

基于 USB-Key 的强口令认证方案设计与分析

于江, 苏锦海, 张永福

(信息工程大学 电子技术学院, 郑州 450004)

(herbert_net@yahoo.com.cn)

摘要:针对 OSPA 强口令认证方案无法抵抗重放攻击、拒绝服务攻击的不足,提出了一种基于 USB-Key 的口令认证方案。该方案使用 USB-Key 进行用户口令的验证并存储认证的安全参数,能够有效地保护安全参数不被窃取。认证方案在认证过程中对用户身份信息进行了保护,使用 Hash 运算计算认证参数,通过用户端和服务器端之间的认证参数的传递实现双向认证。方案的安全性分析表明,它能够防止口令猜测攻击、重放攻击、假冒攻击、拒绝服务攻击,方案系统开销小,适用于运算能力有限的终端用户。

关键词:口令认证; USB-Key; Hash 函数; 双向认证

中图分类号: TP309.2 **文献标志码:** A

Design and analysis of USB-Key based strong password authentication scheme

YU Jiang, SU Jin-hai, ZHANG Yong-fu

(Institute of Electronic Technology, Information Engineering University, Zhengzhou Henan 450004, China)

Abstract: Concerning that the OSPA protocol is vulnerable to the replay attack and the denial-of-service attack, in this paper, a USB-Key based strong password authentication scheme was proposed, which used USB-Key to verify the user's password and store the security parameter. In this scheme, user's identity can be protected by using the temporary identity and the authentication parameters computation by Hash function. This scheme can achieve mutual authentication between user and server by transferring the authentication parameters. The security analysis of the scheme proves that the scheme is resistant to replay attack, impersonation attack and Denial of Service (DoS) attack, and it has high security, and it can be used by users with limited computation ability.

Key words: password authentication; USB-Key; Hash function; mutual authentication

0 引言

随着计算机网络和分布式系统的快速发展及应用,用户在远程访问时身份的有效识别和认证,是远程安全访问的基本前提。为了实现用户的身份识别,提出了多种身份认证方案,最常见最简单的身份认证方法是通过对口令的匹配来确认用户的合法性。传统的口令用户身份认证方式存在很多问题,无法避免窃听、口令猜测、重放、假冒、字典扫描等攻击方式,不适用于安全性要求较高的网络应用系统。一次性口令(One Time Password, OTP)机制在登录过程中加入不确定因素,使用户每次登录系统时传送的口令都不同,以提高系统的安全性。但是,现有的一次性口令机制也越来越多地暴露出易猜测、易被篡改和被分析等缺点。

多数的口令认证方案如 Lamport 方案^[1]、S/Key 系统^[2]、CINON^[3]、SAS^[4]、PERM^[5]等,在认证过程中用户需要将认证请求和用户身份标识传输给认证服务器,这使得攻击者可以监视用户的认证行为和认证过程,更容易对用户的认证过程进行攻击。因此,认证过程中对用户身份标识的保护是十分重要的。

本文提出了一种基于 USB-Key 的安全高效的强口令身份认证方案。该方案在 OSPA 方案^[6]的基础上增加了用户和

服务器之间的双向认证,而且用户的认证信息保存在 USB-Key 中,在认证过程中对用户身份标识进行了保护,方案使用散列函数和异或等轻量的运算操作,有效地防止重放攻击、假冒攻击和拒绝服务攻击等手段,可以在分布式网络环境下实现对用户身份的认证。

1 OSPA 强口令认证方案分析

优化强口令身份认证协议(Optimal Strong Password Authentication, OSPA)方案是一种典型的强口令认证方案,使用 Hash 函数实现用户的认证,方案包括用户注册和认证两个阶段。OSPA 协议由 Lin-Sun-Hwang 提出,Chen 和 Ku 发现协议无法抵抗 stolen-verifier 攻击^[7],Tsuiji 和 Shimizu 指出 OSPA 协议无法抵抗中间人攻击^[8]。基于这些问题,Lin 等人^[9]对 OSPA 方案进行改进,但仍然无法抵抗重放攻击和拒绝服务攻击。

1.1 OSPA 方案

文献[9]的改进方案也包括用户注册和认证两个阶段,并且使用智能卡存储安全参数。方案中 ID_A 表示需要进行认证的用户 A 的标识, S 表示认证服务器, P 表示用户的口令, N 表示产生的随机数, $h(\cdot)$ 表示安全的无碰撞的 Hash 函数, \oplus 表示按位异或运算, \parallel 表示字符串连接操作。

收稿日期:2010-07-29。

作者简介:于江(1978-),男,山东乳山人,讲师,博士研究生,主要研究方向:网络安全; 苏锦海(1963-),男,河北张北人,教授,博士,主要研究方向:信息安全; 张永福(1942-),男,河北昌黎人,教授,主要研究方向:信息安全。

注册阶段,用户 A 产生随机数 N , 计算 $h^2(P \oplus N)$, 并将用户标识 ID_A 和 $h^2(P \oplus N)$ 一起给服务器 S 。服务器 S 保存 $h^2(P \oplus N)$ 并计算 $K = h^2(P \oplus N) \oplus h(x \parallel ID_A)$, 将 K 给用户 A , 用户将 K 和 N 保存在智能卡中。其中 x 是服务器随机选择的保护密钥, 由服务器保存。

认证阶段中的第 n 次认证过程按以下步骤进行。

1) 用户 A 使用智能卡并输入口令 P , 计算 $C_1 = K \oplus h^2(P \oplus N) = h(x \parallel ID_A)$, $C_2 = C_1 \oplus h(P \oplus N)$, $C_3 = h(C_1) \oplus h^2(P \oplus N')$ 。 N' 是用户 A 新生成的随机数, A 将 ID_A, C_2, C_3 发送给服务器 S 。

2) S 收到 ID_A, C_2, C_3 后先计算 $h(x \parallel ID_A)$, 然后计算 $Y_1 = h(x \parallel ID_A) \oplus C_2 = h(P \oplus N)$, 如果 $h(Y_1)$ 和服务器存储的 $h^2(P \oplus N)$ 相等, 那么接受用户认证, 计算 $Y_2 = h^2(x \parallel ID_A) \oplus C_3 = h^2(P \oplus N')$, 服务器用 Y_2 更新认证检验参数, 为用户下次认证使用。

1.2 OSPA 方案的不足

1) 对 OSPA 的拒绝服务攻击。

攻击者在用户的某次认证过程中, 使用相同大小的随机数 R 替换用户计算的参数 C_3 , 不改变 ID_A 和 C_2 。服务器收到参数后先计算 $Y_1 = h(x \parallel ID_A) \oplus C_2 = h(P \oplus N)$, 验证 $h(Y_1)$ 和服务器存储的 $h^2(P \oplus N)$ 是否相等, 且接受用户认证; 然后计算 $Y_2 = h^2(x \parallel ID_A) \oplus C_3 = h^2(P \oplus N')$, 由于 C_3 被攻击者替换为 R , 下次的认证检验参数 Y_2 改变为 $h^2(x \parallel ID_A) \oplus R$, 服务器更新认证参数为 $h^2(x \parallel ID_A) \oplus R$ 。虽然本次认证能正常通过, 但由于 Y_2 的改变将导致合法用户在以后的认证过程中无法被正确认证。

2) 对 OSPA 的重放攻击。

假设当用户 A 第 n 次认证前, 攻击者监听到 A 的前两次认证消息 $(ID_A, C_2^{n-2}, C_3^{n-2})$ 和 $(ID_A, C_2^{n-1}, C_3^{n-1})$ 。当 A 进行第 n 次认证时, 攻击者将第 n 次的认证消息 (ID_A, C_2^n, C_3^n) 中的 C_3^n 替换为 C_3^{n-2} 形成 (ID_A, C_2^n, C_3^{n-2}) 发送给 S , $C_3^{n-2} = h(C_1) \oplus h^2(P \oplus N^{n-1})$, 显然 $N^{n-1} = N^{(n-2)}$ 。 S 收到消息后计算 $Y_1^n = h(x \parallel ID_A) \oplus C_2^n = h(P \oplus N^n)$, 由于 C_2^n 没有被替换, 所以

$h(Y_1^n)$ 与服务器中存储的检验参数 $h^2(P \oplus N^n)$ 相等, 服务器将通过用户的认证继续计算 $Y_2 = h^2(x \parallel ID_A) \oplus C_3^{n-2} = h^2(x \parallel ID_A) \oplus h^2(x \parallel ID_A) \oplus h^2(P \oplus N^{n-1}) = h^2(P \oplus N^{n-1})$, 用 $h^2(P \oplus N^{n-1})$ 更新库中的检验参数。在用户 A 的第 $n+1$ 次认证前, 攻击者便可以使用第 $n-1$ 的认证消息 $(ID_A, C_2^{n-1}, C_3^{n-1})$ 假冒用户 A 进行认证, 因为 $h(h(x \parallel ID_A)) \oplus C_2^{n-1} = h^2(P \oplus N^{n-1})$, 服务器将库中检验参数 $h^2(P \oplus N^{n-1})$ 替换为 $h^2(P \oplus N^n)$, 攻击者还可使用 (ID_A, C_2^n, C_3^{n-1}) 进行再次认证, 这样交替使用 $(ID_A, C_2^{n-1}, C_3^{n-1})$ 和 (ID_A, C_2^n, C_3^{n-2}) 可以不断通过认证。在用户 A 的第 $n+1$ 次认证前攻击者使用 (ID_A, C_2^n, C_3^n) 认证后, 用户 A 将可正常被认证, 无法发现攻击行为。

2 基于 USB-Key 的强口令认证方案

针对 OSPA 方案的不足, 设计的方案从用户 ID 的保护、抗重放攻击、抗假冒攻击等方面入手设计认证协议, 使用保护措施较好的 USB-Key 存储用户认证信息, 实现用户和服务器的双向认证。认证方案也由用户注册过程和用户认证过程两个过程构成。

2.1 用户注册过程

1) 用户 A 生成随机数 N , 并计算 $h(ID_A \parallel N \parallel P)$ 和 $h^2(P \oplus N)$, 将用户身份标识 $ID_A, h(ID_A \parallel N \parallel P)$ 和 $h^2(P \oplus N)$ 发送给服务器 S 。

2) 服务器 S 选择服务器端的安全密钥 k , 与接收的用户注册参数计算 $SV = h^2(P \oplus N) \oplus h(k \parallel ID_A)$, 将 SV 发送给用户 A , 并存储 $ID_A, h(ID_A \parallel N \parallel P)$ 和 $h^2(P \oplus N)$, $h(ID_A \parallel NP)$ 将作为用户每次认证的可变的身份标识。

3) A 将收到的 SV 和 N 存储在 USB-Key 中, 用于以后的认证。

2.2 用户认证过程

当用户 A 第 n 次登录进行身份认证时, 插入 USB-Key 并输入口令 P , USB-Key 验证用户口令正确后进行身份认证的过程如下。

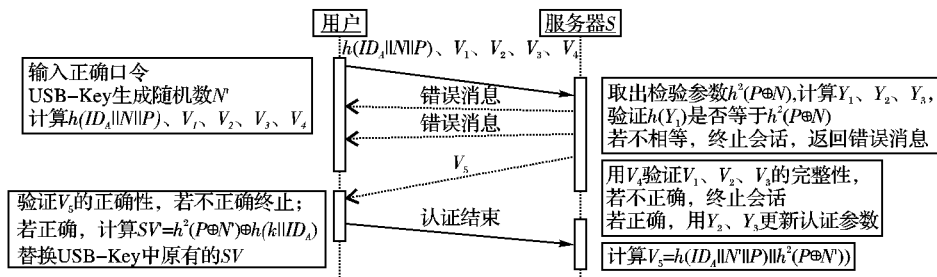


图1 用户的第 n 次认证过程

1) A 的 USB-Key 生成新随机数 N' , 取出 SV 和 N 后计算 $h(ID_A \parallel N \parallel P)$ 和 $h(k \parallel ID_A) = SV \oplus h^2(P \oplus N)$, 然后计算 $V_1 = h^2(k \parallel ID_A) \oplus h(P \oplus N)$, $V_2 = h^2(k \parallel ID_A) \oplus h(ID_A \parallel N' \parallel P)$, $V_3 = h^2(k \parallel ID_A) \oplus h^2(P \oplus N')$, $V_4 = h(h(ID_A \parallel N' \parallel P) \parallel h^2(P \oplus N')) \parallel h(P \oplus N)$ 。用户 A 将 $h(ID_A \parallel N \parallel P), V_1, V_2, V_3, V_4$ 发送给服务器。

2) 服务器收到认证请求后, 用 $h(ID_A \parallel N \parallel P)$ 在口令验证库中检索到用户 A 的身份标识 ID_A 和 $h^2(P \oplus N)$, 计算 $h(k \parallel ID_A)$ 和 $Y_1 = h^2(k \parallel ID_A) \oplus V_1 = h(P \oplus N)$, 验证如果 $h(Y_1)$ 和服务器存储的 $h^2(P \oplus N)$ 相等, 则用户为合法用户; 然后计算 $Y_2 = h^2(k \parallel ID_A) \oplus V_2 = h(ID_A \parallel N' \parallel P)$ 和 $Y_3 = h^2(k \parallel ID_A) \oplus V_3 = h^2(P \oplus N')$, 检验 $h(Y_1 \parallel Y_2 \parallel Y_3)$ 与 V_4 是否相等, 如果相等说明 V_1, V_2, V_3 未被修改, 则用 Y_2, Y_3 替换

证库中的认证参数 $h(ID_A \parallel N \parallel P)$ 和 $h^2(P \oplus N)$ 。

3) 服务器完成用户验证后, 计算 $V_5 = h(h(ID_A \parallel N' \parallel P) \parallel h^2(P \oplus N'))$, 将 V_5 发送给用户 A 。

4) 用户 A 接收到 V_5 后, USB-Key 计算 V_5 是否正确, 如果正确说明服务器没有被假冒, 使用 $SV' = h^2(P \oplus N') \oplus h(k \parallel ID_A)$ 替换 USB-Key 中原有的 SV , 认证过程结束。

3 方案的分析

3.1 安全性分析

1) 口令猜测攻击。

由于方案设计中使用了安全措施的 USB-Key, 采用安全芯片来存储和保护用户认证参数, 只有口令验证通过才能从安全芯片中取出用户认证参数, 但口令只由用户掌握, 攻击者只拿到 USB-Key 是无法获得用户认证参数的。同时 USB-Key 对用户口令的登录验证具有记录功能, 口令错误累计次数到达设定的上限值后 USB-Key 将锁死, 且口令为强口令, 从而防止攻击者的口令猜测攻击。

2) Stolen-Verifier 攻击。

假设攻击者窃取了服务器端的验证参数 $h^2(P \oplus N)$, 并假冒服务器获得用户的登录认证请求, 但由于攻击者没有保护密钥 k , 无法计算 $h(P \oplus N)$ 、 $h^2(P \oplus N')$, 也无法计算 $h(h(ID_A \parallel N' \parallel P) \parallel h^2(P \oplus N'))$ 发送给用户, 所以攻击都无法假冒服务器完成对用户进行认证。

3) 重放攻击。

由于 OSPA 方案的每次认证过程中用户的身份标识 ID 都直接传递, 所以攻击者容易根据用户的 ID 通过以前的认证消息构造用于重放攻击的消息。本方案设计中用户发起认证请求时使用的是经过 Hash 运算的用户身份标识 $h(ID_A \parallel N \parallel P)$, 且每次的运算值有不同的随机数 N 参与, 攻击者无法获得每次受保护的标识, 所以攻击者想重放以前的认证消息时无法直接构造当前用户的认证身份信息。同时, 认证过程中都有服务器生成新的随机数参与认证值的运算, 随机数 N 值够大时保证每次的认证参数计算值都不同, 攻击者重放已经截获的信息无法通过服务器的认证, 且认证参数 V_1 、 V_2 、 V_3 和 V_4 在协议运算过程中是相互关联的, V_4 保证各参数的完整性, 修改任何一个参数都会导致认证的失败。

4) 拒绝服务攻击。

攻击者使用对 OSPA 的拒绝服务攻击的方式时, 会试图通过改变 V_2 、 V_3 来进行拒绝服务攻击, 因为 V_2 、 V_3 中包含以后认证参数 $h(ID_A \parallel N' \parallel P)$ 和 $h^2(P \oplus N')$, 而且由于服务器要通过 V_4 来验证 V_1 、 V_2 、 V_3 的完整性, V_2 、 V_3 改变后将无法通过完整性验证, 此次认证将会失败, 服务器也不会修改口令库中的参数。并且每次完整性验证中还包括随机数 N , 攻击也无法使用以前的认证消息来构造新的有效的认证消息。

3.2 效能分析

1) 方案设计的认证结构以 USB-Key 的硬件平台为基础, 在认证过程中只需要在用户端和服务端进行认证参数计算及验证, 双向的认证过程不需要第三方的参与, 认证结构简单。

2) 方案中用户端的口令验证, 随机数的生成和认证参数 V_1 、 V_2 、 V_3 和 V_4 的计算在 USB-Key 中进行, 在保证安全的前提下能够快速计算。一次完整的双向认证过程用户端只需要两次参数的计算并向服务器传递 1 组认证参数, 服务器端也只需要两次参数的计算和验证, 并向用户端传递 1 组认证参数。

3) 认证过程中的认证参数的计算只用到了 Hash 运算和异或运算, 用户端在第 1 次参数计算时需要执行 5 次 Hash 运算, 服务器端在进行参数验证时需要执行 4 次 Hash 运算, 计算给用户认证服务器的参数时需要执行 1 次 Hash 运算, 最终用户认证服务器时需要 1 次 Hash 运算。从整体来看方案中的运算量较小, 且没有复杂的模运算, 所以认证过程的执行效率较高。

4 结语

针对 OSPA 口令认证方案存在的不足, 本文提出了一种 USB-Key 的双向强口令认证方案。该方案使用 USB-Key 来保护用户的口令登录和认证参数, 设计了双向的认证过程, 能够满足用户和服务器的双向认证要求; 认证过程对用户的身份标识使用变化随机数参与的运算进行保护, 并使用随机数参与口令认证参数的生成, 使认证协议能够防止重放攻击、假冒攻击、拒绝服务等攻击。方案的运算开销小, 具有较高的安全性, 适合轻量级用户使用, 与应用系统集成来实现用户身份认证是十分有效的。

参考文献:

- [1] LAMPORT L. Password authentication with insecure communication [J]. Communications of the ACM, 1981, 24(11): 770-772.
- [2] HALLER N M. The S/KEY one-time password system [C]// Proceedings of the Internet Society Symposium on Network and Distributed Systems. U S: RFC, 1995.
- [3] SHIMIZU A. A dynamic password authentication method by one-way function [J]. Systems and Computers in Japan, 1991, 22(7): 32-40.
- [4] TSUJI T, KAMIOKA T, SHIMIZU A. Simple and secure password authentication protocol (SAS) [J]. IEIC Technical Report, 2002, 102(314): 7-11.
- [5] SHIMIZU A, HORIOKA T, INAGAKI H. A password authentication methods for contents communication on the Internet [J]. IEICE Transactions on Communications, 1998, E81-B(8): 1666-1673.
- [6] LIN L, SUN H M, HWANG T. Attacks and solutions on strong-password authentication [J]. IEICE Transactions on Communications, 2001, E84-B(9): 2622-2627.
- [7] CHEN M, KU W. Stolen-verifier attack on two new strong-password authentication protocols [J]. IEICE Transactions on Communications, 2002, E85-B(11): 2519-2521.
- [8] TSUJI T, SHIMIZU A. An impersonation attack on one-time password authentication protocol OSPA [J]. IEICE Transactions on Communications, 2003, E86-B(7): 2182-2185.
- [9] LIN C W, SHEN J J, HWANG M S. Security enhancement for optimal strong-password authentication protocol [J]. ACM Operating Systems Review, 2003, 37(2): 7-12.