ssh暴力破解日志简要说明

docboon

<olderflower[AT]outlook.com>

2022/01/27

修订历史		
修订 1.0	2022年1月19日	olderflower[AT]outlook.com
发布第一个Draft		

目录

1. 概述

所有直接暴露在互联网上的机器都会或多或少的受到攻击。就比如我们经常使用的ssh服务器，可能随时都会受到来自世界各地的各种暴力破解。如果ssh服务器没有采用比较严格的用户名和密码，那我们的ssh服务器极有可能会被黑客或者暴力破解工具破解并攻陷。所以在我们的ssh服务器被破解之前，及时地知道这些信息，并采取相关的措施就显得非常重要。

幸运地是，在一个日志记录完整的系统中，所有的ssh暴力破解都会被记录在案。通常，在各种linux系统中，ssh服务器使用的一般都是openssh，在默认情况下它的日志都记录在*/var/log/auth.log*文件中。在这个文件中如果我们看到如下样式的日志，那么，我们的系统已经在遭受别人的破解：

```
Jan 19 08:29:46 DocBoon sshd[765073]: Invalid user ari from 165.232.75.221 port 48646 ❶
Jan 19 00:00:53 DocBoon sshd[757795]: Failed password for root from 165.22.88.72 port 49300 ssh2 ❷
```

这里面包含了两种类型的破解：

❶ 暴力破解用户名的日志
❷ 暴力破解密码的日志

通常情况下，黑客是利用暴力破解工具来完成这项工作的，暴力破解工具会利用一个字典，根据字典里的用户名来猜测系统中的可能存在的用户。一旦发现系统中存在某个用户，那破解工具会进一步猜测这个用户的密码。

2. 相关日志解析

2.1. 暴力破解用户名的日志

暴力破解用户名的日志有5个重要的字段：

```
        ❶       ❷                                ❸             ❹           ❺
Jan 19 08:29:46 DocBoon sshd[765073]: Invalid user ari from 165.232.75.221 port 48646
```

❶ `Jan 19 08:29:46` ： 时间戳；
❷ `DocBoon` ： ssh服务器的机器名；
❸ `ari` ： 破解工具猜测的无效的用户名；
❹ `165.232.75.221` ： 破解工具使用的IP地址；
❺ `48646` ： 暴力工具使用的端口。

2.2. 暴力破解密码的日志

暴力破解密码的日志有5个重要的字段：

```
        ❶       ❷                                      ❸           ❹           ❺
Jan 19 00:00:53 DocBoon sshd[7595]: Failed password for root from 165.22.88.72 port 49300 ssh2
```

❶ `Jan 19 00:00:53` ： 时间戳；
❷ `DocBoon` ： ssh服务器的机器名；
❸ `root` ： 破解工具猜测的无效的用户名；

❹ 165.22.88.72 ：破解工具使用的IP地址；
❺ 49300 ：暴力工具使用的端口。

一旦在日志中发现了这样的记录，必须要引起高度重视的。

3. 防范措施

要防止这样的一些恶意行为，可以从两个方面来完善我们的ssh系统。

1. 在配置文件中做好合理的设置；
2. 采取必要的安全措施。

3.1. 设置配置文件

默认情况下，ssh服务器的配置文件是*/etc/ssh/sshd_config*。一些主要的配置项如下：

1. AllowGroups

   这个指令后面跟着一串用空格分隔的组名列表(其中可以使用"*"和"?"通配符)。默认允许所有组登录。如果使用了这个指令，那么将仅允许这些组中的成员登录，而拒绝其它所有组。这里的"组"是指"主组"(primary group)，也就是*/etc/passwd*文件中指定的组。这里只允许使用组的名字而不允许使用GID。相关的 allow/deny 指令按照下列顺序处理：DenyUsers, AllowUsers, DenyGroups, AllowGroups

2. AllowUsers

   这个指令后面跟着一串用空格分隔的用户名列表(其中可以使用"*"和"?"通配符)。默认允许所有用户登录。如果使用了这个指令，那么将仅允许这些用户登录，而拒绝其它所有用户。如果指定了 USER@HOST 模式的用户，那么 USER 和 HOST 将同时被检查。这里只允许使用用户的名字而不允许使用UID。相关的 allow/deny 指令按照下列顺序处理：DenyUsers, AllowUsers, DenyGroups, AllowGroups

3. AuthorizedKeysFile

   存放该用户可以用来登录的 RSA/DSA 公钥。该指令中可以使用下列根据连接时的实际情况进行展开的符号：%% 表示'%'、%h 表示用户的主目录、%u 表示该用户的用户名。经过扩展之后的值必须要么是绝对路径，要么是相对于用户主目录的相对路径。默认值是".ssh/authorized_keys"。

4. Banner

   将这个指令指定的文件中的内容在用户进行认证前显示给远程用户。这个特性仅能用于SSH-2，默认什么内容也不显示。"none"表示禁用这个特性。

5. Ciphers

   指定SSH-2允许使用的加密算法。多个算法之间使用逗号分隔。可以使用的算法如下："aes128-cbc", "aes192-cbc", "aes256-cbc", "aes128-ctr", "aes192-ctr", "aes256-ctr","3des-cbc", "arcfour128", "arcfour256", "arcfour", "blowfish-cbc", "cast128-cbc"默认值是可以使用上述所有算法。

6. ClientAliveCountMax

   sshd(8) 在未收到任何客户端回应前最多允许发送多少个"alive"消息。默认值是 3 。到达这个上限后，sshd(8) 将强制断开连接、关闭会话。需要注意的是，"alive"消息与TCPKeepAlive 有很大差异。"alive"消息是通过加密连接发送的，因此不会被欺骗；而TCPKeepAlive却是可以被欺骗的。如果 ClientAliveInterval 被设为 15 并且将ClientAliveCountMax 保持为默认值，那么无应答的客户端大约会在45秒后被强制断开。这个指令仅可以用于SSH-2协议。

7. ClientAliveInterval

   设置一个以秒记的时长，如果超过这么长时间没有收到客户端的任何数据，sshd(8) 将通过安全通道向客户端发送一个"alive"消息，并等候应答。默认值 0 表示不发送"alive"消息。这个选项仅对SSH-2有效。

8. Compression

   是否对通信数据进行加密，还是延迟到认证成功之后再对通信数据加密。可用值："yes", "delayed"(默认), "no"。

9. DenyGroups

   这个指令后面跟着一串用空格分隔的组名列表(其中可以使用"*"和"?"通配符)。默认允许所有组登录。如果使用了这个指令，那么这些组中的成员将被拒绝登录。这里的"组"是指"主组"(primary group)，也就是/etc/passwd文件中指定的组。这里只允许使用组的名字而不允许使用GID。相关的 allow/deny 指令按照下列顺序处理：DenyUsers, AllowUsers, DenyGroups, AllowGroups

10. DenyUsers

    这个指令后面跟着一串用空格分隔的用户名列表(其中可以使用"*"和"?"通配符)。默认允许所有用户登录。如果使用了这个指令，那么这些用户将被拒绝登录。如果指定了 USER@HOST 模式的用户，那么 USER 和 HOST 将同时被检查。这里只允许使用用户的名字而不允许使用UID。相关的 allow/deny 指令按照下列顺序处理：DenyUsers, AllowUsers, DenyGroups, AllowGroups

11. HostbasedAuthentication

    这个指令与 RhostsRSAAuthentication 类似，但是仅可以用于SSH-2。推荐使用默认值"no"。推荐使用默认值"no"禁止这种不安全的认证方式。

12. HostbasedUsesNameFromPacketOnly

    在开启 HostbasedAuthentication 的情况下，指定服务器在使用 ~/.shosts ~/.rhosts /etc/hosts.equiv 进行远程主机名匹配时，是否进行反向域名查询。"yes"表示 sshd(8) 信任客户端提供的主机名而不进行反向查询。默认值是"no"。

13. HostKey

    主机私钥文件的位置。如果权限不对，sshd(8) 可能会拒绝启动。SSH-1默认是 /etc/ssh/ssh_host_key 。SSH-2默认是 /etc/ssh/ssh_host_rsa_key 和 /etc/ssh/ssh_host_dsa_key 。一台主机可以拥有多个不同的私钥。"rsa1"仅用于SSH-1，"dsa"和"rsa"仅用于SSH-2。

14. ListenAddress

指定 sshd(8) 监听的网络地址，默认监听所有地址。可以使用下面的格式： ListenAddress host|IPv4_addr|IPv6_addr ListenAddress host|IPv4_addr:port ListenAddress [host|IPv6_addr]:port 如果未指定 port ，那么将使用 Port 指令的值。可以使用多个 ListenAddress 指令监听多个地址。

15. LoginGraceTime

    限制用户必须在指定的时限内认证成功，0 表示无限制。默认值是120秒。

16. LogLevel

    指定 sshd(8) 的日志等级(详细程度)。可用值如下：QUIET, FATAL, ERROR, INFO(默认), VERBOSE, DEBUG, DEBUG1, DEBUG2, DEBUG3,DEBUG 与 DEBUG1 等价；DEBUG2 和 DEBUG3 则分别指定了更详细、更罗嗦的日志输出。比 DEBUG 更详细的日志可能会泄漏用户的敏感信息，因此反对使用。

17. MaxAuthTries

    指定每个连接最大允许的认证次数。默认值是 6 。如果失败认证的次数超过这个数值的一半，连接将被强制断开，且会生成额外的失败日志消息。

18. MaxStartups

    最大允许保持多少个未认证的连接。默认值是 10 。到达限制后，将不再接受新连接，除非先前的连接认证成功或超出 LoginGraceTime 的限制。

19. PasswordAuthentication

    是否允许使用基于密码的认证。默认为"yes"。

20. PermitEmptyPasswords

    是否允许密码为空的用户远程登录。默认为"no"。

21. PermitOpen

    指定TCP端口转发允许的目的地，可以使用空格分隔多个转发目标。默认允许所有转发请求。合法的指令格式如下： PermitOpen host:port PermitOpen IPv4_addr:port PermitOpen [IPv6_addr]:port "any"可以用于移除所有限制并允许一切转发请求。

22. PermitRootLogin

    是否允许 root 登录。可用值如下："yes"(默认) 表示允许。"no"表示禁止。"without-password"表示禁止使用密码认证登录。"forced-commands-only"表示只有在指定了 command 选项的情况下才允许使用公钥认证登录。同时其它认证方法全部被禁止。这个值常用于做远程备份之类的事情。

23. PermitTunnel

    是否允许 tun(4) 设备转发。可用值如下："yes", "point-to-point"(layer 3), "ethernet"(layer 2), "no"(默认)。"yes"同时蕴含着"point-to-point"和"ethernet"。

24. PermitUserEnvironment

指定是否允许 sshd(8) 处理 ~/.ssh/environment 以及 ~/.ssh/authorized_keys 中的 environment= 选项。默认值是"no"。如果设为"yes"可能会导致用户有机会使用某些机制(比如LD_PRELOAD)绕过访问控制，造成安全漏洞。

25. PidFile

指定在哪个文件中存放SSH守护进程的进程号，默认为 /var/run/sshd.pid 文件。

26. Port

指定 sshd(8) 守护进程监听的端口号，默认为 22 。可以使用多条指令监听多个端口。默认将在本机的所有网络接口上监听，但是可以通过 ListenAddress 指定只在某个特定的接口上监听。

27. PrintLastLog

指定 sshd(8) 是否在每一次交互式登录时打印最后一位用户的登录时间。默认值是"yes"。

28. PrintMotd

指定 sshd(8) 是否在每一次交互式登录时打印 /etc/motd 文件的内容。默认值是"yes"。

29. Protocol

指定 sshd(8) 支持的SSH协议的版本号。'1'和'2'表示仅仅支持SSH-1和SSH-2协议。"2,1"表示同时支持SSH-1和SSH-2协议。

30. PubkeyAuthentication

是否允许公钥认证。仅可以用于SSH-2。默认值为"yes"。

31. RhostsRSAAuthentication

是否使用强可信主机认证(通过检查远程主机名和关联的用户名进行认证)。仅用于SSH-1。这是通过在RSA认证成功后再检查 ~/.rhosts 或 /etc/hosts.equiv 进行认证的。出于安全考虑，建议使用默认值"no"。

32. RSAAuthentication

是否允许使用纯 RSA 公钥认证。仅用于SSH-1。默认值是"yes"。

33. ServerKeyBits

指定临时服务器密钥的长度。仅用于SSH-1。默认值是 768(位)。最小值是 512 。

34. StrictModes

指定是否要求 sshd(8) 在接受连接请求前对用户主目录和相关的配置文件进行宿主和权限检查。强烈建议使用默认值"yes"来预防可能出现的低级错误。

35. Subsystem

配置一个外部子系统(例如，一个文件传输守护进程)。仅用于SSH-2协议。值是一个子系统的名字和对应的命令行(含选项和参数)。比如"sft /bin/sftp-server"。

36. `SyslogFacility`

    指定 sshd(8) 将日志消息通过哪个日志子系统(facility)发送。有效值是：DAEMON, USER, AUTH(默认), LOCAL0, LOCAL1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6, LOCAL7

37. `TCPKeepAlive`

    指定系统是否向客户端发送 TCP keepalive 消息。默认值是"yes"。这种消息可以检测到死连接、连接不当关闭、客户端崩溃等异常。可以设为"no"关闭这个特性。

38. `UseDNS`

    指定 sshd(8) 是否应该对远程主机名进行反向解析，以检查此主机名是否与其IP地址真实对应。默认值为"yes"。

39. `UseLogin`

    是否在交互式会话的登录过程中使用 login(1) 。默认值是"no"。如果开启此指令，那么 X11Forwarding 将会被禁止，因为 login(1) 不知道如何处理 xauth(1) cookies 。需要注意的是，login(1) 是禁止用于远程执行命令的。如果指定了 UsePrivilegeSeparation ，那么它将在认证完成后被禁用。

40. `UsePrivilegeSeparation`

    是否让 sshd(8) 通过创建非特权子进程处理接入请求的方法来进行权限分离。默认值是"yes"。认证成功后，将以该认证用户的身份创建另一个子进程。这样做的目的是为了防止通过有缺陷的子进程提升权限，从而使系统更加安全。

3.2. 使用密钥登陆

一种更为便捷的方式是使用密钥登陆。网络中有一台SSH-Server（192.168.31.66）、一台SSH-Client，如下图所示：

图 1. 网络示意图

我们要从SSH-CLient登陆到SSH-Server上。在SSH-Server有一个用户bob，在SSH-CLient也建立一个用户bob（当然也可以使用其他的用户），步骤如下：

1. `ssh-keygen`

    在SSH-Client上使用如下命令命令产生密钥对：

    ```
    ssh-keygen
    ```

    命令执行后会有类似如下的反馈：

    ```
    bob@SSH-Client:~$ ssh-keygen
    ```

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/bob/.ssh/id_rsa):
Created directory '/home/bob/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/bob/.ssh/id_rsa
Your public key has been saved in /home/bob/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:u8bm0IR3amluHGnu2XE7iZBliSJepPDEeCOyGBDArC0 bob@SSH-Client
The key's randomart image is:
+---[RSA 3072]----+
|O. o             |
|oo+ = .          |
|o= * +   . .     |
|E . + o.. +      |
| . . o..S=.      |
|    .  +*=       |
|      .=Bo....   |
|       =B+.oo.   |
|       ** . ..   |
+----[SHA256]-----+
bob@SSH-Client:~$
```

这时，我们在SSH-Client的*/home/bob/.ssh/*目录下会看到两个文件：*id_rsa*和*id_rsa.pub*。其中，*id_rsa*是私钥文件、*id_rsa.pub*是公钥文件。

2. ssh-copy-id

使用 如下命令命令将公钥复制到SSH-Server中

```
ssh-copy-id -i .ssh/id_rsa.pub bob@192.168.31.66
```

```
bob@SSH-Client:~$ ssh-copy-id -i .ssh/id_rsa.pub bob@192.168.31.66
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_rsa.pub"
The authenticity of host '192.168.31.66 (192.168.31.66)' can't be established.
ED25519 key fingerprint is SHA256:eoJFz2cx/XsJQqRxPiu5RGVtZ6TAy0tBXeJEdFc+bAo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
                            that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now
                            it is to install the new keys
bob@192.168.31.66's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'bob@192.168.31.66'"
and check to make sure that only the key(s) you wanted were added.

bob@SSH-Client:~$
```

ssh-copy-id将公钥写到远程机器的 *~/.ssh/authorized_key*文件中

3. /etc/init.d/ssh restart

在SSH-Server上重新启动ssh系统：

```
/etc/init.d/ssh restart
```

4. ssh登陆

接下来从SSH-CLient直接登陆的SSH-Server，在SSH-Client执行如下命令：

```
ssh bob@192.168.31.66
```

3.3. 使用安全软件

sshguard