

**本 科 生 毕 业 论 文（设计）**

**开题报告**



**学生姓名: \_\_\_宋志平\_\_\_\_\_\_\_\_\_**

**学生学号: 3120101274**

**指导教师: 卜佳俊**

**年级与专业: 大四 计算机科学与技术**

**所在学院: 计算机学院**

**一、题目： 个人密码的同步管理应用**

**二、指导教师对开题报告、外文翻译和文献综述的具体要求：**

**指导教师（签名）**

**年 月 日**

**毕业论文（设计）开题报告、外文翻译和文献综述考核**

**答辩小组对开题报告、外文翻译和文献综述评语及成绩评定：**

|  |  |  |  |
| --- | --- | --- | --- |
| **成绩比例** | **开题报告**  **占（20%）** | **外文翻译**  **占（10%）** | **文献综述**  **占（10%）** |
| **分 值** |  |  |  |

**开题报告答辩小组负责人（签名）**

**年 月 日**

目 录

[本科毕业论文（设计）开题报告 6](#_Toc446284757)

[1. 课题背景 6](#_Toc446284758)

[2. 目标和任务 9](#_Toc446284759)

[3. 可行性分析 12](#_Toc446284760)

[4. 研究方案和关键技术考虑 12](#_Toc446284761)

[5. 进度计划 13](#_Toc446284762)

[本科毕业论文（设计）文献综述 14](#_Toc446284763)

[本科毕业论文（设计）外文翻译 15](#_Toc446284764)

本科毕业论文（设计）开题报告

1. 课题背景

随着互联网的快速兴起，如何管理自己众多的账号、密码就成为了一个问题。一方面是忘记账号、密码给人带来的麻烦，找回账号、密码需要一定的时间。另一方面是安全性上的挑战，很多人的密码过于简单，并且多个账号共用一个密码，一旦被破解可能会带来连环损失。

在进入了互联网云时代之后，这个问题显得更为重要。与PC时代的单一终端不同，现在的人往往拥有多个互联网终端设备：PC、手机、平板等。因此，在不同的平台上，同步这些私人重要信息显得尤为重要。

另外，还有一些非个人账号密码类的重要信息需要保存，它们往往难以记忆。例如服务器和数据库的登录信息、重要人物的联系方式(手机、Email)、软件序列号等等。

结合当今流行的几款互联网产品和使用经验来看，需要机密保存的对象可以被简单地分为以下几种

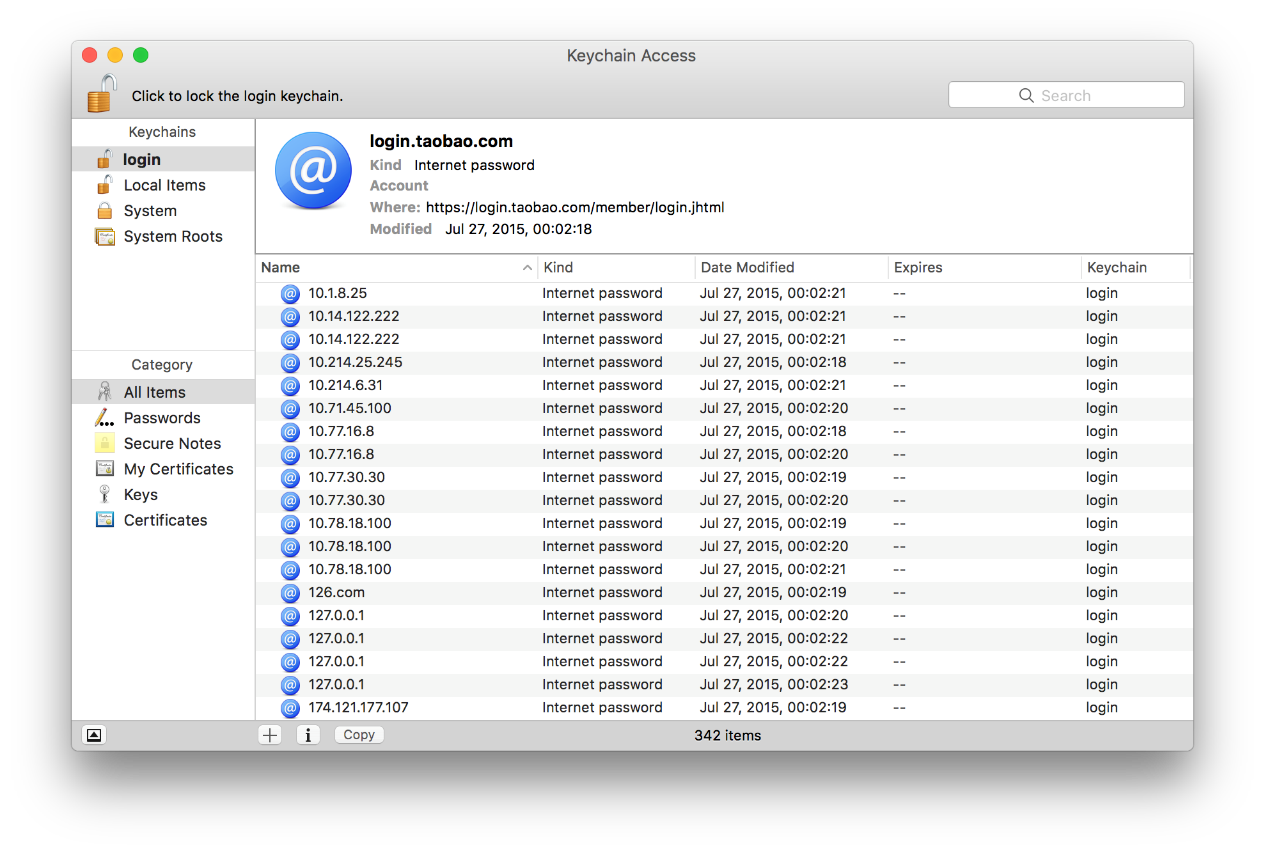
* 非重要的账号、密码。他们多数为一些论坛、网络游戏的账号密码。在数量上，这一类型的账号、密码最多，用户忘记账号、密码可以说是常事。它们往往可以比较容易地被找回，一旦发生账号、密码被盗的情况，对于用户而言损失有限。
* 重要的账号的密码，例如QQ、微信、Facebook等，除了社交账号之外，一些网站的高价值密码也可被归入其中，例如GitHub。这些账号对于个人用户而言价值颇高，一旦被盗，会对用户造成不小的损失。他们往往也是被盗次数最多的账号，例如在公共网络或是网吧等环境下登录QQ，很容易产生盗号情况。
* 关系到个人财产的账号密码。例如银行、证券账户的密码。这些账号的价值很高，一旦被盗往往会带来不可挽回的损失。而银行、证券账户的密码往往比较简单，多为6位数字。银行、证券账户的密码又难以被暴力破解，银行系统往往有输入次数限定。6位的数字方便记忆，一般而言不会被暴力破解，通常而言不适合存储在云端。
* 服务器、数据库系统的登录信息。服务器地址、端口、登录帐号、密码往往难以记忆，被盗也会带来一定的损失。现在，有一些保险箱应用提供了保存这些信息的选项。
* 重要的非密码类信息。例如个人隐私的笔记、银行卡账号、重要人物的联系方式等。

下面介绍部分市面上的类似产品：

* Keychain Access

钥匙串是苹果公司Mac OS中的密码管理系统。它在Mac OS 8.6中被导入，并且包括在了所有后续的Mac OS版本中，包括Mac OS X。一个钥匙串可以包含多种类型的数据：密码（包括网站，FTP服务器，SSH帐户，网络共享，无线网络，群组软件，加密磁盘映像等），私钥，电子证书和加密笔记等。

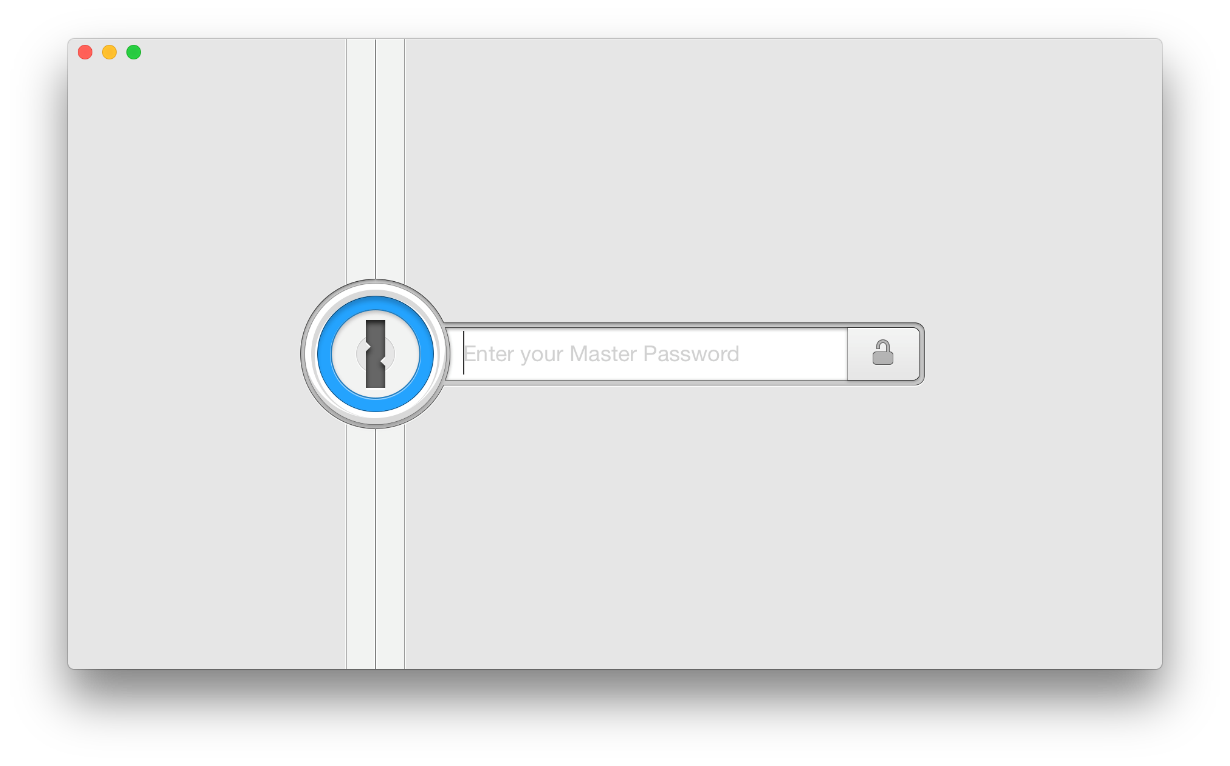
钥匙串访问（Keychain Access）是一个Mac OS X应用程序，它允许用户访问和配置钥匙串的内容（包括网站，FTP服务器，SSH帐户，网络共享，无线网络，群组软件，加密磁盘映像等内容的密码等），加锁或解锁钥匙串，显示系统存储的密码，管理根证书，密钥和加密笔记。

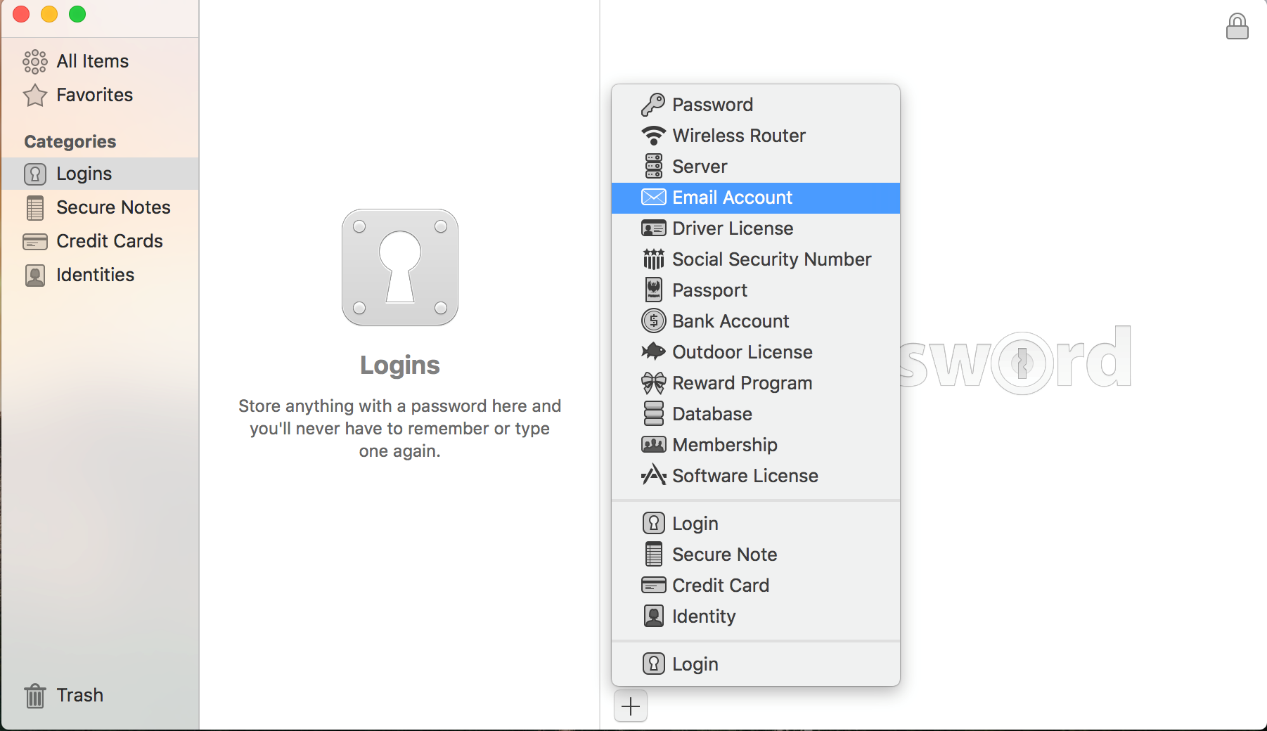


* **1password**

1Password 是一个由 AgileBits 公司开发的密码管理软件。它能用来存放各种不同的密码，除普通登录密码，信用卡、软件许可证等敏感信息也可一同存放在PBKDF2加密的虚拟保险箱里，由主密码保护。1Password支持 Windows、Mac、iOS、Android 等主流操作系统，Mac / iOS 版本已经支持中文。

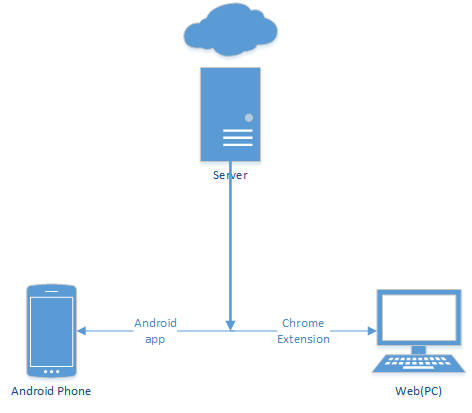
1Password 可以在本地存放密码，不与任何远程服务器接触。也可以选择与 Dropbox（全平台），本地 Wi-Fi（仅 Mac 与 iOS 设备），或 iCloud（仅 Mac 与 iOS 设备）同步。只需记住一个主密码（Master Password），并用其来保护并管理所有密码，是1Password 密码管理方案的主要理念。





1. 目标和任务

制作一个轻量化的个人密码的同步管理应用，它可以跨平台(Android/Web)同步用户的私密个人信息。另外，制作一个在Chrome浏览器上的应用插件，方便用户管理密码。其中Server端使用流行的云服务器，例如腾讯云、阿里云等。



本应用以一个主密码(Master Password)对用户密码进行管理，用户只需要记住Master Password，就可以保护和管理所有密码。

将用户的私密数据分为以下几类：

* 登录帐号(Logins)，用于存放 某网站/某应用的用户名和密码。

其中至少有如下的信息：

-网站名(website)：用于标识在哪个网站

-用户名(username)：用户名

-密码(password)：用户名所对应的密码

-标签(label)：标识它属于哪一类的登录帐号

-注释(notes)：与本帐号相关的一些注释、辅助信息

* 银行卡信息(Credit Cards)，用于存放银行卡相关信息，但并不包含银行卡密码。

其中至少有如下的信息(部分可选)：

-类型(type)：借记卡、信用卡或是其他类型

-银行卡号(number)：记录银行卡号

-信用卡品牌(bands)：可多选，提供常见的信用卡品牌供选择，例如Visa、MasterCard、UnionPay、JCB等。

-起始期和有效截止期(valid from & expiry date)：记录信用卡的日期信息

-标签(label)：标识它属于哪一类的银行卡(工资卡、双币卡等)

-注释(notes)：与本银行卡相关的一些注释、辅助信息

* 身份名片(Indentity)，用于存放人的信息，一般而言是重要人物。

其中至少有如下的信息(部分可选)：

-姓名(name)-性别(sex)

-生日(birth date)

-电话(phone/mobile)

-电子邮箱(email)

-公司(company)

-标签(label)：标识它属于哪一类的人(同学、同事、上司等)

-注释(notes)：与他相关的一些注释、辅助信息

* 服务器信息(Server)，用于存放服务器的登录、服务提供商信息等。

其中至少有如下的信息(部分可选)：

-地址(address)：服务器的地址

-用户名(username)：登录时的用户名

-密码(password)：用户名对应的密码

-端口号(port)：记录端口号

-标签(label)：标识它属于哪一类的登录用户

服务提供商(Hosting Provider)的

-名字(name)

-网址(website)

-联系方式(phone/email)

-注释(notes)：与此相关的一些注释、辅助信息

* 数据库信息(Database)，用于存放数据库的信息。

其中至少有如下的信息(部分可选)：

-数据库类型(type)：例如Mysql、SQL Server、Oracle等

-服务器和端口(server & port)：记录数据库的服务器地址和端口号

-用户名(username)：登录时的用户名

-密码(password)：用户名对应的密码

-数据库名称(database)

-连接方式(connection options)

-标签(label)：标识它属于哪一类的数据库

-注释(notes)：与该数据库相关的一些注释、辅助信息

个人密码的同步管理应用还需要拥有主密码的控制器，当应用在手机上被切出或是一段时间不用后，需要再次输入Master Password。这样做可以减少密码或是关键信息被偷窥的可能，增强了程序的安全性。

除此之外，考虑到有一部分人有使用随机密码的可能性，因为这样可以降低通过字典被暴力破解的概率。个人密码的同步管理应用还需要一个密码生成器，用户可以指定密码的位数、数字的数量和特殊字符的数量来生成随机密码。

1. 可行性分析

* 技术可行性

从技术上来说，本应用主要有4个技术上的部分：加密算法、Android和Web前端、服务器后端和Chrome插件扩展程序。

这4个部分上均有相对成熟的技术和解决方案，具体的关键技术考虑，放在第4节。

* 实用可行性

从实用性的角度看，对于大多数互联网用户而言，有管理密码的需求。

现在的互联网应用更强调轻量化，在PC端通过浏览器交互比通过客户端交互 更为轻量化，今天市面上的大多数产品在Windows/Mac端仍然离不开客户端。 此次毕业设计在应用的轻量化有一定的创新性。

1. 研究方案和关键技术考虑

* 加密算法

考虑使用AES(Advanced Encryption Standard，高级加密标准)加密算法对用户的密码进行加密。AES是美国联邦政府采用的一种区块加密标准。这个标准用来替代原先的DES，已经被多方分析且广为全世界所使用。经过五年的甄选流程，高级加密标准由美国国家标准与技术研究院（NIST）于2001年11月26日发布于FIPS PUB 197，并在2002年5月26日成为有效的标准。2006年，高级加密标准已然成为对称密钥加密中最流行的算法之一。

AES在软件及硬件上都能快速地加解密，相对来说较易于实现，且只需要很少的内存。并且，AES在运行Android系统的ARM上 已经被广泛应用。

* Web前端和加密通信

考虑使用Javascript写前端，并将前后端分离，使用加密通信的方式传输数据。

* 服务器后端

考虑使用互联网云服务器，打算在Linux服务器上搭建MySql数据库系统。

相对NoSQL式数据库，传统的关系型数据库更为稳定，对数据库的ACID要求更为严格，尤其更契合本应用对原子性和一致性的要求。

对数据的一次增加/修改/删除应该被看作一次标准的transcation，因此使用传统的SQL式数据库更为合适。

1. 进度计划

* 在4月中旬完成服务器后端的设计，并有简单的前端页面。
* 在5月上旬完成Android端的设计，并有完成版的前端页面。
* 在5月下旬完成对Chrome插件扩展程序的设计，预留出时间进行bug的修改。

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

\* 注：目录供参考

本科毕业论文（设计）文献综述

本科毕业论文（设计）外文翻译

**原文：On the Security of Password Manager Database Formats**

[**http://link.springer.com/chapter/10.1007/978-3-642-33167-1\_44**](http://link.springer.com/chapter/10.1007/978-3-642-33167-1_44)

**标题：聚焦密码管理器数据库格式的安全性**

**摘要**

密码管理器是一类重要的软件，用户用它去安全地保存高价值和敏感的信息，从网上银行密码、登录信息到护照和社保卡账号。令人惊讶的是，目前鲜有针对这些应用程序提供的安全性所做的学术研究。

本文带来了第一个严谨的分析，就这些流行的密码管理器的存储格式。我们定义了两个现实的安全模型，用于表示真实世界中敌人的能力。接下来，我们将会展示当我们的模型被敌人攻击时，是如何的不堪一击。我们的分析表明了绝大多数的密码管理器数据库格式，甚至会被那些水平低的攻击者攻破。

1. **引言**

互联网上提供的服务数量不断增加，平均每个用户所需要记住的密码数量也相应地不断增加，以至于对于大多数用户而言，记住每一个账户的新高强度密码，成为了不可能。

用户通常有两种方式来解决这一个问题。一种常见的方式是在不用的网站上复用相同的密码。这种方式增加了潜在风险，如果密码被盗、被破解或是有一个服务向攻击者妥协，攻击者就可能在用户的所有在线服务上复用这些密码。另一种方式是使用“密码管理器”，对每一个不同的网站存储高强度的(随机的)密码。密码管理器是一种软件，它要求用户记住一个高强度的密码用来访问用户密码的数据库。记住一个高强度的密码，对于那些在不同的在线服务设置不同的密码为得到安全性的用户而言更为合适。

使用密码管理器还有其他的潜在好处。完整的URL（或者是域名）往往和对应的密码存储在一起，用来自从填写登录表单。因此，那些依赖密码管理器的用户更难以成为网络钓鱼攻击的受害者，即使用户进入了一个与用户目标网站外观相仿的恶意网站，因为密码管理器不会自动登录，提供了一层额外的保护。

用于储存在密码数据库中的信息通常十分敏感，大多数的密码管理器保护其中的内容，禁止未授权的访问。数据库格式一般依赖于数据保护的加密码，通常是通过用户输入的主密码产生的加密/解密码。

这种保护旨在允许用户在未经信任的地方保存密码数据库。大多数密码管理器的开发商建议将密码数据库存储在USB设备、云端或是移动设备上，旨在快速访问保存的密码。这些存储方式也会使潜在的攻击者掌握整个数据库。即使数据库密码存储在本地硬盘上，攻击者也可能通过其他方式获取到一份拷贝。

如果密码管理器的数据库格式是不可信的，那么再好的密码管理器的优点都会被掩盖，用户或许会变得更不安全并且更容易受到个人信息泄露带来的感染：隐私意识强的用户想让他们的浏览习惯变得私密化，因此会经常删除cookie、历史和缓存记录。正相反，密码管理器代表了长期的存储场所，存储着密码的唯一拷贝，因此其中的内容通常永远不会被删除。如果一个密码管理器的数据库通过储存未被加密的URL的方式，泄漏了例如浏览习惯的信息，那么清除缓存和浏览记录并不能阻止攻击者获取敏感信息。

在本论文中，我们将分析几个最流行的密码管理器的数据格式所带来的安全性。我们定义了两个不同的对手：一个被动的攻击者，他只会从密码数据中推断信息；一个主动的攻击者，他会修改原始数据的内容。我们强调，使用“行业标准方法”，例如AES-CBC来保证安全的数据库格式是不够的，即使AES-CBC的实现方式是正确的。我们并不打算提供一个详尽的列表，记录对于所有密码管理器的所有可能的攻击。更确切的说，我们将常见的密码管理器数据格式作为模型，并且给出的典型攻击的例子。

这篇论文的内容如下所示：第2节给出了本研究中，一个密码管理器的概述；第3节介绍了我们的系统和攻击模型；第4节分析了模型中不同的数据库格式。在第5节中，我们将讨论各种数据库格式的一般化问题，第6节是一些相关工作。我们的结论在第7节。

1. **密码管理器概述**

密码管理器们在不同的方面表现各异，包括数据库格式、功能性、开源性、支持的平台和是否支持云存储。表1总结了我们认为的密码管理的主要特点。一些流行的密码管理器创造他们自己的数据库格式，被他们独家使用。尤其是对那些嵌入在浏览器中的密码管理器而言。

表 1 这张表详细展示了我们分析的密码管理器，包括了软件使用的数据库格式、存储选项和支持 的平台。另外我们表示了是否开源和密码管理器是否被集成在浏览器中。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 密码管理器 | 数据库格式 | 存储 | 开源 | 平台 | 浏览器集成 |
| Google Chrome | Chrome | 本地/云 | 是 | Win/Mac/Linux | 是 |
| Mozilla Firefox | Firefox | 本地/云 | 是 | Win/Mac/Linux | 是 |
| Internet Explorer | MSIE | 本地 | 否 | Win | 是 |
| 1Password | 1Password | 本地/云 | 否 | Win/Mac | 否 |
| KeePass 1.x | KDB | 本地 | 是 | Win | 否 |
| KeePass 2.x | KDB/KDBX4 | 本地 | 是 | Win/Mono | 否 |
| KeePassDroid | KDB/KDBX4 | 本地 | 是 | Android | 否 |
| KyPass | KDB/KDBX4 | 本地 | 是 | iOS | 否 |
| PassDrop | KDB/KDBX4 | 本地 | 否 | iOS | 否 |
| PINs | PINs | 本地 | 否 | Win | 否 |
| Password Safe | PasswordSafe | 本地 | 是 | Win | 否 |
| Password Gorilla | PasswordSafe | 本地 | 是 | Win/Mac/Linux | 否 |
| Roboform | Roboform | 本地/云 | 否 | Win/Mac/Linux | 是 |

我们把这些写进我们的分析里，因为这些密码管理器被广泛使用。一些独立的密码管理器共享相同的数据库格式，所以虽然每个密码管理器带给用户不用的体验，构成基础的存储格式却是相同的。

这篇论文的剩余部分将仅仅聚焦于数据库格式和他们提供的安全性，而不是每种密码管理器的实现。我们假设每种密码管理器正确地实现了各自说明的格式。因此，我们不考虑对于加密边信道攻击或者对于具体实现的攻击。相反，我们调查特定存储格式所提供的安全性。因为这个原因，我们的分析主要着眼于提供本地存储的密码管理器。我们将在后面的工作中分析云端的密码管理器。

我们调查了9种流行的密码管理器的数据库格式。3种数据库格式是集成在浏览器中的密码管理器所用：Google Chrom，Mozilla Firefox和微软Internet Explorer；6种格式被大量的独立密码管理器所用：1Password，KDB、KDBX4、PasswordSafe v3，PINs和RoboForm。

1. **对手和系统模型**

我们考虑两种高效率的对手：Advr 拥有密码数据库的访问权限，Advrw拥有读写权限。这两个对手的目标是获取尽可能多的信息，并且对于Advrw而言，还要制造出一个数据库。第一，它不是由用户所创建；第二，数据库一旦被打开，不会触发任何来自密码管理器的警告和错误信息。很显然，Advrw显然比Advr更为强大：任何可以由Advr发起的攻击都可以由Advrw产生。这两个对手都被允许收集多种数据库在不同时间点的快照，用来探测数据库内容的修改。

我们强调了我们的分析不依赖任何对于用户环境的修改，例如篡改密码管理器的源代码或是安装密码记录器。我们仅仅关注由密码管理器的数据格式带来的安全性，并让密码管理器运行在最高安全设置。我们假设用户选择了高强度、高熵的主密码并且所有底层加密算法都被正确执行。此外，我们假设没有额外的机制可以阻止文件篡改。这个允许我们可以比较数据库格式本身带来的安全性。

* 1. **不可信存储**

考虑哪些拥有加密数据库完全访问权限的对手，并且记录它的不同版本。这样的对手明显可以使用任何版本的记录来替换现在的数据库，只要主密码没有被更换。这大体上是一次对积基于云和本地的数据库格式攻击的重演。

我们如下定义的安全概念不会去记录这次攻击，我们也不会通过其他方式尝试去对待它。为了能够保护它免受攻击，一个密码管理器必须在可信的介质上保存一些本地状态（例如对于最近版本的加密数据库散列）。因此，当这次攻击的对象与存储于云上或是无人看管的USB设备中的密码数据库相关时，它并没有因为数据库格式而减少损失。我们将这种情况剔除考虑。

* 1. **安全定义**

我们定义四种算法，代表不同的密码管理器功能：设置(Setup)、创建(Create)、打开(Open)和有效(Valid)。这些算法的定义如下所示：

**定义1.** 密码管理器PM包含以下高效的算法：设置(·)是一种随机算法，给出安全参数 1k ，输出主密码mp；创建(·,·)是一种随机算法，输入mp和三元数组 *RS* = *{*(*r*1*, n*1*, v*1)*, . . . ,* (*r\_, n\_, v\_*)*}*(代表记录集)，输出数据库DB；打开(·,·)是一种确定性的算法，给出mp和数据库DB，输出在DB中编码的记录集RS如果RS是一个存在的记录集，也就是说，存在一个数据集 DB’，使得 DB’<-Create(mp,RS)，并且互相正交；有效(·,·)是一种确定性的算法，输入一个主密码mp和一个数据库DB，如果Open(mp,DB) ≠⊥则返回1.

实际上，有效(Vaild)在密码管理器中，是在打开(Open)功能中实现：如果验证有效失败，则密码管理器会返回一个错误而不是数据库内容。

我们同时定义两种新的游戏，我们分别称呼为不能辨别的数据库游戏(IND-CDBA)和 延展性的选择数据库游戏(MAL-CDBA)。前者获取现实中被动对手的能力，也就是说，这个对手拥有密码数据库的只读权限。后一种模型和主动对手，他拥有密码数据库的读写权限。

**游戏1(IND-CDBAAdvr *,PM*(*κ*)).** 挑战者Ch通过运行PM与Advr交互，如下所示：

-Ch运行 mp<-Setup(1k).

-Advr 输出两个记录集 RS0，RS1.

-Ch 均匀地随机选则 a位b 并且数据库 DBb <- Create(mp,RSb) 返回到 Advr.

- Advr 最终输出一位 b’，当且仅当 b=b’时，这个游戏最终输出1.

我们说Advr获得了IND-CDBA游戏的胜利，如果挑战者能让游戏输出1.

**定义2 (IND-CDBA 安全性).** 密码管理器 PM=(Setup,Create,Vaild,Open)是 IND-CDBA 安全的，如果存在一个可忽略不计的函数negl，对于任意可能的多项式级别时间复杂性的对手Advr而言，我们有Pr[IND*-*CDBAAdvr *,PM*(*κ*) = 1] *≤* 1*/*2 + negl(*κ*)*.*

对于大多数数据库格式而言，一个攻击者无足轻重地可以获得IND-CDBA游戏胜利通过提交两种大小不同的记录集。实际上，这对应着下面的事实：数据库文件的大小通常大致正比于数据库中记录的数量，因此攻击者可以简单地从观察加密数据库大小来获取信息。虽然我们认为这是一次有效的进攻，但会在脆弱性分析之时忽略它。如果数据库格式仅仅是这种攻击的受害者的话，我们认为它是安全的。

附录A展现了IND-CPA和IND-CDBA之间的关系。特别地，这表明了IND-CPA安全性就意味着IND-CDBA安全性。

**游戏2 (MAL-CDBAAdvrw ,PM(κ)).** 挑战者Ch通过运行PM与Advr交互，如下所示：

-Ch运行 mp<-Setup(1k).

-Advrw适应性地输出了 n组记录集 RSi 并且从Ch接收，对应的数据库

DBi <- Create(mp, RSi).

-Advrw最终输出数据库DB’；游戏输出1当且仅当 Valid(DB’) =1 并且对于所有的 i≤n 而言，DB’≠DBi.

我们说Advrw获得了MAL-CDBA游戏的胜利，如果挑战者能让游戏输出1。

**定义3 (MAL-CDBA 安全性)** 密码管理器PM=(Setup, Create, Valid, Open)是MAL-CDBA安全的，如果如果存在一个可忽略不计的函数negl，对于任意可能的多项式级别时间复杂性的对手Advrw而言，我们有Pr[MAL*-*CDBAAdvrw *,PM*(*κ*) = 1] *≤* negl(*κ*)*.*

我们对于MAL-CDBA安全性的定义相当于概念“存在的密文不可伪造”。这个安全性概念表明了，IND-CPA安全就意味着IND-CCA安全。

“密文的完整性”(也被称为 INT-CTXT安全性)是一个相关的安全概念。特别地，MAL-CDBA和INT-CTXT的主要区别是，一个INT-CTXT的对手也会可以验证(Verify)权限。

我们认为，MAL-CDBA安全性(也包括IND-CDBA安全性)是一种合适的安全性概念，对于实际中的密码管理器数据库格式而言。考虑一种非 MAL-CDBA安全的数据库格式，也就是说，Advrw可以计算出他选择的数据集的编码，并且可以生成对应的有效输出 DB’。这种格式在如下的4步攻击面前，是脆弱的：

1. Advrw用一个由Advrw创建的新数据DB’，替换了Alice的密码数据库DB。DB’中包含了amazon.com账号的登录证书。
2. Advrw现在引诱Alice去访问amazon.con，在此时密码管理器自动登录了由Advrw创建的账号信息。
3. Alice购买了商品，在收银台时，Alice被要求往她的账号中添加信用卡。因为Alice信任amazon.com，她照做了。
4. 现在，Advrw用DB’替换了Alice本来的密码数据库。

Advrw现在拥有了一个可以使用Alice信用购物的账号。对于Alice而言，很难察觉到此次攻击，她并没有从她的密码管理器或是amazon.com上收到任何警告信息，因为数据库形状完好并且登录信息对应已有账号。另外，当Alice和amazon.com通信之时，SSL/TLS并不能起到任何作用。而且，在攻击者恢复了Alice的原始数据后，Alice甚至不能找出恶意账号的用户名。

1. **数据库格式漏洞**

现在展示我们的分析，其中包括许多正在被独立的和基于浏览器的密码管理器使用中的数据库格式。对于每一种格式而言，我们提供相关特征的简要说明并且分析它的安全性，基于在第3节中定义的安全模型。如果数据库格式允许不同的安全级别，我们分析最安全的配置。

* 1. **Google Chrome**

**格式描述.** Google Chrome在用户配置文件目录下的SQLite数据库文件中存储用户名和密码。这个数据库既没有提供保密性，也不提供完整性。

Google Chrome可以在Google的服务器上选择性地存储所有浏览器偏好(包括密码)，以允许不同设备间的同步。Chrome的支持页面声称，密码以加密的形式存储在Google的服务器上。

**安全性分析.** 任何能够访问数据库文件的用户可以恢复其所有的内容，并做出任意的修改。因此，用户不可以信赖Chrome密码管理器的对于数据的安全性和完整性，并且应该实施它之外的安全措施。

* 1. **Mozilla Firefox**

**格式描述.** Mozilla Firefox