DISTRIBUTED PROTECTION AGAINST DISTRIBUTED BRUTE FORCE ATTACKS

Jan Richter
Doctoral Degree Programme (1), FIT BUT
E-mail: xricht14@stud.fit.vutbr.cz

Supervised by: Jaroslav Zendulka
E-mail: zendulka@fit.vutbr.cz

Abstract: This paper deals with the analysis of brute force attacks focused on the breaking authentication of common services (especially SSH) on Linux and xBSD operating systems. It also examines real attacks, actual tools and ways of detection of these attacks. Finally, there are designed new mechanisms of tracking distributed brute force attacks in distributed environment.

Keywords: SSH, distributed attack, brute force, tracking, clustering

1 INTRODUCTION

A brute force attack on an authentication service is done by automated attempting to log into a system with various user names and passwords. As it is recently very popular to distribute various tasks and computations to many nodes, it is also popular to perform various attacks from many different hosts, which are usually parts of a botnet. In case of brute force attacks on the system authentication, individual members of the botnet can do only one or a very small number of attempts to log into destination system and thus make the detection of such attack more complicated.

One of the services on unix systems, which are most frequently attacked with brute force attacks, is SSH (Secure Shell), so the project is focussed on this service.

2 STATE OF THE ART

Currently there are few systems that focus on protection against brute force attacks to common services. Best known ones are Sshguard[3] and Fail2ban[4]. These systems and all other similar systems for this purpose use the same principle. They watch and parse the log of authentication module of guarded service and simply blocks access from all IP addresses that have reached threshold value of failed attempts to log in within specified time interval.

3 ANALYSIS

I have done an analysis of last two years' sshd logs from three different linux servers in different locations to get more information on brute force attacks which were performed against those servers.

As the first user existing on most unix systems is the system administrator called root, the attackers can expect that this user account will be present on almost all systems and thus this is the most frequently used user name in brute force attacks. Another frequently used usernames are based on some other usual roles in computer systems such as admin, backup, access, test, user, or based on common persons' names such as james, john, robert etc. Usual passwords used in these attacks can be split into three classes. Dictionary attacks use some dictionary of passwords, which are tried one by one. Hybrid attacks also use the dictionary, but words from this dictionary are concatenated with some additional characters. Very frequent approach is to use the username with some numeric

characters attached to the end as the password[1]. Last class of attacks, generic brute force attacks, doesn't use any dictionary and generates the passwords according to some rules.

My analysis also shows that attacks could be divided into the following classes:

- single attack with a lot of attempts to log in with a few seconds delays from same source IP address,
- attacks with many login attempts in a second from same source IP, these attacks can be part of a distributed attack,
- distributed attack from a small amount of source addresses, where each source performs a new attempt in a few minutes interval,
- distributed attack from a group of source addresses located in the same /24 ipv4 subnet,
- distributed attack from a large number of source addresses when each source attacks only a few times, not more than 4 times,
- single attacks with a large amount of attempts to log into the system from the same source which is a few days later repeated from another address with the same set of usernames and passwords.

4 PREVENTION AND PROTECTION

Common methods of prevention against brute force attacks on the SSH authentication are:

- disabling root login, as it is the most frequently tried user name and compromising this account is very dangerous, it is very good idea to disallow user root to log into the system directly,
- enforcement of using strong passwords, so an attacker shouldn't be able to guess the correct password by dictionary or hybrid attack, generating them in generic attacks is unlikely,
- use of non-standard and non-deducible user names,
- running sshd on non-standard ports,
- allowing only specified source addresses to connect to system,
- use of asymmetric keys for authentication instead of passwords,
- and finally blocking source addresses from which some failed login attempts were detected.

Practices mentioned above usually bring some annoyance and limitations for authorised users and thus they are not widely used in practise, with the exception of blocking source addresses with failed login attempts. In the rest of the paper, we will focus on this practise.

5 PROTECTION AND TRACKING SYSTEM

Nowadays, there is a bunch of simple software applications designated to protect a system from the brute force attacks. The principle of their work is to watch and parse the system log and after the detection of the predefined number of failed attempts to log into the system from the same source address it modifies system firewall by adding the rule to drop SSH packets from that source for a specified time period. This method is very simple and it's purpose is to minimize the attacker's

chance to guess the right password by reducing the number of login attempts. Some examples of such applications are open source projects Fail2ban[4] and Sshguard[3].

Main weaknesses of this simple approach are that:

- they guard only one single computer,
- each host has it's own attackers database and detection is done only by host itself, so it can easily ignore attack which is performed on many network nodes with small number of attempts per node,
- detection is based only on the same source IP address, so distributed attacks aren't detected or they are detected as single attacks, and
- they do not keep any context between single attacks.

I've designed an experimental system called DBFAP (Distributed Brute Force Attack Protection) to test some new approaches of distributed attacks detection and tracking. The system works in similar manner as above mentioned systems, it watches and parses sshd log and it configures host firewall to block the selected source addresses. The difference is in storing the information about failed attempts and in the attack evaluation. DBFAP is intended to protect a small or middle-sized computer network. Each node runs a small application which parses system log and communicates with the central host, which receives information about failed login attempts and provides the list of hosts to be blocked by all nodes. The central host is made of MySQL server, which stores information about the failed attempts and attacks and there is a set of database triggers which are responsible for attack evaluation. The scheme of the system can be seen in Figure 1.

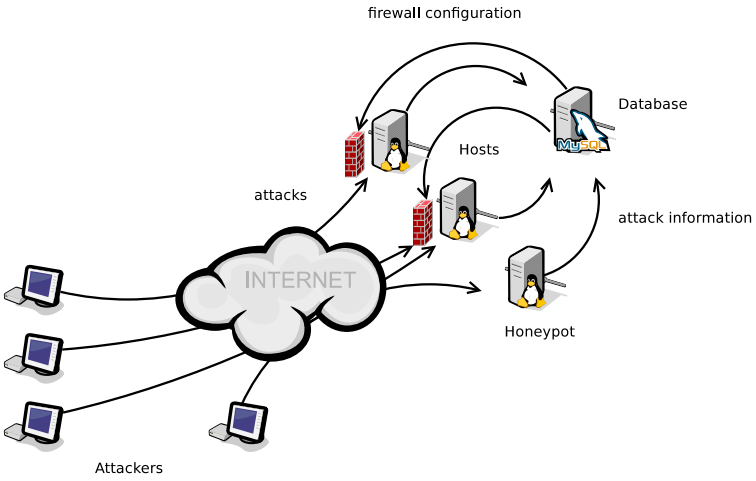

Figure 1: scheme of the system DBFAP

As we can see in the picture, we use a low-interactive honeypot in the network. This honeypot differs from other hosts only by deactivated firewall. It's purpose is to get some more information about ongoing attacks even if all other network nodes have already disabled attacking hosts. It is good practice to forward all unused IP addresses in our network to this honeypot and also to forward all blocked sources from real hosts to this honeypot.

The fact that the system has information about failed login attempts from all network nodes causes that it can detect more sophisticated attacks on the whole network, each node then knows about attack

and can modify it's firewall, so attacker cannot continue with probing any other host in the network. The whole history of login attempts and attacks is stored in database, so it can be compared with actual attacks and the response to the attack can be modified according to this knowledge, the time period for which the attacker is blocked can be adjusted or the attacker could be put into permanent ban list.

Source address is not only guide to detect an attack, the system also takes attention to used usernames and it compares the prefixes of source addresses.

The key feature of DBFAP is the tracking of attacker groups and consecutive blocking of whole groups of hosts. As attacks are detected in sequence, we can compare the time of attacks' occurrence, used credentials and source address prefixes to discover interconnection between particular attacks and then according to these measures we can attach this attacker to actual group or we can close actual group and start with tracking of a new group of attackers.

Currently there is used a simple algorithm which counts average interval $\bar{t}$ between each two consecutive attacks in current group and if the next attack occurs earlier than $2 \cdot \bar{t}$ from previous attack, then this attack is considered to be a part of current group. The value of t for the first attack in a group can be defined in configuration.

Next time when several attacks from previously identified group of attackers are detected, the system automatically blocks the whole group as it can be expected that more of them will try it also. If there is a honeypot present in the network, then system can update the group members according to the hosts that will perform attack also after the whole group has been blocked.

Another kind of groups are previously mentioned network prefixes. Attack analyser engine compares the ipv4 C mask prefix in attack records and if the threshold count of them occurs during specified time window, then the whole prefix is blocked.

6 EVALUATION

The system was successfully tested with real traffic on various networks and also in simulation using sshd logs from one of servers from our faculty. I have prepared a testing environment with log parser which was reading the log line by line, changing the system date and time to the date and time from log entry and passed the rest of the line to tested application and watching what is being done by the tested application.

The testing data set with applications based on standard approach has consecutively generated 39 272 firewall rules which successfully blocked 115 060 other login attempts. Application DBFAP in simulation with the same input data consecutively generated 14 571 firewall rules which caused blocking of 163 595 next login attempts. In addition to these rules there were also generated 89 more rules to block whole networks, which then tended to block 183 more attacks. The module for attacker groups tracking has generated 9 511 firewall rules that successfully blocked 11 708 next attacks from members of these groups.

Together there were blocked 175 486 attack attempts compared to 115 060 attempts blocked by standard approach.

7 CONCLUSION

The experimental system for protection against distributed brute force attacks designed in this paper was tested in real network as well as in simulation using historical data about attacks. The comparison of new approaches with classic approach shows that we could block 50% more attempts to log into the system, which were part of an attack.

In the future work I would like to focus on improving of clustering and botnet tracking algorithm for this project.

REFERENCES

[1] Owens, J., Matthews, J.: A Study of Passwords and Methods Used in Brute-Force SSH Attacks, Department of Computer Science Clarkson University, 2008, Potsdam, New York

[2] Malecot, E.L., Hori, Y., Sakurai, K., Ryou, J.C., Lee, H.:(Visually) Tracking Distributed SSH Brute Force Attacks?, 3rd Int'l Joint Workshop on Information Security and Its Applications (IJWISA), pp. 1-8, February 2008, Korea University, Seoul, Korea

[3] Maggi, F.: Sshguard documentation, `http://www.sshguard.net/docs/`

[4] Jaquier, C.: Fail2ban documentation, `http://www.fail2ban.org/wiki/index.php/README`