测试资源受约束的安全关键软件加速测试方法

张德平[1,2] 聂长海[1] 徐宝文[1]

(东南大学计算机科学与工程学院 南京 210096)[1] (南京航空航天大学理学院 南京 210016)[2]

摘 要 基于马尔可夫链使用模型提出了一种针对安全关键软件测试资源受约束的启发式加速测试方法。该方法利用一种新的随机优化技术——交叉熵方法,以软件投放后软件失效风险损失最小为目标,基于失效风险损失通过修正操作剖面,自动生成测试数据集。实验结果表明该方法能有效地降低软件失效风险,提高测试效率,是一种快速有效的加速测试方法。

关键词 软件测试,安全关键软件,加速测试方,马尔可夫链使用模型,交叉熵方法

中图法分类号 TP311 **文献标识码** A

Acceleration Testing Method of Safety-critical Software with Testing Resource Constraint

ZHANG De-ping[1,2] NIE Chang-hai[1] XU Bao-wen[1]

(Department of Computer Science & Engineering, Southeast University, Nanjing 210096, China)[1]

(College of Science, Nanjing University of Aeronautics & Astronautics, Nanjing 210016, China)[2]

Abstract A heuristic acceleration testing method of safety-critical software with testing resource constraint based on Markov chain usage models was presented. The developed approach makes use of a new stochastic optimization method called Cross Entropy method. By adjusting the operational profile in Markov chain usage models, we considered the minimization of failure risk and tried to automatic generation test data. The experimental results obtained show that this optimization technique is a promising option for tackling this problem.

Keywords Software testing, Safety-critical software, Acceleration testing method, Markov chain usage model, Cross entropy method

统计测试是一种随机测试方法,它基于软件使用模型生成测试用例进行测试,根据测试结果进行统计分析,评估软件质量,是高可靠性软件开发的重要组成部分。统计测试假定每个使用的数据输入都具有相同缺陷检测率,根据转移概率在各个状态下选择下一操作,对使用频繁的操作进行更多的测试,能有效地检测出那些对软件可靠性影响较大的软件缺陷。因此,统计测试充分性判定准则一般都基于测试使用与实际使用的差异程度,以确保根据测试结果估计出的可靠性能代表软件实际使用的可靠性[1,4,5]。

对于特定操作环境这是正确的,但无法贯穿整个软件系统的全部操作集合。典型例子是高可信性软件特别是安全关键软件,其关键操作如核电系统的紧急停堆处理,由于使用概率非常小,在统计测试中往往得不到充分测试,而这类操作的可靠性指标往往很高,其失效经常会造成严重后果。仅从测试使用与实际使用的差异程度判定测试充分性还不能达到统计测试目标。为了充分测试这些关键操作,并获得软件可靠性的无偏估计,需要执行大量的统计测试用例,这会导致统计测试代价过高。基于此,许多统计测试方法被提出并进行理论研究[4-10],其中大多数为"控制"统计测试方法[6-9],主要利用重要抽样技术,在保证软件可靠性或失效风险估计是无偏估计的前提下,通过修正操作剖面来控制估计的方差,提高估计的准确性。尽管这些方法在一定程度上可以降低失效风险,但并不能保证软件投放后失效风险最小。因此能否寻求出既保证测试后软件失效风险最小,又不增加软件测试成本的最佳解决方案,是软件测试人员亟需解决的现实问题。

1 基于成本的马尔可夫链使用模型

马尔可夫链使用模型是一个具有唯一初态和终态的马尔可夫链,可用强连通有向图 $G=(V,A)$ 和函数 $p:V\times V\rightarrow[0,1]$ 表示,它具有如下性质:

· $V=\{1,2,\cdots,n\}$ 是节点集,表示软件系统的使用状态。

· A 为边集,其元素表示在某个状态下选定某个操作时软件状态间的转移。从状态 i 到状态 j 的边 e 定义为一个有序对 (i,j),任意两个状态 i 和 j 之间的一个方向最多只有一条有向边相连。

· 转移概率 $p(i,j)$ 满足 $0\leqslant p(i,j)\leqslant 1$,表示从状态 i 一

到稿日期:2008-10-30 本文受教育部博士点基金(20060286020),国家自然科学基金(60425206, 60773104, 60633010, 60503033),武汉大学软件工程重点实验室开放基金和东南大学优秀青年教师教学科研基金资助。

张德平(1973—),讲师,博士生,主要从事软件测试技术、软件可靠性、数理统计与随机过程等方面的教学、科研工作,E-mail:depingzhang@163.com;徐宝文(1961—),男,教授,博士,博士生导师,主要从事程序设计语言、软件工程、并行与网络软件等方面的教学与科研工作;聂长海(1971—),男,副教授,博士,主要从事软件测试技术、模糊信息处理、神经网络等方面的教学与科研工作。

步转移到状态 j 的概率。整个软件系统状态之间的转移概率 $p(i,j)$可用转移概率矩阵 P 表示，即 $P=(p(i,j))_{n\times n}$。

假定状态 1 为初态，状态 n 为终态，并且状态 n 为吸收态，它表示一旦进入状态 n 就不再离去，即 $p(n,n)=1, p(n,j)=0, \forall j\neq n$。每一个状态 $i\in V$ 都是从初态可达的，即在 G 中总存在一条从状态 1 到状态 i 的有向路径。每条边 e 均可估计出如下参数[10]：失效概率 $f(e)$，投放后的失效风险 $l(e)$，测试成本 $c(e)$，易知 $l(e)>>c(e)$。不失一般性，假定失效边在操作过程中总是失效，一旦经过测试，失效边均看作正确操作，不再造成软件失效。进一步假定[8]：

(a) 边(i,j)的每一次操作运行都有一个先验失效概率 $f(i,j)\geqslant 0$。

(b) 不同操作是否引起软件失效相互独立。

(c) 软件投入运行期间，失效边(i,j)至少历经一次，造成失效风险 $l(i,j)$发生。

(d) 总失效风险由各边失效风险累加形成。

定义一个测试数据集 $x=(e_1,e_2,\cdots,e_m), e_k\in A, (k=1,2,\cdots,m)$为包含初态 1 和终态 n 的一个测试数据。根据规范说明书或用户实际使用生成测试数据集 x 的概率分布称为操作剖面，即转移概率矩阵 P。边 e 可以在测试数据集 x 中出现多次，即测试数据集 x 中可以包含多条从初态 1 到 n 的路径。由假定，运行测试数据集 x 的测试成本为

$$C(x)=\sum_{k=1}^{m}c(e_k) \tag{1}$$

测试前，每一条边是否造成软件失效是不确定的，定义示性函数 $I_f(e)$，对于给定的边 e，如果测试过程中相对于边 e 的操作造成软件失效，则令 $I_f(e)=1$，否则令 $I_f(e)=0$。显然，$E[I_f(e)]=f(e)$，其中 E 表示求期望。

假定软件投入运行后，软件失效均发生在平稳状态下。引入示性函数 $I_r(e)$来描述过程是否达到平稳状态，即如果随机历经边 e 的状态是平稳状态时 $I_r(e)=1$，否则 $I_r(e)=0$。由示性函数 $I_f(e)$与 $I_r(e)$定义知，它们相互独立。令$\overline{A}\subset A$ 表示未经测试边的集合，由假设只有 $\overline{A}$ 中的边才可能因软件失效造成失效风险。则平稳状态下失效风险的期望可表示为：

$$\begin{aligned}L(\overline{A})&=E[\sum_{e\in\overline{A}}I_r(e)\cdot I_f(e)\cdot l(e)]\\&=\sum_{e\in\overline{A}}E[I_r(e)\cdot I_f(e)\cdot l(e)]\end{aligned} \tag{2}$$

由 $I_f(e)$与 $I_r(e)$的独立性及期望的性质可得

$$L(\overline{A})=\sum_{e\in\overline{A}}E[I_r(e)\cdot f(e)\cdot l(e)] \tag{3}$$

上式中 $E[I_r(e)]=q(i,j)=q(e)$为平稳状态下边 $e=(i,j)$被历经的概率。记 π_i 为平稳状态下状态 $i(i=1,2,\cdots,n)$发生的概率(π_i 为长期运行中状态出现的概率)，由马尔可夫过程转移概率与绝对概率之间的关系以及遍历性定理可知，平稳状态下边(i,j)被历经的概率为 $q(i,j)=\pi_i\cdot p(i,j)$。

按照操作剖面 P 生成测试数据集 x 进行测试，当测试数据集足够大时，它有可能测试完所有的边，使得 $\overline{A}=\phi$，此时失效风险的期望 $L(\overline{A})=0$。然而，随着测试数据集 x 边数的增长，相应测试成本 $C(x)$也会增加，因此当测试预算成本受约束时，测试数据集并不能使得所有的边都测试完，软件失效风险不会降低到最小。

本文采用一种导向性启发式方法修正转移概率，以投放后软件失效风险最小为目的确定一个最优测试剖面(即测试中用来生成测试数据的概率分布)，根据最优测试剖面生成测试数据集进行测试。令 $\overline{A}(x)$为测试数据集 x 没有覆盖到的边集，则这类问题可写成如下形式：

问题 P1　确定一个测试剖面 T^*，根据 T^* 生成测试数据集 x 使得在给定预算测试成本条件下，期望失效风险 $S(x)$最小，即

$$\begin{aligned}&\text{minimize } S(x)=L(\overline{A}(x)),\\&\text{满足 } C(x)\leqslant B\end{aligned} \tag{4}$$

其中 B 为给定的预算测试成本。

2　交叉熵方法

应用交叉熵方法[12]需要解决两个问题：如何产生随机样本以及如何在每一次迭代中修正参数。

设 x 为根据转移概率矩阵 P 生成的一个测试数据集，则由马尔可夫链的马氏性知测试数据集 x 出现的概率为

$$f(x;P)=\prod_i p(i,j) \tag{5}$$

其中 i 为测试数据集 x 中所有出现过的状态，j 为与边$(i,j)\in A$ 相依附的状态。令 γ^* 为失效风险 $S(x)$在所有测试数据集组成集合 X 中的最小值，则问题 P1 可改写为确定一个最佳转移概率矩阵(即测试剖面)，由此测试剖面生成测试数据集 x^*，使得

$$S(x^*)=\gamma^*=\min_{x\in X}S(x) \tag{6}$$

定义$\{I_{\{S(x)\leqslant\gamma\}}\}$为 X 上不同$\gamma\in R$ 值的示性函数集合，则式(6)中最优测试剖面确定问题可转化为与之相伴随的概率估计问题来求解：

$$\begin{aligned}\ell(\gamma)&=P_u(S(X)\leqslant\gamma)=E_u[I_{\{S(x)\leqslant\gamma\}}]\\&=\sum_x I_{\{S(x)\leqslant\gamma\}}f(x;u)\end{aligned} \tag{7}$$

其中 P_u 和 E_u 为相对于概率分布 $f(\cdot;u)$的概率测度和期望。

当 $\gamma=\gamma^*$ 时 $\ell(\gamma)$估计的最直接方法是采用重要抽样方法[11]：根据 X 上的概率分布 g 抽取样本 $x_1,x_2,\cdots,x_N$，则 ℓ 的估计为

$$\hat{\ell}=\frac{1}{N}\sum_{i=1}^{N}I_{\{S(x_i)\leqslant\gamma\}}\frac{f(x_i;u)}{g(x_i)} \tag{8}$$

显然，当概率分布 g 取

$$g^*=\frac{I_{\{S(x)\leqslant\gamma\}}f(x;u)}{\ell} \tag{9}$$

时只需抽取一个样本即可得 ℓ 的一个方差为零的无偏估计。

从式(9)可以看出，g^* 依赖于未知参数 ℓ，很难确定。因此一般在概率分布 $f(\cdot;u)$的概率分布簇$\{f(\cdot;v)\}$(v 为参数)中选取概率分布 g 来解决这个问题，即确定推断参数 v 使得 $f(\cdot;v)$与概率分布 g^* 差别达到最小。常用来衡量两个概率分布差别大小的测度是 K-L 距离或交叉熵[11]。

这样，最优测试剖面的确定问题就转化为确定推断参数 v，使得 $f(\cdot;v)$与概率分布 g^* 的交叉熵最小，即

$$v^*=\arg\min_v-\int g^*(x)\ln f(x;v)\mathrm{d}x \tag{10}$$

将式(9)代入式(10)可得如下等价推断参数确定形式：

$$v^*=\arg\max_v E_u[I_{\{S(X)\leqslant\gamma\}}\ln f(X;v)] \tag{11}$$

由于测试数据集 $x=(e_1,e_2,\cdots)$是根据转移概率矩阵 P 生成的，概率矩阵 P 就是推断参数 v，测试数据集 x 的联合概率分布为

$f(x;P)=\prod_i p(i,j)$，满足$\sum_j p(i,j)=1$

由拉格朗日乘子法，则式(11)的优化问题可转化为

$$\max_P\{E_P[I_{\{S(X)\leqslant\gamma\}}\sum_i I_{\{x\in X_{ij}\}}\ln p(i,j)]+\sum_i u_i(\sum_j p(i,j)-1)\} \tag{12}$$

其中事件 $I_{\{x\in X_{ij}\}}$ 表示测试数据集 x 在测试中历经状态 i 并在此状态下一次操作转移到状态 j。

由此可得最优测试剖面中不同状态之间的转移概率

$$p(i,j)=\frac{E_P[I_{\{S(X)\leqslant\gamma\}}\sum_i I_{\{x\in X_{ij}\}}]}{E_P[I_{\{S(X)\leqslant\gamma\}}\sum_i I_{\{x\in X_i\}}]} \tag{13}$$

则最优测试剖面 T^* 的每个转移概率的估计

$$\hat{p}(i,j)=\frac{\sum_{k=1}^{N} I_{\{S(x_k)\leqslant\gamma\}}\sum_i I_{\{x_k\in X_{ij}\}}}{\sum_{k=1}^{N} I_{\{S(x_k)\leqslant\gamma\}}\sum_i I_{\{x_k\in X_i\}}},i,j=1,2,\cdots,n \tag{14}$$

式(14)为不同状态之间转移概率的修正公式，其直观含义为：从状态 i 转移到状态 j 的概率为所有目标函数值不大于 γ 的测试数据集中，从状态 i 一步转移到状态 j 的次数与测试历经状态 i 的次数之比。由此知，在每一个状态中参数修正都是选用最有利于接近最优目标的“精英”样本。

3 应用交叉熵方法加速测试

应用交叉熵方法于统计测试的马尔可夫使用模型，具体由一个两步迭代过程实现：

1. 适应修正 γ_t。对于固定的 v_{t-1}，令 γ_t 为在参数 v_{t-1} 下的目标函数值序列 $S(x)$ 的 $\rho\cdot100\%$-分位数。即 γ_t 满足：

$P_{v_{t-1}}(S(x)\geqslant\gamma_t)\leqslant1-\rho$，并且 $P_{v_{t-1}}(S(x)\leqslant\gamma_t)\geqslant\rho$

其中 $x\sim f(\cdot;v_{t-1})$。最简单的估计是根据概率矩阵 v_{t-1} 抽样出的测试数据集 $x_1,x_2,\cdots,x_N$ 计算出目标函数值 $S(x)$，并将其按从小到大排列：$S_{(1)}\leqslant S_{(2)}\leqslant\cdots\leqslant S_{(N)}$，最后用 N 个目标函数值中的第 $\rho\cdot100\%$ 个来估计 γ_t，即 $\hat{\gamma}_t=S_{\lceil\rho\cdot N\rceil}$。

2. 适应修正 v_t。对于固定的 γ_t 和 v_{t-1}，由式(14)得到修正的 $\tilde{v}_t$，并采用平滑技术得到 v_t 的修正值：

$$\hat{v}_t=\alpha\tilde{v}_t+(1-\alpha)\hat{v}_{t-1},\alpha\in(0.4,0.9)\ [11] \tag{15}$$

步骤 2 中采用平滑技术主要是避免 $\hat{v}_t$ 的某些元素 $\hat{v}_{t,i}$ 为 0，使算法最初阶段一直在局部最优解上搜索。首次循环中，初始化转移概率矩阵为操作剖面 P。由迭代过程可得 $\hat{\gamma}_1$ 和 $\hat{v}_1$，如此循环，可得序列对 $\{(\hat{\gamma}_t,\hat{v}_t),t=1,2,\cdots\}$。由文献[12]易知 $\hat{\gamma}_t\to\gamma^*$，由此可得用于生成最优或近似最优测试数据集的测试剖面，具体算法由算法 3.1 给出。

算法 3.1 最优测试剖面生成算法

1. 初始化 $\hat{v}_0$ 为操作剖面 P，令 t=1。
2. 根据概率矩阵 $\hat{v}_{t-1}$ 及相应的测试充分性判定准则自动生成随机测试数据集 x 的样本 $x_1,x_2,\cdots,x_N$，计算各个测试数据集样本对应的目标函数值 $S(x)$ 并排序 $S_{(1)},S_{(2)},\cdots,S_{(N)}$，找出 N 个目标函数值的 $\rho\cdot100\%$-分位数 $\hat{\gamma}_t=S_{\lceil\rho\cdot N\rceil}$。
3. 用同样的测试数据集样本 $x_1,x_2,\cdots,x_N$，由(14)式解出 $\tilde{v}_t$，应用式(15)的平滑技术得到 $\hat{v}_t$。
4. 令 t=t+1，重复步骤 2—3，直到满足停止条件。(如：对于某个给定的 d(如 d=4)有 $\hat{\gamma}_t=\hat{\gamma}_{t-1}=\cdots=\hat{\gamma}_{t-d}$，或 $\hat{v}_t$ 不再变化，或达到最大迭代步数。)
5. 输出最优测试剖面。

4 实验与分析

为更好说明基于交叉熵方法加速安全关键系统测试的有效性，本节用一个列车调度信息系统软件[7,9,10]实例加以验证。图 1 给出了列车调度的马尔可夫链使用模型。

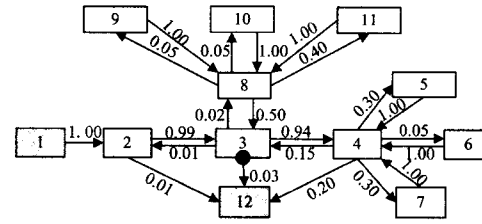

图 1 列车信息系统的马尔可夫链使用模型[7]

软件包括 12 个操作，每次执行从操作 1 开始，操作 12 是软件终止运行时执行的操作，边(8,9)和(8,10)导致关键操作 9 和 10 执行。假定每条边 (i,j) 的失效概率 $f(i,j)$、失效风险 $l(i,j)$ 和测试成本 $c(i,j)$ 已估计出，如表 1 所列。

表 1 马尔可夫使用模型中各边失效概率、失效风险和测试成本

(i,j)	(i,j)	l(i,j)(×10⁵)	c(i,j)
(1,2)(3,2)(3,4)(4,3)			
(4,7)(5,4)(6,4)(7,4)	0.001	10	30
(8,3)(9,8)(10,8)(11,8)			
(2,3)(4,5)	0.01	10	30
(2,12)	0.001	1	30
(3,8)(8,11)	0.001	100	30
(4,6)	0.01	100	300
(4,12)	0.001	1	3
(8,9)	0.001	1000	300
(8,10)	0.1	1000	300
其它	0	0	0

测试预算成本 B 给定为 4,000，每次迭代样本数 $N=1000$，$\rho=0.02$，$d=4$（或最大迭代步数为 50），平滑参数 $\alpha=0.4$，采用测试成本不超过预算成本停止准则。由算法 3.1 生成最优测试剖面 T^*，根据最优测试剖面和操作剖面分别生成的 500 个测试数据集中，按最优测试剖面生成的测试数据集中操作 8 累计遍历了 738 次，关键操作 9 和 10 分别遍历 798 和 709 次，而按操作剖面生成的测试用例集中，操作 8 被遍历的次数为 104 次，关键操作 9 和 10 分别遍历 64 和 31 次。采用最优测试剖面能显著增加关键边的操作次数。

两种情形下平均失效风险随测试边变化曲线如图 2 所示，由此可得当测试预算成本给定时，通过采用最优测试剖面 T^* 生成的测试数据集能显著降低软件失效风险。

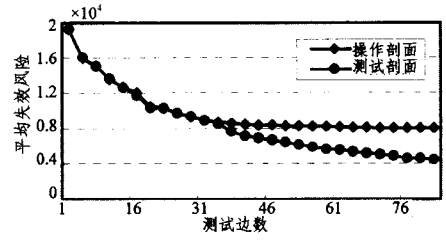

图 2 测试成本受约束时平均失效风险比较

当测试预算成本 B 变化时：B 取值于 100～8,000 之间，每次取值增加量为 $\Delta B=200$，在不同预算成本给定时分别按照相应的最优测试剖面 T^* 与操作剖面 P 生成的 500 个测试数据集中，其平均失效风险的比较曲线如图 3 所示。

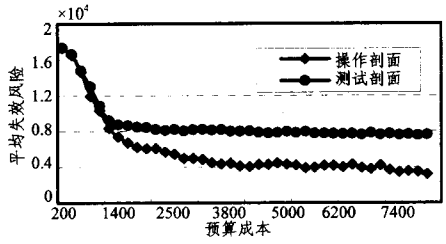

图 3　不同测试预算成本下平均失效风险比较

从图 3 可以看出，当测试预算成本 B 较小时，采用最优测试剖面和操作剖面使得平均失效风险降低幅度差不多，随着测试预算成本的增加，采用最优测试剖面的平均失效风险下降幅度要远远大于操作剖面，并且采用最优测试剖面使平均失效风险一直有下降趋势，而操作剖面则当测试预算成本达到一定值时，平均失效风险不再下降，而是保持不变。

结束语　本文针对安全关键软件系统利用交叉熵方法通过一种修正机制调节操作剖面，增加使用概率小的关键操作的测试机会，加速软件测试，在提高软件系统质量的同时降低软件测试成本。该方法是为实现测试目标而采用的一种导向性测试数据集生成方法，将整个测试数据集作为优化对象，利用“功效”值最好的测试数据集所包含的状态间转移信息，把修正测试剖面归结为一个“伴随”随机优化问题的迭代求解，用以指导整个测试数据集的生成，加速软件测试。

参 考 文 献

[1] Whittacker J A, Poore J H. Markov analysis of software specifications[J]. ACM Transaction on Software Engineering and Method, 1994, 2(1): 93-106

[2] Whittaker J A, Thomason M G. A Markov chain model for statistical software testing[J]. IEEE Transaction on Software Engineering, 1994, 20: 812-824

[3] Walton G H, Poore J H, Trammell J. Statistical testing of software based on a usage model[J]. Software-Practice and Experience, 1995, 25(1): 97-108

[4] Walton G H, Poore J H. Measuring complexity and coverage of software specifications[J]. Information and Software Technology, 2000, 42: 859-872

[5] 冯华，徐锡山，王戟. 统计测试中测试链与使用链的相似性判别[J]. 计算机工程与科学，2003，25(1)：17-19

[6] Gutjahr W J. Failure risk estimation via Markov software usage models[C]// E. Schoitsch, ed. SAFECOMP 96, Proc. of the 15th International Conference on Computer Safety. reliability and security. Springer, 1997: 183-192

[7] Gutjahr W J. Importance sampling of test cases in Markovian software usage models[J]. Probability in the Engineering and Informational Sciences, 1997, 11: 19-36

[8] Doerner K, Laure E. High performance computing in the optimization of software test plans[J]. Optimization and Engineering, 2002, 3: 67-87

[9] 颜炯，王戟，陈火旺. 基于重要抽样的软件统计测试加速[J]. 计算机工程与科学，2005，27(3)：64-66

[10] Doerner K, Gutjahr W J. Extracting test sequences from a Markov software usage model by ACO[J]. LNCS, Springer Verlag, 2003, 2724: 2465-2476

[11] Boer D P-T, Kroese D P, Mannor S, et al. A Tutorial on the Cross-Entropy Method [J]. Annals of Operations Research, 2005, 134: 19-67

[12] Margolin L. On the convergence of the cross-entropy method [J]. Annals of Operations Research, 2005, 134: 201-214

（上接第 120 页）

传恶意 XML 文档，达到发布欺骗性信息、破坏数据库系统等目的[9]。防御结点攻击的一种有效途径是对 XML 中每个元素的类型进行定义，包括数据类型定义、元素可能出现次数的最大/最小值及元素值的取值范围等。由于 XML/DTD 缺乏对文档结构、属性、数据类型等约束的足够描述，而 XML Schema 在此方面更具优势，所以也可以采用 XML Schema 作为 XML 文档模式描述语言，代替 XML DTD。对于提取并置入 SQL 语句中的数据及 SQL 语句本身，也需进行提炼并作安全性分析，以防止 SQL 注入攻击。如，所提取出的 XML 数据不允许含有 select, delete, update 等特定含义的字符。此外，还可以将数据编码成特定的码制（如十六进制），再通过后台系统提取数据并解码。

结束语　XML 正发挥着越来越重要的作用，如何通过有效的安全技术实现 XML 的安全应用已经成为人们研究的热点。虽然已经制定了多个有关的 XML 安全技术规范，但如何应用好这些规范来保证 XML 的安全性，仍是一个具有很大挑战性的课题。本文的工作力图通过一个应用系统的设计，探讨综合应用 XML 加密、签名等安全技术确保 XML 安全的模型以及实现方法。从实践来看，该系统已经建成并投入使用，并已在管理工作中取得了较好效果。

参 考 文 献

[1] W3C XML Core Working Group. Extensible Markup Language (XML) 1.0 (Fourth Edition) W3C Recommendation [EB/OL]. http://www.w3.org/TR/xml/, 2006, 8

[2] BlakeDournaee. XML安全基础[M]. 北京：清华大学出版社，2003

[3] Eastlake D, Reagle J. XML Encryption Syntax and Processing W3C Recommendation [EB/OL]. http://www.w3.org/TR/xmlenc-core, 2002, 12

[4] Eastlake D, Reagle J, Solo D. XML-Signature Syntax and Processing W3C Recommendation [EB/OL]. http://www.w3.org/TR/xmldsig-core, 2002

[5] Ford W, Hallam-Baker P, Fox B, et al. XML Key Management Specification (XKMS) [EB/OL]. http://www.w3.org/TR/xkms/

[6] 顾天竺，沈洁，陈晓红，等. 基于 XML 的异构数据集成模式的研究[J]. 计算机应用研究，2007，24(4)：94-96

[7] Donald M M, Johansson E. C# 数据安全手册 [M]. 北京：清华大学出版社，2006

[8] 丘威，张立臣. XML DTD 规范化处理研究[J]. 计算机科学，2006，33(7)：63-67

[9] Buehrer G, Weide B W, Paolo A, et al. Using parse tree validation to prevent SQL injection attacks [EB/OL]. http://portal.acm.org/citation.cfm? id=1108496, 2007