SSH Password Guessing: Linux compromise and forensics

Author : Jamie Riden (jamesr@europe.com)
Date Created : 21st January 2006
Status : Final
Version : 1.0

This document is copyright the author, but may be freely redistributed provided it is not altered, including this copyright notice. (C) 2006 Jamie Riden

Table of Contents

1 Audience

This document is intended for systems administrators with some knowledge of Linux,TCP/IP networking and the SSH daemon. Although the machine in question was running Debian Linux, the prinicples should apply to any modern UNIX implementation.

2 Introduction

This document describes the compromise of a Debian Linux server on an internal network. We look at how the incident first came to light, the response procedures and an analysis of the actions of the attacker. This leads us to some recommendations on how to secure systems against this kind of exploitation in future. None of this is particularly new or surprising, but hopefully will serve as a welcome reminder, or as useful material when trying to justify particular security policies.

3 Network architecture

The network is Ethernet throughout and is essentially based on a star topology, with a large central router, connected to lots of distribution switches. Most traffic between different subnets, and all traffic to and from the Internet will pass through the central router. On this router, a SPAN port mirrors up to 100Mb/s of traffic to an Intrusion Detection System sensor. Between the router and the Internet, a Cisco PIX firewall allows the majority of outbound traffic to pass but blocks ports 53/udp, 53/tcp (DNS), 25/tcp (SMTP) and 80/tcp (HTTP) from the majority of network hosts. Only official DNS forwarders, mail servers and web caches respectively are allowed to send traffic out on these ports.

4 Initial Evidence of an Intrusion

First thing on Monday morning, a check of the Intrusion Detection System revealed over 200,000 separate alerts; this is substantially higher than usual for this particular network, after careful tuning of the alert rulesets. The main rule which was firing was related to many packets being sent to port 22/tcp – the SSH protocol. Upon further investigation, this had been going on all weekend and the machine 'victim' had looked as if it had been probing for other hosts with vulnerabilities in their SSH daemons.

4.1 Incident Response

After a telephone call to the administrator of the machine, it was quickly disconnected from the network, and rebooted using a Knoppix-based distribution for forensic examination. These days I would use the *Helix* [HLX] bootable CD which is designed specifically for this sort of thing.

(Please note, this step conflicts with one of the points of ITIL, which is to get the user up and running again as soon as possible. You need to know what's happened to the machine and to make sure it doesn't happen again, or to any of your other servers. You may also need to notify administrators of other servers which have appear to have been compromised. Talk this through with your manager, **before** an incident occurs, so you know that you have permission to disconnect the system for as long as necessary.)

4.2 Evidence Gathering

I have to admit I'm not particularly skilled at UNIX forensics; should we judge the incident to be particularly serious – if valuable data had been compromised, or if law enforcement was likely to become involved – we would have kept the machine off and called an expert. In this case, the machine was only a minor file server which was also used for mail pickup and computation jobs. It was also reasonably obvious what was wrong. If you are gathering information officially, remember to keep a log of exactly what you are doing at all times, preserve evidence at each stage, and make sure you have an audit trail (see 'chain of custody') for everything at all times.

Initially, the contents of the /var directory was captured to a laptop via a network cable and the invaluable *netcat* [NC] utility. From memory, it is something like this:

```
server% nc -l -p 31337 > evil.tar
victim% tar cvf /tmp/evil.tar /var ; cat /tmp/evil.tar | nc -p 31337 server.ip.addr
server% tar xvf evil.tar
```

As I say, I'm not an expert, so don't quote me on this. Typically I would *dd* each partition off to another machine and examine it offline. Examination of the /var/log/auth.log revealed password guessing attempts using the SSH daemon. Since the victim machine was attempting to scan other networks for SSH servers, it was reasonable to suppose it in turn was probing for weak passwords. This topic had also been covered on the invaluable ISC diary [ISC].

5 Password Guessing

The sshd log entries looked like this:

```
Jun 26 22:31:04 victim sshd[15384]: Failed password for root from ::ffff:w.x.y.z port
30937 ssh2
Jun 26 22:31:06 victim sshd[15386]: Illegal user network from ::ffff:w.x.y.z
Jun 26 22:31:06 victim sshd[15386]: error: Could not get shadow information for NOUSER
Jun 26 22:31:06 victim sshd[15386]: Failed password for illegal user network from ::
ffff:w.x.y.z port 30951 ssh2
Jun 26 22:31:08 victim sshd[15388]: Illegal user word from ::ffff:w.x.y.z
Jun 26 22:31:08 victim sshd[15388]: error: Could not get shadow information for NOUSER
Jun 26 22:31:08 victim sshd[15388]: Failed password for illegal user word from ::
ffff:w.x.y.z port 30963 ssh2
Jun 26 22:31:10 victim sshd[15390]: Failed password for root from ::ffff:w.x.y.z port
30980 ssh2
Jun 26 22:31:11 victim sshd[15392]: Failed password for root from ::ffff:w.x.y.z port
30992 ssh2
Jun 26 22:31:13 victim sshd[15394]: Failed password for root from ::ffff:w.x.y.z port
31007 ssh2
Jun 26 22:31:15 victim sshd[15396]: Failed password for root from ::ffff:w.x.y.z port
31021 ssh2
Jun 26 22:31:17 victim sshd[15398]: Failed password for root from ::ffff:w.x.y.z port
31031 ssh2
Jun 26 22:31:19 victim sshd[15400]: Failed password for root from ::ffff:w.x.y.z port
31049 ssh2
Jun 26 22:31:20 victim sshd[15403]: Failed password for root from ::ffff:w.x.y.z port
31062 ssh2
Jun 26 22:31:22 victim sshd[15405]: Failed password for root from ::ffff:w.x.y.z port
31073 ssh2
```

We can see that the attacker – IP address w.x.y.z – was probing for weak user accounts, and for weak passwords to the root account. The long and short is that they eventually discovered an account upload/upload and logged in. The *last* command showed:

```
upload   pts/0        Mon Jun 27 07:39 - 07:49  (00:09)     evil.example.com.ro
upload   pts/1        Sun Jun 26 23:10 - 23:10  (00:00)     evil.example.com.ro
upload   pts/1        Sun Jun 26 23:01 - 23:09  (00:08)     w.x.y.z
```

Oops. Obviously, someone had obtained access as the 'upload' user, so it was time to look through the disk for some evidence of what had happened.

6 Privilege Escalation

The attacker had not been clever enough, or had not had time to cover their tracks. The '.bash_history' file in the 'upload' user's home directory had a complete list of the commands that had been executed. Several of these were downloading tar.gz files from a few places on the Internet and then unpacking and executing them. They were obscured by storing them in a directory '/tmp/. ' - that's a dot space as the subdirectory name.

One of these files, local.tar.gz was a good collection of privilege escalation exploits, that is programs that run as a normal user would give root access if the machine was vulnerable to that particular problem. For example, there was exploit code for the do_brk [BRK] issue.

Fortunately the machine was patched up to date, and the attacker had to make do with his/her unprivileged user account.

7 Attempted Phishing

One of the kits downloaded from the net was obviously designed for eBay phishing. Fortunately, the machine was only set up to pick up email and not to send it, and in any case, port 25/tcp was blocked outbound on the firewall. However, there would have been problems had the machine been configured to send email via our outbound mailserver. This is a bounce that ended up in the user's mailbox:

```
From MAILER-DAEMON Mon Jun 27 07:54:24 2005
Return-path: <>
Envelope-to: upload@victim
Received: from mail by victim.fqdn.example.com with local (Exim 3.36 #1 (Debian))
        id 1DmdCy-0005h9-00
        for <upload@victim>; Mon, 27 Jun 2005 07:54:24 +1200
X-Failed-Recipients: entdbiz@yahoo.com
From: Mail Delivery System <Mailer-Daemon@victim>
To: upload@victim
Subject: Mail delivery failed: returning message to sender
Message-Id: <E1DmdCy-0005h9-00@victim.fqdn.example.com>
Date: Mon, 27 Jun 2005 07:54:24 +1200

This message was created automatically by mail delivery software (Exim).

A message that you sent could not be delivered to one or more of its
recipients. This is a permanent error. The following address(es) failed:

  entdbiz@yahoo.com
    unrouteable mail domain "yahoo.com"

------ This is a copy of the message, including all the headers. ------

Return-path: <upload@victim>
Received: from upload by victim.fqdn.example.com with local (Exim 3.36 #1 (Debian))
        id 1DmdCy-0005h5-00
        for <entdbiz@yahoo.com>; Mon, 27 Jun 2005 07:54:24 +1200
From: ***Urgent Safeharbor Department Notice*** <service@eBay.com>
To: entdbiz@yahoo.com
Subject: eBay Fraud Mediation Request
Content-Type: text/html
Message-Id: <E1DmdCy-0005h5-00@victim.fqdn.example.com>
Sender: Upload acct <upload@victim>
Date: Mon, 27 Jun 2005 07:54:24 +1200
```

For thoses that don't deal with email regularly, this is the local mailer daemon complaining that it couldn't deliver the latter part of the text, which seems to be an email purporting to be from 'service@ebay.com', and which was asking for people to enter their account details on some bogus website.

8 SSH scanning

Another kit downloaded was obviously designed to continue the scanning process that led to the compromise of this host. Utilities in it built up a list of Internet hosts which had port 22/tcp open – almost always this is the SSH daemon. Other utilities would take a list of hosts and run through a password dictionary, both against the root account with many different passwords – 'admin', '123', '123123', 'toor', 'password', etc. and also against user accounts, typically with blank passwords, or the same as the username.

The use of the scanning utility was what tripped the alerts on the IDS console and led to the discovery of this incident in the first place.

9 Recovery

Backups were made of data only – no executable content – and the machine was re-installed from scratch. This is standard operational procedure, and ensures that the new operating system is known to be 'good' i.e. that no undiscovered backdoors exist.

Passwords were audited on the newly installed machine, and the ssh daemon was moved to an alternate port as part of a defence-in-depth policy.

10 Prevention

Obviously, the above attack could have been averted in a few ways.

Firstly I prefer to run SSH on an alternate port. This cuts down the amount of rubbish to wade through in log files, and it means that on the offchance on of your users decides to change their password to 'password', or '123', you won't be picked up by the majority of the scanners out there.

If you have users besides yourself on a machine, you can install *pam_cracklib* or *john the ripper* to regularly audit the password strength. If you find weak ones, make sure they get fixed.

It goes without saying that port 22/tcp should not be open through the firewall unless you really need it. You should also regularly patch your system to make sure you aren't running an SSH daemon which is vulnerable to an exploit.

The fact that the machine was not configured to send email, and that the firewall was blocking outbound SMTP which was not routed through our smarthost saved us from an embarassing spam incident. Some form of outbound rate-limiting on the SMTP smarthost would prevent this kind of abuse entirely.

11 Conclusion

We were lulled into a false sense of security in regard to Linux; for a year or more it seemed like the only machines that were compromised were Windows boxes, with the worm *du jour* – first Slammer, then Blaster, Welchia, Korgo, Sasser, etc. It is also quite easy to configure a Linux machine so that it can be easily compromised. We had taken countermeasures once we noticed SSH probing, but other adminstrators had not, due to both limited time to administer their machines and poor distribution of our notices regarding the SSH probing.

SSH is a more secure replacement for telnet; it encrypts data during transit which prevents eavesdropping attacks, barring man-in-the-middle games. It also makes some attempt at ensuring the identity of hosts being connected to is correct, or at least the same as last time. (Although everyone seems to ignore the warnings and not bother checking host key fingerprints.) By itself, it does not automagically make your machine secure, especially if you have weak passwords. VPNs are also perceived in this regard. Your **tunnel** can be secure as you like, but if the ends are not adequately secured, you will have malicious data passing through your SSH daemon. As someone cleverer than me has said, a tunnel is as secure as the least secure of its end points.

IDS is a technology that requires a lot of supervision, but it is definitely worth using, especially on networks which are relatively open. As the old saw goes, an ounce of prevention is worth a pound of cure, so do everything you can on the prevention side. However, detection and cure is a lot better than not knowing you have an incident in progress.

If you see this sort of thing going on, your local CERT may appreciate knowing. For me, that's AusCERT [ACT].

12 Acknowledgements

Thanks to Glen Eustace for this document template and for advice on Linux and firewalls, to Patrick Rynhart for assistance in gathering information. The incident would not have been noticed without the *snort* IDS software[SNT], due to Roesch *et al.,* together with contributed rulesets from the Bleeding Snort project[BLD].

This writeup was greatly assisted by Open Office, Fairtrade coffee, Natalie Merchant and Pink Floyd.

13 References

[ACT] AusCERT, *http://www.auscert.org.au/*
[BLD] Bleeding Snort Project, *http://www.bleedingsnort.com/*
[BRK] *http://isec.pl/vulnerabilities/isec-0012-do_brk.txt*
[HLX] Helix – Incident Response and Computer Forensics Live CD, *http://www.e-fense.com/helix/*
[ISC] Internet Storm Center, *http://isc.sans.org/*
[NC] Netcat, *http://netcat.sourceforge.net/*
[SNT] The snort Intrusion Detection System, *http://www.snort.org/*