基于模型检测的多轮 FR 协议验证*)

郭 华[1] 庄 雷[2] 张习勇[3] 李舟军[1]

(北京航空航天大学计算机学院 北京 100083)[1] (郑州大学理论计算机研究所 郑州 450052)[2]

(解放军信息工程大学 郑州 450002)[3]

摘 要 随着网络的大规模应用,越来越多的协议在并发环境中执行,时间也成为协议中一个重要因素。本文对公平交换协议 Franklin/Reiter 协议加入了时间因素,用时间自动机对其建模,并用自动验证工具 UPPAAL 验证了单轮协议的性质。重点验证了并发环境中多轮协议的执行情况,最后给出了协议在多轮情况下正常执行需满足的条件。

关键词 电子商务协议,模型检测,时间自动机,UPPAAL,多轮执行

Verifying Multiple Runs of FR Protocol Using Model Checking

GUO Hua[1] ZHUANG Lei[2] ZHANG Xi-yong[3] LI Zhou-jun[1]

(School of Computer Science & Engineering, Beihang University, Beijing 100083, China)[1]

(Institute of Information Engineering, Zhengzhou University, Zhengzhou 450052, China)[2]

(Institute of Information Engineering, Information Engineering University, Zhengzhou 450002, China)[3]

Abstract With the growing popularity of the Internet, more and more protocols run concurrently. In this paper, time is added to the FR protocol, and timed automata is used to model Franklin/Reiter protocol. Then UPPAAL is used to verify some properties of the protocol. This paper focuses on multiple runs of FR protocol, and gives some qualifications ensuring that the protocol runs successfully in the concurrent environment.

Keywords E-commerce protocol, Formal verification, Timed automata, UPPAAL, Multiple runs

1 引言

随着电子商务如火如荼地发展,安全电子商务协议的形式化验证方法日益成为研究的焦点。常见的形式化验证方法主要有定理证明和模型检测。前者虽然对协议参加者的数量和消息的数量没有限制,但证明过程比较繁琐,且难以实现协议的自动化验证。

模型检测是把协议形式化为有限状态的自动机模型。验证时通过对模型的状态空间进行探索,确定是否拥有规约所说明的属性。若规约成立,模型检测工具应加以确认。否则,能给出反例说明规约不成立的情况,供调试使用。虽然当状态空间太大时,面临着状态空间爆炸的问题,但检测过程可自动完成。

近几年来,基于模型检测和定理证明出现了一些形式化验证的新方法。文献[1]提出了一种新工具,通过验证发方非否认证据(POO)属于接收者的最终拥有集合,以及收方非否认证据(POR)属于发送者的最终拥有集合是否成立来分析协议的可追究性和公平性。文献[2]将 Kailar 逻辑与 LPC 方法结合来验证协议的可追究性和公平性。文献[3]用归纳证明的方法验证了 SET 协议。文献[4]用时间自动机对 TLS 握手协议进行建模及验证。

上述方法虽然都有较强的能力验证协议的安全性,但大多只验证一轮,即只有一组参与者时协议的性质是否满足。随着网络的大规模应用,越来越多的协议是在并发环境中执行的,因此更有必要验证在多个参与者共同执行协议时协议的性质是否仍然满足。目前关于多轮验证的文献很少。另外,在并发环境中,通信信道的安全与否、参与协议主体的诚实与否都与协议执行的时间有关,因此时间是一个重要因素。

本文用时间自动机(TA)[5]对加入了时间因素的协议建模后,用自动验证工具 UPPAAL[6]不仅验证了单轮协议的性质,还重点验证了多轮协议的执行。具体组织方式如下:第 2 节介绍 Franklin/Reiter(FR)协议;第 3 节用 UPPAAL 验证了 FR 协议单轮及多轮的执行情况,并给出了协议在多轮情况下正常执行需满足的条件;最后进行了小结。

2 Franklin/Reiter 协议

Franklin/Reiter 协议[7]是电子商务中一个重要的公平交换协议,它在两个参与者 X 和 Y 之间通过第三方 Z 交换消息。协议描述如下:

1) X→Y:Xsecret1;

2) Y→X:Ysecret1;

3) X→Z:Xsecret2correct 或 Xsecret2incorrect;

4) Y→Z:Ysecret2correct 或 Ysecret2incorrect;

5) Z→X:Ysecret2correct 或 ExchangeAbort;

6) Z→Y:Xsecret2correct 或 ExchangeAbort。

在该协议中,参与者 X 与 Y 相互交换秘密,第三方 Z 控

*)国家自然科学基金项目(60473057, 69873040),河南省教育厅基础研究项目(2003520256)。郭 华 博士研究生,研究方向为安全协议分析及验证;庄 雷 博士,教授,研究方向为自动机理论及信息安全;张习勇 博士,讲师,研究方向为信息安全;李舟军 博士,教授,研究方向为安全协议分析。

制两个参与者或者接收相互的秘密，或者均得不到彼此的秘密。X，Y 把各自的秘密分为两部分。第一二步，两者各自把自己秘密的第一部分传给对方；三四步中，各自把秘密的第二部分传给第三方 Z。在实际的 FR 协议的第四步之后，Z 对 X 和 Y 秘密的第二部分进行基于 HASH 函数的检验。之后，Z 要么把秘密的第二部分正确传送，要么通知 X，Y 交换失败。

时间是设计协议时不可忽略的因素，本文对原协议做如下限制：参与者 X，Y 在发完第二部分消息后，若等待时间超过 $t1$ 秒，则取消协议；整个交换过程在 $t2$（$t2>t1$）秒之内完成。为方便起见，本文设 $t1$ 为 5 秒，$t2$ 为 15 秒，且通信信道是安全且无堵塞的。

3 用 UPPAAL 验证 FR 协议性质

3.1 UPPAAL 简介

UPPAAL 是由瑞典的 Uppsala 大学与丹麦的 Aalborg 大学联合研发的一种自动验证工具，已成功用于通信协议和实时控制器的验证。

UPPAAL 有一个易于用户操作和使用的集成环境，它的图形用户主界面主要由一个系统编辑器（system editor）、一个模拟器（simulator）和一个验证器（verifier）组成。使用该工具时，首先在系统编辑器中将要验证的协议用自动机来建模，之后可在模拟器中检查所建模型可能的执行是否有错，以便在验证前发现一些错误；最后在验证器中将要验证的协议的性质用 BNF 语法来描述，通过快速搜索系统的状态空间来检查协议是否满足这些性质。如果性质得不到满足，UPPAAL 会自动生成一个诊断序列，利用该诊断序列提供的信息，协议设计者可对协议进行修改，直至协议性质得到满足。

3.2 单轮协议验证模型

首先为协议的各参与者建模。

3.2.1 符号约定

m1　Z 收到的 X 的第二部分秘密

m2　Z 收到的 Y 的第二部分秘密

xi　参与者 X 发送的自身的第 i 部分秘密

yi　参与者 Y 发送的自身的第 i 部分秘密

x2-ok　第三方 Z 发送 X 的第二部分正确秘密给 Y

y2-ok　第三方 Z 发送 Y 的第二部分正确秘密给 X

abort1，abort2 第三方 Z 告知 X、Y 协议取消

a?　通道

a!　通道

x，y，z　时钟

3.2.2 为协议参与者建模

为参与者 X 建模：进程 X 发送自己的第一部分秘密给 Y，并将时钟 x 置为零，进入状态 s1，在 s1 的停留时间不超过 5 秒。收到进程 Y 的第一部分秘密后，转至 s2，并立即发送第二部分秘密给第三方 Z，进入等待状态 s3。在 s3，若等待时间不超过 5 秒，则有两种情况发生：(1)若收到参与者 Y 的第二部分正确秘密，转至 s4，并正常终止协议；(2)若收到由 Z 发出的取消协议的 abort1 信号，转至 s5，并取消协议。否则，若收到 abort2 信号，表明 X 的等待时间超过 5 秒，转至 s6，并取消协议。该过程用 TA 建模如图 1 所示。

为参与者 Y 建模：进程 Y 收到进程 X 的第一部分秘密后，将时钟 y 置为零，转至 s1，在 5 秒内发送自己的第一部分秘密给 X，进入 s2。然后在 5 秒之内发送第二部分秘密给第三方 Z，并进入等待状态 s3。在 s3，若等待时间不超过 5 秒，则有两种情况发生：(1)若收到参与者 X 的第二部分正确秘密，则转至 s4，并正常终止协议；(2)若收到由 Z 发送的取消协议的 abort1 信号，则转至 s5，并取消协议。否则，若收到 abort2 信号，表明 Y 的等待时间超过 5 秒，则转至 s6，并取消协议。该过程用 TA 建模，如图 2 所示。

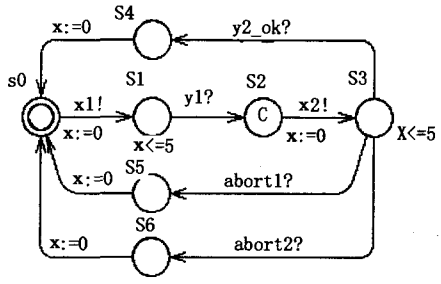

图 1　X 进程

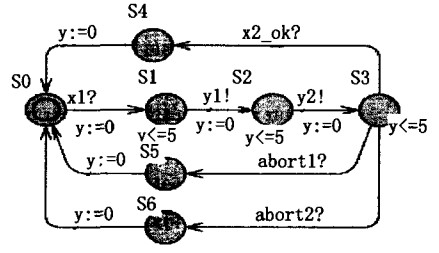

图 2　Y 进程

为第三方 Z 建模：第三方 Z 收到参与者 X，Y 发送的各自的第二部分秘密后，在 s2 对 X 的第二部分秘密进行哈希验证。若为正确秘密，则 m1：＝1，否则 m1：＝－1，并转至 s3。在 s3 对 Y 的第二部分秘密进行哈希验证，若为正确秘密，则 m2：＝1，否则 m2：＝－1，并转至 s4。Z 在该位置对验证后的结果检查后，有两种情况发生：(1)若两者均为正确秘密，转至 s5，在 s5 判断时钟 z 的值。若 $z>5$，则隐含着 X 与 Y 的等待时间超过 5 秒，Z 通过信号 abort2 广播通知 X，Y 取消协议；若 $z\leqslant 5$，则将 X 的第二部分正确秘密传送给 Y，将 Y 的第二部分正确秘密传送给 X；(2)若有一个为错误秘密，转至位置 s8，在该位置判断时钟 z 的值。若 $z>5$，则隐含着 X 与 Y 的等待时间超过 5 秒，Z 通过信号 abort2 广播通知 X，Y 取消协议；若 $z\leqslant 5$，通过 abort1 来通知 X，Y 取消协议。该过程用 TA 建模，如图 3 所示。

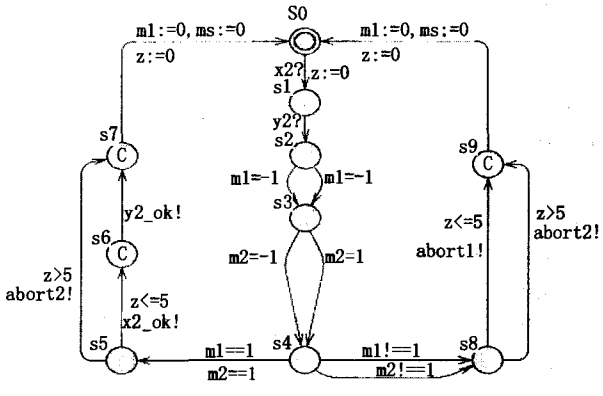

图 3　Z 进程

在上述模型中，整个系统为三者之积：X || Y || Z。大致工作过程为：X 发送 x1(Xsecret1)给 Y，Y 收到后发送 y1

(Ysecret1)给 X。之后,X,Y 分别向第三方 Z 发送 x2,y2,第三方 Z 在位置 s2,s3 对收到的 x2,y2 进行检验,若均为正确秘密,则将 x2,y2 发送给对方;否则,通知 X 与 Y 秘密交换失败,取消协议。

3.2.3 用 UPPAAL 验证

建立模型后,通过在模拟器中观察各进程之间的交互情况,初步判定模型符合协议要求。下面在验证器中使用 BNF 语法对模型进行验证。本节主要验证保护个人利益的性质,即如果 Y 得到了 X 的秘密,X 就必须得到 Y 的秘密;反之,即使 Y 想欺骗 Z,X 的利益也不会受到损失。验证如下:

A[] not deadlock 通过验证,表明系统没有死锁。

E<> Y.s4 and X.s4 通过验证,表明如果 Y 得到了 X 的秘密,X 也必须得到 Y 的秘密。

E<> Z.s6 imply X.s4 通过验证,表明若 X 发出了第二部分正确秘密,X 应得到 Y 的秘密。

E<>Z.s8 imply X.s5 通过验证,表明如果 Y 想欺骗 Z(即 Y 发假秘密给 X),则 X 的利益也不会受到损失(即 X 取消该协议)。

下面验证一些关于时间的性质。

E<> (X.s4 and X.x≤5) 通过验证,表明 X 在 5 秒内成功得到 Y 的第二部分秘密。

E<> (X.s5 and X.x≤5) 通过验证,表明若 Y 欺骗 X,则 X 在 5 秒内取消协议。

E<> (Y.s4 and Y.y≤5) 通过验证,表明 Y 在 5 秒内成功得到 X 的第二部分秘密。

E<> (Y.s5 and Y.y≤5) 通过验证,表明若 X 欺骗 Y,则 Y 在 5 秒内取消协议。

图 4 是在验证器中得到的结果。

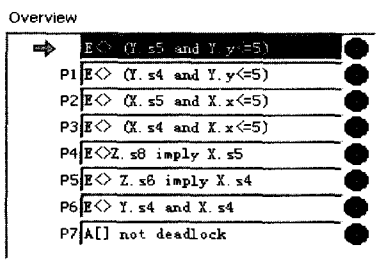

图 4 验证器中的结果

在上述过程中,通过使用 UPPAAL 语义中约束位置(committed location)的概念,确保了 Z 在发送 x2,y2 的过程中满足原子性,即如果 Z 发送 X 秘密的第二部分,必须不可中断地紧接着发送 Y 秘密的第二部分,从而确保协议满足保护个人利益的属性。

3.3 多轮协议的实现模型

上面验证的是单轮协议执行时的情况。实际应用中,往往是多对参与者并发执行协议,因此验证多轮情况下协议是否仍满足性质显得尤为重要。

3.3.1 对协议的各参与者建模

在建模时,首先把协议的多对参与者 X_i,Y_i($i\geq 2$)抽象为一对参与者 X,Y,然后用 TA 在系统编辑器中为 X,Y 建模。通过对模板中的 X,Y 传递参数,在模拟器中可得到多对参与者 X_i,Y_i,从而可在验证器中进行协议的多轮验证。下面分别为 X,Y,Z 建模。

为参与者 X 建模:进程 X_i 在 s0 将自己的 id 值 i 赋给 s_id。s_id 不仅用于查找对应的 X_i,Y_i,并且用于记住当前 X_i 的 id 值,以便第三方 Z 将当前发送第二部分消息的 X_i,Y_i 的 id 值放入队列 m1_idL[tail1], m2_idL[tail2]。在 s0,X_i 还要检查自己的 flag 标记,若 flag=1,表明该参与者已经执行过协议,在 s0 不变;若 flag=0,则转至 s1,开始执行协议。首先发送自己的第一部分秘密给 Y_i,并将时钟 x 置为零,到达 s2。当接收到进程 Y_i 的第一部分秘密后,立即发送第二部分秘密给第三方 Z,并进入等待状态 s4。在 5 秒内若收到参与者 Y_i 的第二部分正确消息,则正常终止协议,并令 flag 为 1,表明该对主体已经执行过协议,以后不再执行;若收到由 Z 发送的取消协议的信号 abort1,则取消协议,并令 flag 为 1,表明该对参与者中有一方不诚实,以后该对参与者不再相互交互秘密,即以后不再执行该协议。否则,若收到 abort2 信号,表明 X 的等待时间超过 5 秒,则取消协议,并回到位置 s0。该过程用 TA 建模如图 5 所示。

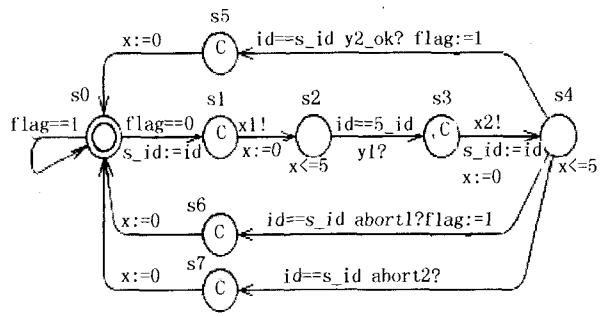

图 5 X 进程

为参与者 Y 建模:进程 Y_i 通过 s_id 来判断自己是否与 X_i 对应,若对应,则接收进程 X_i 的第一部分秘密后,将时钟 y 置为零,并立即发送自己秘密的第一部分给 X_i。然后在 5 秒内发送秘密的第二部分给第三方 Z,并进入等待状态 s3。在 5 秒内若收到参与者 X_i 的第二部分正确秘密,则正常终止协议;若收到由 Z 发送的取消协议的信号 abort1,则取消协议。否则,若收到 abort2 信号,表明 Y 的等待时间超过 5 秒,则取消协议,并回到位置 s0。该过程用 TA 建模如图 6 所示。

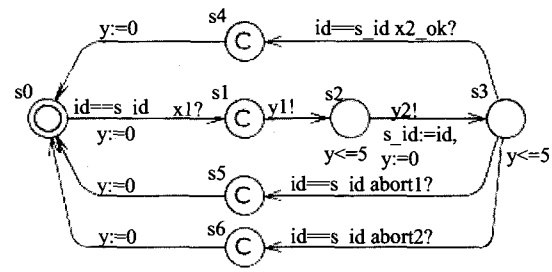

图 6 Y 进程

为第三方 Z 建模:第三方 Z 接收到 X_i,Y_i 发送的各自的第二部分秘密后,分别将其 id 号入队并转至 s2。在 s2 有两种情形:(1)或者处理其它参与者 X_j($i\neq j$)发送秘密的第二部分,并转至 s1;(2)或者从队列中取出队头消息转至 s3。在 s3,s4 对它们进行哈希验证。在 s3,根据 X_i 发送的第二部分秘密的正确与否,将 m1_descL[tail1]赋值为 1 或 −1,转至 s4。同理,根据 Y_i 发送的第二部分秘密的正确与否,将 m2_descL[tail2]赋值为 1 或 −1,并转至 s5。在 s5 比较两部分秘密,此时有两种情况:(1)若均为正确的秘密,转至 s6,在 s6 判断时钟 z 的值,若 $z>5$,则隐含着 X_i 与 Y_i 的等待时间超过 5 秒,Z 广播通知它们取消协议,转至 s8;若 $z\leq 5$,则将 X_i 的第

二部分正确秘密传送给 Yi，将 Yi 的第二部分正确秘密传送给 Xi，转至 s8；(2)若有一个为错误秘密，转至 s9，在 s9 判断时钟 z 的值，若 z>5，则隐含 Xi 与 Yi 的等待时间超过 5 秒，Z 通过信号 abort2 广播通知两参与者取消协议，并转至 s10；若 z≤5，则通过信号 abort1 广播通知两参与者取消协议，转至 s10。在 s8 与 s10，有三种情形：(1)处理其它参与者 $Xj(i\neq j)$ 发送秘密的第二部分，并转至 s1，(2)若此时循环队列非空，从队列中取出下一对参与者 Xj，Yj 并转至 s3，(3)若此时循环队列为空，则转至起始位置 s0。该过程用 TA 建模，如图 7 所示。

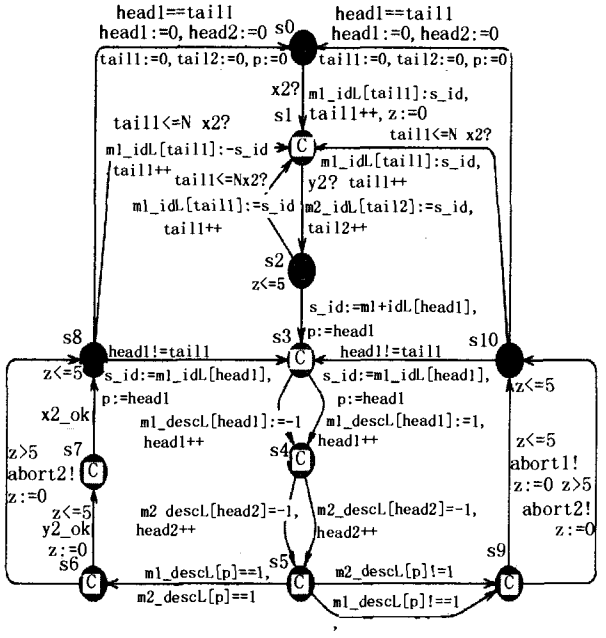

图 7　Z 进程

3.3.2　用 UPPAAL 验证

假如有 i 对参与者同时交换秘密，则整个系统为一乘积：Xi || Yi || Z。为简化起见，我们以 X1 与 Y1 为例，其余参与者对有类似结论。验证如下：

A[] not deadlock 通过验证，表明系统没有死锁。

E<> X1. s5 and Y1. s4 通过验证，表明若 X1 得到了 Y1 的秘密，Y1 也必须得到 X1 的秘密。

E<> Z. s6 imply X1. s5 通过验证，表明若 X1 发出了正确秘密，则 X1 应得到 Y1 的秘密。

E<>Z. s9 imply X1. s6 通过验证，表明若 X1 (或 Y1)发出错误秘密，最终协议将被取消。

下面验证关于时间的一些性质。

E<> (X1. s5 and X1. x≤5) 通过验证，表明 X1 在 5 秒内成功得到 Y1 的第二部分秘密。

E<> (X1. s6 and X1. x≤5) 通过验证，表明若 Y1 欺骗 X1，则 X1 在 5 秒内取消协议。

E<> (Y1. s4 and Y1. y<5) 通过验证，表明 Y1 在 5 秒内成功得到 X1 的第二部分秘密。

E<> (Y1. s5 and Y1. y≤5) 通过验证，表明若 X1 欺骗 Y1，则 Y1 在 5 秒内取消协议。

图 8 是在验证器中得到的结果。

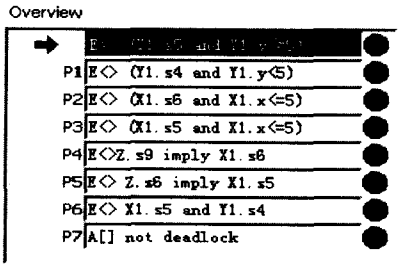

图 8　验证器中的结果

通过上述验证发现，在单轮模型中，若要实现协议的顺利执行，只需第三方 Z 不可中断地发送 X 和 Y 的第二部分秘密即可。但对于多轮协议的执行，除上述条件外，还需满足以下两个条件：(1)一对参与者发送的第一部分秘密也必须满足原子性，即一对协议参与者必须把各自第二部分秘密发送给第三方 Z 以后，才允许其它参与者执行协议；(2)当第三方 Z 无中断地处理过一对参与者之后，方允许其它参与者执行协议。只要满足上述条件，协议即可由多对参与者同时执行，并满足保护个人利益的属性。

结束语　电子商务协议是电子商务的重要组成部分。本文用自动验证工具 UPPAAL 验证了带有时间的 FR 协议的一个新属性，重点给出了多轮协议的验证。验证结果表明，只要稍加条件限制，原协议就可由多个参与者同时执行，并且仍然满足保护个人利益的属性及一些时间限制。用自动机对协议建模后可借助工具实现协议的自动验证，但各参与者的加解密信息及签名信息不能充分表示出来。因此今后的工作是，将定理证明与自动机有机结合，增强自动机的表达能力，尽可能多地验证协议的其他安全属性。

参 考 文 献

[1] 周典萃，卿斯汉，周展飞. 一种分析电子商务协议的新工具[J]. 软件学报，2001,12(9)：1318-1328

[2] 王彩芬，葛建华. 一种分析电子商务协议的新方法[J]. 计算机学报，2004，27(4)：507-515

[3] Bella G, Massacci F, Paulson L C. Verifying the SET registration protocols [J]. IEEE Journal on Selected Areas in Communications, 2003, 21(1): 77-87

[4] Diaz G, Cuartero F, Ruiz V, et al. Automatic verification of the TLS handshake protocol [C]. Nicosia, Cyprus: SAC, 2004: 789-794

[5] Alur R, Dill D L. A theory of timed automata [J]. Theoretical Computer Science, 1994, 126: 183-235

[6] Larsen K G, Petterson P, Wang Yi. UPPAAL in a nutshell [J]. Journal on Software Tools for Technology Transfer, 1997, 1(1/2): 134-152

[7] Franklin M, Reiter M. Fair exchange with a semi-trusted third party[C]//Proceedings of the 4th ACM Conference on Computer and Communication Security. Switzerland: ACM Press, 1997: 1-5