

**解密macOS / OS X上的Google Chrome密码**

**译者：freecsw**

**原文链接：<http://bufferovernoah.com/2016/10/17/chrome/>**

**原文作者：noah**

** **

**微信公众号：看雪iOS安全小组 我们的微博：weibo.com/pediyiosteam**

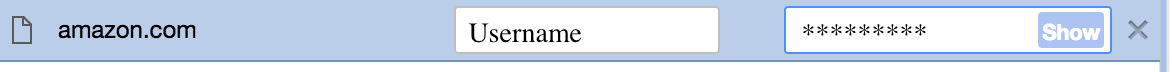
**我们的知乎：zhihu.com/people/pediyiosteam**

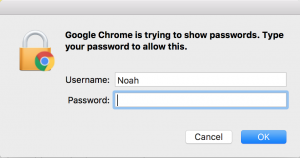
**加入我们：看雪iOS安全小组成员募集中：http://bbs.pediy.com/showthread.php?t=212949  
[看雪iOS安全小组]置顶向导集合贴： http://bbs.pediy.com/showthread.php?t=212685**

**解密macOS / OS X上的Google Chrome密码**

现在有一些开源程序，可以解密存储在Google Chrome上面的密码。 但是，所有这些程序仅适用于Windows操作系统。当我们MacOS用户需要从Google Chrome中快速转储所有存储的密码时，应该怎么办？

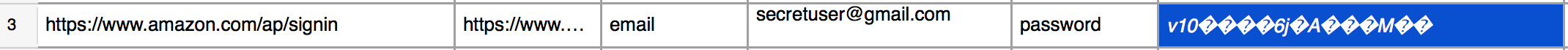
目前从Google Chrome浏览器中导出密码的方式是，打开Chrome浏览器，进入设置，然后点击“管理密码”，然后可以看见您要访问的每个密码显示如下。





谁需要它！ 这是非常不方便的，它展现给你看所有登录过的本地用户的密码。

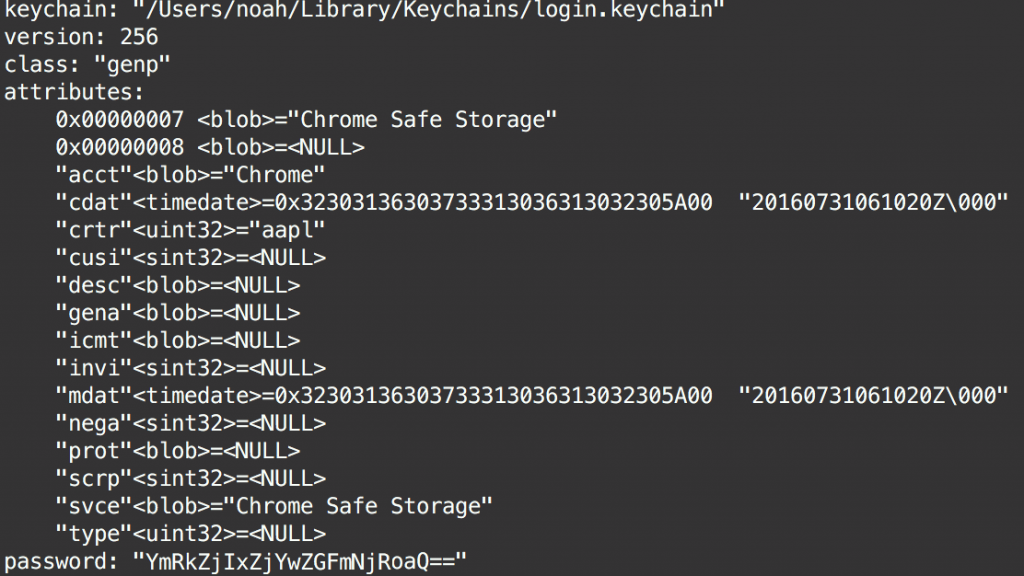
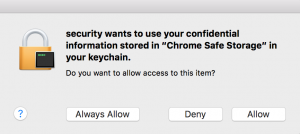
在MacOS上，我们可以通过直接查询Google Chrome SQL数据库来解决这个问题，数据库位于**“~/Library/Application Support/Google/Chrome/Profile \*/Login Data**”。

对于每个保存在Google Chrome中的密码，在“Login Data”数据库中都有一个类似下面的字段：

对于这个用户，我们可以得到以v10开头的加密数据。这个密文用的是什么加密算法？ 好吧，让我们查看[谷歌的Chromium源代码](https://cs.chromium.org/chromium/src/components/os_crypt/os_crypt_mac.mm)。 在此源代码中，我们找到以下信息：

加密方案：AES-128 CBC使用恒定salt和恒定迭代。 解密密钥是一个PBKDF2密钥，其生成过程如下：

1. **Salt** ：salt是'saltysalt'（常量）
2. **迭代** ：1003（常量）对称密钥推导。
3. **IV** ：16位。
4. **哈希函数** ：SHA1
5. **密码** ：这是重要的变化的部分。 我们需要的是存储在用户的钥匙串中的“Chrome 安全存储”密钥。 令人感到美妙的是，我们实际上不需要用户的钥匙串密码来访问此安全存储密钥。 我们可以通过使用命令行'security find-generic-password -ga Chrome'来直接调用MacOS的安全进程，调用后弹出如下对话框：

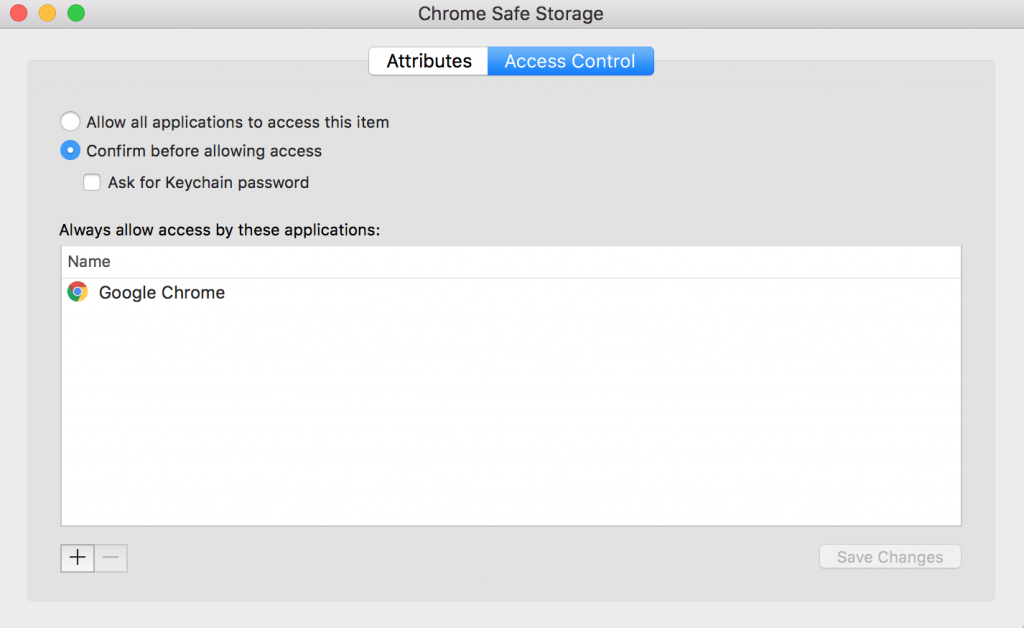


哦也，我们得到了Chrome安全存储密钥，并而无需确认用户的密码！ （注意，不是我的密码）

所以，**AES解密密钥**是通过PBKDF2（'SHA1'，安全存储密钥，'saltysalt'，1003）生成的。 好吧，得到所有这些信息是伟大的，但我们如何使用它来做一个通过遍历我们的数据库，并解密每一条数据的自动化的程序。 让我们使用Python，因为Python是最好的。 更重要的是，我们使用Python可以做到零依赖，使得我们可以在任何的MacOS系统上运行此脚本，并且在不到5秒钟内搞定！ 通常情况下，Python的 AES解密是通过完美的[加密库](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=zh-CN&ie=UTF8&prev=_t&rurl=translate.google.com.hk&sl=en&tl=zh-CN&u=https://cryptography.io/en/latest/&usg=ALkJrhgxYy5YfSLPCr04LZMuAr26_41gHA)来实现的，但很可惜，它不是一个本地自带的库。 我们可以利用OpenSSL的命令行工具，使用一些巧妙的管道和重定向，来发送我们的加密密码的base64副本和我们的PBKDF2解密密钥的十六进制副本。

我已经实现了以上功能，[代码在我的在Github上](https://github.com/manwhoami/OSXChromeDecrypt)。 它打印出这样的东西：

**值得注意的一点是** ：这个之所以会如此神奇，是因为谷歌不保护它的钥匙串项！如果用户必须在安全过程中验证其密码，而不是在对话框上单击“接受”，那么可以阻止99％的恶意行为。当我们尝试在Google Chrome外部访问钥匙串时， Google可以通过弹出“请输入钥匙串密码”来解决此问题...对吗？ 其实，不，他们不能。



如果Google想要这样做，那么他们需要修改对安全系统的访问过程，当用户第一次创建安全存储密钥时，以及要访问钥匙串时，都要求用户验证他们的安全存储密码。 如果我们看看“security add-generic-password”手册，我们可以看到，没有一种方法可以实现这样的编程需求。 苹果公司应该**认真**考虑向开发者提供这样的功能，使他们能够改善他们的用户的数字安全。

**另外请注意：**几行*非常*微不足道的shell / Python编写的代码，实质上将迫使用户多次点击弹出的“接受”提示来破解，直到允许访问。 具体的过程请查看Github上面的代码。

**太长，别读：**你可以使用[这个](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=zh-CN&ie=UTF8&prev=_t&rurl=translate.google.com.hk&sl=en&tl=zh-CN&u=https://github.com/manwhoami/OSXChromeDecrypt/&usg=ALkJrhjLRSCZV_jrTM1jkGwc2V35g_oCtQ)程序，去解密/导出MacOS / OS X上的谷歌浏览器的密码。这是唯一的MacOS / OS X上可以使用的程序，并且是100％的原始调用，没有任何依赖。 你可以（为了乐趣）添加几行代码，提取密码，将它们上传到远程FTP服务器，然后从计算机中删除程序的任何痕迹。 伟大的取证：一下子全部抓取你所有的密码。