**密码存储中MD5算法的安全性分析**

摘要 - 哈希算法通常用于转换密码理论上进入散列是不可能的解密 本文分析了哈希算法MD5密码存储的安全隐患和讨论不同的解决方案，如盐和迭代散列。 我们提出了一种在密码存储中使用MD5的新方法使用外部信息，计算盐和随机密钥在MD5计算之前加密密码。 我们建议使用按键拉伸进行哈希计算较慢并使用XOR密码来制作最终的哈希值不可能在任何标准的彩虹表中找到。

关键词组件; MD5; 密码存储安全; 数据安全; 数据攻击 rainbow tables;

1. 介绍

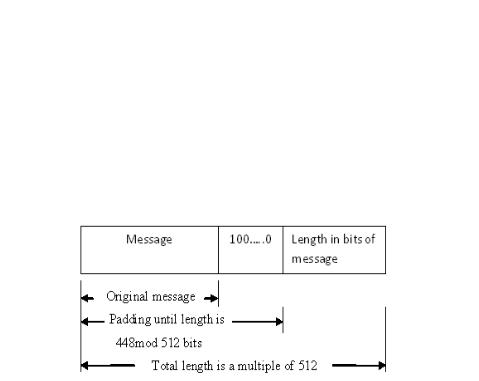
随着计算机技术的出现，它变得越来越多有效地将信息存储在数据库中而不是存放在纸质文件中。 Web应用程序，需要用户身份验证通常通过验证输入的密码将它们与通常的真实密码与存储在公司的私人数据库中的密码进行比较。 如果数据库因此这些密码将被破坏，攻击者可以无限制地访问这些用户个人。 现在，数据库使用哈希算法保护存储的密码，但仍然有安全性威胁 最近在2012年，俄罗斯黑客发布了一个大的破解密码列表来自社会的有名的网站包括LinkedIn。 这些袭击是成功被发现是由于使用弱散号算法。

1. 哈希功能

散列函数是单向加密功能采用可变大小的输入明文m，大小哈希输出。在计算上难以破译哈希和任何试图破解它实际上是不可行的。一个“安全”哈希函数应该能够抵制前图像攻击和碰撞攻击。由于鸽子的原理和生日悖论，将会有一些投入产生相同的哈希结果。输出长度固定大小128位，共计2128个可能的输出哈希值。这个值可能看起来很大，并不是无限的，从而导致碰撞。

A. MD5算法

MD5（Message Digest Algorithm 5）由Ron Rivest于1991年设计。MD5处理可变长度消息转换成128位的固定长度输出。MD5是流行哈希算法。它适用于512位的消息块，每个进程通过4轮进行，每轮进场处理16个子块（每个32位）。512位的消息在处理之前分为16个子块。如果一个消息块不高达512位，如图所示填充图。 可以找到该算法的详细说明在[1]。



图片1.填充后的消息长度（以位为单位）

III. MD5算法在密码安全存储中的应用

在非常不安全的数据库中存储密码是非常不安全的。 为了增加密码的安全性，MD5算法用于哈希原始密码和哈希值，而不是明文存储在数据库。在认证期间，输入密码也是以类似MD5的方式散列，结果为用户哈希值与数据库中的特定哈希值相比较。

IV.MD5安全分析

MD5算法容易出现两种主要的攻击类型：字典攻击和彩虹表。

1. 字典攻击

在字典攻击中，攻击者尽可能地尝试名为字典的详尽列表中的密码。 的

攻击者从字典中删除每个密码对受损的散列进行二进制搜索密码 这种方法可以通过预处理快得多，计算这些可能密码的哈希值将它们存储在哈希表中。

1. 彩虹表

彩虹表由哈希链组成，比哈希表更有效，因为它们优化了存储要求，虽然查找做得稍慢一些。彩虹表与哈希表不同使用还原和哈希函数创建。 减少明文是

不是哈希值所在的原始明文生成，但另一个通过交替散列函数具有缩减功能，交替密码链并形成哈希值。只有第一（连锁起点）生成的最后一个明文（链的终点）存储在表中。 要解密散列密码，我们先进行处理通过减少功能的哈希密码直到我们找到一个匹配链的终点。然后我们拿这条链相应的起点和重新生成哈希链和

找到散列密码的原始明文。彩虹表现在很容易在网上找到。有许多密码破解系统和使用彩虹的表网站也是，例如:OphCrack，当然，利用彩虹表破解密码系统不能保证100%的成功率。然而，用于创建彩虹表的字符集越大是哈希链长度越长，彩虹表越大。

V. 对策研究

A. 信息熵

密码强度通常是根据信息熵。 简单来说，越高信息熵越大，密码越大更难的是破解它。 6个字符的密码一个暴力攻击，一个密码为42个字符将需要242次尝试。 可以看出，越长密码和它所在的字符集越大派生，密码越强。 作为最佳做法和初步要求，应用要坚持用户在注册期间使用强密码处理。 强密码运行的风险较低字典 常见的简单密码如“123456”已被Microsoft Hotmail禁止。

B. 盐

成功密码的最常见原因之一像LinkedIn这样针对LinkedIn的破解攻击是因为他们使用无盐哈希。 这使得它变得更加容易黑客通过使用彩虹表破解系统，特别是考虑到许多用户使用非常普遍的事实，简单的密码和这些类似的密码导致

类似哈希 盐是次要的信息由附加到的字符串组成明文然后哈希。 盐渍使密码更加丰富对彩虹桌的耐盐渍加密密码会具有更高的信息熵，因此更不可能存在于预先计算的彩虹表中。 通常，盐应至少48位。 盐可以使用

以下方式：

1）

所有密码的单盐：鉴于盐是足够长而复杂的标准彩虹台，不能用来破译盐渍的哈希。 但是，两个相同的密码仍然会产生相同的散列。

2）

不同的随机盐为每个密码和存储数据库内的盐：如果我们使用不同的盐对于每个密码，两个相同的密码将有所不同哈希 攻击者必须生成不同的彩虹表对于每个用户来说，使计算更难为攻击者破解散列。 这些盐可以储存在数据库中的明文。 盐的目的不是是秘密的，但要随意足以打败使用彩虹表。

3）

使用现有列值：现有列像用户名一样的值可以用作盐。 这个解决方案是类似于上面讨论的第二个解决方案。 它击败了使用标准彩虹表，但黑客可能会生成一个彩虹表的特定用户名，例如“root”或“管理员”。

4）

使用不同位置的计算盐：盐位置可以是前缀（盐附加在密码前面），中缀（盐附加在密码中）或后缀（盐附加在密码的末尾）。 如果盐的位置是做随机，然后破解密码变得更加困难。例如，我们可以将盐位置设置为等于密码的长度模数3.盐可以通过计算使用随机字符序列（存储在数据库中）并使用其他字符（嵌入代码中）。 对于例如，盐可以作为第一个的组合两个字母的用户名，随机的字符序列最后2个字母的用户名。

5）使用可变的计算盐包括数据库外的信息和应用程序代码：黑客现在必须打破数据库和服务器包含应用代码。 他也需要获得破解密码所需的其他信息。

C. 改进MD5处理

可以使用以下方法来改善MD5加工：

1）

功能被更改，例如使用以下之一功能如（1），（2）和（3）所示：

hash = Hash（password + salt） （1）；

hash = Hash（Hash（password）+salt） （2）；

hash = Hash（password + salt + key） （3）；

2）

迭代散列：密码散列数字的时代。 MD5是一个快速散列函数，就是它计算速度快。 迭代散列使得计算速度较慢，因此计算速度较慢难以破解 迭代次数通常可以是等于1000。

3）

关键拉伸：这使得密码更具抵抗力通过使攻击工作量进行预计算攻击大。 使用迭代散列，在那里提供弱键进入哈希算法，输出结果更强键。 有3个按键拉伸方法取决于用于迭代散列的输入：

1. 简单按键拉伸：只有键反复地被散列，如（4）。 不涉及盐

key = Hash（key） （4）；

b）密码键拉伸：密码随附关键都在循环中使用。

c）盐密钥拉伸：密钥，密码和盐在循环中使用。 这种方法是三大关键之一拉伸方法。

4）

在哈希之前转换密码：在计算密码的MD5哈希值之前，后者是使用简单的密码方法进行转换。

5）

链接方法和异或（异或）密码：If使用迭代散列，从当前的哈希输出迭代用于下一次迭代的输入。 我们用一个简单的XOR密码通过“XORing”计算最终的散列来自所有迭代的哈希输出值。 一个简单的异或密码通常是（5）所示的形式。 如果是关键做了随机的，密文将会几乎不可能破解。

plaintext XOR key = ciphertext （5）；

d）改进的MD5处理的例子

我们现在将演示如何在其中填入密码数据库使用MD5的改进版本。有五个主要步骤涉及首先，随机长度的随机密钥串是第一个生成。其字符集为{0-9，a-z，A-Z}密钥字符串用于生成复杂的密码也存储在数据库中供以后在密码期间使用认证。其次，密码被转换成一个复杂的密码通过柱状转置密码。假设随机密钥是“YDCiA”，密码是“crazyrichard”，密码首先转换为矩阵5列（与随机键的长度相同）和空白电池交替填充“\*”和“@”，如图所示图2使用柱状转置密码，复合体生成的密码是“ya \* ac \* ridcrrzh @”。第三，通过找到XOR值计算盐的随机密钥字符串与复合密码。在我们的例子中，盐是“％i \ b \”s6 \* \ r \“”。

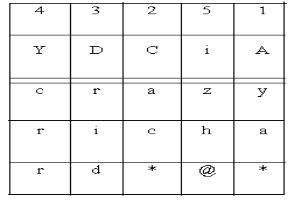


图2：通过柱状转置生成复杂的密码

第四，一个128的附加随机信息字符串为每个用户生成位并存储在外部文件中，例如 在闪存驱动器。最后，使用基于公式的哈希密码钥匙拉伸。 散列过程类似于密码块链接方法，其中一轮的输出为用于另一轮的输入，如图1所示。 来自一次计算哈希值的异或结果与前一轮的一个，由此产生的散列值在任何标准的彩虹表中都不可能找到。然后将最终散列密码存储在数据库中。系统通过计算哈希值对用户进行认证（随机密钥从数据库检索以供使用），然后将其与存储的散列密码进行比较。 一个哈希密码如何出现的例子数据库如图1所示。 4。

finalHash = HV0 ^ HV1 ^ ……^ HVN ,

HV0 = Hash (CpxPassword, additionalInfo)

HV1 = Hash (CpxPassword, HV0, salt);

HVN= Hash (CpxPassword, HVN-1, salt);

N is the number of iterations and ^ is XOR.

HV: Hash Value and

CpxPassword: Complex Password

图3.使用改进的数据库中的散列密码示例

MD5

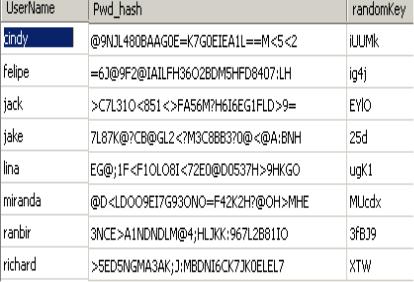


图4.使用改进的数据库中的散列密码示例MD5

总体算法可以总结在图1中。 5.这里使用的初始化向量是附加信息，它是128位的随机串。 每个用户都有一个不同的初始化向量值。

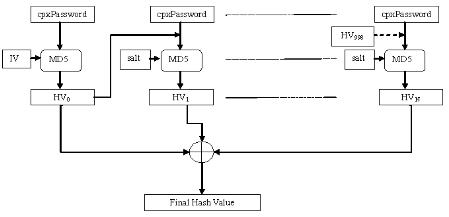


图5.改进的MD5处理（IV：初始化向量）

VI. 性能分析

首先，使用标准的彩虹表进行攻击会失败，因为哈希密码存储在数据库不是十六进制形式，因此不会存在于任何标准的彩虹表中。 我们在网上试了几个MD5解密工具，如http://www.md5decrypter.co.uk/并下载了像Cain和Abel这样的软件工具。 但，因为所有这些都使用MD5散列的彩虹表是十六进制形式，我们存储的哈希值是ASCII，攻击从一开始就已经失败了，如图所示图. 6。

另外，通过异或运算来自每个的输出哈希值迭代使得几乎不可能找到原来的第一轮的哈希输出。 一般来说，如果我们XOR的话明文用密钥计算密文，我们可以得到只需使用异或密文来返回明文用钥匙 但是，如果密钥是随机的，那么几乎是这样不可能得到明文。 在我们的例子中，关键是完全随机的，因为我们散列中间哈希输出值，因此给出最终的哈希值，这是不可能的解密它。



图6 Cain和Abel中MD5哈希中的字符无效

另外，即使两个用户有两个相同的密码，用于加密密码的随机密钥将不同，

导致不同的复杂密码。 此外，初始化向量和盐对于每个用户是不同的。 对于

两个用户具有相同的密码，最终的哈希值会完全不一样。

VII. 结论

密码存储安全是数据的一个重要方面安全性，因为大多数系统现在需要验证方法使用密码。哈希算法如MD5通常用于加密明文密码理论上不能被黑客破坏的字符串由于它们的单向加密功能。但随着时间的流逝，通过使用字典表可以实现攻击

和彩虹桌。在本文中，我们讨论了不同的阻止这些攻击的方法：（1）使用强力密码减少它存在的可能性词典，（2）使用盐，（3）键拉伸和迭代散列使​​MD5计算速度更慢，（4）链接方法，其中在输入中使用一个迭代的输出的下一次迭代和使用不同的初始化每个密码的向量，（5）加密密码之前散列和（6）XOR密码来做出最终的哈希值不可能在任何彩虹表中找到。一个实现所提出的方法是使用C＃进行的编程语言和Microsoft SQL Server as数据库。随着我们提出的方法，攻击者现在将会必须入侵数据库，服务器包含应用程序代码以及外部文件。

**致谢**

本文报告的工作得到了支持中国北京自然科学基金（GrantNo. 4112037）。作者要感谢并承认亲戚朋友和同事的支持和协助**。**

**参考文献**

[1] Rivest, R. The MD5 message-digest algorithm. RFC 1321, 37 (April 1992)

[2] Zhang Shaolan,Xing Guobo,Yang Yixian,Improvement and Security Analysis on MD5 [J].Computer Application, 2009, vol. 29(4):947-949.

[3] Xiaoling Zheng,JiDong Jin,Research for the Application and Safety

of MD5 Algorithm in Password Authentication,9th International

[4] H. Mirvaziri,Kasmiran Jumari,Mahamod Ismail,Z.Mohd Hanapi, A

new Hash Function Based on Combination of Existing Digest Algorithms ,The 5th Student Conference on Research and Development–SCOReD 2007,11-12 December 2007, Malaysia.

[5] Md.Didarul Alam Chawdhury,and A.H.M.Ashfak Habib,Security Enhancement of MD5 Hashed Passwords by using the Unused Bits of

TCP Header,Proceedings of 11th International Conference onComputer and Information Technology (ICCIT 2008)25-27 December, 2008, Khulna, Bangladesh