安全协议的时限责任分析

梁 坚, 敖青云, 尤晋元
(上海交通大学计算机科学与工程系, 上海 200030)

摘 要: 本文针对安全协议中的时限责任问题, 提出结合责任性与新鲜性来分析时间标记. 这种思路体现在我们新的时限逻辑框架中, 比较 Kudo[8] 的方法, 新的逻辑体系因为有对消息完整性的判断, 能更有效地防止消息的篡改与重发攻击, 且更加简洁和实用.

关键词: 时限责任; 形式化分析; Kudo 逻辑
中图分类号: TP309 文献标识码: A 文章编号: 0372-2112 (2002) 10-1450-05

Analyzing the Temporal Accountability of Secure Protocols

LIANG Jian, AO Qing-yun, YOU Jin-yuan
(*Dept of Computer Science & Engineering, Shanghai Jiao Tong University, Shanghai 200030, China*)

Abstract: To analyze the time-critical secure protocols, we propose a new method that combines temporal accountability and timestamp freshness verification. Comparing to Kudo's logic, our method can prevent tempar and replay attacks by adding the integrity verification based on timestamps. In addition, our method benefits intuition and practicality.

Key words: temporal accountability; formal analysis; Kudo logic

1 引言

加密协议利用密码技术为我们在公开信道中建立起可信、保密的通道, 为开放环境下的安全通信提供了基础设施. 在具有一定程度时钟同步的分布式环境下, 协议的运行被加上时间限制以保证在规定时间范围内完成某项事务或阶段性事务处理. 例如认证协议中(如 Wide-Mouth-Frog protocol, Kerberos Protocol[1]), 时间标记(timestamp)被用来标识消息的新鲜(freshness), 这种新鲜性标记是相对于接收方某一时间段内有效的. 某些电子商务协议, 如电子投递, 订单协议[2]中, 这种时间限定成为协议正确执行的一个重要前提, 时间标记除了标识消息的完整(integrity)外, 更用于跟踪主体(principal)的时敏动作(time-sensitive actions)(如在何时有消息的收、发操作). 对于标志新鲜的时间标记, 有效时间段内的消息重发攻击可能破坏协议的完整(如对 Denning-Sacco 协议的攻击[3]). 对于后者, 任何篡改和重发时间标记的可能都不能保证对主体行为的有效追究. 我们把这一类使用时间标记的安全协议叫做时限安全协议, 对他们的分析和设计要求对于违反时限要求的主体行为具有时限责任的可追究性(temporal accountability).

Syverson[4] 首先在 BAN 逻辑[1]中引入时态公式(temporal statement), 能对认证协议的因果一致性攻击(causal consistence attacking)做有效分析, 由于 Syverson 的逻辑体系以 AT 逻辑[5]作为语义模型, 所以非常复杂, 难于实际操作. Stubblebine[6] 在 Syverson 工作的基础上对时限协议的分析中加入了同步语义, 并提出密钥期限和信任公式可撤消(revocation)的概念, 使传统 BAN 逻辑具有了非单调的操作. 但是 Stubblebine 并没有在其文献中给出完整的分析方法, 其语义模型也很复杂. 而且, 两人的工作都是 BAN 信任逻辑的扩展, 不能对时限责任进行分析. Kailar[7] 运用了主体对自己签名消息负责的特点提出了可证明性逻辑, 适合于责任性问题的分析, 信任逻辑的分析目的是推演主体相信某一公式成立, 而责任性问题是某主体向第三方证明某个公式是成立的. Kudo[8] 在 Kailar 逻辑的基础上引入时态构造(timestamp, at, before, after)和相应假设, 提出了时限责任问题(A CanProve x generated at t, A CanProve x generated before t, …). 但这种逻辑混淆了两个重要原语: x generated before t by A 和 A Says$_t$ x, 分别表示 A 在时间 t 以前(包含 t 时刻)产生了 x 和 A 在 t 时刻发出 x. Kudo 认为对于 B receives$_{t1}\{t2, x\}Ka^{-1}$(假设 A 有同步时钟), 有 B CanProve A Says$_{t2}$ x. 而实际上, $\{t2, x\}Ka^{-1}$可能为已被篡改或重发的消息, 所以应为 x generated before $t2$ by A.

总结已有的相关工作, 我们认为对安全协议进行时限责任的分析时, 应加入对消息新鲜性的判断, 因为协议中的时间标记有这两重作用. 为此我们在 Kailar 逻辑中加入时态公式(x At t), 新鲜性公式(x Freshbefore t), 和相应的 2 条时态假设. 简化了 Kudo 的逻辑体系, 同时又增强了对时限责任的分析能力. 本文结构如下: 第二节是我们的时限逻辑体系, 并与 Kudo 逻辑进行了分析比较; 第三节给出了一个时限协议设计

收稿日期: 2000-09-18; 修回日期: 2001-01-12
基金项目: 国家自然科学基金(No. 69973032)

与分析的例子.

2 时限逻辑

我们的逻辑体系也是对可证明性逻辑的扩展, 扩展中引入基本的时态公式 x At t, timestamp. 同时也加入了对时间标记新鲜性的判断: Freshstamped, Freshbefore 两个原语. 在我们的推演中, 对时间标记的分析不仅包括时限责任分析, 同时也分析其新鲜性. 为此我们先简要介绍 Kailar 可证明性逻辑:

Kailar 逻辑的基本符号:

A, B, …: 参与协议的各个主体.

M, Q: 消息, 由一个主体发送给另一个主体的消息.

TTP: 可信任第三方(trusted third party, 简称 *TTP*).

Ka: A 的公钥, 用于验证 A 的数字签名. Ka^{-1}是与 Ka 对应的A 的私有密钥.

Kailar 逻辑的公式如下:

A CanProve x: 对于任何主体 B, A 能执行一系列操作使得通过这些操作以后, A 能使B 相信公式x, 而不泄漏任何秘密 $y(y \not\Rightarrow x)$给 B.

Ka Authenticates A: Ka 能用于验证A 的数字签名.

x in M: x 是M 中的一个或几个可被理解的域, 它的含义是由协议设计者明确定义的. 可被理解的域通常是明文或者主体拥有密钥的加密域.

A Says x: A 声明公式x 并对x 以及x 能推导出的公式负责. 通常, 隐含地假设以下推论成立,

R1: A Says $(x, y) \Rightarrow A$ Says x

A Receives M SignedWith K^{-1}: A 收到一个用K^{-1}签名的消息 M. 通常, 隐含地假设以下推论成立,

R2: A Received M SignedWith K^{-1}; x in $M \Rightarrow A$ Received x SignedWith K^{-1}

A IsTrustedOn x: A 对公式x 具有管辖权, 即A 被协议其他主体所相信A 声明的公式x 是正确的.

R3 连接规则:

A CanProve x; A CanProve $y \Rightarrow$

A CanProve $(x \wedge y)$

如果 A 能够证明公式x, 并且 A 能够证明公式y, 那么 A 能够证明公式$x \wedge y$.

R4 推理规则:

A CanProve x; $x \Rightarrow y$

则有 A CanProve y

如果 A 能够证明公式x, 而由公式x 能推导公式y(即公式x 蕴涵有公式y 的含义), 那么 A 能够证明公式y.

R5 签名规则:

A Received$(M$ SignedWith $K^{-1})$; x in M;

$\underline{A \text{ CanProve}(K \text{ Authenticates } B)}$

A CanProve$(B$ Says $x)$

如果 A 收到一个用私钥K^{-1}签名的消息 M, M 中包含A 能理解的公式x, 并且 A 能够证明公钥K 能用于验证 B 的签名, 那么 A 能证明B 声明了公式x.

R6 信任规则:

A CanProve$(B$ Says $x)$;

$\underline{A \text{ CanProve}(B \text{ IsTrustedOn } x)}$

A CanProve x

如果 A 能够证明B 对x 有管辖权, 并且 B 声明了公式x, 那么 A 能证明公式x.

利用 Kailar 逻辑来分析协议共有 4 个步骤: ①列举协议要达到的目标; ②对协议的语句进行解释, 使之转化为逻辑公式. 在这一步中, 只对那些包含签过名的明文消息并且和分析可追究性相关的语句进行解释; ③列举分析协议时需要用到的初始假设; ④对协议进行分析.

我们的时限公式扩展:

(1) x At t: 在时间 t 发生了x, x 为任意公式. 如A Says x At t(A 在t 时间发出x, 记为 A Says$_t$ X), A Receives x At t(A 在 t 时间接收到x, 记为 A Receives$_t$ x).

(2) x TimestampedWith t: 对 x 打上时间标记 t.

(3) x FreshstampedWith n: x 打上新鲜标记 n, 在某些协议中, 时间标记同时也作为新鲜性的标志.

(4) x Freshbefore t: x 在时间 t 之前是新鲜的(包含 t 时刻), 即 x 在 t 时刻以前是唯一的. 推理规则如下:

R7 新鲜性规则

A CanProve$(x$ FreshstampedWith n SignedWith $K^{-1})$;

A CanProve$(x$ TimestampedWith t SignedWith $K^{-1})$;

A CanProve$(K$ Authenticates *TTP*);

$\underline{A \text{ CanProve}(TTP \text{ IsTrustedOn } t)}$

A CanProve$(x$ Freshbefore $t)$

R8 时限规则

A Receives $(x$ TimestampedWith t SignedWith $K^{-1})$;

$(x$ in $y)$SignedWith K_b^{-1};

A CanProve $(x$ Freshbefore $t)$;

A CanProve$(K_b$ Authenticates $B)$;

A CanProve$(K$ Authenticates *TTP*);

$\underline{A \text{ CanProve}(TTP \text{ IsTrustedOn } t)}$

A CanProve$(B$ Says y At $t)$

Kudo 的方法中有 before, after 的公式和相应的语义, 但没有新鲜性的判断. 下面给出 Kudo 逻辑中的两个主要规则:

K1:

A Receives$(x$ TimestampedWith t SignedWith $K^{-1})$;

A CanProve$(K$ Authenticates $T)$;

$\underline{A \text{ CanProve}(T \text{ IsTrustedOn } t)}$

A CanProve$(x$ Generatedbefore $t)$

x Generatebefore t 指公式x 在时间t 以前(包含时刻 t)产生.

K2:

A CanProve $(x$ Generatedbefore $t)$;

$(y$ SignedWith $K^{-1})$ in x;

$\underline{A \text{ CanProve}(K \text{ Authenticates } B)}$

A CanProve$(B$ Says y At $t)$

B Says y At t 的语义是 B 在 t 时刻发出了 y, $(y$ SignedWith $K^{-1})$ in x 指x 中含有$\{y\}K^{-1}$的内容. 我们可以发现虽然

x 在 t 时刻已经产生, 但包含 B 签名的公式 y 并不能说明 y 是 B 在 t 时刻发出的. 原因是: 时间签名者没有判定 $\{y\}K^{-1}$ 是否满足新鲜性.

举例说明如下: 对于公式: A Receives$\{t,(n,y)K_b^{-1}\}K_t^{-1}$, 其中 K_b^{-1}、K_t^{-1} 分别为 B、同步时钟 TTP 的签名, t 为时间, n 为 nonce 随机数, $(n,y)K_b^{-1}$ 可能为重发的字段, TTP 只对 t 的签名负责, 而 B 也只对保证新鲜性的 n 的签名负责, 所以 TTP 不能识别 $(n,y)K_b^{-1}$ 的唯一性. 而按 K2 规则的推演可得到 A CanProve(B Says y At t). 在我们的方法中, 由于不能得到 A CanProve((n,y) Freshbefore t) 的条件, 所以不能导出 A CanProve(B Says y At t) 的结论, 防止了可能的重传攻击. 另外, 对于 Kudo 提出的电子投递协议(具体协议请参考文献[8]), 我们发现对于其中的关键消息 $M8$ 的重传攻击就能破坏该协议对时限责任的保证.

……

$M8$: Alice $\rightarrow$ Bob: $\{xi, A, B, \{ts, \{desc\}K_b^{-1}\}K_t^{-1}\}K_a^{-1}$

$M9$: Bob $\rightarrow$ TTP: $\{yi\}K_b^{-1}$

$M10$: TTP $\rightarrow$ Bob: $\{t1, \{yi\}K_b^{-1}\}K_t^{-1}$

$M11$: Bob $\rightarrow$ Alice: $\{t1, \{yi\}K_b^{-1}\}K_t^{-1}$

……

其中, $\{desc\}$ 为投递的起止时间 $desc = (ts, te)$, $yi = M8$, xi 为投递的内容, Bob 通过 $M9$ 向可信时钟 TTP 申请投递收到的时间签名, $M11$ 是 Bob 对 Alice 投递收到的确认, 确认消息经 TTP 的时间签名, 所以 Alice 可以向第三方证明其投递已被 Bob 在 $t1$ 以前收到. 这时如果有 $M8$ 的重传消息, 由于 TTP 和主体 Bob 并没有对消息的唯一性进行判断, 可能使 Alice 收到的 $M11$ 中含有 $t2(t1 < t2)$ 时间标记, 即证明 Bob 在比应收到时间 $t1$ 晚的时间 $t2$ 收到. 使得时限责任难以追究.

3 时限协议的设计与分析

由第 2 节的讨论我们可以看到, 安全的时限协议中时间标记的作用不仅作为主体时限责任可追究的依据, 而且也应作为判断消息新鲜性的标志. 所以在协议的设计中我们对消息的时间签名应能保证这两点, 最好由具有提供时钟服务的可信主体完成. 下面的例子中, 我们通过对 NetBill 协议实例[9]的修改和分析说明这一问题.

NetBill 是 CMU 在 1996 年提出的一个安全电子交易协议, 特点是保证交易过程的原子性要求: 即在交易的任何时刻终止, 协议保证各方的利益不受损. 另外 Netbill 系统能在网上完成整个电子化商品的交易过程, 但协议本身并不支持时限交易, 我们在协议中加入时间标记以支持对主体时限责任的可追究. 原有的和修改后的 Netbill 体系结构如图 1.

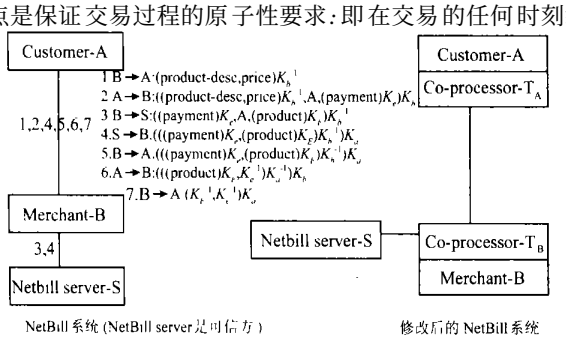

图 1

我们在原有系统中 A、B 的两端都加上 Cσ processor[10] T_A, T_B 作为同步的时钟对消息 1, 2, 4, 5, 6, 7 加上时间标记. Cσ processor 是一种防篡改设备, 由可信的权威机构(如时钟的根服务器)发行, 并保持分布环境下的时钟同步. Cσ processor 作为双方可信的设备对收发的消息打时间标记以作为可信的时限追究依据, 同时也作为消息新鲜性的标志. 修改后的协议如下:

1. $B \rightarrow A$: $(T_{b1}, (\text{product desc, price})K_b^{-1})K_{cb}^{-1}$
   $\{T_A$ 对接收的消息 1 签 $T_{a1}\}$

2. $A \rightarrow B$: $(T_{a2}, ((\text{product desc, price})K_a^{-1}), A, (\text{payment})K_e)K_b, T_{a1})K_{ca}^{-1}$
   $\{T_B$ 对接收的消息 2 签 $T_{b2}\}$

3. $B \rightarrow S$: $(T_{a2}, ((\text{payment})K_e, A, (\text{product})K_E)K_b^{-1}, T_{b2})K_{cb}^{-1}$
   $\{B$、S 间的处理时间忽略$\}$

4. $S \rightarrow B$: $(((\text{payment})K_e, (\text{product})K_E)K_b^{-1})K_a$

5. $B \rightarrow A$: $(T_{b3}, (((\text{payment})K_e, (\text{product})K_E)K_b^{-1})K_a, T_{b2})K_{cb}^{-1}$
   $\{T_A$ 对接收的消息 5 签 $T_{a3}\}$

6. $A \rightarrow B$: $(T_{a4}, (((\text{product})K_E, K_e^{-1})K_a^{-1})K_b, T_{a3})K_{ca}^{-1}$
   $\{T_B$ 对接收的消息 6 签 $T_{b4}\}$

7. $B \rightarrow A$: $(T_{b5}, (K_E^{-1}, K_e^{-1})K_a, T_{b4})K_{cb}^{-1}$
   $\{T_A$ 对接收的消息 7 签 $T_{a5}\}$

其中 K_{cb}^{-1}, K_{ca}^{-1} 分别为 T_B, T_A 的私钥, (K_e, K_e^{-1}), (K_E, K_E^{-1}) 为 A、B 为电子支付和电子商品产生的密钥对, 每个消息前面的 timestamp 为发送方 Cσ processor 标记本次消息发送的时间, 后面的 timestamp 是由接收方 Cσ processor 产生, 标记为上一消息接收的时间. 各时间标记满足 $T_{b1} < T_{a1} < T_{a2} < T_{b2} < T_{b3} < T_{a3} < T_{a4} < T_{b4} < T_{b5} < T_{a5}$ 先后关系, Cσ processor 根据来自同一主体的最近消息时间标记判断当前消息的新鲜性, 保证收发消息的完整.

我们对交易中的几个关键阶段进行时限分析, 消息 2、5 分别是定单和商品投递阶段, 消息 6 和 7 是密钥交换的支付阶段. 假设各消息传送的时间限定为 To, Td, $Tp6$, $Tp7$, 而总的交易期限为 Ts, 协议要求满足 $To + Td + Tp6 + Tp7 \leqslant Ts$, 如图 2. 如果违反, 则协议应能追究是哪个阶段和哪个主体超时.

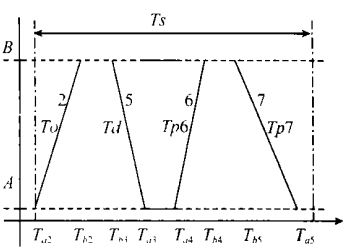

图 2

根据逻辑分析的步骤, 我们先确定我们的分析目标: (本文只讨论时限责任问题)

$G1$: B CanProve(A can send order during To);

$G2$: A CanProve(B can deliver product during Td);

$G3$: B CanProve(A can send payment key during $Tp6$);

$G4$: A CanProve(B can send product key during $Tp7$);

然后是协议的逻辑描述:(我们将非 T_A, T_B 处理的字段省略为$M1$, $M2$, …$M7$, 即原 Netbill 协议中的消息内容).

(1) A Receives ($M1$ TimestampedWith T_{b1} SignedWith K_{cb}^{-1}) TimestamepedWith T_{a1} SignedWith K_{ca}^{-1};

(2) B Receives ($M2$ TimestampedWith T_{a2}, T_{a1} SignedWith K_{ca}^{-1}) TimestampedWith T_{b2} SignedWith K_{cb}^{-1};

(3) S Receives $M3$ TimestampedWith T_{a2}, T_{b2} SignedWith K_{cb}^{-1};

(4) B Receives $M4$;

(5) A Receives ($M5$ TimestampedWith T_{b3}, T_{b2} SignedWith K_{cb}^{-1}) TimestampedWith T_{a3} SignedWith K_{ca}^{-1};

(6) B Receives ($M6$ TimestampedWith T_{a4}, T_{a3} SignedWith K_{ca}^{-1}) TimestampedWith T_{b4} SignedWith K_{cb}^{-1};

(7) A Receives ($M7$ TimestampedWith T_{b5}, T_{b4} SignedWith K_{cb}^{-1}) TimestampedWith T_{a5} SignedWith K_{ca}^{-1}.

本协议的初始假设条件:(A 为一般的主体符号)

A1: A CanProve(T_A IsTrustedOn T_{a1}, T_{a2}, T_{a3}, T_{a4}, T_{a5});

A2: A CanProve(T_B IsTrustedOn T_{b1}, T_{b2}, T_{b3}, T_{b4}, T_{b5});

A3: A Receives(x TimestampedWith t SignedWith K_{ca}^{-1}); (Mi SignedWith K_{cb}^{-1}) in x {同时推出}

⇒A CanProve(A Receives x At t);

A CanProve(A Receives Mi At t);

A CanProve(x FreshstampedWith t SignedWith K_{ca}^{-1});

A4: A Receives(Mi TimestampedWith $t2$, $t1$ SignedWith K_{cb}^{1})

⇒A CanProve(Mi FreshstampedWith $t2$ SignedWith K_{cb}^{1});

A CanProve(Mi TimestampedWith $t2$ SignedWith K_{cb}^{-1})

A CanProve(B Receives M_{i-1} At $t1$)

A5: K_{ca} Authenticates T_A, A; {T_A, T_B 的签名也代表了A, B 的身份}

A6: K_{cb} Authenticates T_B, B;

A7: K_a Authenticates A;

A8: K_b Authenticates B;

A9: B CanProve(B Receives{消息 2} At $t2$);

B CanProve(A Says{消息 2} At $t1$);

$t2$-$t1$ ⩽ To

⇒(B CanProve(A can send order during To);

A10: A CanProve(A Receives{消息 5} At $t2$);

A CanProve(B Says{消息 5} At $t1$);

$t2$-$t1$ ⩽ Td

⇒A CanProve(B can deliver product during Td);

A11: B CanProve(B Receives{消息 6} At $t2$);

B CanProve(A Says{消息 6} At $t1$);

$t2$-$t1$ ⩽ $Tp6$

⇒B CanProve(A can send payment key during $Tp6$);

A12: A CanProve(A Receives{消息 7} At $t2$);

A CanProve(B Says{消息 7} At $t1$);

$t2$-$t1$ ⩽ $Tp7$

⇒A CanProve(B can send product key during $Tp7$);

对 G1 的分析过程:

消息 2= ($M2$ TimestampedWith T_{a2}, T_{a1} SignedWith K_{ca}^{-1})

由 B Receives{消息 2} TimestampedWith T_{b2} SignedWith K_{cb}^{-1}; {A3}

⇒B CanProve(B Receives{消息 2} At T_{b2};

B CanProve (B Receives ($M2$ TimestampedWith T_{a2}, T_{a1} SignedWith K_{ca}^{-1})); {A4}

⇒B CanProve(B CanProve($M2$ FreshstampedWith T_{a2} SignedWith K_{ca}^{-1}))

B CanProve(B CanProve($M2$ TimestampedWith T_{a2} SignedWith K_{ca}^{-1}))

A5; A1; R7

⇒B CanProve($M2$ Freshbefore T_{a2});

B CanProve($M2$ TimestampedWith T_{a2} SignedWith K_{ca}^{-1}) {R2}

($M2$ in{消息 2}) SignedWith K_{ca}^{-1};

K_{ca} Authenticates A; {A5}

K_{ca} Authenticates T_A; {A5}

B CanProve(T_A IsTrustedOn T_{a2}); {A1}

R8

⇒B CanProve(A Says{消息 2} At T_{a2});

A9; if T_{b2}-T_{a2} ⩽ To

⇒B CanProve(A can send order during To)

同样的方法我们可以证明 G2(消息 5, A3, A4, A6, A2, R2, R7, R8, A10), G3(消息 6, A3, A4, A5, A1, R2, R7, R8, A11), G4(消息 7, A3, A4, A6, A2, R2, R7, R8, A12). 从我们的证明过程中可知, A、B 都能相互证明对方收发消息的时间. 所以在超时情况下, 各方的时限责任是可追究的.

4　结论

本文针对安全协议的时限责任问题, 提出对时间标记的分析应同时考虑责任性和新鲜性两个方面. 我们总结当前研究现状, 发现文献[6、8]中的逻辑方法只对其中一个方面进行分析, 从而导致分析能力受限并有出错的可能. 为此, 我们在 Kailar 逻辑中加入时间公式 x At t, 新鲜性标志 Freshstamp, 公式 x Freshbefore t, 和相关的两个假设规则. 从责任性和新鲜性两方面分析时间标记. 比较 Kudo 的方法, 能防止可能的重发攻击, 且逻辑体系更加简单、实用(Kudo 逻辑有 6 种语义扩展和 12 个基本假设).

参考文献:

[1] Burrows M, Abadi M, Needham R M. A logic of authentication. Report. 39[R]. Palo Alto, CA: DEC System Research Center, 1989.

[2] Subramanian S. Design and verification of secure e-commerce protocols [D]. USA: The Ohio State University, 1998.

[3] Denning D E, Sacco G M. Timestamps in key distribution protocols[J]. Comm. of ACM, 1981, 24(8): 533-536.

[4] Syverson P. Adding time to a logic of authentication[A]. In Proc. of the

1st ACM Conf. on Computer and Communications Security[C]. Fairfax: ACM, 1993. 97-101.

[5] Abadi M, Tuttle M R. A semantics for a logic of authentication[A]. Proc. of the 10th Annual ACM Symposium on Principles of Distributed Computing[C]. 1991. 201-216.

[6] Stubblebine S G, Wright R N. An authentication logic supporting synchronization, revocation, and recency[A]. Proc. of the 3rd ACM Conf. on Computer and Communications Security[C]. New Delhi: ACM, 1996. 95-105.

[7] Kailar R. Accountability in electronic commerce protocols[J]. IEEE Trans. on Software Engineering, 1996, 22(5): 313-328.

[8] Kudo M. Electronic submission protocol based on temporal accountability[A]. Proc. of 14th Annual Computer Security Application Conf. [C]. Phoenix: ACSA, 1998. 353-363.

[9] Sirbu M, Tygar J D. NetBill: an internet commerce system optimized for network delivered services[J]. IEEE Personal Communications, 1995, 2(4): 34-39.

[10] Yee B, Tygar J D. Secure coprocessors in electronic commerce applications[A]. Proc. of the 1st USENIX Workshop on Electronic Commerce [C]. New York: USENIX, 1995. 155-170.

作者简介:

梁 坚 男, 1973 年出生于湖北鄂州, 分别于 1995 年, 1998 年在湖南大学计算机系获本科、硕士学位, 1998 年进入上海交通大学计算机系, 上海分布计算技术中心, 现在的研究方向为网络安全协议分析, 分布式计算, 参与多项国家级科研项目, 已发表论文 10 余篇.

敖青云 男, 1973 年出生于江西, 1990 年进入上海交通大学, 现为计算机系博士研究生, 研究方向为密码学, 网络安全协议设计, 于 2001 年毕业. 参与多项国家级科研项目, 发表论文 10 余篇.

尤晋元 男, 1939 年出生于江苏常州, 上海交通大学教授, 博士生导师, 曾在加拿大里贾纳大学, 美国 SFSU 计算机科学系为高级访问学者, 现任上海分布计算技术中心主任, 国务院学位委员会计算机学科评议组成员, 上海计算机学会软件工程专业委员会主任, 研究方向主要为分布式操作系统、分布对象计算, 信息安全, 先后承担多项国家七五、八五、九五国家科技攻关课题, 国家自然科学基金, 863 课题.