文章编号:1007-5321(2013)05-0030-06 **DOI**:10.13190/j.jbupt.2013.05.007

基于滑动窗口机制的 RFID 自同步可扩展所有权变更协议

王 昕[1], 袁超伟[1], 黄 晨[2]
(1. 北京邮电大学 信息与通信工程学院, 北京 100876; 2. 工业和信息化部电信研究院 信息通信安全研究所, 北京 100191)

摘要: 针对可扩展射频标签(RFID)认证鉴权子协议 P_1 存储代价与寻签查找效率的矛盾和密钥更新子协议 P_2 的非同步攻击缺陷,提出了一种改进的 RFID 轻量级、可扩展、自同步所有权变更协议(LSDARP). 该协议采用滑动窗口机制,仅存储和维护标签新旧相邻特征值,既降低了存储代价又可避免产生特征值组耗尽后的寻签效率退化问题,在遭受连续非同步攻击后自行恢复同步. 在存储代价、计算复杂度、后向安全、抵御非同步攻击和 Tag Killing 攻击等方面,LSDARP 协议比 P_1、P_2 协议更优.

关 键 词: 所有权变更; 可扩展性; 非同步攻击; 射频标签协议
中图分类号: TN929.53 **文献标志码**: A

Scalable and Self-Synchronizable RFID Ownership Transfer Protocol Based on the Sliding Window Mechanism

WANG Xin[1], YUAN Chao-wei[1], HUANG Chen[2]
(1. School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China;
2. Institute of Information and Communication Security, China Academy of Telecommunication Research, Beijing 100191, China)

Abstract: Aiming at the conflict of the storage cost, the update frequency in P_1 and the de-synchronizing defect in P_2, an improved scalable radio frequency identification tag (RFID) ownership transfer protocol-lightweight scalable desynchronizing attack resistant protocol (LSDARP) was proposed. By the mechanism of sliding windows, the scheme only stores the tag's old and new adjacent eigenvalues to reduce the storage cost and avoid search efficiency degeneration when a group of eigenvalues was used out. It can be also re-synchronized when suffering de-synchronizing attacks continuously. Comparative analysis shows that LSDARP performs very well in aspects of the storage cost, computational complexity, backward security, de-synchronizing attack resistant and Tag Killing resistant.

Key words: ownership transfer; scalability; desynchronizing attack; radio frequency identification tag protocol

RFID 的所有权变更主要由寻签、鉴权和密钥更新 3 部分组成. 为了更好地提高 RFID 所有权变更协议的安全性和系统容量的可扩展性[1], Osaka 等[2-5]在前后向安全、防重放攻击、抵御 DoS 攻击、防位置追踪等方面做了深入研究,但仍无法抵御非同步攻击[6]. Song 等[7]基于预先计算的哈希链特征值组提出了可扩展的 RFID 认证鉴权协议 P_1 和密钥更新协议 P_2. P_1 协议通过预先计算并存储 $m+1$ 个特征值实现了标签匿名条件下的可扩展认证鉴权,并可通过计数器判断特征值耗尽时的临界情况.

收稿日期: 2012-12-27
作者简介: 王 昕(1982—), 男, 博士生, E-mail: wangxinsmile@gmail.com; 袁超伟(1956—), 男, 教授, 博士生导师.

研究发现，P_1 协议中存储代价同寻签查找效率存在矛盾：大 m 值可以减少特征值耗尽的频率但存储代价增加；小 m 值节省存储空间并可提高查找效率，但特征值耗尽后的效率退化频率增加. P_2 协议以 P_1 为基础，实现了标签的所有权变更，但由于缺乏最后一条消息丢失情况下的协议恢复处理，P_2 协议无法抵御非同步攻击.

笔者针对 P_1 协议存储代价与寻签查找效率的矛盾[7]和 P_2 协议的非同步攻击缺陷，提出了存储代价低、计算复杂度小，且能够抵御连续非同步攻击的可扩展 RFID 所有权变更协议 LSDARP，并使用 Java 语言对 LSDARP 协议的模拟仿真，实现了遭受非同步攻击后的同步恢复，验证了协议在抵御非同步攻击方面的良好特性.

1　LSDARP 协议

LSDARP 协议系统主要由后台系统 S 和 RFID 标签 T 组成. 其他符号和标记参见表 1.

表 1　协议中的符号

符号	描述
M	协议参与方所发送的消息
r	协议参与方生成的随机数
$\oplus$	环和运算符
$\|\|$	连接运算符
Φ,Φ'	所有 X_ϕ 的集合，所有 $X_{\phi'}$ 的集合
Λ,Λ'	所有 X_λ 的集合，所有 $X_{\lambda'}$ 的集合
$f_k(x),e_k(x)$	对 x 的带密钥哈希算法：$\{0,1\}^* \times \{0,1\}^l \rightarrow \{0,1\}^l$
$X \leftarrow$ EXP	计算表达式 EXP 的值，并覆盖存储至地址 X
M = EXP	计算表达式 EXP 的值，并用 M 代表
EXP1 = = EXP2	表达式 EXP1 的值与表达式 EXP2 的值相等
exl(x)	l bit 的字符串 x 的抽取函数：$\{0,1\}^l \rightarrow \{0,1\}^n$

LSDARP 协议主要包括初始化、寻签/鉴权和密钥更新 3 部分，分别描述如下.

1.1　初始化

后台系统 S 为每个标签新建一条包含[K_s，R_s，(X_ϕ，$X_{\phi'}$，X_λ，$X_{\lambda'}$)]字段的记录，其中 K_s 存储该标签的密钥；R_s 长度为 l bit，存储上次协议执行过程中用来计算新密钥的随机数. X_ϕ、$X_{\phi'}$、X_λ 和 $X_{\lambda'}$ 的长度均为 l bit，其中 X_ϕ 和 $X_{\phi'}$ 存储标签上次更新失败后的标签旧特征值，而 X_λ 和 $X_{\lambda'}$ 存储上次更新成功后标签的特征值.

对于每个标签，系统 S 随机生成的长度为 l bit 的密钥 k 存储在相应记录的 K_s 字段；生成 l bit 的随机数 r_0 并存储于 R_s；计算 $h_{r_0}(K_s)$，并通过安全信道将其写入对应的标签 T 并存储，地址记为 K_t；随机生成 l bit 长字符串 x_0，存储于 X_ϕ 和 X_λ；计算 $e_{h_{r_0}(K_s)}(x_0)$ 并存储于 $X_{\phi'}$ 和 $X_{\lambda'}$. 最后，S 通过安全信道将 x_0 写入标签 T，标签 T 侧的存储地址记为 X_t.

1.2　寻签和鉴权

同 P_1 协议不同，LSDARP 协议采用滑动窗口机制，通过存储和匹配新旧两组相邻标签特征值的方法，可以 $O(1)$ 的复杂度完成寻签，无需预先计算和存储标签所有可使用的特征值，无需使用计数器记录标签特征值偏移量，避免了特征值耗尽而产生的临界效率退化. 寻签和鉴权流程如图 1 所示.

1）S 生成一个 l bit 长度的随机数 r，发送至标签 T.

2）T 收到 r 后，计算 $M_1 = f_{K_t}(r \| X_t)$，$M_2 = e_{K_t}(X_t)$，发送 M_1、M_2 至 S.

3）S 收到 M_1、M_2 后，先后在集合 Λ' 和 Φ' 中查找 M_2.

情况 1　找到一个 $X_{\lambda'} \in \Lambda'$，且 $X_{\lambda'}$ = = M_2

使用 $X_{\lambda'}$ 对应的 X_λ，计算并校验 $f_{h_{R_s}(K_s)}(r \| X_\lambda)$ = = M_1. 若相等，鉴权通过，继续执行 1.3 节的密钥更新子协议；否则鉴权失败，协议终止.

情况 2　找到一个 $X_{\phi'} \in \Phi'$，且 X_ϕ = = M_2

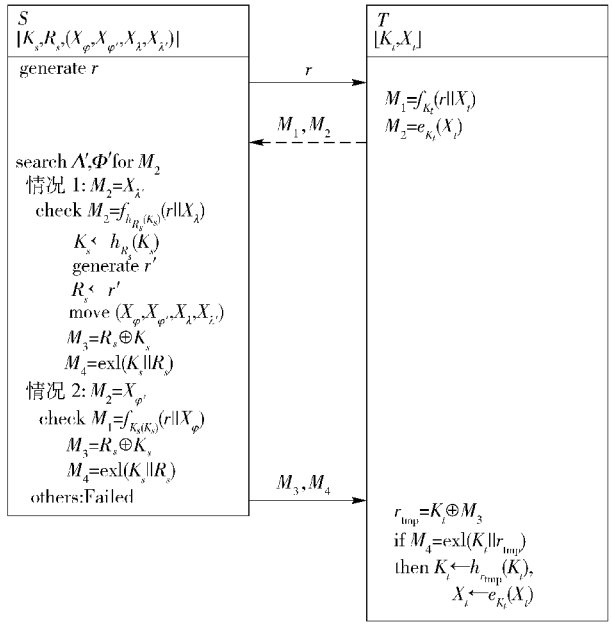

图 1　LSDARP 协议交互流程

使用 $X_{\phi'}$ 对应的 X_ϕ，计算并校验 $f_{K_s}(r \parallel X_\phi) = = M_1$ 是否成立. 若相等，鉴权通过，继续执行1.3节的密钥更新子协议；否则鉴权失败，协议终止.

其他情况：寻签失败，协议终止.

1.3 密钥更新

1）对应后台系统 S 寻签和鉴权的结果，密钥更新也相应地分为2种情况，描述如下.

情况1

S 计算并更新 $K_s \leftarrow h_{R_s}(K_s)$；生成长度为 l bit 的随机数 r'，$R_s \leftarrow r'$；移动该标签的特征值：令 $X_\phi \leftarrow X_\lambda$，$X_{\phi'} \leftarrow X_{\lambda'}$，$X_\lambda \leftarrow e_{h_{R_s}(K_s)}(X_\lambda)$，$X_{\lambda'} \leftarrow e_{h_{R_s}(K_s)}(X_\lambda)$；计算 $M_3 = R_s \oplus K_s$，$M_4 = \mathrm{exl}(K_s \parallel R_s)$，发送 M_3 和 M_4 至 T.

情况2

计算 $M_3 = R_s \oplus K_s$，$M_4 = \mathrm{exl}(K_s \parallel R_s)$，发送 M_3 和 M_4 至 T.

2）T 收到 M_3 和 M_4 后，计算 $r_{\mathrm{tmp}} = K_t \oplus M_3$，计算并校验 $\mathrm{exl}(K_t \parallel r_{\mathrm{tmp}}) = = M_4$. 若相等，则计算并更新 $K_t \leftarrow h_{\mathrm{tmp}}(K_t)$，$X_t \leftarrow e_{K_t}(X_t)$；否则协议终止.

2 对比分析

2.1 存储代价及查找效率

为提高后台系统的查找效率，P_1 协议中后台系统为每个标签存储 $m+1$ 个哈希值作为标签的特征值，在标签上送特征信息后通过全局查找来定位标签记录信息. 哈希链的长度 $m+1$ 是后台为每个标签预先计算出的特征值的个数，对应的后台系统的总存储代价为 $n(m+5)l$ bit. 在建立索引的条件下使用折半查找算法，时间复杂度为 $O(\mathrm{lb}nm)$. 可见，m 的大小直接影响协议存储代价和查找效率：大 m 值可增加标签的可用特征值，减少预先计算的特征值耗尽的频率，但存储代价增加、查找效率降低；小 m 值可减少存储代价、增加查找效率，但特征值将更新频繁.

如图2所示，所提出的协议中，哈希链上的滑动窗口机制可较好地解决这一矛盾.

LSDARP 协议不再需要计数器 c 和预先计算的 $m+1$ 个特征值，而是计算并存储新旧哈希链上相邻2个哈希值作为标签的特征值，其中 X_ϕ 和 $X_{\phi'}$ 为旧密钥哈希链上2个相邻哈希值，且 $X_{\phi'} = h_k(X_\phi)$；X_λ 和 $X_{\lambda'}$ 是新密钥哈希链上的相邻哈希值，且 $X_{\lambda'} = h_{k'}(X_\lambda)$. 集合 $\boldsymbol{\Phi}$、$\boldsymbol{\Phi}'$、Λ 和 Λ' 中的 X_ϕ、$X_{\phi'}$、X_λ、$X_{\lambda'}$ 构成了标签特征值的滑动窗口. 随着标签密钥的每次鉴权与更新，集合 $\boldsymbol{\Phi}$、$\boldsymbol{\Phi}'$、Λ 和 Λ' 向前滑动一格，以

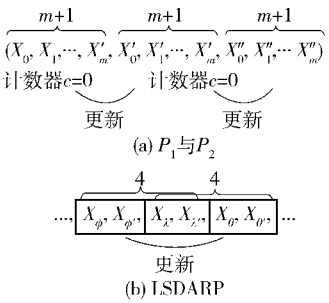

图2 特征值更新机制的对比

更小的代价来保证后台系统存储标签特征值的历史连续性. 此时后台系统特征值的存储代价由 P_1 协议的 $n(m+5)l$ bit 缩减至 $6nl$ bit，而查找特征值的时间复杂度因此由常态下的 $O(\mathrm{lb}nm)$、临界情况下的 $O(n)$ 降低至稳定的 $O(\mathrm{lb}n)$. 不仅如此，存储代价的降低和查找效率的提高还能大幅降低代理模式下对终端设备的硬件要求.

2.2 寻签及鉴权效率

P_1 协议使用计数器 c 来标识标签特征值的使用情况，计数器 c 达到阀值，即 $c=0$ 时，协议寻签需要针对后台系统的每一标签记录的 x_m 和 x_0 计算 $f_k(r \parallel M_2 \oplus x)$，并与 M_1 比较，计算量巨大，查找的时间复杂度会从正常的 $O(\mathrm{lb}nm)$ 退化到线性查找的 $O(n)$，效率退化明显. 虽然协议对于每个标签发生效率退化的概率为 $1/m$，但在实际应用中随着大批量标签的正常使用，标签特征值用尽而产生退化的情况会在某一时间段大量出现，成为系统稳定运行的隐患.

LSDARP 协议的寻签过程使用滑动窗口的机制，无需预先生成所有标签特征值，避免了计数器耗尽后的协议效率退化. 下面将以协议初始化后首次执行和正常状态2种情况分析和证明协议均可寻签及鉴权成功. 遭受非同步攻击时的寻签和鉴权过程在2.3节详细讨论.

2.2.1 初始化后首次执行

1）系统 S 及标签 T 初始化后为

$S[k, r_0, x_0, e_{h_{r_0}(k)}(x_0), x_0, e_{h_{r_0}(k)}(x_0)]$，$T[h_{r_0}(k), x_0]$

2）当 T 收到 r 后，计算 $M_1 = f_{k_{r_0}(k)}(r \parallel x_0)$，$M_2 = e_{h_{r_0}(k)}(x_0)$，并返回至 S.

3）S 先依据 $M_2 = e_{h_{r_0}(k)}(x_0)$ 查找 Λ'，将找到一个 $X_{\lambda'} = = e_{h_{r_0}(k)}(x_0) = = M_2$，寻签完毕.

4）S 使用 $X_{\lambda'}$ 对应的 $X_\lambda = x_0$，计算 $f_{h_{R_s}(K_s)}(r \parallel X_\lambda)$，且 $f_{h_{R_s}(K_s)}(r \parallel X_\lambda) = = M_1$，鉴权成功.

2.2.2　正常状态下上次执行成功

1）正常状态下系统 S 及标签 T 上次更新成功后为：$S[k, r_i, x_{i-1}, e_k(x_{i-1}), x_i, e_{h_{r_i}(k)}(x_i)]$，$T[h_{r_i}(k), x_i]$.

2）当 T 收到 r 后，计算 $M_1 = f_{h_{r_i}(k)}(r \| x_i)$，$M_2 = e_{h_{r_i}(k)}(x_i)$ 并返回至 S.

3）S 先依据 $M_2 = e_{h_{r_i}(k)}(x_i)$ 查找 Λ'，将找到一个 $X_{\lambda'} = = e_{h_{r_i}(k)}(x_i) = = M_2$，寻签完毕.

4）S 使用 $X_{\lambda'}$ 对应的 $X_\lambda = X_i$，计算 $f_{h_{R_s}(K_s)}(r \| X_\lambda)$，且 $f_{h_{R_s}(K_s)}(r \| X_\pi) = = M_1$，鉴权成功.

可见，LSDARP 协议采用存储新旧 2 次协议的相邻哈希链值的方式，避免了 P_1 协议的预先计算所有哈希链值和计数器的耗尽临界时寻签效率退化的问题. 从表 2 中可以看出，LSDARP 协议在存储代价、寻签时间复杂度和计算复杂度等方面均优于 P_1 协议.

表 2　LSDARP 协议与 P_1 协议的性能比较

协议	存储空间		正常状态寻签效率	临界状态寻签效率	正常状态计算代价		临界状态计算代价	
	S	T	S	S	S	T	S	T
P_1	$(m+5)l$	$2l+l_m$	$O(\text{lb}\, nm)$	k	1H 3PRNG $(m+4)$KHF	1H 0PRNG 4KHF	1H 3PRNG $>(n+m+4)$ KHF	1H 1PRNG 3KHF
LSDARP	$6l$	$2l$	$O(\text{lb}\, n)$	$O(n)$	1H 2PRNG 3 KHF	1H 0PRNG 3KHF	1H 2PRNG 3 KHF	1H 0PRNG 3KHF

H：哈希运算；PRNG：伪随机数生成；KHF：带密钥的哈希运算.

2.3　前后向安全

前向安全方面，LSDARP 协议中标签的每次交互信息均具有新鲜性，且标签 T 的密钥 K_t 和特征值 X_t 均由单向哈希函数生成，攻击者在获得当前标签密钥的情况下，获取标签历史信息的复杂度与破解单向哈希函数相同. 协议具有较好的前向安全特性.

P_1 协议在标签的一组特征值耗尽后进行密钥和特征值更新操作，这可以保证标签在更新前后的后向安全，即攻击者在获取到当前密钥的情况下，无法获知标签密钥更新后的后续行为. 但在标签的一组特征值使用过程中，密钥并未更新，攻击者可利用已破解的密钥跟踪标签的后续交易，直至下一次密钥更新，因此，P_1 协议在此阶段不具有后向安全特性.

LSDARP 协议中，标签 T 在每次接收 M_3 和 M_4 并验证成功后，都会更新密钥 $K_t \leftarrow h_{r_{\text{tmp}}}(K_t)$，标签的特征值哈希链将会从原来以 K_t 为密钥的哈希链跳跃至新的以 $h_{r_{\text{tmp}}}(K_t)$ 为密钥的哈希链. 在未持续跟踪和监听标签的情况下，攻击者即使获得当前标签的密钥，也无法跟踪和获知标签的下一次行为. 协议具有较好的后向安全特性.

2.4　抵御重放攻击及位置追踪

LSDARP 协议中，r 为随机数，M_3 和 M_4 的运算均依赖于随机数 R_s，所有消息对攻击者均具有随机性，攻击者无法通过重用历史交互信息而通过后台和标签的验证，因此协议能较好地抵御重放攻击.

同时，LSDARP 协议中标签 T 返回的消息 M_1 的运算中有随机数 r 参与，消息 M_2 在正常状态下每次更新，因此 M_1 同 M_2 均具有新鲜性. 攻击者无法通过记录和匹配标签的历史消息而跟踪和辨识标签的位置信息. 在遭受非同步攻击时，标签返回的 M_2 不变，攻击者可以通过记录 M_2 而标识 T，但由于标签未同步成功而无法更新密钥，所有权变更不会完成，位置追踪无效.

2.5　抵御 Tag Killing 攻击

P_1 协议中标签 T 在每次收到查询请求 r 后都计算并更新特征值 $x \leftarrow e_k(x)$. 在 Tag Killing[8] 攻击下，标签 T 将快速消耗掉所有后台预先计算并存储的特征值，且计数器 $c=0$，导致下一次与后台交互时触发密钥更新操作，后台须执行复杂度高的线性查找，系统负担加重，存在隐患.

LSDARP 协议中标签 T 在收到查询请求 r 后不更新特征值，只临时计算 $M_1 = f_{K_t}(r \| X_t)$，$M_2 = e_{K_t}(X_t)$ 并返回至后台，而只在接收后台的更新成功消息 M_3 和 M_4 并验证后，才更新密钥和特征值，避免了 Tag Killing 攻击下标签特征值的消耗对协议执

行的影响,能有效抵御 Tag Killing 攻击.

2.6 抵御非同步攻击

研究发现,所有权变更协议 P_2 对于文献[6]中所描述的非同步攻击仍有以下缺陷,描述如下.

1) 在文献[7]中6.2.1节所描述的密钥更新子协议中,后台系统 S 选择新的密钥 s' 和 r,计算并发送 (r, M_s).

2) T 接收 M_s 后,计算 $(s \parallel k' \parallel m') = M_s \oplus g_k(r \parallel r_T)$,验证 $h(s) == k$ 成功,更新密钥 k 为 k',更新计数器 c 为 m',计算 $M_T = f_{k'}(r \parallel x)$ 并发送.

3) 攻击者拦截消息 M_T,而将篡改后的消息 M'_T 发送至 S;S 验证 M'_T 失败后协议终止,S 未更新密钥 k 为 k',未更新 x_i,$1 \leqslant i \leqslant m'$.

在下一次协议执行时有:

4) 此标签 T 接收查询请求 r 后,计算 $x \leftarrow e_{k'}(x)$ 并返回至 S.

5) S 将无法再从未更新的由 k 生成的 $(x_0, \cdots, x'_i, \cdots, x'_m)$ 中查找匹配到 T 返回的 $x = e_{k'}(x)$,标签将无法再与 S 同步.

P_2 协议中,后台系统 S 使用标签 T 返回的 M_T 来判断标签是否更新密钥成功,攻击者可通过拦截和篡改关键信息 M_T 的方法,使后台 S 同标签 T 失去同步而无法继续执行协议. P_2 协议忽略了最后一条消息 M_T 的丢失对下次协议执行的影响.

实际上,RFID 的所有权变更协议是一类具有时间相关性的协议,研究其在抵御非同步攻击方面需要综合考虑2次甚至3次不同执行结果情况下的健壮性. 2.2节中详细分析并证明了 LSDARP 协议在初始化后首次执行和正常状态下均可成功寻签和鉴权,下面将分析并证明在遭受非同步攻击后,LSDARP协议仍可完成寻签鉴权和密钥更新,即在单次执行失败的情况下,标签仍可在下一次执行时自行恢复同步.

1) 正常状态下系统 S 及标签 T 上次更新失败后为:$S[k, r_i, x_{i-1}, e_k(x_{i-1}), x_i, e_{h_{r_i}(k)}(x_i)]$,$T[k, x_{i-1}]$.

2) 当 T 收到 r 后,计算 $M_1 = f_k(r \parallel x_{i-1})$,$M_2 = e_k(x_{i-1})$ 并返回至 S.

3) S 先依据 $M_2 = e_k(x_{i-1})$ 查找 Λ',由于上次协议执行失败,将无法在 Λ' 中找到一个 $X_{\lambda'} == e_k(x_{i-1}) == M_2$;之后 S 将在 Φ' 中找到一个 $X_{\phi'} == e_k(x_{i-1}) == M_2$,寻签完毕,协议执行图1中的情况2分支.

4) S 使用 $X_{\phi'}$ 对应的 $X_\phi = x_{i-1}$,计算 $f_{K_s}(r \parallel X_\phi)$,且 $f_{K_s}(r \parallel X_\phi) == M_1$,鉴权成功.

5) S 不再更新密钥而重新计算 $M_3 = R_s \oplus K_s = r_i \oplus k$,$M_4 = \mathrm{exl}(K_s \parallel R_s) = \mathrm{exl}(k \parallel r_i)$,发送 M_3 和 M_4 至 T.

6) T 收到 M_3 和 M_4 后,计算 $r_{\mathrm{tmp}} = K_t \oplus M_3 = k \oplus M_3$,并校验 $\mathrm{exl}(K_t \parallel r_{\mathrm{tmp}})$ 即 $\mathrm{exl}(k \parallel r_{\mathrm{tmp}}) == M_4$. 若相等,则计算并更新 $K_t \leftarrow h_{r_{\mathrm{tmp}}}(K_t)$,$X_t \leftarrow e_{K_t}(X_t)$;否则协议终止.

表3 P_1、P_2 协议与 LSDARP 协议的安全性比较

协议	I1	I2	I3	I4	I5	I6
P_1	√	√	*	√	×	*
P_2	√	√	*	√	×	×
LSDARP	√	√	√	√	√	√

I1:重放攻击; I2:前向安全; I3:后向安全; I4:位置追踪; I5:Tag Killing 攻击; I6:非同步攻击;

√:能够保证; *:有条件的保证; ×:无法保证.

3 结束语

针对 P_1 子协议中存储代价与寻签效率之间的矛盾以及 P_2 子协议无法抵御非同步攻击的缺陷,提出了一种改进的 RFID 所有权变更协议 LSDARP. 该协议通过滑动窗口机制减少了后台系统的存储代价,并降低了寻签的时间复杂度,避免了 P_1 协议特征值耗尽后的寻签效率退化,在保证寻签效率平滑稳定的前提下,实现了系统容量的可扩展. 同时,协议增加了对于非同步状态的判决和处理,使得后台系统和标签能在失去同步后自行恢复,解决了 P_2 协议无法抵御非同步攻击的问题,也能在连续遭受非同步攻击后有效抵御 Tag Killing 攻击. 通过对比分析可以发现,LSDARP 协议在存储代价、时间复杂度、计算复杂度和后向安全方面均优于 P_1、P_2 协议.

参考文献:

[1] Bondi A. Characteristics of scalability and their impact on performance[C]// Proceedings of the Second International Workshop on Software and Performance-WOSP 2000. Ottawa: ACM Press, 2000: 195-203.

[2] Osaka K, Takagi T, Yamazaki K, et al. An efficient and secure RFID security method with ownership transfer[C]// Proc International Conference on Computational Intelligence and Security, Volume 2. Piscataway: IEEE Press, 2006: 1090-1095.

[3] Lei Hong, Cao Tianjie. RFID protocol enabling ownership transfer to protect against traceability and DoS attacks[C]// Proc the First International Symposium on Data, Privacy, and E-Commerce. Washington DC: IEEE Computer Society Press, 2007: 508-510.

[4] Wang C H, Chin S. A new RFID authentication protocol with ownership transfer in an insecure communication environment[J]. Proc Hybrid Intelligent Systems, 9th International Conference, 2009, 1(8): 486-491.

[5] Yang M H. Across-authority lightweight ownership transfer protocol[J]. Electronic Commerce Research and Applications, 2011: 375-383.

[6] Peris-Lopez P, Hernandez-Castro J C, Tapiador J M E, et al. Vulnerability analysis of RFID protocols for tag ownership transfer [J]. Computer Networks, 2010, 54 (9): 1502-1508.

[7] Song B, Mitchell C J. Scalable RFID security protocols supporting tag ownership transfer [J]. Computer Communications, 2011, 34(4): 556-566.

[8] Han D G, Takagi T, Kim H W, et al. New security problem in RFID systems tag killing[C]// Proc in ACIS 2006. Springer-Verlag: LNCS 3982, 2006: 375-384.