Network Security Auditing with *aHack*

Simon J. Gerraty <sjg@quick.com.au>

Quick Consulting

ABSTRACT

This paper describes *aHack*, a re-implementation (with permission) of Alec Muffet's *AutoHack* tool. Like *AutoHack*, *aHack* is a tool for performing wide scale network security audits. It is aimed at testing the host security of general UNIX systems not firewall bastions. The tool is described briefly and the tests currently implemented detailed.

1. Introduction

Alec Muffet's *AutoHack* [Muffet1995a] is an engine for implementing wide scale security auditing of hosts attached to a network. It is designed in such a way as to be easy to introduce and integrate new tests and more importantly, when probing large networks, to do so in an efficient manner. *AutoHack* is used by Sun Microsystems to regularly check the security of hosts on their corporate network.

Since *AutoHack* is not available outside of Sun Microsystems, the author decided to implement a similar tool. With Alec Muffet's permission and his description of *AutoHack*, the *aHack* engine was soon ready. Providing it with attack and exploit code is an on-going exercise. The *aHack* engine allows for new exploits to be implemented, and run against a network without necessarily re-running previous tests.

2. The *aHack* implementation

AHack implements the final structure described in Muffet's paper. It is written in a mixture of C, perl, expect/tcl and shell scripts. Most of the tool is simply an engine to drive the attack and exploit code in the directory hack.d and to store the results in a directory per host. The storage tree is broken up at each octet of a hosts IP address so that no directory contains more than 255 entries. The engine is not described in detail as it closely matches that described by Muffet.

The *aHack* engine is actually at the end of a pipeline of commands.

genaddr.pl
: Produces on its standard output all the IP addresses described by its command line arguments.

avoidaddr.pl
: Allows certain IP addresses to not be tested.

mux.sh
: Is a multiplexer. It runs multiple copies of the command line given as an argument and feeds each one a slice of its input.

testaddr
: Checks whether IP addresses read from stdin are valid by sending either (or both) an ICMP echo packet or a packet to the UPD echo port. Valid addresses are copied to stdout.

sortaddr.sh
: Sorts the input from the multiple **testaddr** processes back into a coherent stream.

engine.sh
: Again run via **mux.sh** this is where the real work begins. Each module in hack.d is tried against each IP address read from stdin.

2.1. Engine

For each IP address read from stdin, *aHack* creates a directory named for the IP address (by replacing the ".."s with "/") and then run all the modules in hack.d that match the criteria given

Copyright © 1995, Quick Consulting

(default is all).

The basic model is that the output from an module such as A30.sendmail is stored in a file called sendmail. If no output is produced, the empty file is removed. Otherwise if there is a script named exploit.sendmail then it will be invoked.

3. Attack modules.

This section describes the attack modules currently implemented as part of *aHack*. Most of these modules are based on vulnerabilities described by Alec Muffet[Muffet1995], Dan Farmer and Wietsa Venema [Farmer1992a] and others.

Presently *aHack* implements the following types of modules.

- Probe only modules. These simply gather information.
- Attack modules. These attack network services known to be potentially vulnerable.
- Exploit modules. These modules exploit data gathered by the other modules.

The purpose of the modules is to gather data that would facilitate an attack or directly attack known vulnerabilities. Obtaining the content of the /etc/passwd file for offline password cracking is a common goal.

3.1. Information probes.

The following checks are made against all hosts.

- A TCP port scan.
- Check for rpc services advertised.
- Check for NFS file-systems exported.
- Check for HTTP daemons.
- Check for SNMP read community of "public".

Most of the attack modules, check the results of the above probes to see whether an attack is feasible.

3.2. FTP

Many sites still set up anonymous FTP so as to allow anyone to retrieve the real /etc/passwd or create/modify ~ftp/.rhosts and hence rlogin to the system. We list the content of ~ftp so that this can be checked.

There are a number of bugs in various FTP servers that allow normal authentication to be bypassed. The common **site exec** and bad password holes are checked.

3.3. Finger

Many sites provide a wealth of useful data via the finger daemon.

3.4. Sendmail attacks

Sendmail is a large, complex program and a favorite of system attackers. The first steps taken attempt to identify the sendmail binary. The greeting from the SMTP daemon is logged as a source of useful information. The following tests are then performed.

3.4.1. Debug and wizard

Two commands that should never be enabled in production sendmail binaries. The debug command allowed the Morris [Eichina] worm to break into a large number of Sun systems.

3.4.2. EXPN

A list of dangerous aliases are tested for support. A name that should not succeed is also tested as SMTP daemons such as smap from the TIS fwtk fake the answers to EXPN queries. If smap is detected most subsequent tests are skipped.

3.4.3. Sendmail security hole #1

This hole is flagged if sendmail does not complain about mail addressed to programs. Some SMTP daemon's do not complain and silently drop the mail.

If access is gained to the system via rcmd, then the tag file left by this test is checked for and if present, the sendmail hole is flagged as confirmed.

3.4.4. Sendmail security hole #2

Some sendmail's check for mail addressed to programs, but do not make similar checks on the return address. Thus mail addressed to an invalid user with a return address that is a program will often succeed where the more direct attack fails.

Copyright © 1995, Quick Consulting

Again we attempt to confirm the bug by looking for the presence of a tag file on the target system.

3.5. rexd

This service is so hopeless, that we merely look for its presence and do not even bother attacking it.

3.6. tftp

Many sites offer their /etc/passwd file via TFTP.

3.7. X11

Any host that runs an X11 display that allows us to connect to it, allows us to control the display. We simply check if the display can be opened.

3.8. ypserv

Many sites with old or poorly setup yp servers will gladly hand out their passwd files to anyone who asks. We use the ypx tool and some reasonable guesses at the YP(NIS) domain name to attempt to retrieve the passwd map.

3.9. NFS

Exporting file-systems that contain home directories to the world means that anyone that can mount the file-system can login to the host - unless rcmd/rlogin is disabled.

Even if file-systems are not exported, many NFS servers use predictable file handles which if given to the server in a request will be honored.

3.10. telnet – bad user passwords

This test checks a number of common user id's such as root, nobody, games, guest, play and any obtained via fingerd, to see whether any have blank or trivial passwords. The success rate is generally high!

3.11. rcmd

The same list of users tested via telnet plus some others are tested for rcmd access. A modified rcmd client (in perl) obviates the need to have matching accounts on the attacking system.

3.12. Buffer overrun.

These tests are not yet implemented due to their very specific nature and thus the effort required. However, many network programs are vulnerable. Any sendmail binary prior to sendmail-8.6.10 is vulnerable. The morris worm used a buffer overrun attack against fingerd on VMS.

As stated above these attacks are tedious to initially setup, but are virtually guaranteed to succeed on all but the most modern systems.

There are also many buffer overrun exploits that can be used once access to the system has been obtained and are much simpler to implement. Given a small amount of appropriate machine code, and a willing architecture (i386, sparc and most others), a root shell can be guaranteed.

4. Report

After running all the tests, a report script is run to summarize the results. Each comment in the report is tagged with 1–5 "*". A single "*" indicates information only, while 5 means the host is directly and readily vulnerable – usually as root. Some examples are shown in Appendix A.

5. Availability

Unlike many security checking tools, *aHack* actually exploits the vulnerabilities it finds. While *aHack* does this in benign ways, it can be easily modified to be quite unfriendly. For this reason, *aHack* is not publicly available.

References

Muffet1995a.
Alec Muffet, "WAN-hacking with *AutoHack* – Auditing security *behind* the firewall," *Proceedings of The 5th USENIX UNIX Security Symposium,* Network Security Group – Sun Microsystems (June 1995).

Farmer1992a.
Dan Farmer and Wietsa Venema, *Improving the security of your UNIX system by breaking into it* (1992).

Eichina.
Mark W. Eichin and Jon A. Rochlis, *With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988* ..

Appendix A

A secure hosts may show a result like (only the addresses have been changed):

```
host 192.168.12.4 not
date Fri Sep 29 17:27:24 EST 1995

192.168.10.4    *       version='Sendmail 8.6.12/8.6.9'
192.168.10.4    *       sendmail='sendmail-8'
192.168.10.4    *       sendmail protected. smap?
192.168.10.4    *       info: last probed Thu Sep 28 23:33:54 EST 1995
192.168.10.4    *       hostname='not'
```

while an insecure host may show a result more like:

```
host 192.168.10.2 rex
date Fri Sep 29 17:27:20 EST 1995

192.168.10.2    *****   sendmail security hole #1
192.168.10.2    *****   direct access as root
192.168.10.2    *****   confirm sendmail security hole #2
192.168.10.2    *****   confirm sendmail security hole #1
192.168.10.2    ****    passwd file available via tftp
192.168.10.2    ****    passwd file available via rcmd
192.168.10.2    ****    might have sendmail security hole #2
192.168.10.2    ****    got 12 passwd entries to crack
192.168.10.2    ****    direct access as uucp
192.168.10.2    ****    direct access as sys
192.168.10.2    ****    direct access as sjg
192.168.10.2    ****    direct access as operator
192.168.10.2    ****    direct access as nobody
192.168.10.2    ****    direct access as games
192.168.10.2    ****    direct access as ftp
192.168.10.2    ****    direct access as daemon
192.168.10.2    ****    direct access as bin
192.168.10.2    ****    direct access as adm
192.168.10.2    ****    access via telnet as games with NO passwd
192.168.10.2    *       version='Sendmail 4.0/SMI-4.0'
192.168.10.2    *       uname: SunOS rex 4.0  sun
192.168.10.2    *       sendmail='sendmail-5'
192.168.10.2    *       info: last probed Thu Sep 28 23:00:20 EST 1995
192.168.10.2    *       hostname='rex'
```