,责任授权由独立于访问控制模块的程序进行控制和运行.该方法是一种扩展的简单责任授权方案,无法表达责任授权中的时间、事件驱动因素和责任补偿等概念.Manuel 等人[7]使用分布式时态逻辑(DTL)来表达责任授权,表达力较强,但其方法仍无法表达部分的事件驱动性责任,如无法表达事件的开始时刻、结束时刻或持续期间等概念.其他方法如 Irwin 等人的方法[8]和 Dougherty 等人的方法[9]在表达力方面,Barth 等人的方法[10]在算法复杂度和可执行性方面都存在不足.OSL 语言[11]能够表达时态、事件驱动和责任补偿等丰富的责任授权概念,但该语言未能将责任授权与访问控制机制分离,而是通过在传统授权语言中增加责任授权的方式实现.类似的方法也包括 PONDER[12]和 PDL[13]等策略表达语言.现有的权利描述语言如 XrML,ODRL 和 MPEG-21 REL 等,也能表达一定的时态驱动责任,但都未能将责任机制与访问控制机制分离,也不存在完整的责任补偿等概念.在责任实施方面,现有的实施方法[14−16]也不是专门针对 DRM.事实上,现有的 DRM 机制中缺乏真正的责任授权描述和实施能力.

针对上述问题,本文提出一种应用于 DRM 的责任授权模型 OUC(obligation-based usage control),对模型的语法语义进行了分析和说明,描述了模型的责任实施机制,并对模型的实现、应用和表达力进行了说明和示例.

1 OUC 责任授权模型的语法和语义

OUC 模型基于我们提出的 LucScript(logic-based usage control license script)权利描述语言(rights expression language,简称 REL)[17].该语言基于 Active-U-Datalog[18]逻辑,其具有触发功能的授权机制起源于我们前期提出的集中式使用控制授权框架 LUC[19].Active-U-Datalog 是一种结合主动规则具有可更新能力的 Datalog 程序,其谓词原子包括表示插入和删除的更新原子$\pm p(t_1,t_2,\ldots,t_n)$.Active-U-Datalog 事务是不含规则头的具有如下形式的规则:

$$u_1,\ldots,u_k, l_1,\ldots,l_m \tag{1}$$

其中,$u_i(0\leqslant i\leqslant k)$是任意更新原子,$l_i(0\leqslant i\leqslant m)$是不含更新原子的文字.

OUC 模型的时态操作符基于分布式时态逻辑(distributed temporal logic,简称 DTL)的扩展.DTL 的未来时态操作符包括一元操作符 X(*next*)和 G(*always*)、二元操作符∪.DTL 的过去时态操作符包括一元操作符

Y(previous),P(sometime in the past)和 H(always in the past)以及二元操作符 S(since).另外,X^n 代表 n 个重复的应用 X,如定义:

$$F^{\leqslant n}\varphi \equiv \vee_{i=0}^{n} X^i\varphi,\ G^{\leqslant n}\varphi \equiv \wedge_{i=0}^{n} X^i\varphi.$$

1.1 语 法

责任是如公式(1)所示不含规则头的 Active-U-Datalog 事务,本文也将责任表示为

$$l_1\wedge\ldots\wedge l_m\wedge u_1\wedge\ldots\wedge u_k \tag{2}$$

称不包含更新原子的责任 $l_1\wedge\ldots\wedge l_m$ 为条件责任,包含更新原子的责任 $l_1\wedge\ldots\wedge l_m\wedge u_1\wedge\ldots\wedge u_k(n\geqslant 1)$为更新责任.

定义 1(责任公式). 责任公式定义如下:

(1) 任何一个责任 $l_1\wedge\ldots\wedge l_m\wedge u_1\wedge\ldots\wedge u_k$ 都是一个责任公式.

(2) 如果 obl 是一个责任,那么 $X^i(obl)(i>0),F^{\leqslant i}(obl)(i>0),G(obl),G^{\leqslant i}(obl),GX^i(obl)(i>0),GF^{\leqslant i}(obl)(i>0)$也是责任公式.

(3) 如果 F 是上述形式(1)或形式(2)的责任公式,那么 $F\cup l_1\wedge\ldots\wedge l_m$ 也是责任公式.

简单起见,本文只考虑过去时态的结果,责任公式中不含过去时态操作符.在实际应用中,时态操作符也可以加上 S(秒)、M(分)、H(小时)和 D(日)等时间步单位,如 $X^{\mathrm{H}}T(i>0)$表示第 i 个小时.

定义 2(责任规则). 责任规则包含如下形式的两种规则:

(1) I 型责任规则

$$p^{\pm}(x_1,\ldots,x_n)\Rightarrow\varphi_1,\ldots,\varphi_m \tag{3}$$

其中:$\varphi_i(1\leqslant i\leqslant n)$是责任公式;$p(x_1,\ldots,x_n)$是内涵操作谓词原子,$p^+(x_1,\ldots,x_n)$表示 $p(x_1,\ldots,x_n)$开始执行时刻,$p^-(x_1,\ldots,x_n)$表示 $p(x_1,\ldots,x_n)$结束执行时刻.I 型规则说明,在 $p(x_1,\ldots,x_n)$开始执行或结束时刻需要接受 φ_1 到 φ_m 的责任.

(2) II 型责任规则

$$p^+(x_1,\ldots,x_n),\ p^-(x_1,\ldots,x_n)\Rightarrow\varphi_1,\ldots,\varphi_m \tag{4}$$

其中:$\varphi_i(1\leqslant i\leqslant n)$是责任公式;$p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)$表示 $p(x_1,\ldots,x_n)$从开始到结束的执行期.II 型责任规则说明,在执行 $p(x_1,\ldots,x_n)$期间内需要承担 φ_1 到 φ_m 的责任.

另外,称操作 $p(x_1,\ldots,x_n)$为责任规则的触发操作.责任规则说明,在执行 $p(x_1,\ldots,x_n)$的开始或结束时刻或执行期内需要接受 φ_1 到 φ_m 的责任.如下面的规则(5)说明,用户应在使用数字内容 D 结束后的 5 个时间步内缴费:

$$play^-(D)\Rightarrow F^{\leqslant 5}(pay(n)\wedge price(D,n)) \tag{5}$$

定义 3(补偿规则). 补偿规则包含如下两种形式的补偿规则:

(1) I 型补偿规则

$$\begin{aligned} &p^{\pm}(x_1,...,x_n)\Rightarrow\neg r,\ \varphi_1,...,\varphi_m\\ &p^+(x_1,...,x_n),\ p^-(x_1,...,x_n)\Rightarrow\neg r,\ \varphi_1,...,\varphi_m \end{aligned} \tag{6}$$

其中:$\varphi_i(1\leqslant i\leqslant n)$是责任公式;$p(x_1,\ldots,x_n)$是补偿规则的触发操作;$r$ 是除该规则外具有同一触发操作谓词的任意责任或补偿规则,并称该规则是 r 的补偿规则,r 是该规则的被补偿规则.I 型补偿规则说明,如果被补偿规则 r 中有责任公式被违反,则使用该规则的所有责任公式进行补偿.

(2) II 型补偿规则

$$\begin{aligned} &p^+(x_1,...,x_n)\Rightarrow\neg r::\varphi,\ \varphi_1,...,\varphi_m\\ &p^+(x_1,...,x_n),\ p^-(x_1,...,x_n)\Rightarrow\neg r::\varphi,\ \varphi_1,...,\varphi_m \end{aligned} \tag{7}$$

II 型补偿规则在 I 型补偿规则的基础上增加对被违反责任公式的说明,其中,φ是被补偿规则 r 的责任公式.II 型补偿规则说明,如果被补偿规则 r 中的责任公式φ被违反,则使用该规则的责任公式补偿φ.另外,称φ是该补偿规则的被补偿责任.

补偿规则说明,在被补偿规则中的责任被违反的情况下,应接受补偿规则中的责任.如下面的补偿规则(8)

中:r_1 说明数字内容 D 在使用期间应每个时间步付费一次(*pay_by_step*())并打开广告窗口(*openadvwindows*());r_2 是 I 型补偿规则,说明 r_1 任意责任的违反将会在 D 使用结束时刻发送报告给服务端;r_3 是 II 型补偿规则,说明如果未在使用期间每个时间步付费一次,则在使用结束时按使用次数付费(*pay_by_time*()).

$$\begin{aligned} r_1 &: play^+(D), play^-(D) \Rightarrow GX(pay\_by\_step()), openadvwindow() \\ r_2 &: play^-(D) \Rightarrow \neg r_1, sendreport(server) \\ r_3 &: play^-(D) \Rightarrow \neg r_1 :: GX(pay\_by\_step()), pay\_by\_time() \end{aligned} \tag{8}$$

定义 4(补偿树). 对任意责任规则 $r:p^+(x_1,\ldots,x_n)\Rightarrow\varphi_1,\ldots,\varphi_n$,如果以 r 作为根结点,r 的规则体中的责任公式作为责任子结点(包括∨结点),补偿规则作为被补偿责任结点的补偿子结点,其中,∨结点作为 I 型补偿规则父结点,则可建立如图 1 所示的补偿树 $r(t)$.

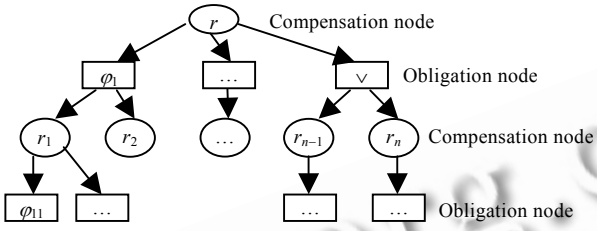

Fig.1 Compensation tree

图 1 补偿树

对规则 r 的补偿树 $r(t)$,如果 $r(t)$的任一补偿结点 r_i 都不存在祖先补偿结点 r_j,使得 $r_i=r_j$,则称补偿树 $r(t)$是分层的;如果补偿树的深度是有限的,则称 $r(t)$是有限分层补偿树.

不分层补偿树中存在循环引用,导致规则体无法评价(evaluation)[18].补偿树的层级如果不是有限的,规则体评价就无法结束,因而有限分层补偿树才是可评价的.

定义 5(规则集的语法正确性). 对任意规则集 O,如果任意责任规则 $o\in O$ 的补偿树均是有限分层的且不存在补偿规则 $p\in O$,而 p 不属于 O 中任何责任规则的补偿树,则称规则集 O 是语法正确的,也称规则集 O 是语法正确集.

对任意有限规则集,由于规则数量的有限性,责任规则的分层补偿树都是有限的.对系统假定变量有限集χ,常量有限集 C 和谓词符号集 $Q=Q_{ext}\cup Q_{int}$,其中,Q_{ext} 和 Q_{int} 分别是外延符号集和内涵符号集.OUC 责任模型的语法结构如图 2 所示.

Term: $t::=X|a$, $X\in\chi$, $a\in C$
Atom: $P_T::=p(t_1,\ldots,t_n)$, $p\in Q$, $T\in\{ext,int\}$
Update: $U::=+P_{ext}|-P_{ext}$
Literal: $L::=P_{int}|P_{ext}|\neg P_{ext}$
Authorization rule: $AR::=P_{int}\leftarrow L_1\wedge\ldots\wedge L_m\wedge U_1\wedge\ldots\wedge U_k$
…
Obligation formula: $OBL::=L_1\wedge\ldots\wedge L_m\wedge U_1\wedge\ldots\wedge U_k$
Obligation function:
$O::=OBL|X^i(OBL)|F^{\leqslant i}(OBL)|G(OBL)|G^{\leqslant i}(OBL)|GX^i(OBL)|GF^{\leqslant i}(OBL)$, $i\in\mathbf{N}$, $i>0$
$F::=O|O\cup L_1\wedge\ldots\wedge L_m$
Obligation rule:
$H::=p^+_{int}\mid p^-_{int}\mid p^+_{int},p^-_{int}$
$OR::=H\Rightarrow F_1,\ldots,F_n|H\Rightarrow\neg OR,F_1,\ldots,F_n|H\Rightarrow\neg OR::F,F_1,\ldots,F_n$

Fig.2 Syntactic structure of OUC obligation model

图 2 OUC 责任模型语法结构

1.2 语 义

1.2.1 责任执行域

定义 6(终结算子ξ). 对责任规则或补偿规则 r,定义 r 的终结算子ξ返回 r 的触发操作 $p(x_1,\ldots,x_n)$终结时距离开始时刻的时间步,也即 $p(x_1,\ldots,x_n)$执行的时间步.

终结算子ξ应用在规则中,增加规则的时态表达力.如下述规则(9)说明,播放 D 必须在播放结束之前付费.

$$play^+(D),\ play^-(D)\Rightarrow F^{\leqslant\xi}(pay()) \tag{9}$$

定义 7(补偿算子ω). 对补偿规则 r_2,r_1 是 r_2 的被补偿规则,定义 r_2 的补偿算子ω返回 r_1 被违反时的当前时刻.

补偿算子ω用来说明补偿规则的责任执行域.

对于 I 型责任规则,责任执行域的起始步在操作 $p(x_1,\ldots,x_n)$的执行开始或结束时刻,不存在执行域的终止步,也即执行域为$[p^+(x_1,\ldots,x_n),\infty)$或$[p^-(x_1,\ldots,x_n),\infty)$.II 型规则的责任执行域为$[p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)]$.

对补偿规则,责任执行域的起始步应从补偿规则被激发时刻(ω时刻)开始,也即$[\omega,\infty)$与上述责任执行域的交集,如对 I 型责任规则分别为$[\omega,\infty)$或$[\max(\omega,p^-(x_1,\ldots,x_n)),\infty]$,max 函数取最大数作为起始步.II 型补偿规则的责任执行域为$[\omega,\infty]\cap[p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)]$.

在责任规则和补偿规则中,责任公式均以责任执行域的起始步作为责任执行的起始步,如以下规则:

$$p^+(x_1,\ldots,x_n),\ p^-(x_1,\ldots,x_n)\Rightarrow X^{30}(sendreport()),... \tag{10}$$

上述规则(10)的执行域为$[p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)]$,操作符 X^{30} 的起始步从 $p^+(x_1,\ldots,x_n)$执行的时间步开始,即 X^{30} 是指从 $p(x_1,\ldots,x_n)$开始执行时间步后的第 30 个时间步.

下面的规则(11)说明,数字内容 D 的使用规则是每隔 10 分钟收费一次,每隔 15 分钟打开一次广告窗口.如果规则被违反,数字内容 D 就将立即停止播放.

$$\begin{aligned} r_1&: play^+(D),play^-(D)\Rightarrow GX^{10M}(pay()),GX^{15M}(openadvwindow())\\ r_2&: play^+(D),play^-(D)\Rightarrow r_1,-play\_a() \end{aligned} \tag{11}$$

其中,补偿规则 r_2 的责任执行域应为$[\omega,p^-(x_1,\ldots,x_n))$,也即退出播放操作–*play_a*()应在 r_1 被违反的当前时间步立即执行.

定义 8(责任公式违反). 对在责任执行域内的责任公式$\varphi=\sigma(obl)$,如果在时态操作符σ表示的当前时刻,责任 *obl* 不能在当前程序 P 中被成功地执行(p 可满足 *obl* 且 *obl* 引发的更新是一致的[18]),则称责任公式被违反.

责任公式违反只发生在责任执行域内,故如果上述规则(10)中如果未达到 30 个时间步 $p(x_1,\ldots,x_n)$已结束,则 $X^{30}(sendreport())$不需要执行.

1.2.2 基本规则语义

如图 2 所示的责任公式包括 8 种类型,称只包含单个责任公式的责任规则为基本规则.基本规则包括如下 8 种类型,其中,*obl* 表示责任:

基本规则 1. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow obl$ 和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow obl$.

基本规则 1 中不含时态操作符,说明在责任执行域的起始步有执行 *obl* 的义务.

例 1:数字内容 D 开始播放时应打开广告网页(http://www.advertise.com),结束播放时应记录日志:

$$\begin{aligned} r_1&: play^+(D)\Rightarrow open(\text{http://www.advertise.com}),\\ r_2&: play^-(D)\Rightarrow log(). \end{aligned}$$

基本规则 2. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow X^i(obl)$和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow X^i(obl),i>0$.

基本规则 2 说明,在规则责任执行域的第 i 个时间步有执行责任 *obl* 的义务.

例 2:数字内容 D 在下载后必须在第 30 个时间步发送报告到服务端:

$$r_1{:}download^-(D)\Rightarrow X^{30}(sendreport()).$$

基本规则 3. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow F^{\leqslant i}(obl)$和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow F^{\leqslant i}(obl),i>0\vee i=\infty$.

基本规则 3 说明,在规则责任执行域的第 i 个时间步内有执行责任 *obl* 的义务.

例 3:数字内容 D 在使用开始后的第 10 个时间步内必须被删除:

$$r_1{:}play^+(D)\Rightarrow F^{\leqslant 10}(delete(D)).$$

基本规则 4. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow G(obl)$和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow G(obl)$.

基本规则 4 说明,在规则责任执行域的任何时间步均有执行责任 *obl* 的义务.在基本规则 4 中,一般情况下,*obl* 应是条件责任,说明规则在责任执行域内应满足 *obl* 的条件.

例 4:数字内容 *D* 在使用前应打开广告网页,使用期间必须保持广告网页打开:

$$r_1 : play^+(D) \Rightarrow open(\text{http://www.advertise.com}),$$

$$r_2 : play^+(D), play^-(D) \Rightarrow G(is\_open(\text{http://www.advertise.com})).$$

其中,谓词 *open*(http://www.advertise.com)打开网页,*is_open*(http://www.advertise.com)判断网页是否打开.

基本规则 5. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow G^{\leqslant i}(obl)$和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow G^{\leqslant i}(obl),i>0$.

基本规则 5 说明,在规则责任执行域 *i* 个时间步内的任意时刻都有执行责任 *obl* 的义务.一般情况下,*obl* 应是条件责任,说明规则在责任执行域的 *i* 个时间步内应满足 *obl* 的条件.

例 5:免费试用的数字内容 *D* 要求下载后至少连续打开广告网页 10 个时间步:

$$r_1{:}download^-(D)\Rightarrow G^{\leqslant 10}(is\_open(\text{http://www.advertise.com})).$$

基本规则 6. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow GX^i(obl)$和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow GX^i(obl),i>0$.

基本规则 6 说明,在规则责任执行域内总是每隔 *i* 个时间步有执行责任 *obl* 的义务.

例 6:数字内容 *D* 在播放期间必须每隔 5 个时间步付费 1 次:

$$r_1{:}play^+(D),play^-(D)\Rightarrow GX^5(pay()).$$

基本规则 7. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow GF^{\leqslant i}(obl)$和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow GF^{\leqslant i}(obl),i>0$.

基本规则 7 说明,在规则责任执行域内有每隔 *i* 个时间步内执行责任 *obl* 的义务.

例 7:数字内容 *D* 的使用需要使用口令密码,用户应至少在每 30 个时间步内改变一次口令密码:

$$r_1{:}play^-(D)\Rightarrow GF^{\leqslant 30}(changepassword()).$$

基本规则 8. $p^{\pm}(x_1,\ldots,x_n)\Rightarrow O\cup L_1\wedge\ldots\wedge L_m$ 和 $p^+(x_1,\ldots,x_n),p^-(x_1,\ldots,x_n)\Rightarrow O\cup L_1\wedge\ldots\wedge L_m$.

O 的定义见图 2,基本规则 8 说明,在规则责任执行域内有在 $L_1\wedge\ldots\wedge L_m$ 状态成立之前按照责任公式 *O* 执行责任的义务.

例 8:在下载数字内容 *D* 后,用户在删除 *D* 之前有每隔 10 个时间步付费的义务,责任规则如下:

$$r_1{:}download^-(D)\Rightarrow GX^{10}(pay())\cup(\neg object(D)).$$

上述 r_1 说明,在对象 *D* 不存在(删除)之前有每隔 10 个时间步付费的义务.

OUC 责任模型中的所有责任规则和补偿规则均由上述 8 种基本规则组成,其语义是基本规则语义的复合.

1.2.3 合法补偿集

定义 9(谓词原子的相似性). 设 $p(x_1,\ldots,x_n)$和 $p(y_1,\ldots,y_m)$分别是谓词原子,对 $p(x_1,\ldots,x_n)$的任意常量项 $x_i(1\leqslant i\leqslant n)$,$p(y_1,\ldots,y_m)$中的相应项 $y_i=x_i$,反之亦然,则称 $p(x_1,\ldots,x_n)$和 $p(y_1,\ldots,y_m)$是相似的.

假定 *X*,*Y* 和 *Z* 分别表示变量,那么谓词原子 *p*(*Z*,5,*Y*)和 *p*(*X*,5,*X*)是相似的,*p*(*X*,5,*Y*)和 *p*(*X*,3,*Y*)由于常量不相等而不是相似的.

操作原子的相似性是为了保证补偿树的可实例化.如下定义补偿树的实例化条件:

定义 10(可实例化补偿树). 对任意合法规则集 *O*,*r*(*t*)是责任规则 $r\in O$ 的补偿树.如果对 *r*(*t*)上的任意补偿结点上的补偿规则 $c\in O$,$p(x_1,\ldots,x_n)$和 $p(y_1,\ldots,y_m)$分别是 *c* 和 *r* 规则的触发操作原子,均有 $p(x_1,\ldots,x_n)$和 $p(y_1,\ldots,y_m)$是相似的,则称补偿树 *r*(*t*)是可实例化的.

定义 10 是保证补偿树上的补偿规则之间可通代(unifiable)的必要条件,谓词原子的相似性说明,谓词原子除了项变量的名称不同和原子名称可不相同以外,其余都相同,因而对可实例化补偿树.易证如下定理:

定理 1(补偿树通代定理). 对任意合法规则集 *O*,责任规则 $r\in O$ 的补偿树 *r*(*t*)是可实例化补偿树.假定 $p(x_1,\ldots,x_n)$是 *r* 规则头的操作谓词原子,对任意谓词原子 $p(y_1,\ldots,y_m)$,如果存在最广通代符 θ 使得 $\theta p(x_1,\ldots,x_n)=p(y_1,\ldots,y_m)$,则对 *r*(*t*)补偿结点上的任意补偿规则 $c\in O$,其中,$q(z_1,\ldots,z_n)$是 *c* 规则头的操作谓词原子.必定存在最广

通代符θ_1,使得$\theta_1(z_1,\ldots,z_n)=(y_1,\ldots,y_m)$.

定义 11(合法补偿集和合法规则集). 对任意规则集 O,如果任意责任规则 $o\in O$ 的补偿树都是可实例化补偿树,则称规则集 O 是合法补偿集.另外,如果规则集 O 既是语法正确集又是合法补偿集,则称 O 是合法规则集.

在下面的责任实施机制中,均要求规则集是合法补偿集和合法规则集.

2 责任实施算法

定理 2(补偿成立定理). 对任意合法补偿规则集 O,如果补偿规则 $c\in O$ 是责任规则 $r\in O$ 补偿树 $r(t)$上的补偿结点规则,则补偿规则 c 在补偿树 $r(t)$成立的必要条件是所有 c 在 $r(t)$树中的祖先责任结点中的责任公式均在当前时刻能够判断已被违反,其中,$\vee$结点被违反说明其父补偿结点中存在能够判断已被违反的责任公式.

补偿成立定理说明,任意补偿规则只有在所有祖先结点上相应的责任均已被违反的前提下才能成立.由定理 2,补偿过程按照补偿树由顶自下进行,主要步骤如下:

(1) 如果责任公式φ被违反,那么φ停止运行,假定φ属于规则 r.

(2) 如果 r 的补偿树中结点φ只包含 I 型补偿规则 r_1,则用 r_1 的责任公式替代φ.如果存在多个 I 型补偿规则,则由用户选择一条补偿规则.

(3) 如果 r 的补偿树中结点φ只包含 II 型补偿规则 r_1,则用 r_1 的责任公式替代 r 当前运行的所有责任公式;如果存在多个 II 型补偿规则,则由用户选择补偿规则.

(4) 如果 r 的补偿树中结点φ包含 I 型和 II 型多个补偿规则,则由用户选择相应的补偿方法.

每个操作实例包括一个工作区域,工作区域包含责任区域和补偿区域两部分.责任区域表示用户正在承担的责任,用 $p^{\pm}(x_1,\ldots,x_n)=\delta(\varphi_1\wedge\ldots\wedge\varphi_m,ts)[r]$或 $p^{+}(x_1,\ldots,x_n),p^{+}(x_1,\ldots,x_n)=\delta(\varphi_1\wedge\ldots\wedge\varphi_m,ts)[r]$表示,其中,$r$ 是引发责任的规则,ts 是责任执行区域起始步的时刻.$\varphi_1\wedge\ldots\wedge\varphi_m$ 是用户需要承担责任公式的合取式,δ函数用起始步 ts 时刻实例化$\varphi_1\wedge\ldots\wedge\varphi_m$ 中的相应时态操作符.补偿区包括补偿责任或补偿规则:补偿责任用实例化δ公式表示,代表用户正在承担的责任;补偿规则表示按照补偿成立定理用户需要承担该补偿规则的责任,但该补偿规则的责任执行区域尚未激活($p^{-}(x_1,\ldots,x_n)$尚未执行),当该补偿规则的责任区域激活时,将该补偿规则实例化为δ公式.

设 R 表示责任规则集,C 代表补偿规则集,操作 $p(a_1,\ldots,a_n)$是实例化的基原子.责任实施算法过程如下:

(1) 操作 $p^{+}(a_1,\ldots,a_n)$发生.当用户开始 $p(a_1,\ldots,a_n)$操作时,产生 $p^{+}(a_1,\ldots,a_n)$操作触发,包含以下步骤:

① R 集中存在 I 型责任规则 $r{:}p^{+}(x_1,\ldots,x_n)\Rightarrow\varphi_1,\ldots,\varphi_m\in R$ 且存在最广通代符θ使得$\theta p(x_1,\ldots,x_n)=p(a_1,\ldots,a_n)$.将 $p^{+}(a_1,\ldots,a_n)=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$加入责任区域,其中,$ts$ 是 $p^{+}(a_1,\ldots,a_n)$发生的时刻.如果 R 中存在多个满足条件的责任规则,则由系统提供用户选择以确定最终加入责任区域的δ公式.

② 同样,对 R 集中存在的 II 型责任规则 $r{:}p^{+}(x_1,\ldots,x_n),p^{-}(x_1,\ldots,x_n)\Rightarrow\varphi_1,\ldots,\varphi_m\in R$ 作上述过程①的处理并将选定的δ公式加入 $p^{+}(a_1,\ldots,a_n),p^{-}(a_1,\ldots,a_n)=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$加入责任区域.

(2) 操作 $p^{-}(a_1,\ldots,a_n)$发生.当用户退出 $p(a_1,\ldots,a_n)$操作时,产生 $p^{-}(a_1,\ldots,a_n)$操作触发,包含以下步骤:

① R 集中存在 I 型责任规则 $r{:}p^{-}(x_1,\ldots,x_n)\Rightarrow\varphi_1,\ldots,\varphi_m\in R$ 且存在最广通代符θ使得$\theta p(x_1,\ldots,x_n)=p(a_1,\ldots,a_n)$.将 $p^{-}(a_1,\ldots,a_n)=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$加入责任区域,其中,$ts$ 是 $p^{-}(a_1,\ldots,a_n)$发生的时刻.如果 R 中存在多个满足条件的责任规则,则由系统提供用户选择以确定最终加入责任区域的δ公式.

② 如果补偿区中存在 $r{:}p^{-}(a_1,\ldots,a_n)\Rightarrow\theta\varphi_1,\ldots,\theta\varphi_m$ 形式的规则,则将该规则转换成δ公式 $p^{-}(a_1,\ldots,a_n)=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$形式加入补偿区域,其中,$ts$ 是 $p^{-}(a_1,\ldots,a_n)$发生时刻.

③ 如果责任区或补偿区存在 $p^{+}(a_1,\ldots,a_n),p^{-}(a_1,\ldots,a_n)=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$形式的$\delta$公式,则将该$\delta$公式转换成结束$\delta$公式 $p^{+}(a_1,\ldots,a_n),p^{-}(a_1,\ldots,a_n)=\delta(\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts),es)[r]$,其中,外围$\delta$函数用结束步 es 时刻实例化$\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)$中的相应时态操作符.

④ 对当前操作实例工作区域中所有包含 $p(a_1,\ldots,a_n)$操作公式的终结算子ξ,均使用当前时间步实例化.

(3) 每个时间步的责任执行.

① 在当前时间步按照责任区域和补偿区域中的δ公式 $H=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$执行在当前时间步应执

行的责任,包括以下情形:

1) 当前时间步成功执行责任公式$\theta\varphi_i(1\leqslant i\leqslant n)$的责任.如果该责任公式未完成,则继续保持该责任公式;如果该责任公式已完成,则从δ公式中删除该责任公式;如果该责任公式是该δ公式中的最后责任公式,则将该δ公式也从当前区域删除.
2) 如果当前时间步未成功执行责任公式$\theta\varphi_i(1\leqslant i\leqslant n)$的责任,则由用户在规则 r 的实例树上选择(如果存在多条)补偿$\theta\varphi_i$的 II 型补偿规则或补偿 r 的 I 型补偿规则.如果选择的补偿规则 r_1 包括责任公式$\gamma_1,\ldots,\gamma_k$,则按照 r_1 的规则头将δ公式 $H=\delta(\gamma_1\wedge\ldots\wedge\gamma_n,ts)[r\xrightarrow{\theta\varphi_i}r_1]$插入补偿区,其中,$ts$ 是当前时刻.但如果 r_1 的规则头是$p^-(a_1,\ldots,a_n)$且$p^-(a_1,\ldots,a_n)$尚未执行,则直接将规则$p^-(a_1,\ldots,a_n)\Rightarrow\gamma_1\wedge\ldots\wedge\gamma_n[r\xrightarrow{\theta\varphi_i}r_1]$插入补偿区.最后,如果用户选择的是 I 型补偿规则,则将 $p^+(a_1,\ldots,a_n)=\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts)[r]$从责任区域删除;否则,只删除$\theta\varphi_i$.
3) 如果当前时间步存在多个同时执行的责任发生更新冲突[18],则可按照一定的冲突解决策略执行,如撤销所有的责任公式执行或由用户选择责任公式执行或按事先确定的执行策略,撤销的责任公式按照上述步骤 2)进行补偿操作.
4) 执行上述步骤 2)和步骤 3)产生的补偿责任,如果在当前时刻的执行责任多次被违反,则一直执行补偿处理过程,直到补偿树的叶结点.

② 在当前时间步按照责任区域和补偿区域中的结束δ公式 $p^+(a_1,\ldots,a_n),p^-(a_1,\ldots,a_n)=\delta(\delta(\theta\varphi_1\wedge\ldots\wedge\theta\varphi_m,ts),es)[r]$执行当前时间步应执行的责任,执行方法如上述①,然后从工作区域中删除该公式.

3 应用和实现示例

假定数字内容 D 试用的相关责任政策:

(1) 数字内容 D 的使用需要使用口令密码,用户在下载数字内容后至少应每隔 30 天改变一次口令密码.
(2) 数字内容 D 在下载后的 120 天内必须被删除,数字内容删除后用户应立即通知服务端.
(3) 用户在数字内容 D 被删除前必须每隔 10 天发送报告给服务端,报告数字内容使用情况(使用次数).
(4) 用户在使用数字内容 D 的同时必须打开广告网页(http://www.advertise.com),并在使用期间保持广告网页的打开,否则将导致数字内容 D 的关闭.
(5) 数字内容 D 的付费方式有 3 种:按使用期间每个时间步付费一次,或在使用结束时按次付费,否则最迟也应在使用结束的 10 个时间步内付费并增加额外的滞纳金,否则用户将会被取消会员资格.

上述责任政策按如图 3 所示的方式表达.

r_0: $download^-(D)\Rightarrow GF^{\leqslant 30D}(changepassword()),F^{\leqslant 120D}(delete(D)\wedge notice(server)),GX^{10D}(sendreport())\cup(\neg object(D))$
r_1: $play^+(D)\Rightarrow open(\text{http://www.advertise.com})$
r_2: $play^+(D),play^-(D)\Rightarrow G(is\_open(\text{http://www.advertise.com})),GX(pay\_by\_step())$
r_3: $play^+(D),play^-(D)\Rightarrow\neg r_2::G(is\_open(\text{http://www.advertise.com})),\neg play\_a(D)$
r_4: $play^-(D)\Rightarrow\neg r_2::GX(pay\_by\_step()),pay\_by\_time()$
r_5: $play^-(D)\Rightarrow\neg r_4,F^{\leqslant 10}(pay\_with\_extra())$
r_6: $play^-(D)\Rightarrow\neg r_5,demember()$

Fig.3 An example of obligation policy

图 3 责任政策示例

责任执行的过程如下:

(1) 用户首先执行 $download(D)$从服务器下载 D,在下载完毕后,当前责任区域如下(当前时刻假定是 0):

$$E_1:download^-(D)=\delta(GF^{\leqslant 30D}(changepassword())\wedge F^{\leqslant 120D}(delete(D)\wedge notice(server))\wedge GX^{10D}(sendreport())\cup(\neg object(D)),0)[r_0].$$

(2) 在每个时间步,系统检查上述δ公式执行相应的责任.如果相应的责任执行完毕,则从δ公式中删除该责任.假定用户在时刻 10 开始执行播放操作时,当前责任区域如下:

$$E_1{:}download^{-}(D)=\delta(GF^{\leqslant 30D}(changepassword())\wedge F^{\leqslant 120D}(delete(D)\wedge notice(server))\wedge GX^{10D}(sendreport())\cup(\neg object(D)),0)[r_0],$$

$$E_2{:}play^{+}(D)=\delta(open(\text{http://www.advertise.com}),10)[r_1],$$

$$E_3{:}play^{+}(D),play^{-}(D)=\delta(G(is\_open(\text{http://www.advertise.com})\wedge GX(pay\_by\_step()),10)[r_2].$$

(3) 假定用户在进入执行 D 的当前时间步完成 E_2 的责任并删除该规则,同时假定用户在执行期间未执行按时间步付款的责任,那么将执行补偿责任 r_4.当前责任区域(E)和补偿区域(C)如下(其中,C_1 由于 $play^{-}(D)$尚未执行,因而以规则的形式表示):

$$E_1{:}download^{-}(D)=\delta(GF^{\leqslant 30D}(changepassword())\wedge F^{\leqslant 120D}(delete(D)\wedge notice(server))\wedge GX^{10D}(sendreport())\cup(\neg object(D)),0)[r_0],$$

$$E_3{:}play^{+}(D),play^{-}(D)=\delta(G(is\_open(\text{http://www.advertise.com})),10)[r_2],$$

$$C_1{:}play^{-}(D)\Rightarrow pay\_by\_time()[r_2 \xrightarrow{GX(pay\_by\_step())} r_4].$$

(4) 假定用户在时刻 15 结束播放 D,将触发上述 C_1 的按次付款补偿规则转换成δ公式并执行.如果用户在结束播放 D 时未执行按次付款,将导致补偿规则 r_5 的执行,则当前责任区域和补偿区域如下:

$$E_1{:}download^{-}(D)=\delta(GF^{\leqslant 30D}(changepassword())\wedge F^{\leqslant 120D}(delete(D)\wedge notice(server))\wedge GX^{10D}(sendreport())\cup(\neg object(D)),0)[r_0],$$

$$C_1{:}play^{-}(D)=\delta(F^{\leqslant 10}(pay\_with\_extra()),15)[r_2 \xrightarrow{GX(pay\_by\_step())} r_4\rightarrow r_5].$$

(5) 如果用户在播放结束 10 个时间步内没有付费并增加额外滞纳金,将导致补偿规则 r_6 的执行(注:这时的补偿算子ω返回时刻 25).当前责任区域和补偿区域如下:

$$E_1{:}download^{-}(D)=\delta(GF^{\leqslant 30D}(changepassword())\wedge F^{\leqslant 120D}(delete(D)\wedge notice(server))\wedge GX^{10D}(sendreport())\cup(\neg object(D)),0)[r_0],$$

$$C_1{:}play^{-}(D)=\delta(demember(),25)[r_2 \xrightarrow{GX(pay\_by\_step())} r_4\rightarrow r_5\rightarrow r_6].$$

(6) 最后,r_6 执行后将导致用户会员资格被取消,当前责任区域返回到上述过程(1)的责任区域.

4 模型表达力

在表达力上,我们已经说明 LucScript 能够表达 UCON 的所有基本模型[19].本文在 LucScript 模型的基础上引入时态逻辑,进一步加强了 LucScript 语言的表达力.通过扩展的分布式时态操作符,OUC 模型能够表达完整的时态因素;通过责任规则的触发操作和责任执行域概念,OUC 模型能表达责任的事件驱动性和持续概念;通过补偿规则,OUC 模型能够表达复杂的补偿概念.

Marco 等人[20]在讨论企业数据隐私保护的责任处理中,将企业的数据隐私保护分为长期的隐私责任、短期的隐私责任和正在执行(ongoing)的隐私责任 3 类.Marco 提出责任驱动包括:(1) 时态驱动,如由特定的日期和时间或固定周期等触发责任的承担;(2) 使用驱动,如在一定使用次数后应承担一定的责任;(3) 环境事件驱动,如由数据的使用、传输和删除等操作事件触发责任的承担.Marco 提出的方法易于使用本文的方法来表达,这也充分说明了本文方法的表达力.Marco 使用的责任触发方法以及使用本文方法的表达方式见表 1.

5 结束语

OUC 模型存在表达完整的责任表达和实施机制,能够表达时态驱动、事件驱动和责任补偿等因素,并且将责任规则与访问控制规则分开运行和表达.责任的实施需要满足访问控制规则的要求.OUC 模型建立了与访问控制相关但独立运行的责任表达和实施机制.

在 LucScript 体系中,OUC 模型作为数字内容使用控制的补充和并行部分,部署在客户端作为 LucScript 语言解释器的补充部分.LucScript 客户端控制器包含可信部分和不可信部分.可信部分(可信计算基)包括 LucScript 的解释器、许可证的可信存储.控制器可信部分可与硬件令牌(如智能卡、USBKey 等)绑定发布.控制器拥有自己的公私钥对,公钥保存在硬件令牌中,私钥保存在可信部分.硬件令牌对软件的可信部分的完整性进

行验证.公私钥对用于控制器与服务端的通信和验证.限于篇幅(类似方法可参考文献[21]),在此不再赘述.

本文的不足之处在于,在补偿责任方面,要求补偿责任的触发操作谓词与被补偿规则的触发操作谓词相同,未考虑不同操作之间的补偿问题.该问题可以作为本文的后续研究方向.

Table 1 Obligation types of enterprise privacy and mode of expression

表 1 企业隐私责任类型和表达方式

Obligation types	Events triggering obligations		Expression of the paper
Long-Term privacy obligations	Time-Driven	*T* being executed at a specific date and time	X^iT, *T* being executed at *i*-th time steps.
		T being executed after a certain period of time	$G^{\leqslant i}T'$, here *T'* can mean the action that keeps the status of the data before operation. For example, $G^{\leqslant 3D}(object(D))$ means that the object *D* only can be deleted after three days.
	Driven by usage and counters	*T* being executed after the data has being used for a certain number of times in a specific time frame	$G^{\leqslant i}T'$, here *T'* can mean that the usage times of users are lower than *n* where usage times are recorded by the licenses, then $G^{\leqslant i}T'$ means the usage times of the users must be lower than *n* during *i* time steps, or else a compensation rule can be put to execute *T* immediately.
Ongoing privacy obligations	Time-Driven	*T* being executed periodically (e.g. every month)	$GX^i(T)$, *T* always being executed every *i* time steps.
	Driven by contextual events	Such as when the data being used, transferred, deleted, retrieved or any actions predefined by the data subject	The obligation concepts at the time of or during the time of an operation can express the obligations driven by contextual events distinctly.
	Others	Such as data access being stopped or obligation update being triggered when the privacy policies changed	By the triggering rules in LucScript language or triggering methods in the obligation model, the method that data access can be stopped or obligation update can be triggered when the privacy policies change can be implemented.
Short-Term and interactive privacy obligations	Driven by time or context or others discussed above	Such as obligations needed to be executed immediately or interactive obligations needed to be participated by users	The concept of obligations being executed immediately can be expressed by the basic rule 1 without temporal operators. The execution mechanism of the model and the open management mechanism of LucScript both can express and implement short-term and interactive privacy obligations.

References:

[1] Sandhu R, Park J. Usage control: a vision for next generation access control. In: Gorodetski V, Popyack L, Skormin V, eds. Proc. of the MMM-ACNS-2003. LNCS 2776, Heidelberg: Springer-Verlag, 2003. 17−31.

[2] Jajodia S, Samarati P, Sapino ML, Subrahmanian VS. Flexible support for multiple access control policies. ACM Trans. on Database System, 2001,26(2):214−260. [doi: 10.1145/383891.383894]

[3] Bertino E, Bettini C, Ferrari E, Samarati P. An access control model supporting periodicity constraints and temporal reasoning. ACM Trans. on Database Systems, 1998,23(3):231−285. [doi: 10.1145/293910.293151]

[4] Cholewka DG, Botha RA, Eloff JHP. A context-sensitive access control model and prototype implementation. In: Qing SH, Eloff JHP, eds. Proc. of the IFIP TC11 15th Annual Working Conf. on Information Security for Global Information Infrastructures. Deventer: Kluwer Academic Publisher, 2000. 341−350.

[5] Pucella R, Weissman V. Reasoning about dynamic policies. In: Walukiewicz I, ed. Proc. of the FoSSaCS 2004. LNCS 2987, Heidelberg: Springer-Verlag, 2004. 453−467.

[6] Bettini C, Jajodia S, Wang XS, Wijesekera D. Provisions and obligations in policy management and security applications. In: Bernstein PA, Loannidis YE, Ramakrishnan R, eds. Proc. of the 28th Int'l Conf. on Very Large Data Bases. Hong Kong: Morgan Kaufmann Publishers, 2002. 502−513.

[7] Hilty M, Basin D, Pretschner A. On obligations. In: Capitani VS, Syverson P, Gollmann D, eds. Proc. of the 10th European Symp. on Research in Computer Security. LNCS 3679, Heidelberg: Springer-Verlag, 2005. 98−117.

[8] Irwin K, Yu T, Winsborough WH. On the modeling and analysis of obligations. In: Rebecca NW, Sabrian CV, Vitaly S, eds. Proc. of the 13th ACM Conf. on Computer and Communication Security. New York: ACM Press, 2006. 134−143.

[9] Dougherty DJ, Fisler K, Krishnamurthi S. Obligations and their interaction with programs. In: Joachim B, Javier L, eds. Proc. of the 12th European Symp. on Research In Computer Security. LNCS 4734, Heidelberg: Springer-Verlag, 2007. 375−389.

[10] Barth A, Datta A, Mitchell JC, Nissenbaum H. Privacy and contextual integrity: Framework and application. In: Proc. of the 27th IEEE Symp. on Security and Privacy. New York: IEEE Press, 2006. 184−198.

[11] Hilty M, Pretschner A, Basin D, Schaefer C, Walter T. A policy language for distributed usage control. In: Joachim B, Javier L, eds. Proc. of the 12th European Symp. on Research in Computer Security. LNCS 4734, Heidelberg: Springer-Verlag, 2007. 531−546.

[12] Daoampi N, Dulay N, Lupu E, Sloman M. The ponder policy specification language. In: Broy M, Jahnichen S, eds. Proc. of the Policy 2001. LNCS 1995, Heidelberg: Springer-Verlag, 2001. 18−39.

[13] Lobo J, Bhatia R, Naqvi S. A policy description language. In: Proc. of the 16th National Conf. on Artificial Intelligence. Menlo Park: American Association for Artificial Intelligence, 1999. 291−298.

[14] Gama P, Ferreira P. Obligation policies: An enforcement platform. In: Proc. of the 6th IEEE Int'l Workshop on Policies for Distributed Systems and Networks. New York: IEEE Press, 2005. 203−212.

[15] Sailer M, Morciniec M. Monitoring and execution for contract compliance. Technical Report, HPL-2001-261R1, Bristol: Hewlett Packard Laboratories, 2001. http://www.hpl.hp.com/techreports/2001/HPL-2001-261R1.html

[16] Skene J, Skene A, Crampton J, Emmerich W. The monitorability of service-level agreements for application-service provision. In: Proc. of the 6th Int'l Workshop on Software and Performance. New York: ACM Press, 2007. 3−14.

[17] Zhong Y, Qin XL, Liu FY. Logical implementation mechanism for ODRL rights expression language. Computer Science, 2009, 36(4):133−139 (in Chinese with English abstract).

[18] Bertino E, Catania B, Gervasi V, Raffaetà A. Active-U-Datalog: Integrating active rules in a logical update languages. In: Decker H, Freitag B, Kifer M, Voronkov A, eds. Proc. of the Int'l Seminar on Logic Databases and the Meaning of Change. LNCS 1472, Heidelberg: Springer-Verlag, 1998. 107−133.

[19] Zhong Y, Qin XL, Zheng JP, Lin DM. A flexible usage control authorization language framework. Chinese Journal of Computers, 2006,29(8):1408−1418 (in Chinese with English abstract).

[20] Mont MC. Dealing with privacy obligations in enterprises. Technical Report, HPL-2004-109, Bristol: Hewlett Packard Laboratories, 2004. http://www.hpl.hp.com/techreports/2004/HPL-2004-109.pdf

[21] Chong CN, Ren B, Doumen J, Etalle S, Hartel PH, Corin R. License protection with a tamper-resistant token. In: Chae HL, Moti Y, eds. Proc. of the 5th Int'l Workshop Information Security Applications. LNCS 3325, Heidelberg: Springer-Verlag, 2004. 223−227.

附中文参考文献:

[17] 钟勇,秦小麟,刘凤玉.ODRL 权利描述语言逻辑实施机制研究.计算机科学,2009,36(4):133−139.

[19] 钟勇,秦小麟,郑吉平,林冬梅.一种灵活的使用控制授权语言框架.计算机学报,2006,29(8):1408−1418.

钟勇(1970－),男,江西峡江人,博士,副教授,主要研究领域为访问控制,数字版权保护,数据库安全,信息安全.

秦小麟(1953－),男,教授,博士生导师,CCF 高级会员,主要研究领域为安全数据库,时空数据库,信息安全.

刘凤玉(1943－),女,教授,博士生导师,主要研究领域为网络性能保护,信息安全.