Farewell Syskey!

Passcape Software

1 简介

在Windows 10和Windows Server 2016操作系统的下一个大更新正式发布前不久，微软公布了 新版本的变化清单 在替换和删除的组件中，有syskey.exe工具。
微软这样解释 为什么删除syskey.exe：

- Syskey加密密钥和Syskey.exe的使用不再被认为是安全的
- Syskey是基于薄弱的加密技术，在现代很容易被破解。
- 被syskey保护的数据是非常有限的，并不包括所有的文件或数据。
- syskey.exe工具被黑客用来作为勒索软件骗局的一部分。

尽管这是一个里程碑式的事件，但许多新闻门户网站和网站并不关注这条新闻。

从内部了解Syskey的功能，我们将试图澄清并向您展示，反对Syskey大多数论点都是不正确的。至少还有具体的含沙射影。你很感兴趣，想知道为什么吗？那就去阅读吧。

2 Syskey是如何工作的

2.1 什么是Syskey

首先，让我们弄清楚Syskey是什么。Syskey 是一个易被人所知的软件，旨在加强关键数据保护，如用户哈希、密码、系统机密等。它还旨在防止SAM数据库的脱机攻击，并保护用户哈希不受彩虹表的攻击。

2.2 syskey.exe工具

Syskey最初是在Windows NT 4.0 SP3中引入的。默认情况下，此额外保护包含在Windows XP和更高版本的操作系统中，无法关闭。您可能想知道最后一个短语为什么用粗体显示？好吧许多网站曲解了Syskey，或者发布了Syskkey被完全拒绝的消息。这是错误的。Microsoft拒绝使用syskey。只有exe实用程序，但Syskey保护将保持原样。如下所示，这是一个很大的区别。

2.3 Syskey加密密钥的存储

在物理层面上，Syskey只是一种具有唯一128位密钥的加密算法。还有系统密钥。exe实用程序只提供系统密钥加密密钥在系统中的存储方式。有三个可能的存储区域：

- System registry
- Startup Diskette
- Startup password

2.3.1 系统注册表

默认情况下，系统密钥存储在注册表中，并分散在以下四个分支中：

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\JD
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Skew1
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Data
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\GBG

除系统外，所有人都无法访问这些分支。但拥有管理员权限，不难阅读，提取并解密 它.

2.3.2 启动软盘

如果你将syskey.exe设置为使用这种存储方式，每次启动电脑时都会提示你需要一张装有Syskey的软盘。为了选择不使用这种存储方式，也将需要这张磁盘。

2.3.3 开机密码

在这种情况下，密钥不存储在任何地方，而是在每次系统启动时生成。一旦你选择了这个选项，在每次系统启动时，在用户登录屏幕出现之前，都会提示用户输入启动密码。然后，系统密钥将使用MD5算法对输入的密码进行hash生成。如果不提供正确的启动密码，用户将无法继续前进并加载系统。

2.4 再见了syskey.exe

因此，新版本的Windows 10和Windows Server 2016将没有syskey.exe工具。因此，将没有办法选择存储Syskey的方法，因为它将永远驻留在注册表中！ 但有什么问题呢？陷阱是，最后两种存储Syskey的方法，即要扔掉，确保对用户的数据有更强的保护。尽管没有那么多用户知道，但计算机专家和安全大师们却很享受这种更强的保护。例如，一旦用户将Syskey切换为需要启动软盘，即使入侵者或黑客获得了对PC的物理访问权，也无法获得用户的个人信息、密码或EFS加密的文件。除非提供启动软盘，否则没有机会解密数据。

3 Syskey是否已经过时了？

让我们进一步了解微软提到的删除syskey.exe的所有原因。

3.1 syskey.exe不再被认为是安全的

"syskey加密密钥和syskey.exe的使用不再被视为安全"。

这是不对的。拒绝在注册表中存储Syskey，可以大幅度提高你的数据的安全性，防止离线攻击。例如，当设置了Syskey启动密码后，即使攻击者能够进入并读取数据，也无法解密你的Skype/IE/Chrome/WiFi/LAN密码（无需提供启动密码）。在Windows 10秋季造物主更新出来后，对于潜在的恶意者来说，这将变得更加容易。

3.2 脆弱的密码学

"Syskey是基于弱的密码学，在现代很容易被破解"。

并非如此。从Syskey密钥中推导出SAM加密密钥的算法 在文章的最后显示. 当然，这种算法是二十年前发明的，有一些缺点：哈希函数MD5被认为是妥协的，弱流密码RC4也是过时的。然而，在最近的Windows 10版本中，加密算法被相应地替换为SHA-256和AES。让我们试着计算一下，人们能以多快的速度破解启动时的Syskey密码或Syskey软盘。旧算法的启动密码猜测速度大约为每秒1000-2000万个密码（新算法则低得多），这取决于使用的硬件。让我们采取最悲观的情况，把这个值四舍五入为100。但即使在这种情况下，与Windows账户密码相比，破解速度也会慢100倍左右。使用完全的暴力破解法来猜测密码'Letmein123'将需要超过260年。如果你想对启动软盘进行暴力破解，你将不得不等待太阳系的死亡。

3.3 有限的数据保护

"受syskey保护的数据非常有限，不包括所有文件或数据"。

这不是很清楚是什么意思。最有可能的是无法用EFS加密保护一些文件。例如，系统注册表。作为一个替代方案，微软建议使用BitLocker来代替。有趣的是，当你为微软账户启用BitLocker时，数据恢复密钥会自动发送到微软服务器。在一个域中，所有BitLocker恢复密钥都存储在活动目录中，任何能够访问服务器的人都可以 轻松解密 域中的任何PC. 我们已经在上一篇专门介绍 域账户安全缺陷 的文章中概述了这个安全弱点。

3.4 被黑客使用

"syskey.exe工具被黑客用来作为勒索软件骗局的一部分"。

在2014-2017年，Windows用户面临着使用内置syskey.exe工具的最广泛攻击之一。一个用户通常会接到一个电话。另一端有人带着浓重的印度口音和一个令人难以置信的英式名字，如约翰-史密斯或类似的名字，告诉说电脑已被破坏，需要修复。在毫无戒心的受害者打开他/她的电脑访问权限后，攻击者运行了syskey.exe工具并设置了启动Syskey密码。在遭受系统攻击后，恢复计算机的主要问题是，仅仅重置Syskey或将其切换回注册表中存储是不够的。要完全恢复系统，需要知道Syskey的启动密码。

毫无疑问，微软新的更新将使这种欺诈计划归零。但是，如果希望或声称骗子永远不会使用其他的欺骗计划，那就太天真了。微软的安全人员应该告诉我们，怎样才能阻止骗子重设用户的登录密码而不

是Syskey？要做到这一点甚至要容易得多。你知道吗，当骗子进入受害者的电脑后，他只需使用具有管理员权限的命令提示符，然后键入
NET USER <USERNAME> <NEWPASSWORD>
在不知道当前密码的情况下设置新密码。

4 总结

因此，在新发布的Windows 10秋季创意者更新和syskey.exe的死亡之后，所有用户的文件和密码将对骗子和情报机构变得更加开放。如果微软继续其简化用户个人数据访问的不经意政策，这最终将使普通用户、组织和整个国家迁移到其他操作系统。

剩下的最后一根弦是停止对普通本地账户的支持，已经只剩下微软账户。在这一步之后，用户将不再能够认为他们的私人数据是完全安全的。另一方面，对于普通的网络用户来说，这更像是一个加分项。意识到你的整个私人生活不仅可以成为你的，而且不完全是私人的，这是令人沮丧的。

Syskey是好是坏？ 自己决定吧。

5 附录1. 提取SAM加密密钥

```
BYTE hash[16], pDecodedData[32];
static BYTE samc1[]="!@#$%^&*()qwertyUIOPAzxcvbnmQQQQQQQQQQQQ)(*@&%";
static BYTE samc2[]="0123456789012345678901234567890123456789";

//Generate decrypt key
Md5Init();
Md5Update(pSamSessionKey+8,16);
Md5Update(samc1,0x2F);
Md5Update(m_pSyskey,16);
Md5Update(samc2,0x29);
Md5Final(hash);

//Decrypt
memcpy(pDecodedData,pSamSessionKey+24,32);
Rc4SetKey(hash,16);
Rc4Decrypt(pDecodedData,32);

//Check sign
Md5Init();
Md5Update(pDecodedData,16);
Md5Update(samc2,0x29);
Md5Update(pDecodedData,16);
Md5Update(samc1,0x2F);
Md5Final(hash);

if( memcmp(pDecodedData+16,hash,16)==0 )
{
   memcpy(pDecodedSamSessionKey,pDecodedData,16);
```

```
    return TRUE;
}
return FALSE;
```

6 附录2. 解密Syskey

```
BOOL DecryptSyskey(LPCTSTR szJD, LPCTSTR szSkew1, LPCTSTR szGBG, LPCTSTR szData)
{
    BYTE p[]={0xb,0x6,0x7,0x1,0x8,0xa,0xe,0x0,0x3,0x5,0x2,0xf,0xd,0x9,0xc,0x4};
    BYTE pEncryptedKey[16];

    if( !szJD || !szSkew1 || !szGBG || !szData )
        return FALSE;

    //convert to binary
    _stscanf_s(szJD,_T("%X"),(LPDWORD)&pEncryptedKey[0]);
    _stscanf_s(szSkew1,_T("%X"),(LPDWORD)&pEncryptedKey[4]);
    _stscanf_s(szGBG,_T("%X"),(LPDWORD)&pEncryptedKey[8]);
    _stscanf_s(szData,_T("%X"),(LPDWORD)&pEncryptedKey[12]);

    //Permutate
    for( int i=0; i<16; i++ )
        m_pSyskey[i]=pEncryptedKey[p[i]];

    return TRUE;
}
```