一种可扩展访问控制标记语言的策略优化算法

卢秋如 陈建平 马海英 陈韦旭
(南通大学计算机科学与技术学院 南通 226019)

摘 要 可扩展访问控制标记语言XACML得到了广泛应用。为提高XACML策略的评估效率,提出一种基于韦恩图法的XACML策略优化算法。将XACML策略规则的组成结构用集合论中的韦恩图表示,在设定合并算法优先级的基础上,借助集合间的交并关系,检测和消除策略规则间的冲突与冗余,提高策略评估效率。实验测试表明,该算法在各主流引擎下将请求评估时间平均缩短10%~20%,同时能减少占用的存储空间,达到策略优化的目的。

关键词 访问控制,XACML,策略评估,韦恩图
中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.12.022

Optimization Algorithm for Extensible Access Control Markup Language Policies

LU Qiu-ru CHEN Jian-ping MA Hai-ying CHEN Wei-xu
(School of Computer Science and Technology, Nantong University, Nantong 226019, China)

Abstract Extensible access control markup language XACML is widely used. To improve the efficiency of XACML policy evaluation, an XACML policy optimization algorithm based on Venn graphic method was proposed. The XACML policy and rule structure are expressed as the Venn diagrams in the set theory. On the basis of setting the combination algorithm priorities, the conflicts and redundancies among the policies and rules are detected and eliminated according to the intersection and union relations between the sets. The experimental tests show that the algorithm reduces the evaluation time by 10% to 20% for the mainstream engines and decreases the occupied memory space at the same time, which hence achieves the purpose of the policy optimization.

Keywords Access control, XACML, Policy evaluation, Venn diagram

1 引言

访问控制是保障网络信息安全的一项关键技术,它使得数据资源能够在合法范围内得到有效管理和使用。作为一种描述安全访问控制策略的标准语言,可扩展访问控制标记语言XACML得到了广泛应用。XACML[1]不仅可以描述访问控制策略,而且具有策略执行机制,可用于访问控制决策的执行,当用户申请访问资源时,访问控制模块通过评估访问策略对用户进行授权。随着用户和资源数量的急剧增长,访问控制策略包含的规则越来越多,访问结构变得复杂,访问策略间的冗余与冲突问题愈加突出,从而影响到访问控制策略的评估和执行效率。因此,消减访问策略间的冗余与冲突,优化访问控制策略,对提高XACML的策略评估效率具有重要的意义。

在访问控制策略优化方面,Sun XACML开发包[2]提供了一个基于文件策略模式的判定器实例,但没有实现复杂的策略匹配的索引结构。文献[3-7]对策略引擎优化进行了研究。其中,Enterprise XACML[3]提供了一个索引结构,缩减了策略检索空间。XEngine[4]将规则属性数值化,在其转换的扁平结构上应用首次适用算法来优化策略匹配。MLOBEE[5]则在设定多级缓存的基础上,通过调整策略顺序来建立两级索引,以提高策略匹配效率。在策略优化上,Kolovski等人[8]使用描述逻辑(DL)一阶谓词对XACML策略进行逻辑形式化,采用DL验证分析工具检测策略中的规则冗余。Fisler等人[9]提出的策略变更影响分析工具也可用来分析规则冗余。Mourad等人[10-12]用语义来描述策略,提出了规则评估中出现的缺陷、冲突和冗余判定方法。王雅哲等人[13]分析了规则冗余的原因和规则冲突检测问题,提出了不同规则评估合并算法下的冗余判定定理。文献[14-16]围绕规则权重和统计度量进行了分析。以上主要从策略规则冗余和冲突的语义描述方面进行研究,对如何有效去除策略间的冗余并消解冲突尚未涉及。针对这一问题,本文提出一种冲突冗余检测与消解算法,该算法将XACML策略结构映射为集合,并用韦恩图表示。其利用韦恩图的直观性检测冲突和冗余,根据集合间

到稿日期:2016-12-04 返修日期:2017-03-15 本文受国家自然科学基金项目(61402244),南通市应用基础研究项目(GY2015012),南通大学自然科学基金项目(15z06)资助。

卢秋如(1990-),女,硕士生,主要研究方向为信息安全;陈建平(1960-),男,硕士,教授,主要研究方向为信息安全、数值计算,E-mail:chen_jp@ntu.edu.cn(通信作者);马海英(1977-),女,博士,副教授,主要研究方向为信息安全、密码学;陈韦旭(1992-),男,硕士生,主要研究方向为信息安全。

的交并关系去除冲突和冗余，以达到优化策略、提高评估效率的目的。

2　XACML 的相关概念

一个 XACML 策略由一个策略集和一个合并算法构成。XACML 策略集包含若干策略或其他策略集。一个策略由规则集组成，每个策略包含一个目标，该目标决定了该策略与请求的相关性。规则集则由若干条规则组成，规则中包含主体(Subject-Sj)、资源(Resource-Re)和动作(Action-Ac) 3 类基本属性以及适用该条规则的判定结果(Effect-E)。

XACML 支持 4 种策略/规则合并算法。针对一个请求、多个匹配的情况，XACML 利用策略/规则评估合并算法计算最终的评估结果。这 4 种典型的策略/规则合并算法为：允许优先(permit-override)，即只要有一条规则的评估为 permit，最终授权决策即为 permit；拒绝优先(deny-override)，即只要有一条规则的评估为 deny，最终授权决策即为 deny；首次适用(first-applicable)，即将遇到的第一条相关规则的评估结果作为最终授权决策的评估结果；唯一适用(only-one-applicable)，即将唯一一条适应请求规则的评估结果作为最终的授权决策。

XACML 虽然可以通过定义多种合并算法来约束决策结果，但仍容易导致复杂的冲突和冗余。策略评估的过程是根据用户发出的请求，依次匹配策略规则。如何快速匹配直接影响着策略评估的效率，尤其是在多次判定的情况下，能否快速匹配适用的策略规则尤为重要。

在运用上述合并算法时，XACML 允许设定算法的优先级。结合本文提出的冗余冲突消解算法，对上述 4 种合并算法设定相应的优先级：在允许优先条件下，允许的优先级最高，拒绝的优先级最低；反之，在拒绝优先条件下，拒绝的优先级最高，允许的优先级最低；首次适用始终优先于唯一适用。

根据上述设定，对于某一策略序列的优先级执行顺序如下：

$Queue_{ps}=\{P_1,P_2,\cdots,P_n\}$

其中，$prior(P_i)\geqslant prior(P_{i+1}),1\leqslant i\leqslant n-1$。

对于某一规则序列的优先级执行顺序如下：

$Queue_p=\{R_1,R_2,\cdots,R_n\}$

其中，$prior(R_i)\geqslant prior(R_{i+1}),1\leqslant i\leqslant n-1$。

3　XACML 策略的韦恩图表示

XACML 策略具有分层嵌套的结构，顶层是策略集(Policy Set)，包含一个或多个策略或子策略集，每个策略包含一个或多个规则(Rule)，规则中包含主体、资源、动作等属性。每一个策略集或策略都有合并算法，针对策略或规则之间的联系做出最后的决策。上层对下层具有包含关系，执行过程从底层开始，自底向上逐层进行。

从抽象的角度看，XACML 策略的每一层包含的是一组或多组数据，可以看作是集合，对于细粒度的属性值，会产生重叠或交叉关系(表现为策略和规则之间的冗余与冲突)，这种关系与集合之间的相交、相离、包含关系类似，可以采用集合理论中的韦恩图形象且直观地表示。XACML 策略的韦恩图表示如图 1 所示。

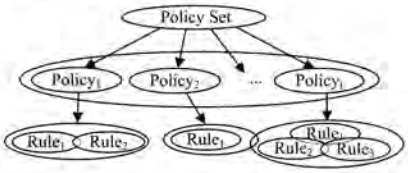


图 1　XACML 策略的韦恩图表示

4　基于韦恩图的策略优化算法

在将 XACML 策略用韦恩图表示后，策略和规则之间的冗余与冲突就表现为集合之间的包含与相交，从而可利用韦恩图法，根据集合之间的交并关系检测和消除冗余与冲突，进而减少参与匹配的策略条数，避免策略间的冲突，提高策略评估效率，优化评估过程。

策略评估中规则层与策略层之间存在规则合并算法，策略层与策略集层之间存在策略合并算法。本文重点针对允许优先和拒绝优先进行讨论。

在 XACML 环境中，相离的两集合的数据不会发生冲突或冗余，因此仅需针对策略集/规则集相交的情形进行讨论。冲突或冗余存在两种形式：真包含和部分相交，其韦恩图表示如图 2 所示。其中 E_i，E_j 表示对某一请求进行不同策略评估后返回的决策结果 $Effect$，$Effect$ 可取值 permit 或 deny。

(a)真包含　　(b)部分相交

图 2　用韦恩图描述冲突与冗余

根据上述集合关系，可以得知在真包含和部分相交的情况下如何判定冲突和冗余。当发生冲突时，这两种情况的决策结果不同，一种取值为 permit，另一种取值为 deny，并且二者存在交集关系；当发生冗余时，这两种情况的决策结果相同，两种取值同为 permit 或同为 deny，并且二者存在包含关系。冲突和冗余的检测与消解的具体表述如下。

4.1　冲突和冗余的检测

冲突检测的过程如图 3 所示(图中虚线的左侧部分为冲突的检测，右侧部分为冲突的消解)。对于同一请求所能够匹配的策略，当决策结果不同，即 $E_i\neq E_j$ 时，检测到冲突。

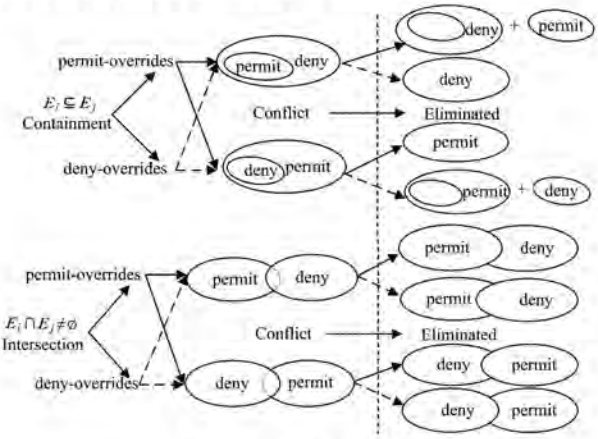


图 3　冲突的检测与消解

冲突在规则集层表现为规则内部属性的交集关系，可用如下表达式表示：

$(R_i.Sj \cap R_j.Sj \neq \emptyset \wedge R_i.Ac \cap R_j.Ac \neq \emptyset \wedge R_i.Re \cap R_j.Re \neq \emptyset) \rightarrow (R_i, R_j) = \text{Conflict}$

冲突在策略集层表现为策略内部规则的交集关系，可表示为：

$P_i.R \cap R_j.R \neq \emptyset \rightarrow (P_i, P_j) = \text{Conflict}$

冗余检测的过程如图 4 所示(图中虚线的左侧部分为冗余的检测，右侧部分为冗余的去除)。对于同一请求所能够匹配的策略，当决策结果相同，即 $E_i = E_j$ 时，判定为冗余。

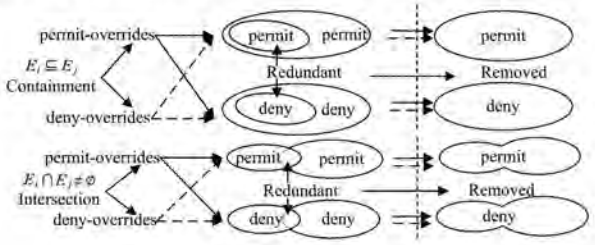


图 4 冗余的检测与去除

冗余在规则集层表现为规则内部属性的包含关系，用如下表达式表示：

$(R_i.Sj \subseteq R_j.Sj \wedge R_i.Ac \subseteq R_j.Ac \wedge R_i.Re \subseteq R_j.Re) \rightarrow R_i = \text{Redundant}$

冗余在策略集层表现为策略内部规则的包含关系，表示为：

$P_i.R \subseteq P_j.R \rightarrow P_i = \text{Redundant}$

4.2 冲突和冗余的消除

图 3 的右半部分描述了在 permit-overrides(带箭头的实线)及 deny-overrides(带箭头的虚线)情况下消解冲突的过程。在 permit-overrides 情况下，$Effect$ 优先取值为 permit，在 deny-overrides 情况下，$Effect$ 优先取值为 deny，根据当前算法的优先级进行消解，相交部分取优先值。当 $E_i \subseteq E_j$ 时，若外部状态的优先级较高，则取外层状态的结果；若内部状态的优先级较高，则保留内部状态的决策结果，消解外部状态的相应部分。当 $E_i \cap E_j \neq \emptyset$ 时，存在状态相交部分，此时相交部分取优先级较高的决策结果，并以此作为冲突部分的结果。

图 4 的右半部分描述了在 permit-overrides(带箭头的实线)及 deny-overrides(带箭头的虚线)情况下去除冗余的过程。当上层合并算法是 permit-overrides 时，$Effect$ 值为 permit，当合并算法是 deny-overrides 时，取 $Effect$ 值为 deny，去除相交部分。当 $E_i \subseteq E_j$ 时，由于外部状态与内部状态的决策结果相同，保留外部状态的决策结果，去除内部状态的冗余部分。当 $E_i \cap E_j \neq \emptyset$ 时，存在状态相交部分，此时相交部分仍取原状态的决策结果，去除相交部分。

4.3 策略优化算法的描述

根据上述冲突与冗余的判定方法，本节给出策略优化算法的具体描述。优化过程分策略层和策略集层，层间的策略优化遵循自底向上的执行顺序，即先执行策略层再执行策略集层。

策略层优化算法有 3 个输入，分别是规则合并算法($P.RCA$)、规则层 R_i 的 $Effect(R_i.E)$ 和规则层 R_j 的 $Effect(R_j.E)$。输出为策略层的决策结果集(P.E)。算法的具体描述如下。

```
if Ri⊆Rj//真包含
  if P.RCA=permit-overrides
    if Ri.E=Rj.E return Rj.E //冗余,取大集合
    if Ri.E=permit and Rj.E=deny
      return Ri.E+(Rj.E−Ri.E) //冲突,取并集
    if Ri.E=deny and Rj.E=permit return Rj.E
  else if P.RCA=deny-overrides
    if Ri.E=Rj.E return Rj.E //冗余,取大集合
    if Ri.E=permit and Rj.E=deny
      return Rj.E //冲突,取并集
    if Ri.E=deny and Rj.E=permit
      return Ri.E+(Rj.E−Ri.E)
  else
    Queuep={Ri.E,Rj.E},prior(Ri) ≥ prior(Rj)
else if Ri∩Rj≠∅ //部分交集
  if P.RCA=permit-overrides
    if Ri.E=Rj.E return Ri.E+(Rj.E−Ri.E∩Rj.E) //冗余,取大集合
    if Ri.E=permit and Rj.E=deny
      return Ri.E+(Rj.E−Ri.E∩Rj.E) //冲突,取并集
    if Ri.E=deny and Rj.E =permit
      return Rj.E+(Ri.E−Ri.E∩Rj.E)
  else if P.RCA=deny-overrides
    if Ri.E=Rj.E return Ri.E+(Rj.E−Ri.E∩Rj.E) //冗余,取大集合
    if Ri.E=permit and Rj.E=deny
      return Rj.E+(Ri.E−Ri.E∩Rj.E) //冲突,取并集
    if Ri.E=deny and Rj.E=permit
      return Ri.E+(Rj.E−Ri.E∩Rj.E)
  else
    Queuep={Ri.E,Rj.E},prior(Ri) ≥ prior(Rj)
else//相离
  Queuep={Ri.E,Rj.E},prior(Ri)≥prior(Rj)
```

策略集层优化算法同样有 3 个输入，分别是策略合并算法($PS.PCA$)、策略层 P_i 的 $Effect(P_i.E)$ 和策略层 P_j 的 $Effect(P_j.E)$。输出为策略集层的决策结果集($PS.E$)。算法的具体描述与上述策略层优化算法类似，不再具体给出。

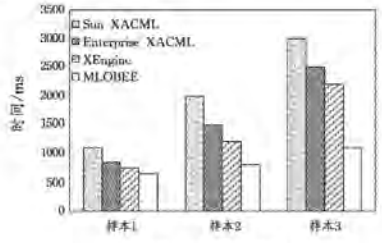
5 实验测试结果

通过仿真实验测试本文提出的优化算法在目前主流评估引擎上的性能。使用的评估引擎包括 Sun XACML，Enterprise XACML，XEngine 和 MLOBEE。测试所需用例在 XACML 官方测试包[17]的基础上进行修改和扩充。实验环境为 Inter Pentium 2.7GHz 双核 CPU，2GB 内存，Windows 7 操作系统，Java Runtime Environment 1.6.10。

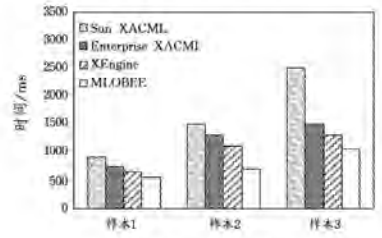
5.1 时间性能

选取上述 4 种评估引擎，计算策略优化前后平均处理一

次请求的时耗，并比较它们的差异。实验采用 3 组测试样本，第一组由 500 条原始策略组成，其中包含 100 条冗余规则和 20 条冲突规则；第二组由 1000 条原始策略组成，包含 300 条冗余规则和 50 条冲突规则；第三组由 2000 条原始策略组成，包含 500 条冗余规则和 100 条冲突规则。3 组样本中的每条策略均包含 5 条规则，根据策略包含的属性信息，分别随机生成 1000 次不同的访问请求。在 4 种引擎下先使用原始策略（即未优化的策略）对访问请求进行评估，再使用本文提出的优化算法对访问请求进行评估，最后比较两者访问请求响应的平均时间，结果如图 5 所示。



(a)优化前的评估时间



(b)优化后的评估时间

图 5　时间性能比较

对比图 5(a)与图 5(b)可以看出，在 3 组样本情况下，评估时间平均减少 10%～20%，随着策略规则数的增加，时间减少的幅度加大，在 Sun XACML 上的效果尤为明显。上述实验表明，本文所提算法通过消除策略间的冲突和冗余，缩短了请求的评估时间，提高了评估效率。

5.2　存储性能

本文提出的优化算法能去除和消解策略间的冗余和冲突，在缩短评估时间的同时还减小了策略占用的存储空间。在 Sun XACML 系统上，比较了采用本文优化算法所占用的存储空间与使用原始方法所占用的存储空间。逐渐增加规则数，测试二者占用空间的大小，对比结果如图 6 所示。当策略规则数量达到一定规模(1000 条左右)时，采用本文优化算法所占用的存储空间开始减小，且随着策略规则数目的增加，减小的效果愈加显著。

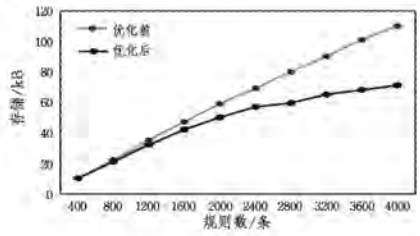


图 6　存储性能比较

最后需要说明的是，本算法旨在消除策略间的冗余与冲突，提高评估效率，并不对原有策略进行修改，因此不影响策略评估的准确性，这在实验中也得到了验证。

结束语　本文通过深入分析 XACML 策略层级架构和构成特点，采用集合的方法，用韦恩图形象且直观地对 XACML 策略进行表示；利用韦恩图法，根据集合间的交并关系检测消除策略间的冲突和冗余，形成一种基于韦恩图的 XACML 策略优化算法。该算法不仅能缩短策略评估时间，而且能减小策略占用的存储空间，有效提高策略的评估效率。目前该算法只是策略优化的多种方式之一，在后续工作中，可整合多种方式以进一步提高策略评估效率。

参 考 文 献

[1] SALDHANA A, TAPPETLA A, ANDERSON A, et al. Extensible Access Control Markup Language (XACML) Version 3.0 [OL]. http://docs.oasis-open.org/xacml/3.0/xacml_3.0_core-spec_en.html.

[2] Sun XACML[OL]. http://sunxacml.sourceforge.net.

[3] Enterprise XACML[OL]. http://code.google.com/p/enterprise-java-xacml.

[4] LIU A X, CHEN F, HWANG J H, et al. Designing Fast and Scalable XACML Policy Evaluation Engines[J]. IEEE Transactions on Computers, 2011, 60(12): 1802-1817.

[5] WANG Y Z, FENG D G, ZHANG L W, et al. XACML Policy Evaluation Engine Based on Multi-Level Optimization Technology[J]. Journal of Software, 2011, 22(2): 323-338. (in Chinese)
王雅哲，冯登国，张立武，等. 基于多层次优化技术的 XACML 策略评估引擎[J]. 软件学报，2011，22(2)：323-338.

[6] NIU D H, MA J F, MA Z, et al. HPEngine: high performance XACML policy evaluation engine based on statistical analysis [J]. Journal on Communications, 2014, 35(8): 206-215. (in Chinese)
牛德华，马建峰，马卓，等. 基于统计分析优化的高性能 XACML 策略评估引擎[J]. 通信学报，2014，35(8)：206-215.

[7] BULTER B, JENNINGS B, BOTVICH D. XACML policy performance evaluation using a flexible load testing framework[C]// ACM Conference on Computer and Communications Security. Chicago, USA, 2010: 648-650.

[8] KOLOVSKI V, HENDLER J, PARSIA B. Analyzing web access control policies[C]// International Conference on World Wide Web. Banff, Canada, 2007: 677-686.

[9] FISLER K, KRISHNAMURTHI S, MEYEROVICH L A, et al. Verification and change-impact analysis of access-control policies [C]// International Conference on Software Engineering. St. Louis, USA, 2005: 196-205.

[10] MOURAD A, JEBBAOUI H. SBA-XACML: Set-based approach providing efficient policy decision process for accessing Web services[J]. Expert Systems with Applications, 2015, 42(1): 165-178.

(下转第 125 页)

$\Delta X=\alpha)\approx 3.483878560829191E-5$，因此 REESSE3＋(16)算法在经过 16 轮迭代后就能抵抗差分攻击；同样地，$3/2^{16}\approx 4.57763671875E-5\geqslant P_{max}(\Delta Y(7)=\beta\mid\Delta X=\alpha)=2.497408028320769E-5$，因此 16 位输入的改进算法在经过 8 轮迭代后就能够抵抗差分攻击。

由以上分析可知，16 位输入的改进算法在经过 8 轮迭代后就能够抵抗差分攻击；而 REESSE3＋(16)算法要经过 16 轮迭代才能抵抗差分攻击。因此，在 16 位输入时，改进算法比 REESSE3＋算法更能抵抗差分攻击。

结束语　本文采用了马尔科夫密码模型，通过实验对比直接证明了：16 位输入的改进算法达到安全性要求所需要的轮数比原 REESSE3＋(16)算法所需轮数更少。由此可见，16 位输入时，改进算法比 REESSE3＋算法更能抵抗差分攻击。至于在输入位数为 128 时，改进算法是否更加安全，还有待进一步论证。

参 考 文 献

[1] SU S H, LÜ S W, DONG D Q. A 128-bit Block Cipher Based on Three Group Arithmetics[DB/OL]. http://eprint.iacr.org/2014/704.pdf.

[2] SU S H. The REESSE Symmetric Key Cryptosystem[J]. Computer Engineering and Applications, 2004, 40(19): 84-86. (in Chinese)
苏盛辉. REESSE 对称密钥密码体制[J]. 计算机工程与应用, 2004, 40(19): 84-86.

[3] BIHAM E, SHAMIR A. Differential cryptanalysis of DES-like cryptosystems[J]. Journal of Cryptology, 1991, 4(1): 3-72.

[4] BIHAM E, SHAMIR A. Differential Cryptanalysis of the Full 16-Round DES[C]// International Cryptology Conference on Advances in Cryptology. Springer-Verlag, 1992: 487-496.

[5] HEYS H M. A Tutorial on Linear and Differential Cryptanalysis [J]. Cryptologia, 2001, 26(3): 189-221.

[6] BIHAM E, SHAMIR A. Differential Cryptanalysis of the Data Encryption Standard[M]. Springer-Verlag, 1993.

[7] LAI X J. On the design and security of block ciphers[C]// Series in Information Processing. 1992.

[8] LAI X J. International Data Encryption Algorithm[J]. Hepatology, 2007, 60(6): 2125-2126.

[9] STINSON D R. Cryptography: Theory and Practice[M]. CRC Press, 1995.

[10] LAI X J, MASSEY J L, MURPHY S. Markov ciphers and differential cryptanalysis[C]// International Conference on Theory and Application of Cryptographic Techniques. Springer-Verlag, 1991: 17-38.

[11] O'CONNOR L, GOLIC J D. A Unified Markow Approach to Differential and Linear Cryptanalysis[C]// International Conference on the Theory and Applications of Cryptology: Advances in Cryptology. Springer-Verlag, 1994: 387-397.

[12] KATZ J, LINDE Y. Introduction to modern cryptography[OL]. http://www.pdfsdocuments.com/introductions-to-modern-crypotography-principles-and-protocols.pdf.

[13] SCHNEIER B, SUTHERLAND P. Applied Cryptography: Protocols, Algorithms, and Source Code in C[M]. John Wiley & Sons, 2015.

[14] BURGIN M. Theory of information: fundamentality, diversity and unification[M]. World Scientific, 2010.

[15] KEILSON J. Markov Chain Models — Rarity and Exponentiality[M]. Springer New York, 1979.

[16] MINC H. Nonnegative matrices[M]. John Wiley & Sons, 1988: 318-330.

[17] HARPES C, HARPES C. Cryptanalysis of iterated block ciphers [C]// ETH Series in Information Processing. Hartung-Gorre, 1996.

[18] BIHAM E, DUNKELMAN O, KELLER N. New cryptanalytic results on IDEA[C]// International Conference on Theory and Application of Cryptology and Information Security. Springer-Verlag, 2006: 412-427.

[19] BIRYUKOV A, NAKAHARA J, YILDIRIM H M. Differential entropy analysis of the IDEA block cipher[J]. Journal of Computational & Applied Mathematics, 2014, 259(6): 561-570.

[20] LAI X J, MASSEY J L. A Proposal for a New Block Encryption Standard[M]// Advances in Cryptology — EUROCRYPT '90. Springer Berlin Heidelberg, 1999: 389-404.

(上接第 113 页)

[11] JEBBAOUI H, MOURAD A, OTROK H, et al. Semantics-based approach for detecting flaws, conflicts and redundancies in XACML policies[J]. Computers & Electrical Engineering, 2015, 44(C): 91-103.

[12] MOURAD A, TOUT H, TAHLI C, et al. From model-driven specification to design-level set-based analysis of XACML policies[J]. Computers & Electrical Engineering, 2016, 52(C): 65-79.

[13] WANG Y Z, FENG D G. A Conflict and Redundancy Analysis Method for XACML Rules[J]. Journal of Computers, 2009, 32(3): 516-530. (in Chinese)
王雅哲，冯登国. 一种 XACML 规则冲突及冗余分析方法[J]. 计算机学报, 2009, 32(3): 516-530.

[14] CHEN W H, WANG N N. Research on XACML policy evaluation optimization technology[J]. Application Research of Computers, 2013, 30(3): 900-905. (in Chinese)
陈伟鹤，王娜娜. 基于 XACML 的策略评估优化技术的研究[J]. 计算机应用研究, 2013, 30(3): 900-905.

[15] QI Y, CHEN J, LI Q M. XACML policy evaluationoptimization method based on recording[J]. Journal of Nanjing University of Science and Technology, 2015, 39(2): 187-193. (in Chinese)
戚湧，陈俊，李千目. 一种基于重排序的 XACML 策略评估优化方法[J]. 南京理工大学学报, 2015, 39(2): 187-193.

[16] MAROUF S, SHEHAB M, SQUICCIARINI A, et al. Adaptive Reordering and Clustering-Based Framework for Efficient XACML Policy Evaluation[J]. IEEE Transactions on Services Computing, 2012, 4(4): 300-313.

[17] XACML 2.0 conformance test[OL]. http://www.oasis-open.org/committees/download.php/14846/xacml2.0-ct-v.0.4.zip.