



Master's Thesis

# Development of a modular extensible Framework for Penetration Testing

carried out at

Information & Software Engineering Group  
Vienna University of Technology

under instructions of

O.Univ.Prof. Dipl.-Ing. Dr.techn. A Min Tjoa  
Univ.Ass. Dipl.-Ing. Dr.techn. Mag.rer.soc.oec. Edgar Weippl

by

David Huemer, Bakk.  
Karl-Vogelsangasse 4/2/2  
2000 Stockerau

Wien, 20. Jänner 2007

---

Unterschrift

# Abstract

This master's thesis deals with Penetration Testing, which is part of the computer security sector.

First some basics about computer security and legal aspects of Penetration Testing are given. Then a few definitions are made, followed by the most common threats to computers.

The main part afterwards introduces into the Penetration Testing methodology. It is divided into the different phases of a Penetration Test and also gives a short introduction to the most important protocols used nowadays.

In the following section the reporting software, which was implemented, is introduced and explained. Some code examples of it are given too.

At the end of this work some common Penetration Testing tools are listed and a conclusion is drawn.

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Initial Situation . . . . .	5
1.2	Legal Aspects . . . . .	6
1.2.1	§ 118a StGB: Illegal access to a computer system . . .	6
1.2.2	§ 119 StGB: Violation of the telecommunications secret	6
1.2.3	§ 119a StGB: Abusive interception of data . . . . .	6
1.2.4	§ 126a StGB: Damage of data . . . . .	6
1.2.5	§ 126b StGB: Disturbance of the functionality of a com- puter system . . . . .	7
1.2.6	§ 126c StGB: Abuse of computer programs or access data . . . . .	7
1.3	History . . . . .	7
1.3.1	1960s . . . . .	7
1.3.2	1970s . . . . .	7
1.3.3	1980s . . . . .	7
1.3.4	1986 . . . . .	8
1.3.5	1988 . . . . .	8
1.3.6	1990s . . . . .	8
1.3.7	2000 . . . . .	8
1.4	Definitions . . . . .	8
1.4.1	Definition of some stakeholders . . . . .	8
1.5	Threats . . . . .	11
1.5.1	Digital Threats . . . . .	11
1.5.2	Physical Threats . . . . .	14
<b>2</b>	<b>Penetration Testing</b>	<b>15</b>
2.1	Introduction . . . . .	15
2.1.1	Types of Tests . . . . .	16
2.1.2	ISO and OSI model . . . . .	19
2.1.3	TCP . . . . .	25
2.1.4	UDP . . . . .	26

2.1.5	IP	28
2.1.6	Data structures	28
2.2	Reconnaissance	29
2.2.1	Domain Name System (DNS)	29
2.2.2	Querying Sub Domains	31
2.2.3	Types of DNS records	32
2.2.4	DNS tools	35
2.2.5	Google Hacking	38
2.2.6	Social Engineering	38
2.3	Scanning	39
2.3.1	Port Scanning	39
2.4	Enumeration	43
2.4.1	Null Sessions	44
2.4.2	Banner grabbing	44
2.4.3	Operating System	45
2.4.4	SNMP	46
2.4.5	BGP	46
2.4.6	LDAP	46
2.4.7	Unix RPC Enumeration	47
2.4.8	SQL Resolution Service Enumeration	47
2.4.9	NFS Enumeration	47
2.5	Penetration	48
2.5.1	Windows	48
2.5.2	Trojans	56
2.5.3	WLAN Hacking	57
2.5.4	Web Application Vulnerabilities	57
2.6	Privilege	57
2.7	Access	58
2.7.1	Rootkits	58
2.7.2	Physical Security	58
2.7.3	Denial of Service	59
2.8	Backdoors	60
2.9	Reports	61
2.10	VulnTester	62
2.10.1	Architecture	63
2.10.2	Data structure (XML)	70
2.10.3	Stakeholders	71
2.10.4	Reports	72
2.10.5	Extending the Software	73
2.10.6	Programming Language	73

<b>3</b>	<b>PT Tools</b>	<b>75</b>
3.1	Reconnaissance . . . . .	75
3.1.1	Online query tools . . . . .	75
3.1.2	Samspade . . . . .	75
3.2	Scanning & Enumeration . . . . .	76
3.2.1	Nmap . . . . .	76
3.2.2	Paketto Keiretsu 1.10: Advanced TCP/IP Toolkit . . .	76
3.2.3	Amap . . . . .	77
3.2.4	fping . . . . .	77
3.2.5	strobe . . . . .	77
3.2.6	Superscan . . . . .	78
3.2.7	Nessus . . . . .	79
3.2.8	Pwdump . . . . .	79
3.2.9	Dsniff . . . . .	79
3.3	Penetration . . . . .	79
3.3.1	Cain & Able . . . . .	79
3.3.2	John the Ripper . . . . .	80
3.3.3	Metasploit . . . . .	80
3.4	Access . . . . .	81
3.4.1	DreamPackPL . . . . .	81
3.4.2	Netcat . . . . .	81
3.4.3	FU Rootkit . . . . .	81
3.4.4	Netbus . . . . .	81
<b>4</b>	<b>Conclusio</b>	<b>82</b>
<b>5</b>	<b>Abbreviations</b>	<b>83</b>
<b>6</b>	<b>Glossary</b>	<b>84</b>

# Chapter 1

## Introduction

### 1.1 Initial Situation

The computer technology is a very young discipline but nevertheless it has penetrated our lives in such a strong way. Almost everyone depends on computers, they are used in the super market, by architects, in offices and even for diagnosing cars. They have made our lives much more comfortable but in some way we got addicted to them. Just imagine you work in an office and cannot start your computer to write invoices, check mails, or just access the Internet to obtain useful information. Some companies like 'Amazon', totally depend on the Internet for doing their business. They do not have selling places where people can walk in look at the books or other goods, buy them and carry them home. Everything is sold through the Internet, so if such a web site is not accessible, a lot of money is lost, because a competitor is just one click away. Unfortunately some criminals have specialized on stealing information or just misuse this technology for their criminal activities.

This shows us that computers and computer networks are very essential and need to be protected. A complete new working field, the computer security, has developed. Computer security is nowadays a very important field and its aim is to make computers and computer networks as secure as possible. Of course complete security is not possible but with the concept of layered security good possibilities exist to protect the digital assets of companies and persons.

Penetration testing is part of the computer security sector, but goes one step further as the other security concepts. The idea behind it is, if hackers try to break into my computers I need to understand how a hacker thinks, that is I have to know my enemy. Thus penetration testing is the process, where computer security specialists get hired for trying to break into

computer systems and networks, just like hackers. At the end a report is generated which clearly states where security problems exist. After the flaws have been revealed they can be dealt with.

## 1.2 Legal Aspects

As a penetration tester one thinks and acts like a hacker, although a penetration tester would never do anything that could damage a computer system or any task he is not authorized for. Nevertheless it is worth looking at some paragraphs of the Austrian law before doing penetration testing seriously [51]. In the following sections 'StGB' is used for 'Strafgesetzbuch' which is best translated with 'Criminal Code' and 'Datenschutzgesetz' is 'Data protection law' in English.

### 1.2.1 § 118a StGB: Illegal access to a computer system

This paragraph principally says if someone gets access or enables another person access to a computer system illegally and profits from that the person can be punished with up to six months or 360 day rates.

### 1.2.2 § 119 StGB: Violation of the telecommunications secret

If a person tries to obtain information not intended for his or her and the information was transmitted using some kind of telecommunications channel, the person can be punished with up to six months or 360 day rates.

### 1.2.3 § 119a StGB: Abusive interception of data

This paragraph, in principle, is the same as §119 but it additionally prohibits to intercept electromagnetic radiations to obtain information. This is also known as 'tempest' in computer security, where radiations from computer monitors, for example, are intercepted and information is stolen.

### 1.2.4 § 126a StGB: Damage of data

If a person damages data that are processed in an automated way by deleting, manipulating or changing them, he can be punished with up to six months or 360 day rates. If the damage is worth more than Euro 1,817.00 the punishment can even be harder.

### **1.2.5 § 126b StGB: Disturbance of the functionality of a computer system**

If a person influences a computer system in a way that influences others' work negatively he can be punished with up to six months or 360 day rates.

### **1.2.6 § 126c StGB: Abuse of computer programs or access data**

It is neither allowed to create computer programs for manipulating other systems in a way that someone can access them illegally, nor to introduce a password in an application or software that enables a person to commit any of the other crimes listed above.

## **1.3 A brief history of hacking**

### **1.3.1 1960s**

The first computer hackers emerge at the 'Massachusetts Institute of Technology' (MIT) [42]. This is where the name 'hacker' comes from because they borrowed it from a model train group at the campus who 'hacked' model trains to drive faster.

### **1.3.2 1970s**

Phreakers misuse their knowledge to make free phone calls by hacking the telephone switches. John Draper, also known as 'Cap'n Crunch', figures out that a toy whistle included in 'Cap'n Crunch' cereal generates a 2600-hertz signal. This is the same frequency that makes it possible to access AT&T's long-distance switching system. He builds a 'blue box' that can be used to make free phone calls.

### **1.3.3 1980s**

Hacker message boards and groups arise in the 1980s. Message boards are used to share know how, serial and credit card numbers. Hacker groups get formed, one of the earliest are 'Legion of Doom' and 'Chaos Computer Club'.

The movie 'War Games' introduces the public to hacking. In the movie a teenager wants to break into a computer game manufacturer to play a game but unintentionally activates a computer which simulates war scenarios.



### 1.3.4 1986

The Government in the United States recognises a lot of break-ins into computer systems. As a result Congress passes the Computer Fraud and Abuse Act, which makes it a crime to break into computer systems. The law, however, does not cover juveniles.

### 1.3.5 1988

Robert T. Morris, Jr. launches a self-replicating worm on the government's ARPAnet. He wants to test its effects on Unix systems. The worm gets out of control and infects about 6000 computers.

### 1.3.6 1990s

In 1993 a radio station conducts several call-in contest with two Porsches, vacation trips and \$20,000 as prizes. Kevin Poulsen and two friends hack the telephone system of the radio station and let only their phone calls pass to win the prizes. Poulsen serves five years in prison for computer and wire fraud.

In this year the first Def Con hacking conference takes place in Las Vegas. It is meant to be a one-time party but it gets such a success that it becomes an annual event.

In 1995 Kevin Mitnick is captured for computer fraud. He is one of the most popular hackers as he not only has excellent technical skills but also carried out some social engineering attacks very successfully [38].

### 1.3.7 2000

In 2000 several Denial of Service attacks were launched against big companies such as eBay, Yahoo!, CNN.com., Amazon and others. Hackers break into the network of Microsoft and steal the source code of the Windows operating system.

## 1.4 Definitions

### 1.4.1 Definition of some stakeholders

This section describes some stakeholders and terms used within computer security and Penetration Testing. First a classification of hackers regarding their skill level is made. Afterwards some other terms are explained, which

are commonly used in literature. For details about hacker classification [44] is a good source of information.

### **Third-Tier Hackers or Script Kiddies**

This category of hacker does not have a detailed technical knowledge. Script Kiddies just run scripts they find on the Internet against arbitrary targets. The targets are most often selected because they have a lot of vulnerabilities. The weaker the security of a system the more interesting it is for Script Kiddies. They do not even try to cover their tracks or to understand the code behind the programs they run. They just find, download and run them without thinking if someone recognizes their activity or if they could damage the systems. Very often they even do not have enough knowledge to distinct between scripts written for Unix/Linux and scripts written for Windows. Everyone who is able to use the Internet is a potential Script Kiddy because the only know how needed for starting this kind of attack is to know how to download a program and run it afterwards.

### **Second-Tier or Dedicated Hackers**

Dedicated Hackers typically have a broad but not very deep level of knowledge. They know a little bit of everything and are not specialized in one particular operating system or technology. Very often a second-tier hacker works as system administrator and has some programming knowledge. He has developed a tool set for several tasks and knows how to use the programs but seldom codes exploits himself. He is able to automate attacks and knows how computers and computer programs work. He understands TCP/IP and tries to avoid 'noise' scans which means to run scans and attacks which are easily detectable by the administrator of a system.

### **First-Tier or Skilled Hackers**

These hackers have a very high technical skill and know several programming languages. They are able to code their own exploits and to find new vulnerabilities and weaknesses. They exactly know the ins and outs of a computer system and are able to read the code of it, so they really understand how it works. If they want to attack a system, they carefully plan the scanning and penetration phase and try to hide their tracks as good as possible. Often systems are owned months or years before somebody notices the break in of a First-Tier hacker. Very few hackers ever reach this high level of knowledge.

**Super Hackers**

This is the very highest level of knowledge a hacker can gain. If a Super Hacker wants to own a system, he will find a way to do so without ever being recognized. He knows everything about the targets he is attacking and perfectly hides his tracks. He has knowledge in all kinds of programming languages especially in C and assembler. Just a handful of hackers world wide can be classified into this category.

**White hat or Ethical Hacker**

A White hat Hacker performs ethical hacking with the aim to help organizations to secure their IT infrastructure. He always has an authorization and plans his steps very carefully. White hat hackers often find new bugs and vulnerabilities in software and operating systems. They never use these bugs to compromise other computers, but they report the problem or even publish self-written bug fixes.

**Black hat Hacker**

Black hat Hackers are the opposite of White hat Hackers. They hack computers and networks with malicious intentions. If they find a new bug they try to keep it secret as long as possible, so the systems cannot be patched. This makes it possible to misuse these new unknown vulnerabilities to compromise other computer systems.

**Gray hat Hacker**

Gray hat Hackers exist between Black hat Hackers and White hat Hackers. They usually follow the rules of ethics like a White hat Hacker but they sometimes do things a real White hat Hacker would never do. For example they sometimes hack into systems without permission but usually they just look what is going on on the system and do not implement Trojan horses or other malicious software.

**Cracker**

This is an abbreviation for a criminal hacker. The term 'Cracker' is often used to entitle a person who modifies software in a way that no license key is needed, so people do not have to pay for the software they use.

**Phreakers**

Phreakers were the hackers of the early digital networks. They usually hacked telecommunications systems to make free phone calls and played around with the capabilities of these networks. For example, if a phreaker hacks a telephone switch he is able to redirect phone calls or make free phone calls.

**Whackers**

A whacker is a newbie with very limited technical skills who is trying to attack Wireless Local Area Networks (WLANs). This term is not used very often.

## 1.5 Threats

As computer systems are very complex and as they are connected to each other through networks, a lot of threats exist. This section gives a short introduction to them. A thorough understanding of the threats is essential for planning and conducting Penetration Tests. Threats can be divided into two categories: digital threats and physical threats.

### 1.5.1 Digital Threats

Digital threats are everything that can damage or manipulate a computer system's software. There also exist some techniques, which do not damage a computer system in its initial nature, but can be used to gather information not intended to be publicly available. Viruses, Worms, Key loggers and Trojan Horses are in the damaging category and SQL injections, Cross Site Scripting attacks and Buffer overflows belong to the second one. The most important ones of both digital threat categories are described in the following paragraphs.

**Computer Viruses**

A computer virus is a piece of code which is spread with the help of another computer program, called host. Viruses are hidden in executable programs and are activated by executing them. Some viruses just display messages while others delete files or even format the whole hard disk. Nowadays viruses are distributed mostly via e-mail attachments.

**Worms**

A computer worm is a piece of self-replicating software which does not depend on any interaction to get distributed. This is the difference to viruses because a virus needs some interaction like a user sending an email. A Worm does not even depend on another service like email or FTP to infect other computers. Typically a worm uses a bug in an operating system or a program to infect other systems. The name 'worm' originates from the science fiction novel 'The Shockwave Rider' published 1975 by John Brunner. John F Shoch and John A Hupp chose the name in the paper 'The Worm Programs' 1982.

**Trojan Horses**

Trojan horses are computer programs with hidden functionality. Most often a user downloads a piece of software from an internet site and installs it. The program runs normally but another functionality is installed too, which works in background and enables a hacker to use the compromised system for its actions.

**Back door**

A program installed on a hacked system to maintain access to the system even if the vulnerability gets fixed is called a back door. A hacker typically installs it on a system right after he has gained root or admin access.

**SQL injections**

In an SQL injection attack input fields are used to enter SQL commands which are then interpreted by the SQL engine and useful information is responded back. It is also possible to bypass logins with SQL injections or gain higher rights in an application. If no SQL errors are returned by the application in an attack the hacker does not know exactly how the original statement looks like and how it can be misused. This then is called a blind SQL injection.

**Buffer Overflows**

A buffer overflow can occur in programs where the input length is not checked for its length. If more data is put into a variable than the programmer reserved for it, some other data on the stack data structure is overwritten. This can be used to gain unauthorized access to a computer system.

**Rootkits**

A rootkit is a piece of software which manipulates a computer system to do malicious things the user does not recognize. Typically some system programs are replaced by rootkits. The user does not recognize it because everything works normal but in addition the programs log user names and passwords or provide some kind of back door functionality for the hacker.

**Denial of Service (DoS)**

Denial of Service is an attack where access from users to services is made impossible. Several techniques exist but the most common is to overload a system with dummy requests so that it cannot handle legitimate requests any more. Most often this attack is launched by a hacker who has brought a lot of computer systems under his control. He then uses these systems, called zombies or bot nets, to attack his target by sending requests to it from all systems at the same time.

**Spam**

This term describes unsolicited mails which make advertisement for dubious things mostly pharmacy or porn. Some criminals have specialized on spamming and get payed by companies to advertise their products. The spammers hack a lot of computers often with the help of worms and then use them to dispatch their ads.

**Dialers**

A dialer is a piece of software which replaces the phone number used to dial into the Internet with a pay service number. Nowadays this attack is very seldom because most people have broad band access and do not dial into the Internet by phone any more.

**Spoofing**

Spoofing is a technique where an attacker pretends to be another one. This can be done in many different ways. For example IP addresses, MAC (Mandatory Access Control) addresses, or DNS (Domain Name System) entries can be spoofed.

**Session Hijacking**

Session Hijacking is similar to spoofing as an attacker authenticates to the system with wrong credentials. The difference is that in a session hijacking attack, the attacker takes over an active session a user created. This means the legitimate user cannot use the session any more and is therefore deauthenticated.

**1.5.2 Physical Threats**

As these threats do not play an important part in a penetration test, I am going to list just two of them.

**Hardware Failure**

As computers are mechanical devices errors can always occur. Especially hard disks have a limited life time because they are very stressed while a computer system is running. That is why manufacturers developed fail over techniques such as RAID (Redundant Array of In-dependable Disks).

**Natural accidents**

Earth quakes, tsunamis and other natural accidents are threats for computer systems too. Another point to mention here are terroristic attacks.

# Chapter 2

## Penetration Testing

### 2.1 Introduction

The purpose of Penetration Testing is to test computer systems and networks to learn about their weaknesses and vulnerabilities. This is done because most valuable assets are digitally stored and the companies depend on their IT infrastructure. The problem lies in the connection to other non-trustable computer systems through the Internet. Companies depend on doing business through the Internet. For example think of Amazon. They sell books, and a lot of other things on their web shop. Therefore they depend on their web site to be online 24/7. If the web shop cannot be reached, customers will go to another shop to buy their goods because the next shop is just one click away.

For doing business, online companies must save a lot of information about their customers. At a minimum they need the name, the address and the credit card information of each customer. If this information gets into the hands of the wrong people the image of the company will be damaged and customers may even claim legal issues. So the protection of these data and the availability is an absolute must for these companies.

Hackers continually try to gain unauthorized access to data or try to steal other valuable information. To fully protect the systems the idea of Penetration Testing is to think like an attacker. Attackers usually hack systems in a methodical way. A Penetration Test is also structured in phases. In the following sections I am going to describe the process of hacking a system divided into these phases.



### 2.1.1 Types of Tests

In literature [1], [52], especially on the Internet, terms like Penetration Testing or Vulnerability Assessment and others are used interchangeably but in the following document I will distinct them as following:

#### **Vulnerability Assessment**

A vulnerability assessment scans for and points out vulnerabilities but does not exploit them. It can be completely automated with tools such as Retina, ISS or Nessus. The strength of this Assessment is that it can be carried out without detailed technical knowledge, fast and automated.

#### **Penetration Test**

Penetration Testing focuses on exploiting vulnerabilities. Of course the vulnerabilities must be found before they can be exploited which is also part of Penetration testing. The whole process from information gathering over scanning and enumeration up to exploiting and writing a report is included in a Penetration Test.

#### **Red Teaming**

Red Teaming includes everything a Penetration Test does but additionally covers some more techniques like war dialing or application testing. To better understand the difference between Penetration Testing and Red Teaming Figure 2.1 is provided.

#### **Blue Teaming**

This is a very technical process which checks systems and networks to identify security vulnerabilities. Sometimes blue teaming is divided into two Levels, level one and level two. Level two additionally includes analysis of firewalls, intrusion detection systems, guards and routers.

#### **White Teaming**

As described by 'SecurEye'<sup>1</sup> white teaming focuses on the non-technical security functions within an organization. Typically security policies, procedures, architectures and organizational structures are examined.

---

<sup>1</sup>[www.cebic.com/SecurEye.pdf](http://www.cebic.com/SecurEye.pdf)

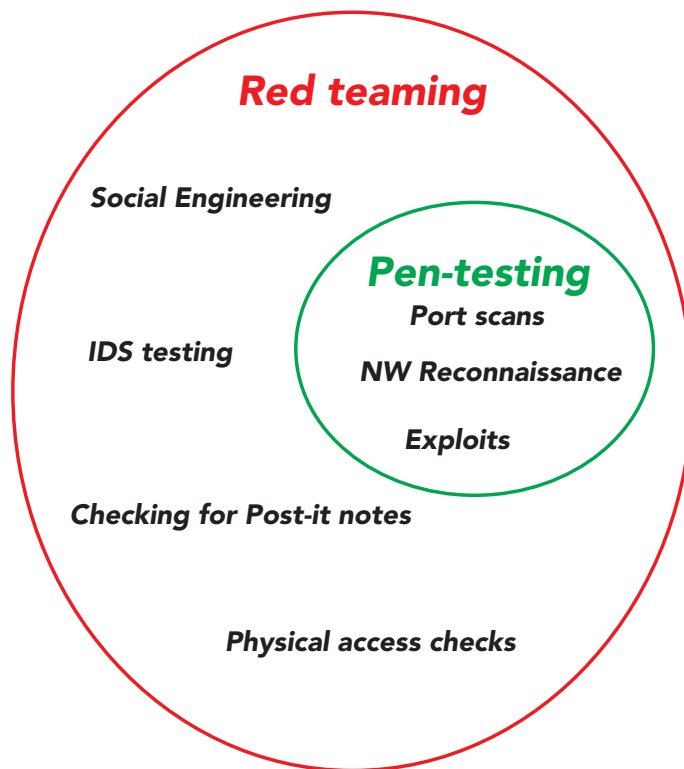


Figure 2.1: Difference between Red Teaming and Penetration Testing.

### Black Box Testing

Black box testing, also known as 'zero knowledge testing', is a type of test which best simulates an attack from outside. No information is given to the tester, he just has access to the computer system, network, or web service like everyone else. This means the tester must try to gain as much information as possible while testing or he can test what happens if he sets some kind of action and then can think what the tested object might consist of.

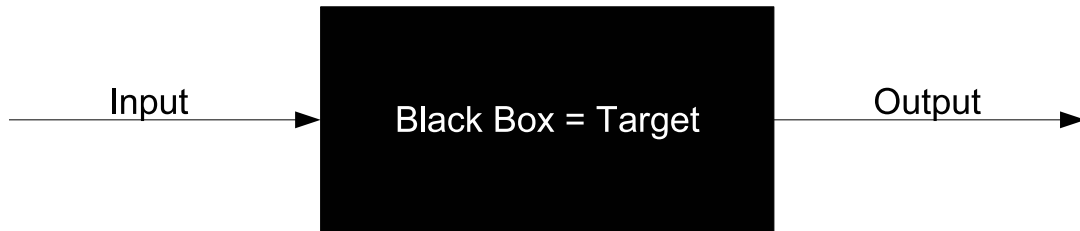


Figure 2.2: Black Box testing.

### White Box Testing

Within a white box test, all information available about the test object is provided to the tester. For example if the tester must check a web application, the code is given to him and he can analyze it for vulnerabilities and then tries to use them to do something he is not intended to do.

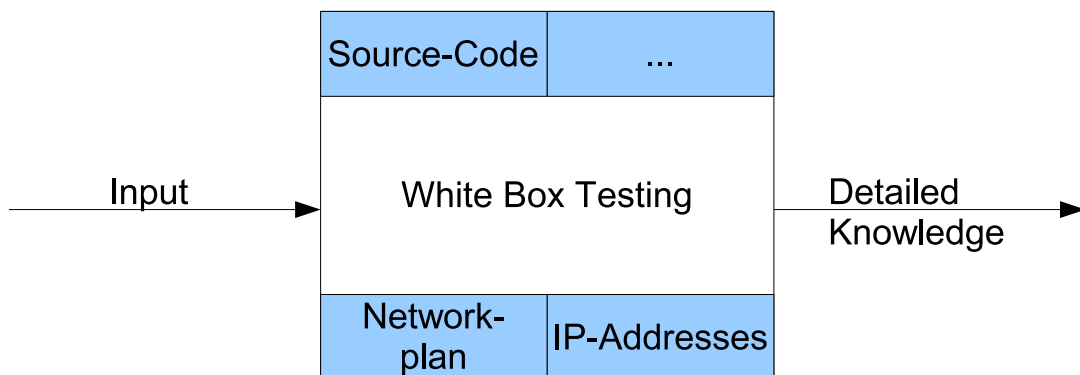


Figure 2.3: White Box testing.

### Gray Box Testing

Another very interesting test is the gray box test, because it simulates an attack from an insider. The insider has some knowledge about his target and is already located in the internal network, but he does not have the full information such as network structure or source code of applications. As a lot of attacks come from inside this type of test is very useful and informative. It reveals what an internal attacker really can do. Companies often protect their networks against outside attacks but completely forget that the real danger might sit right in the company itself.

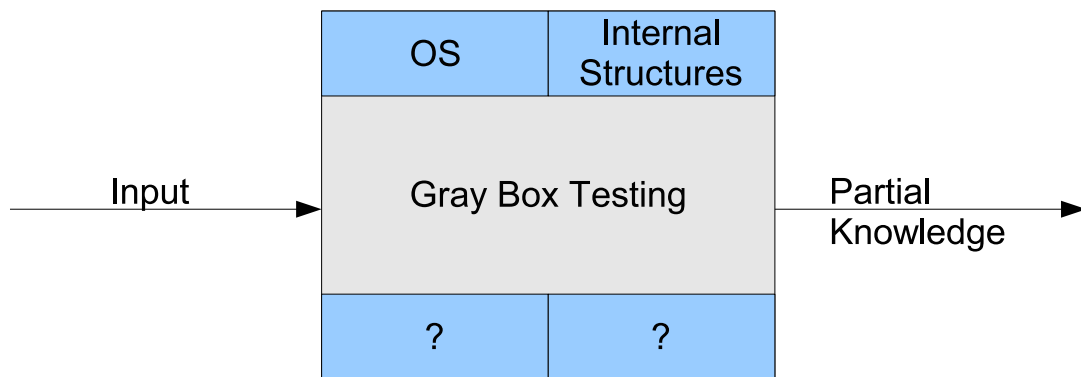


Figure 2.4: Gray box testing.

#### 2.1.2 ISO and OSI model

The 'Open Systems Interconnection' (OSI) Reference Model was developed 1980 by the 'International Organization for Standardization' (ISO) as a set of protocols that should be used by all vendors throughout the world to allow the interconnection of network devices. The protocol set did not get a standard but the model got adopted and is used as an abstract framework to which most operating systems and protocols adhere. The OSI model is described by the ISO standard 7498 and segments networking tasks, protocols and services into different layers. The model helps vendors to develop their products in way that they can be used in an open network architecture, which means that no vendor owns this architecture but the communication across vendor products is possible. One important point of the OSI model is 'encapsulation' which means, that each protocol at a specific OSI layer on one computer communicates with a corresponding protocol operating at the

same OSI layer on another computer. Each layer adds its own information to the data packet. This process is shown in Figure 2.5.

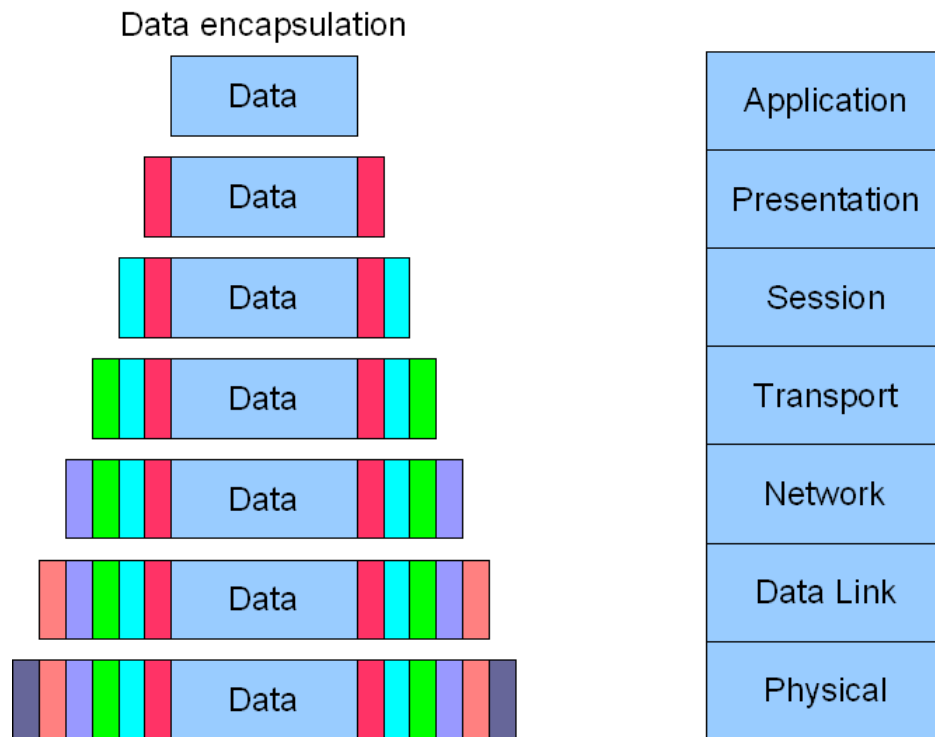


Figure 2.5: Data encapsulation.

It is interesting to mention that each layer of the 'Transmission Control Protocol/Internet Protocol' (TCP/IP) can be matched to a layer of the OSI model. This shows Figure 2.6.

### Application layer (Layer 7)

This layer works closest to the user and provides message exchange functionalities, terminal sessions and much more. If a program wants to send data over the network, it passes it to the protocol that supports it at the application layer. The layer communicates by using 'Application Programming Interfaces' (APIs) with the application. The application layer formats the data and passes it down to the presentation layer. Some examples of protocols working at this layer are the following:

- Simple Mail Transfer Protocol (SMTP)

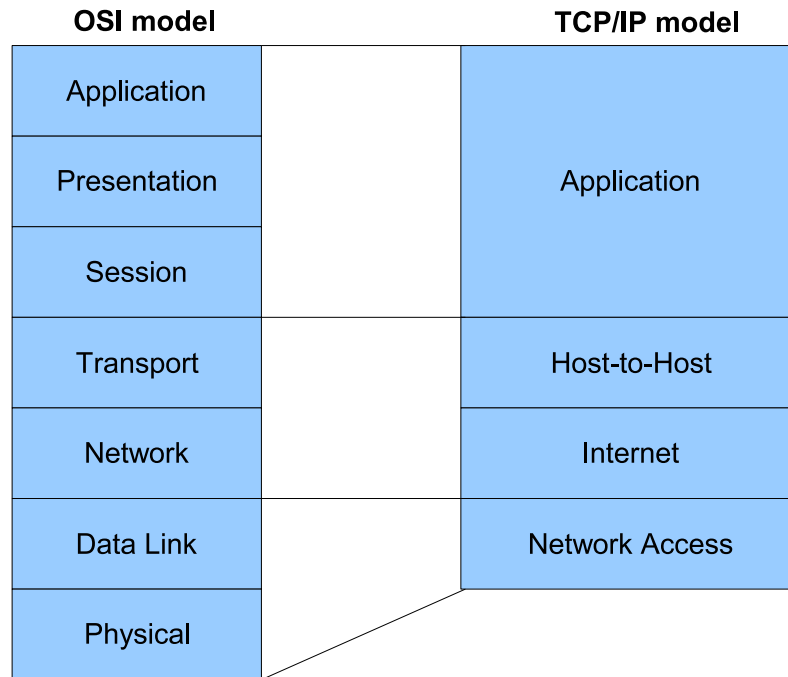


Figure 2.6: OSI versus TCP/IP.

- Hypertext Transfer Protocol (HTTP)
- Line Printer Daemon (LPD)
- File Transfer Protocol (FTP)
- Trivial File Transfer Protocol (TFTP)

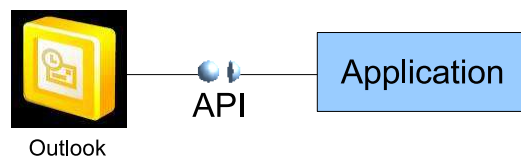


Figure 2.7: Layer and program communicate using an API.

### Presentation layer (Layer 6)

The presentation layer takes the information from the application layer and transforms it into a format understood by all computers following the OSI

model. This layer is not concerned with the meaning of data, but with the syntax and format of it. Layer 6 also handles encryption and compression of data. The following list gives some examples of layer 6 formats:

- ASCII
- JPEG
- TIFF
- GIF

### Session layer (Layer 5)

If two applications want to exchange data, a connection between them is needed. Here the session layer is used. Its job is to establish a connection between them, maintaining it during the transfer of data and then to release the connection again. This layer works in three phases – connection establishment, data transfer, and connection release. Layer 5 also provides session start and recovery and the overall maintenance of the session. This function is also known as 'dialog management'. Figure 2.8 depicts this.

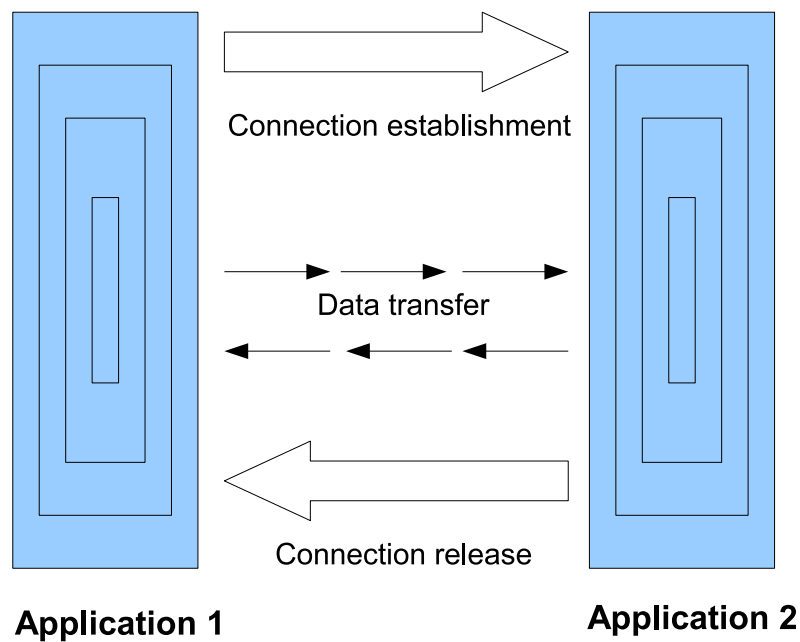


Figure 2.8: Dialog management of the Session layer.

Some examples of the session layer are the following:

- Secure Socket Layer (SSL)
- Network File System (NFS)
- Structured Query Language (SQL)
- Remote Procedure Call (RPC)

### Transport layer (Layer 4)

Before two computers communicate through a connection-oriented protocol, they will first agree on how much information each computer will send at a time, how to verify the integrity of the data and how to determine a package loss. This all is done during the handshaking process taking place at this layer. Transport layer and session layer have similar functionality but the session layer sets up connections between applications and the transport layer sets up connections between computer systems. The transport layer receives data from different applications and assembles the data into a stream that can properly be transmitted over the network. Example protocols at this layer are:

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)
- Sequenced Packet Exchange (SPX)

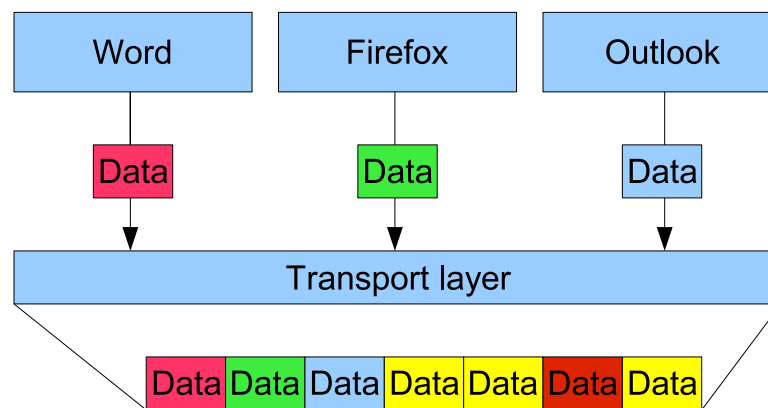


Figure 2.9: Data gets transformed into a stream by the Transport layer.



**Network layer (Layer 3)**

At the Network layer routing information is inserted into the packet's header, so it can be properly addressed and routed. The protocols at this layer are responsible for finding the best route to deliver a package and therefore hold their routing tables at this layer. These tables hold routing information for several destinations and when a package needs to be sent the routing table is consulted to find the best route. The following protocols work at this layer:

- Internet Protocol (IP)
- Internet Control Message Protocol (ICMP)
- Internet Group Management Protocol (IGMP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol (BGP)
- Routing Information Protocol (RIP)

**Data link layer (Layer 2)**

This layer translates the data packet format into 'local area network' (LAN) or 'wide area network' (WAN) technology binary format. These two technologies differ in the data structure format their interpretation of the voltage used and several other things. At this layer the network stack knows what format the data frame must be in to transmit properly over Token Ring, Ethernet, ATM or Fiber Distributed Data Interface (FDDI). The data link layer is divided into two functional sub layers – the 'Logical Link Control' (LLC) and the 'Media Access Control' (MAC). The IEEE 802.2 specification defines the LLC, which communicates with the network layer. IEEE 802.3 specifies the MAC. The list below gives some examples of protocols used at this layer.

- Serial Line Internet Protocol (SLIP)
- Point-to-Point Protocol (PPP)
- Reverse Address Resolution Protocol (RARP)
- Layer 2 Tunneling Protocol (L2TP)
- Integrated Services Digital Network (ISDN)

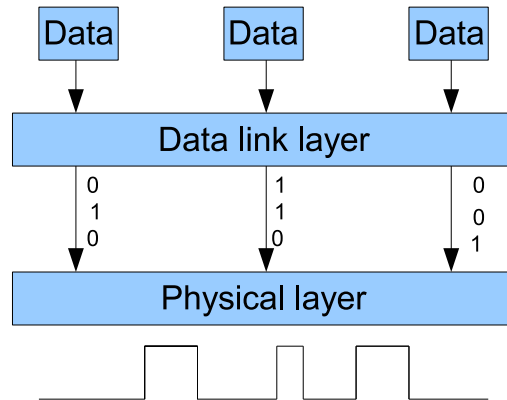


Figure 2.10: Data link layer.

### Physical layer (Layer 1)

This last layer converts bits into voltage for transmission. Depending on the technology used, LAN or WAN, signals and voltage schemes have different meanings. This layer also controls synchronization, data rates, line noise and medium access. The following list gives some examples of technologies used at this layer.

- RS232
- SONET
- HSSI
- X.21

### 2.1.3 TCP

The 'Transmission Control Protocol' (TCP) is a connection-oriented protocol which sits at the transport layer. Connection-oriented means that a specified sequence, the so called '3-way handshake' must be completed before data can be interchanged. After the 3-way handshake a virtual connection is established between the two communication partners.

TCP provides a full-duplex, reliable communication mechanism and re-sends packets if they get lost or damaged. TCP pays with a lot of overhead for this functionality because a TCP packet needs to carry a lot of additional information to provide this error correction features. This can be seen when looking at the parts of the TCP header.



Figure 2.11: TCP 3-way handshake.

TCP flag	Description
ACK	Acknowledge: Acknowledges the receipt of data.
FIN	Finished: Torning down a connection.
FIN-ACK	Finished Acknowledged: Connection gets torned down.
PSH	Push: Tells to push data up to the application.
RST	Reset: Resets connection after error.
SYN	Synchronize: Establishing a new connection.
SYN-ACK	Synchronize Acknowledged: Answer to SYN.
URG	Urgent: Offset pointer, marking a special byte position.

Table 2.1: TCP flags.

Flags in TCP are used to manage the sessions. Technically these flags are just bits, which can be set to mark a package with a certain flag. Whenever a TCP 3-way handshake is made, the SYN, SYN/ACK and ACK flags are used. Other flags can reset a connection or raise an error. The following table shows the TCP flags and their meaning.

#### 2.1.4 UDP

User Datagram Protocol is a connection-less non-reliable protocol, which means it has no error correction. The advantage of UDP is, that much more data can be transported with each package. The following table describes the most important differences between TCP and UDP.

The UDP header is very small. It just consists of four data fields, with 16 bit length each. The only additional information in an UDP header are the source port, the destination port, the length and the checksum.

Service	TCP	UDP
Reliability	Ensures delivery	Delivery not guaranteed
Connection	connection-oriented	connection-less
Packet sequencing	Sequence numbers	No sequence numbers
Congestion controls	Slow down of rate possible	No flow control
Usage	For reliable delivery	For time critical applications
Speed and overhead	A lot of overhead	Less overhead than TCP

Table 2.2: Differences between TCP and UDP.

**TCP header**

Source port		Destination port	
Sequence number			
Acknowledgment number			
Offset	Reserved	Flags	Window
Checksum		Urgent pointer	
Options		Padding	
Data			

**UDP header**

Source port	Destination port
Length	Checksum
Data	

Figure 2.12: TCP and UDP packet format.

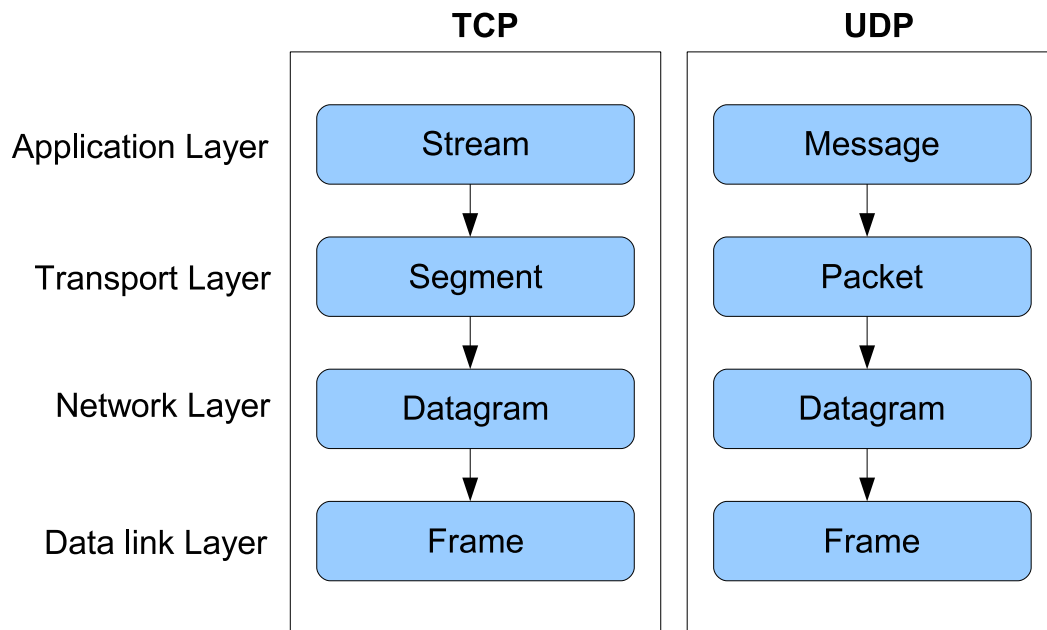


Figure 2.13: Data structures of TCP and UDP.

### 2.1.5 IP

Internet Protocol (IP) is a network protocol used for the Internet, for example. It is an implementation of the Internet layer of the TCP/IP model or an implementation of the network layer of the OSI model. It can be used to divide a network logically into subparts by the help of IP address and netmask. The netmask specifies, how many bits are used to specify the network part. IP nowadays plays a very important part in combination with TCP because these two protocols are used for the Internet.

### 2.1.6 Data structures

As mentioned above data passes down the stack and each protocol adds its information to it. This process is called data encapsulation and each stage has a specific name that indicates what happens. The names and stations are depicted in Figure 2.13.

## 2.2 Reconnaissance

The most important thing for a penetration tester is to gather information about his target. If he does not have enough information the test will fail because he will not be able to inspect all computers or applications. The information is also needed for Social Engineering attacks. Therefore he has to gather as much information as possible which includes searching the company homepage, searching the web and news groups and lots of other resources and services such as Domain Name System (DNS). The so called 'Seven-Step Information Gathering Process' is very often used to acquire this knowledge. As supposed by its name the information gathering process is divided into seven steps as can be seen in Figure 2.14. In literature Reconnaissance is often called 'Information Gathering' or 'Footprinting'. The following section describes all these steps.

The 'Domain Name System' (DNS) is a very good starting point for IP addresses and a penetration tester should be familiar with this service and its internals. Every domain name must be registered with an authorized registrar. For being able to acquire a domain name, one must have access to at least two DNS servers. The IP addresses of these servers must be entered during the registration process. These two DNS servers later are the authorized DNS servers for the domain being registered and manage all sub domains and all other services regarding resolving IPs. Everyone querying for a sub domain owned by this registered domain is going to use one of these DNS servers to get hold of the IP address the service is waiting for. Therefore the IP addresses of these two domain servers are publicly available and are the first target in a Penetration Test. To get a better understanding the following section describes the internals of DNS.

### 2.2.1 Domain Name System (DNS)

DNS is one of the most important services of the Internet. It resolves domain names, such as 'www.google.at', to IP addresses with which a computer can work. DNS is a distributed database with a tree structure very similar to the tree structure of the Unix file system. Every node on this tree is called a zone because it manages a certain zone of the Domain Name System. Zones are areas which describe the responsibility of the web servers. For example the top level domains such as '.com', '.net', '.org', '.at' and all the others are zones but one can also entitle another more specific zone such as 'ac.at', for example. Figure 2.15. shows how zones can be thought of.

If someone enters a domain name, such as 'www.google.at', into the browser the query process starts as following: If the IP address for this do-

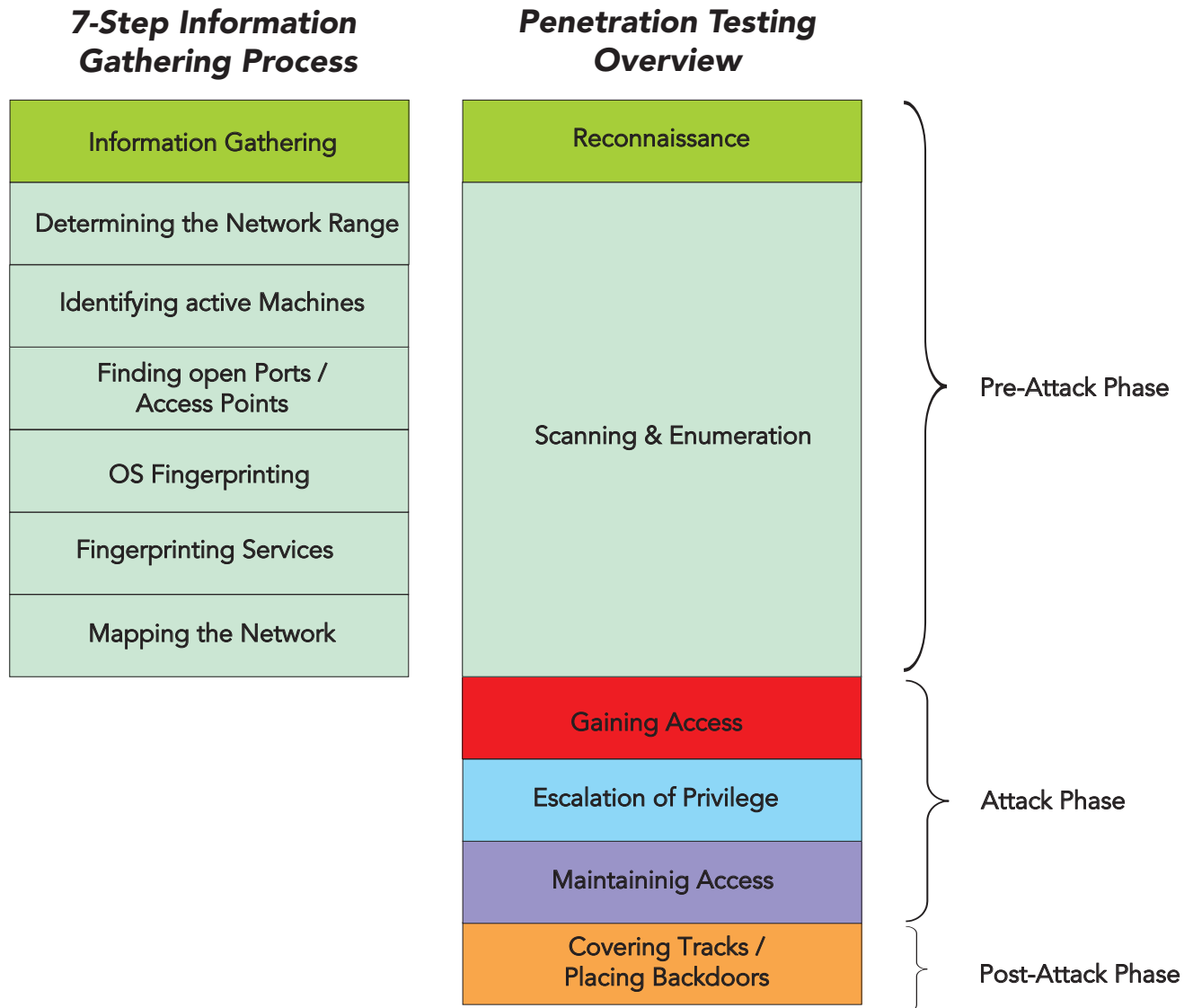


Figure 2.14: Overview of Penetration Testing.

main name is not cached, the computer asks his configured name server for the foreign IP. This DNS server recognizes that the domain has '.at' as its top level domain. If the DNS server of the client does not know the IP address too, he asks the DNS name server authoritative for '.at'. This domain server tells the client's DNS server the IP address of the DNS server responsible for 'google.at'. Now the first DNS server can directly ask google's DNS server which IP address to use for connecting to 'www.google.at'. As soon as the

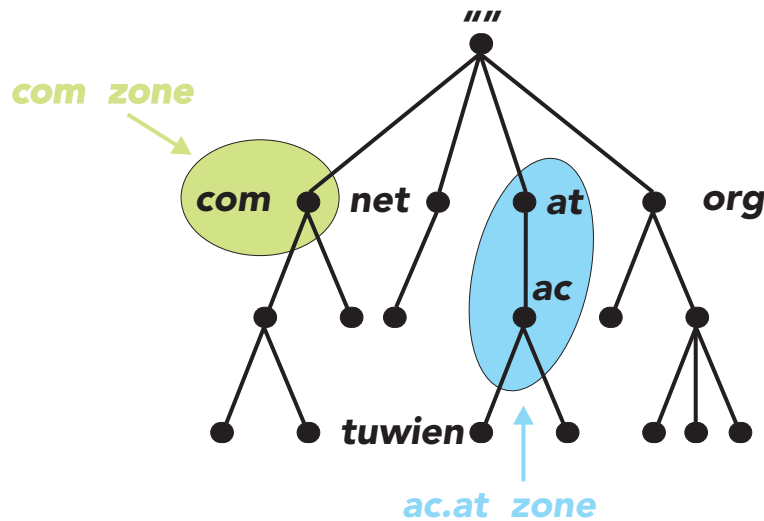


Figure 2.15: DNS zones.

first DNS server knows this IP, it puts the IP down to our computer willing to connect to 'google' and now this client can establish a connection using the IP obtained. If we want to contact google later again, this IP has been cached and does not have to be required as long as the caching limit set by 'google' has not been exceeded.

### 2.2.2 Querying Sub Domains

By now it should be clear how DNS works. The problem lies in the information leakage character of the Domain Name System. Everyone is allowed to send queries to a DNS system. As this is a very normal activity which guarantees the functionality of the Internet, such queries are not regarded as suspicious. Therefore collecting information about sub domains from companies is very easy and stealthy. Several tools for querying special information from DNS exist. The first thing in the information gathering phase is to try a zone transfer on the DNS servers of the target.

A zone transfer was developed for redundancy purposes. Every domain must have at least two DNS servers. One of these servers must be denoted as primary name server, meaning that this is the first DNS server to be contacted in a DNS query. If the primary DNS server fails, then the secondary DNS server will be contacted and afterwards, if existing, the tertiary DNS server. This means the other DNS servers must have the same level of infor-



mation as the primary DNS server. Zone transfers are used for keeping the other DNS servers up to date. At specified time intervals a zone transfer from the primary DNS server to all others takes place, so all the servers are able to serve requests. Normally zone transfers should be limited to exactly specified servers owned by the company. If an administrator has not secured or mis-configured just one of these servers a zone transfer could also be possible from outside. In such a case the hacker has a complete list of sub domains and mail servers with all IP addresses at once. This would be the most wishable situation for the hacker and the most problematic one for the company. As big companies own several thousand IP addresses the penetration tester has to filter them because not all IPs are really used. Therefore an address is only considered as relevant if the IP:

- Belongs to the organization
- Is used by the organization
- Is Registered by the organization
- Serves the organization in some way
- Is closely associated with the organization

A good starting point for querying DNS information are the following sites:

- AFRINIC - African Network Information Centre
- APNIC - Asia Pacific Network Information Centre
- ARIN - American Registry for Internet Numbers
- LACNIC - Latin America & Caribbean Network Information Centre
- RIPE - Reseaux IP Européens – Network Coordination Centre
- IANA - Internet Assigned Numbers Authority

### 2.2.3 Types of DNS records

DNS uses a special structure for managing its entries. A penetration tester must know what all these records stand for and how they can be queried for. Table 2.3 is based on [31] and gives an overview of the most important records.

Record	Purpose
SOA	Start of Authority
MX	Mail Exchanger
NS	Name Server
A	Address (Name-to-address mapping)
PTR	Pointer (Address-to-name mapping)
HINFO	Host Information
TXT	Text String

Table 2.3: DNS records

## SOA

SOA stands for 'Start of Authority'. It indicates that the name server is the best source of information for data within a zone.

## MX

This stands for Mail Exchanger and indicates which Mail server should be used when a mail has to be delivered to a mail account of a certain domain. Additionally some integer values are specified with the MX record to set the priorities of the mail servers. As an example consider the following entry was made in the DNS file: 'abc.com IN MX 0 first.abc.com'. The '0' indicates, that 'first.abc.com' is the first mail server used for delivering mails to the domain 'abc.com'. Other MX records with higher values such as '10' or '20' could be specified. If the delivery to the mail server with the lowest number, '0' in this case, fails the next number '10' is tried.

## NS

NS is short for 'Name Server' and is used to set the name servers for domains. Name Server records are used in conjunction with sub domains in which case they point to the name servers which manage these sub domains. For example if 'abc.com' has two sub domains 'subdomain.abc.com' and 'secondsub.abc.com' these two sub domains could have their own name servers. If someone queries for computers on one of these domains the DNS server of 'abc.com' can redirect to the name server of the sub domain, because it has more details about the domain it is responsible for.

## **A**

An address entry provides the name-to-address mapping. This means that if a client wants to know the IP address for a certain domain name, he queries for the 'A' record of the domain. A special variant is a 'CNAME' record which stands for 'Canonical Name'. That means if a client finds a CNAME record while it is resolving a domain name, he replaces the 'CNAME' record with the matching 'A' record and resolves the DNS name again. As an example take a look at 'mp.abc.at IN CNAME masterplan.abc.at'. If a client wants to resolve 'mp.abc.at' it is resolved to 'masterplan.abc.at' and this is tried to be resolved again. If 'masterplan.abc.at' is an 'A' record, then this can be resolved to an IP address and the client can make a connection to this server.

## **PTR**

'PTR' is used for 'Pointer' in the DNS record sets. Pointers enable address-to-name mapping. In Penetration Testing one often talks about 'Reverse DNS lookup' which means the tester has a given IP address and wants to know the DNS name for it. This is where the 'PTR' records are used, because to find the related IP the tester technically makes a query for the 'PTR' record for this address. As a result he gets the DNS name for it.

## **HINFO**

'HINFO' stands for 'Host Information' and intentionally was planned to provide information about the server hardware. This should have made it possible to provide services like FTP with information about how to deal with services. Unfortunately it was not used by many sites so it nearly no practical relevance but if some sites for some reason have provided information in their DNS through 'HINFO' then this is a valuable piece of information for a penetration tester.

## **TXT**

Text is abbreviated by 'TXT' and is used for messages up to 256 characters in length. It can be used for additional information such as to list the host's location. Probably the DNS administrator uses this field to remember something important about the host and this could be worth a look for a penetration tester too. And as this information is just one query away the existence of this record should be checked for.

### 2.2.4 DNS tools

There are a lot of tools and online services to extract the information of DNS. Typically all the online sites that can be used for DNS queries use the same tools, so this document only describes the most important tools. A more complete list of tools can be found in several hacking books especially [27], [8] and [59] . Using online tools is a good idea as no direct communication between the penetration tester and the target takes place which means that no tracks lead to the penetration tester. This is not the primary concern in a penetration test, but a real hacker, with enough knowledge would try to do his scans as stealthy as possible. So a penetration tester would normally use the following tools directly.

#### Whois

Whois is a standard tool to query the information registered for the domain. This is usually the first thing a penetration tester would query for. The tool is used by its name and the domain we are interested in. As an example consider the output for my own domain 'dhuemer.at'.

#### Host

A tool often used is 'host' which is available for Linux. A simple 'host -l example.com' [18] can be used to try a zone transfer on 'example.com'.

#### Nslookup

Nslookup is a program to query Internet domain name servers. It displays information that can be used to diagnose Domain Name System (DNS) infrastructure and helps to find additional IP addresses if authoritative DNS is known from whois.

#### Dig

The Linux manpage best describes the programs so I am going to cite it here:

dig (domain information groper) is a flexible tool for interrogating DNS name servers. It performs DNS lookups and displays the answers that are returned from the name server(s) that were queried. Most DNS administrators use dig to troubleshoot DNS problems because of its flexibility, ease of use and clarity of output. Other lookup tools tend to have less functionality than dig.

domain:	dhuemer.at	
registrant:	DH2490125-NICAT	
admin-c:	AN1331690-NICAT	
tech-c:	AN1331690-NICAT	
zone-c:	AN1331690-NICAT	
nserver:	ns1.mpca.at	<b>Name servers</b>
remarks:	83.65.5.194	
nserver:	ns2.mpca.at	
remarks:	83.65.5.195	
nserver:	ns3.mpca.at	
remarks:	83.65.5.203	
changed:	20061016 08:43:31	
source:	AT-DOM	

personname:	David Huemer	
organization:		
street address:	Karl-Vogelsanggasse 4/2/2	
postal code:	A-2000	
city:	Stockerau	
country:	Austria	<b>Registrant</b>
e-mail:	public@dhuemer.at	
nic-hdl:	DH2490125-NICAT	
changed:	20061016 08:43:30	
source:	AT-DOM	

personname:	Admin Networkpartners	
organization:	Network Partners GmbH	
street address:	Flossgasse 9	
street address:	A-1020 Vienna	
street address:	AUSTRIA	
postal code:	A-1020	
city:	Wien	
country:	Austria	<b>Administrative Contact</b>
phone:	+43012199010	
fax-no:	+43012199010	
e-mail:	admin@networkpartners.at	
nic-hdl:	AN1331690-NICAT	
changed:	20040326 12:04:24	
source:	AT-DOM	

Figure 2.16: Whois overview.

## Traceroute

Traceroute is used to see what route the packages from the source to the destination take. The program sends out packages with increasing TTL (Time To Live) values. As the error messages come from the servers routing

the packages the program can so show each server between the package passes.

## Nmap

Nmap is a very powerful and universal tool. Although the main purpose of nmap is to make port scans one can also use it for reverse dns walks. Therefore the syntax 'nmap -sL <IP range>' is used. Nmap then performs a reverse DNS lookup for the whole IP range and lists all the domain names found.

## Methodology for DNS

To find as much domains and IPs as possible it is always easier to search in a methodical way. As described in [33] there are two assumptions to be made:

- Some IP/name mapping must exist for a domain to be functional. These include the name server records (NS) and the mail exchanger records (MX). If a company is actually using a domain, you will be able to request these two special entries; immediately, you have one or more actual IP addresses with which to work.
- Some IP/name mappings are very likely to exist on an active domain. For example, www is a machine that exists in just about every domain. Names like 'mail', 'firewall', and 'gateway' are also likely candidates - there is a long list of common names we can test.

Considering these two points a good methodology is to first try a zone transfer, followed by trying to extract the Domain Records, then to brute-forcing Forward DNS and last but not least to send a Bouncing mail.

A zone transfer can be initiated with the 'host' tool. For example, a simple 'host -l |domain|' tries to list all sub domains but several other tools for automating DNS zone transfers exist. If this succeeds the mission is completed at once. To query for special Domain Records 'host' can be used with the parameter '-t' and the Record. For example 'host -t MX <domain>' queries for the Mail Servers of <domain>. To brute force Forward DNS tools such as 'jarf-dnsbrute.pl' exist which only try to resolve commonly used DNS names such as 'gateway', 'vpn' and other ones often used. To gain knowledge of the internal network Bouncing mails can be used. A Bouncing mail is an ordinary mail addressed to a non-existing recipient in the target domain. For example a mail is sent to 'asdf1234@abc.com' where 'abc.com' is the target domain. The mail server of the target tries to deliver the mail but because the recipient specified does not exist the mail is deleted and an error

message is returned to the sender. This error message often contains a lot of information about the inner working of the target domain in the mail header.

### **2.2.5 Google Hacking**

Everyone knows 'Google' as a very powerful search engine which helps to find information in the digital information jungle. Far fewer people know that this powerful mechanism can be misused to find information not intended for the public. This is possible because Google searches through the Internet and builds an index of the documents it finds. If information is not protected adequately or was published accidentally it gets indexed by the Google bots and becomes search-able. As Google allows to refining searches a penetration tester can use the search engine to query especially for password lists or other interesting things by using special functions and filters provided by Google. This is called 'Google Hacking'. Johnny Long has written a whole book [35] about Google hacking and his homepage<sup>2</sup> contains a lot of useful search strings.

### **2.2.6 Social Engineering**

Social Engineering is a hacking technique where no technical skills are needed. At this stage a penetration tester tries to gain access of obtain information from a company by just asking for it. This sounds trivial because hopefully no one would give out sensible information to a foreigner. So the penetration tester uses tricks such as deceiving his real identity and trying to use the 'language' spoke in the company. With 'language' hidden expressions are meant which are typically used in the company. As an example long names of branches are often abbreviated by their initials, so no one in the company would say 'Research and Development' but 'R&S' is used instead. If the penetration tester knows about the 'language' used, he can use it to be accepted as an employee of the company. This will make it much more easier to get the information he is interested in. The art here is not to ask the counterpart directly for the information but ask for something harmless to develop a position of trust. Then casually the person is asked for sensible information. Not only one person is asked, but several employees of the company are called so every person just gives away a little piece of information but these pieces result in a detailed picture.

---

<sup>2</sup>[johnny.ihackstuff.com/](http://johnny.ihackstuff.com/)

The most popular technique is calling the employees by telephone and if the penetration tester could manage to call from an inside telephone wire he has almost won. Other common techniques are to identify as a technician who wants to repair something or dumpster diving where the trash of the company is searched for valuable information. The most popular person regarding Social Engineering is Kevin Mitnick. He was a hacker in the 1980s and has used a lot of social engineering attacks to hack companies and government institutions. He wrote a whole book about Social Engineering [39] and now owns a computer security firm.

## 2.3 Scanning

Scanning is one of the three components of intelligence gathering. The aim of this phase is to obtain knowledge about the specific IP addresses, the operating system, the architecture and the services running at the target's side. Herefore three types of scans are used: Target Scanning, Network Scanning, and Vulnerability Scanning.

### 2.3.1 Port Scanning

Every computer has 65535 ports which can be thought of as doors into the computer. On every port a service can listen for connections to service a user. Ports 1-1024 are called the privileged or well known ports because here the most common services listen. For example port 80 is used for Web Servers, port 21 for FTP Servers, and Port 22 for Secure Shell Service. It is not specified to which port a certain service has to be bound but it is common practice for these services to use the ports mentioned above and nearly everyone does it because otherwise the services may not be found. This is especially interesting for penetration testers because ports are the entry points to computer systems. As a result a penetration tester should scan for open ports in this stage of a penetration test. All ports stated as open are potential attack points and will later on be investigated more on. The most powerful and quasi standard for port scanning is 'nmap' [45]. It is freely available and has developed its potential through the development and nowadays is not only able to do port scans of different nature but also to fingerprint systems and a lot of more. All the port scanning techniques following can be made with nmap.



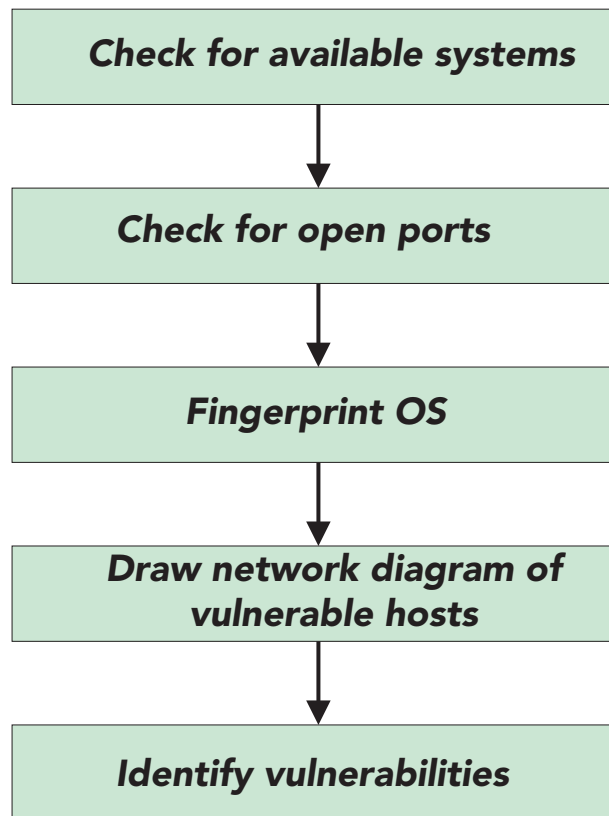


Figure 2.17: Scanning phases.

### Connect Scan

A connect scan is a typical scan in which the TCP/IP 3-way handshake is completed. For every port a scan is performed a full connection must be completed. This is very time-consuming and thus very slow.

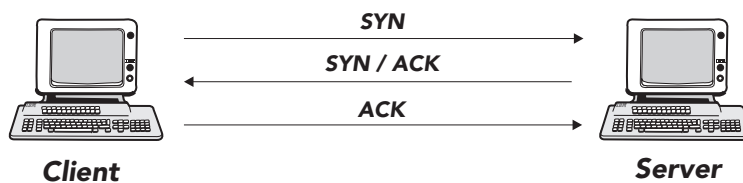


Figure 2.18: The connect Scan.

### SYN Scan

The SYN Scan is also called Half-open scan. The 3-way handshake is not completed, because the client turns down the connection by sending a package with the RST (Reset) flag set, as soon as he receives the SYN/ACK package from the server.

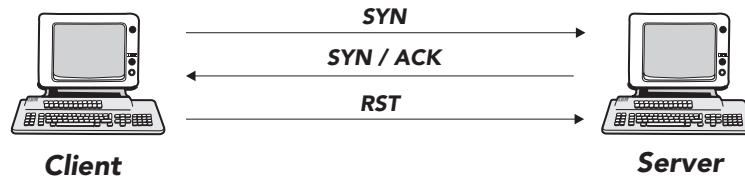


Figure 2.19: The SYN scan.

### FIN Stealth Scan

The FIN scan does not use the 3-way handshake at all. It just sends a package with the SYN flag set to the port which should be scanned. If the port is closed, the server sends back a package with the RST (reset) flag set and aborts the communication.

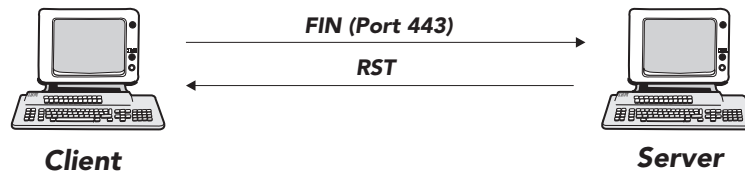


Figure 2.20: The FIN scan when the port is closed.

If the port is open, the server does not respond at all. After a certain time period the connection times out and because no response was received by the client the port is considered to be in state open.

### List Scan

I have already described the nature of the list scan above. The main purpose of this scan is to do a reverse DNS resolution of a specified IP range. This is very useful if an organization has a lot of IP addresses registered to find out the corresponding DNS names. This can be very time consuming as for every IP address the whole resolution process must be completed.

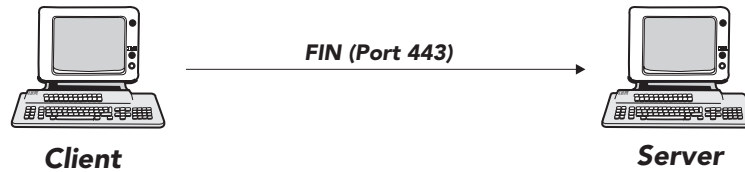


Figure 2.21: The FIN scan when the port is open.

### Source Port Scan

Some firewalls block all communications besides the ones from certain source ports. Normally the source port from which a connection is made is determined randomly by the client computer. With a Source Port scan this port can be specified. For example the user can tell its computer that it should use port 33333 as the source port.

### XMAS Scan

This is a very dubious scan which should represent all scan techniques not conform to the TCP/IP specification. In the XMAS scan, all TCP flags are set, which means this package, regarding to the specification, is complete nonsense but some operating systems are vulnerable for such packages and will crash or will let the packages pass.

### Fingerprinting Scan

This scan misuses the implementation differences of the various operating systems. Every vendor has implements the TCP stack, for example, a little bit different and therefore the different operating systems have different responses on certain packages. Nmap, for example, has a list of operating systems and their responses so it can guess the operating system. There are two types of fingerprinting an operating system. The active scan releases packages and analyzes the response whereas a passive scan just sniffs packages and tries to guess the operating system. Passive scanning is completely stealth.

### Ping sweep

Here an ICMP (Internet Control Message Protocol) ECHO request is sent to a multiple hosts. If a host is active it will return an ICMP ECHO reply. This does not always work because most networks filter such requests nowadays.

### **Vulnerability Scanning**

Vulnerability Scanning is used to find the weaknesses of computer systems on a network. This can be done by hand but a lot of good tools exist which do this automatically. The most popular tool is Nessus, which is freely available. This tool, like all others, scan the systems by sending special queries to them and matching the results against an internal vulnerability database. The providers of these programs continually bring out new vulnerability updates for their databases so the scanners are able to find the newest flaws. After a scan is completed, a report is created which describes the problems found. The report usually contains the severity, a description and a BugTrack ID of the vulnerabilities. This makes it possible to patch the most dangerous flaws first and also to get more detailed information.

### **Service Mapping**

I already described that some services are nearly ever bound to well known ports such as 80 for Web Servers or 22 for Secure Shell. This is not always true, because sometimes the services are just provided for few people. They then must know where the service needed is bound to. This makes no sense from the usability point but it increases the security. As an example consider Secure Shell. Port 22 is normally used to publish this service but several tools exist where brute force attacks against Secure Shell are tried, which all use the standard port. Therefore if a system administrator keeps port 22 closed and runs Secure Shell on another port, all the script Kiddie attacks do not work any more. This method is called security by obscurity because it does not fool a more sophisticated attacker or a penetration tester. A penetration tester would first scan the available ports and would first detect an unknown service bound to an higher port. He then would use a tool such as 'Amap' or 'Netcat' to find out which service is running at the port. This again shows how important it is to operate methodically in a penetration test because such unusual things can be overseen very easily.

## **2.4 Enumeration**

In the enumeration phase the penetration tester tries to find out as much about the services and operating systems as possible and in addition he is trying to obtain usernames and passwords from the users of the system

### 2.4.1 Null Sessions

In Windows environments Null sessions can have an information leakage character if they are not protected. Null sessions are shares which are accessible without a password. If a penetration tester finds a system on which a Null session is possible, he can connect to it by using the command

```
net use \\<IP>\IPC$ " /user:" .
```

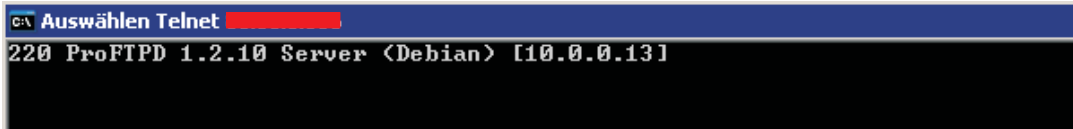
Windows creates some standard shares, such as IPC\$ and C\$, which are commonly used in Null session attacks. Once a null session connection has been established, the attacker can use tools to query for password policies, usernames and a lot more. He even is able to find out to which name the administrator account has been renamed because he can look at the 'System Identifiers' (SIDs). The last three digits of the SID describe the role of a certain user account. The numeric value '500' stands for the administrator account, '501' for guest and so on. Using this knowledge it is enough to ask for the user name which has '500' as its last three digits and the user name of the administrator is put out.

### 2.4.2 Banner grabbing

Banner grabbing was mentioned before but here I want to give some more details about this technique. Banner grabbing can be thought of as the process of connecting to services with the purpose to find out which software and which version of this software the service runs. This can be quite informative, because if a penetration tester is able to determine a special service he then is able to exactly search for vulnerabilities for this particular software and version. This is a very important step as some versions and softwares are vulnerable against certain exploits and some are not. Thus if the version is known a lot of time can be saved and no trial-and-error is necessary.

Although most port scanners have some banner grabbing functionality, two tools are used to banner grab remote services by hand. The first and most primitive one is 'telnet' and the other one is 'netcat'. Telnet is a very simple protocol which only makes a connection to the address and the port the user specifies. That is all it can do, it does not support encryption or anything else, but it is enough to grab banners of services. After 'telnet' has made a connection to the remote service the user just presses 'Enter' several times and the service mostly creates an error message and outputs the name and version of the service. Sometimes this even reveals which operating system the server is running on and the internal IP address. As an example

Figure 2.22 is provided, where telnet was used to connect to a File Transfer Protocol (FTP) server on port 21.

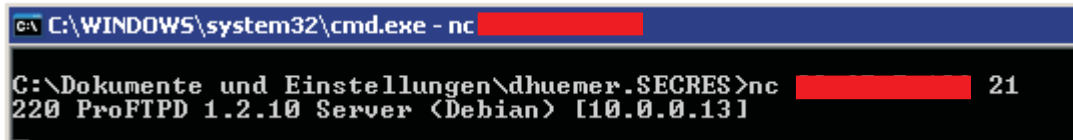


```
C:\ Auswählen Telnet [redacted]  
220 ProFTPD 1.2.10 Server (Debian) [10.0.0.13]
```

Figure 2.22: Banner grabbing with Telnet.

Another tool for banner grabbing is 'netcat' (nc). It is very often called the 'swiss army knife' for TCP, because it can make everything regarding TCP/IP. Here 'netcat' is only used for connecting to the FTP server without any option but a lot more can be done with 'netcat' so some other options of netcat will be shown later on.

Figure 2.23 shows 'netcat' in action while grabbing the banner from the same server as with 'telnet'.



```
C:\ WINDOWS\system32\cmd.exe - nc [redacted]  
C:\Dokumente und Einstellungen\dhuemer.SECRES>nc [redacted] 21  
220 ProFTPD 1.2.10 Server (Debian) [10.0.0.13]
```

Figure 2.23: Banner grabbing with Netcat.

### 2.4.3 Operating System

We already should know which operating system the servers are running. The matching step in the Enumeration Phase is to analyze the version and the patch level of them. It is interesting, which service packs are installed, on Windows operating systems for example or which version of Windows NT is used. Another point of interest on Windows systems is the 'RPC endpoint mapper'. This usually runs on TCP port 135 and can yield information about applications and services available on the target. A tool called 'epdump' is included into the Windows resource kit which can query for such information. Windows up to Windows 2000 used a service called NetBIOS which was later replaced by DNS, but is still enabled by default by all Windows distributions. This service runs on 'Universal Datagram Protocol' (UDP) port 137 and enables every user in the domain to enumerate computers and other domains.

Misusing this service also makes it possible to find the 'Domain Controller' (DC), which is the 'holy grail' of a Windows network.

On Linux systems the Kernel version and the modules loaded are of interest to the penetration tester. If the Linux system is poorly designed the 'finger' daemon may be installed which would make it possible to obtain information about the users of the system.

#### 2.4.4 SNMP

The 'Simple Network Management Protocol' (SNMP) is used to retrieve information about network devices and computers over a standardized protocol, which makes it much easier for the administrator to keep track of them. It runs on UDP port 161 and is a primary target of attackers. Unfortunately this service is most often poorly and not protected at all, but it is implemented on nearly all network devices. Very often standard passwords can be used to retrieve this information and several tools exist to enumerate the information. What makes the situation even worse is the fact, that a lot of manufacturers, such as Microsoft, have implemented their own extension to SNMP, called 'Management Information Base' (MIB). These include custom information, in the case of Windows all user names of the Windows accounts can be enumerated.

#### 2.4.5 BGP

'BGP' stands for 'Border Gateway Protocol' and is the defacto standard routing protocol on the Internet. It is used by routers to propagate information necessary to route IP packets to their destination and uses TCP port 179. With the BGP routing tables one can find out which networks are associated with a particular corporation. First the 'Autonomous System Number' (ASN) of the target organization is needed. This is a 16 bit integer which organizations buy from ARIN to identify themselves on the Internet. Second a query on the routers is executed to find out all networks where the AS Path terminates with the ASN of the organization. With this number the penetration tester can query the ARIN database and gets back an IP range belonging to the company which otherwise would have never been found. This range is added to the information overview of the target.

#### 2.4.6 LDAP

The 'Lightweight Directory Access Protocol' (LDAP) is a protocol for managing and organizing similar things in an ordered way. 'Active Directory'

(AD), developed by Microsoft, is a kind of LDAP which manages users, roles and access controls. This makes it a primary target for all attackers. If a user can compromise AD, he is able to retrieve user names, passwords and all other kind of interesting information. 'Active Directory' uses TCP and UDP ports 389 and 3268. If the compatibility mode for older versions of Windows was activated, it is very easy to enumerate these information. The tool for that gets already shipped with Windows server and is called 'ldp'. All a penetration tester has to do is to connect to the LDAP service and extract the information wanted.

### **2.4.7 Unix RPC Enumeration**

Unix uses TCP and UDP ports 111 and 32771 for the 'Remote Procedure Call' (RPC) services. RPC was developed to give operating systems to execute applications on other computer systems. 'Rpcinfo' is a tool equivalent to finger which can be used to enumerate RPC applications listening on remote hosts. If the penetration tester learns that the operating system is Unix, he should check for the existence of these services.

### **2.4.8 SQL Resolution Service Enumeration**

Microsoft's 'Structured Query Language' (SQL) server prior to the MS SQL 2000 server traditionally listens on TCP port 1433. The problem was that only one instance can listen on one port, but Microsoft wanted the MS SQL 2000 server to be able to host multiple instances on one server. They solved this problem by using UDP port 1434 as the listening one for SQL. On port 1434 the SQL Server Resolution Service manages which ports serve which client, so several instances at the same machine were possible. The tool 'SQLPing' can be used to find SQL servers listening on UDP port 1434 and the classical combination 'sa/null password' can be used to gain administrator access on the SQL server.

### **2.4.9 NFS Enumeration**

'Network File System' (NFS) uses TCP and UDP port 2049 and provides to export partitions onto the network. 'Showmount', Unix tool, can be used to enumerate all exported NFS file systems on a network.



## 2.5 Penetration

After all the preparation work has been done the next step in a Penetration Test is to use the information to gain access to the vulnerable systems. Several possibilities exist how this can be done. The following subsections explain the methods and tools used for that and how and why they work. The declared aim is to hack one or more systems and then to use these system to get more information from the network so that at the end the whole network will be owned by the penetration tester. Therefore the steps of the previous phases recur slightly adapted as soon as access to a system inside the network exists. It is tried to gain administration access to this system, then all information possible is collected to compromise more systems. For a better overview the following sections are divided into a Windows part and a Linux part.

### 2.5.1 Windows

#### Password guessing

As passwords are the security concept mainly used nowadays I want to discuss some basics first before introducing some ways how to crack them. It is very important to use strong passwords as this is the most effective way to protect against the different password cracking techniques. Details regarding strong passwords can be found at [4]. Windows uses two different types of password encryption, the LM hash and the NT hash.

The LM hash is very old and thus insecure. It is created by taking the password input by the user and pads it with NULL bytes until it consists of 14 characters. Then the password is converted into uppercase characters and split into two seven byte pieces which reduces the possible key space to 68 characters. Each chunk is used to encrypt a given string, the result of both is concatenated and stored as the LM hash. As this hash only consists of uppercase characters and two password with length seven are used it is much less. The whole procedure is described in Figure 2.24.

The second type of Windows hash is the NTLM hash. It uses the 'Message Digest 4' (MD4) algorithm to create a hash out of a password, is case sensitive and can handle up to 128 or even 256 characters, but is only available on Windows 2000 or later.

Both of these password hashes are not 'salted'. A 'salt' kind of an initialisation vector which has the effect that although two users have the same password, the resulting hash is not the same because the users have different salts.

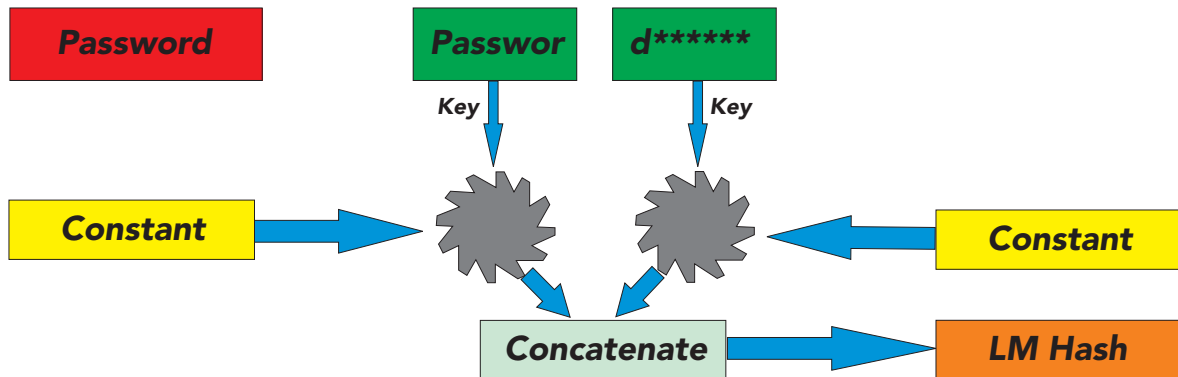


Figure 2.24: LM Hash creation.



Figure 2.25: NTLM Hash creation.

Almost always a Windows system exports some kind of file or print sharing service, which uses the 'Server Message Block' (SMB) protocol [50]. Windows versions up to Windows 2000 just used TCP port 139, but the current Windows XP and Windows 2000 use TCP port 445 for SMB too. As described before the Windows standard shares, like C\$ or IPC\$, can be used to obtain access to a system. A script using the 'net use' command can be written for that which tries every combination of user name and password from a given file. For example the following script could be used, whereas 'credentials.txt' contains the usernames and passwords separated by a whitespace such as a tabulator.

```
FOR /F 'token=1, 2*' %i in (credentials.txt)
type net use \\target\IPC\$ %i /u: %j
```

[55]

### Dictionary Attacks

A Dictionary Attack is one of the first things used to break passwords. The penetration tester or hacker provides a lot of files with tens of thousands

or even millions of words to a computer program which tries every single word with every user name. If it was successful the program outputs the user name and the matching password. This process can be done in several ways. Maybe the usernames and their password hashes are offline available which means it was able to save them from the real computer system into a file, which can be saved to the attackers machine. This would be great for the attacker as he can do the comparing process very fast. The program takes the words one after another from the word list, uses the hash function to create the hash and then compares it with the hash in the list obtained from the machine he attacks. If every word of the wordlist must first be sent through the network and then the response of failure or success must travel through the network too the cracking process takes very long and consumes a lot of bandwidth.

### Brute Force Attacks

A Brute Force Attack is a very primitive attack. It is automated and often takes a very long time, depending on the character set used. The hacker simply defines which characters may be used in the passwords and a program tries every possible combination of them. It is very unlikely to break a good password with this naive method because the number of possible passwords is  $(keyspace)^{length}$ . This means if only the 26 characters of the alphabet are allowed and the password length is 5 there already exist

$$26^5 = 11881376 \quad (2.1)$$

possible passwords. Table 2.4 shows how fast the number of possibilities grow with increasing key space. The problem is, that a lot of users tend to have very weak passwords so it is worth a try if all the other methods fail. Very often used passwords are 'password', using the user name as password, 'god', or no password at all. A lot of dictionaries with often used passwords can be downloaded on the Internet already divided into several different languages.

### Hybrid Attacks

If a dictionary attack did not work the next best method is a hybrid attack. Here the words from a dictionary are taken but are slightly modified. For example if the dictionary contains the word 'password' words like 'password123', 'pass' or 'wordpass' are tried. The logic for this is built into the programs and the better the programs implemented it, the faster the cracking

Key space	Possible characters	Length	Number of passwords
Case-insensitive alpha characters (a-z)	26	7	8.03E+09
Case-sensitive alpha characters	52	7	1.03E+12
Alphanumeric characters	62	7	3.52E+12
US English keyboard characters	94	7	6.48E+13
Case-insensitive alpha characters (a-z)	26	15	1.68E+21

Table 2.4: Growing of possible passwords with increasing key space. [30]

process is. 'John the Ripper' is an excellent tool which is capable of doing all kind of attacks including hybrid attacks. Its logic is one of the best at the moment and in addition this tool is freely available.

### Rainbow Tables

The fastest method for cracking passwords is by attacking them with so called 'rainbow tables'. Rainbow tables are just lists with word like normal word lists but the computation of the hash has already been done. This means not the words themselves are saved in the file, but the hashes of the words. All an attacker has to do is to take a password hash he obtained from the user account he wants to crack and search for it in his Rainbow tables. This means no computation must be done, only a lookup in a table which is enormously fast. Not all passwords can be cracked with rainbow tables, since some systems like Linux use salts for their passwords. Windows passwords for example do not use salted passwords so rainbow tables can be used to crack them.

### Obtaining Passwords

As mentioned above password cracking is much faster if it can be done offline, thus the attacker must be able to retrieve the files storing the password hashes somehow. This can be done in several ways but it should be mentioned that it is very easy if the attacker has physical access to the computer. The system can be booted with a Linux Live distribution and the files can be stored on an USB-Stick, for example. Under Windows the attacker must get rid of two files, the 'SAM' and the 'SYSTEM' file to crack the passwords.

The 'SAM' file contains the password hashes but they are encrypted with a 'System key' (Syskey), which is stored in the 'SYSTEM' file [34]. Another way is to steal backup tapes from a Windows computer system, because if the backup is not encrypted these files are contained in the backup too. Logging keystrokes with a key logger is also possible to obtain passwords directly, and it is also possible to sniff passwords directly from the network. Windows even caches some passwords which can be dumped directly with some programs and other tools such as 'pwdump' are able to dump password hashes directly over the network using 'Null Sessions'.

On Linux systems the '/etc/passwd' and '/etc/shadow' files are of interest. The first one contains the user names and is readable by every user, the second one contains the hashed passwords and is only readable by the user 'root'.

### SMB redirect

Another way to obtain the user credentials is to trick the user into clicking a link sent by the attacker. The link points to a SMB server manipulated by the attacker. As soon as the victim clicks on the button his computer tries to connect to the server the link points to and tries to log in with the credentials stored. The server under the attacker's control sniffs the user's password hash and is now able to crack them offline.

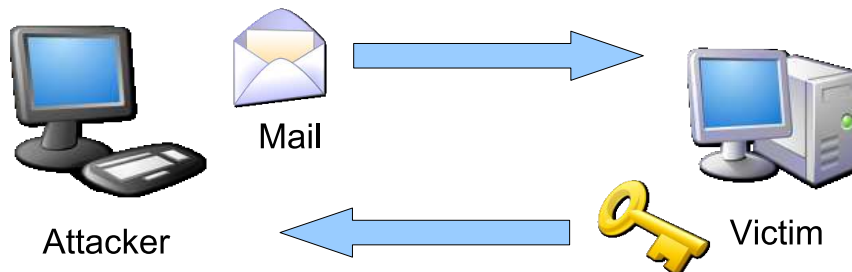


Figure 2.26: Link manipulation.

### Sniffing

Sniffing is a very popular technique to eavesdrop information and passwords. This can be done in a lot of different ways but the most common one is in combination with a spoofing attack which will be described afterwards or with an 'Address Resolution Protocol' (ARP) attack. In short the attacker sets

his network interface into promiscuous mode. This mode tells the interface to accept all packets it recognizes on the network even if they are not sent to his address. The next step is to spoof an IP address or to change the ARP information of a switch, in a way that enables the attacker to 'see' all network traffic from or to the victim. If the user enters passwords and they are not encrypted, the attacker sniffs them and knows them at once.

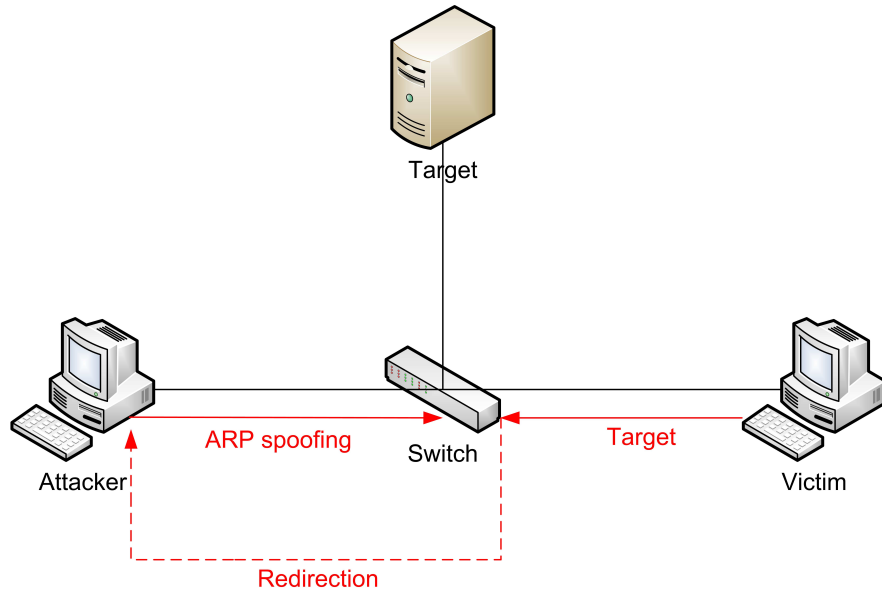


Figure 2.27: ARP poisoning.

## Spoofing

In computer communication, especially when using the Internet, some standards were developed which are used, so two or more computers can communicate with each other although they use different operating systems. The problem with all these standards is that they were created decades ago and without security considerations. These old protocols are still in use because they work just fine except of security concerns. That is the reason why spoofing attacks on IP addresses, MAC addresses, DNS, and some other protocols is possible. As spoofing attacks can be very complex I am going to describe just a few of them and not all the technical details. The interested reader may consult [46], [11], and [41] for more details.

The first spoofing attack I want to describe is IP spoofing. This in fact is very trivial: before the IP packet leaves the computer of the hacker, he

just changes the IP address in the header of the package and then dispatches the package. How the structure of an IP header is manipulated can be seen in Figure 2.28. A lot of tools exist to manipulate IP packets or even craft a whole IP package at all. The problem with these attacks, as with nearly all spoofing attacks is, that the response to the packet is sent to the source IP, which was specified in the packet. The receiver does not check if the IP packet really came from this address, which in fact raises two problems. The first is if an attacker spoofs an IP address and wants to see the response to the package, he must combine this attack with another one, for example sniffing the traffic. Otherwise the response will be sent to the IP address specified and the attacker will never see it. The second one is that this behaviour can be used to launch DoS attacks.

DNS spoofing is one attack method often used by internal attackers. Again the problem lies in the trust of the client because if a computer wants to resolve a DNS name into an IP address he sends out a DNS request to his DNS server. If an attacker sniffs the traffic from the client and recognizes, that the client queries for an IP address he immediately sends a response to the victim with a wrong IP address. The client accepts the first response he is receiving for a DNS name resolution and drops all other responses. That means if the attacker responses first he can redirect the victim to every IP address he wants to.

The last spoofing attack that will be mentioned here is spoofing MAC addresses. Companies often use some security features in their networks which should guarantee, that only allowed clients can access the internal network by plugging the network cable into the computer's plug. The only security feature used for this is often a check of the MAC address of the network interface, which is stored into a table. If a computer with another MAC address is used at this network cable later on, the switch of the network recognizes the different MAC address and denies access. Unfortunately the MAC address of a network card can be spoofed, which, for example, is done very easily in Linux by just typing the following command:

```
ifconfig eth0 hw ether <new MAC>
```

This is enough to change the MAC address into a new one. If checking the MAC address is the only protection of the network, an attacker can use this technique and immediately has access to the internal network.

### **Buffer Overflows**

A buffer is an array of bytes used by programs to store information into them. For example C uses a stack structure for local variables. A stack is a 'last

Version	Header Length	Type of Service
Packet Length		
Packet Identifier		
Fragmentation Data		
Time to Live	Protocol	
Header Checksum		
Source Address		
Destination Address		

Figure 2.28: IP address spoofing.

in, first out' (LIFO) data structure, which means the value most recently pushed on the stack is the first that is going to be popped from the stack. In addition function arguments, return addresses and registers are stored on the stack. Several ways exist to build a program stack but for historical reasons the stack grows down, that is it starts at a fixed value then allocates local variables and pushes parameters into lower addresses.

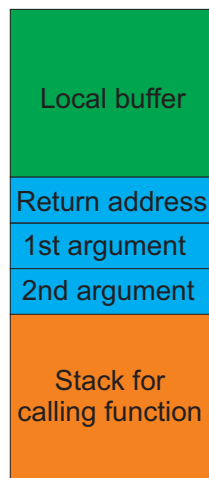


Figure 2.29: Structure of a buffer.

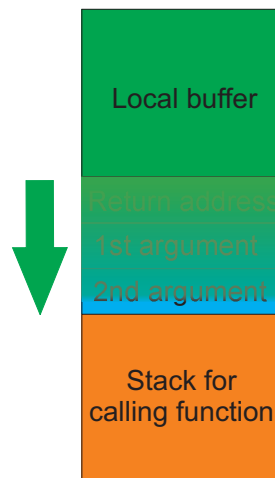


Figure 2.30: Buffer gets overflowed.

Program languages such as C or C++ do not check automatically if a



user puts more information into a variable than it can store. If this is done, the adjacent memory locations are overwritten. This can lead to very problematic situations because right after the local buffer return address is stored. This address tells the program what should be executed next. If an attacker finds out how much data he must write into the local buffers to overflow the return address, he can put some shell code into the program's memory, overflow the buffer and then the shell code he put into the memory gets executed by the program. Most often this attacks provide the attacker with root or administrator access which is the worst situation that could occur. Several different buffer overflow attacks, like heap overflow or stack overflow, can be distinguished but this is behind the scope of this work. More details can be found at [14] and [25], for example.

### **Keyloggers**

Key loggers are a very efficient way to get user passwords or record other information. A key logger simply copies every key stroke the user makes on the keyboard. Two types of them exist – software key loggers and hardware key logger. Software key loggers are installed as software on a computer system and copy the data stream into a file stored on the hard disk. The file and the key logger are hidden from the user. Then, for example at predefined intervals, this file is sent to the hacker automatically. A hardware key logger on the other side gets installed between the computer and the keyboard. Two connection interfaces, 'Universal Serial Bus' (USB) and 'PS/2', are currently used. The key logger has build in some storage chip on which the information typed gets stored. The hacker must then get back to the computer and collect his key logger again.

For both versions of a key logger the hacker must have some kind of privileged access to a system. For software key loggers it is enough being able to install software, but for hardware key loggers he must have physical access to a machine two times at least. If such an attack succeeds the passwords and accounts of the users are revealed.

### **2.5.2 Trojans**

Trojans, or trojan horses, have their name from greek history. They are programs that have some functionality useful for computer users and are freely available for download on the Internet. As soon as the user installs them onto the system, the trojan also installs another program or hidden functionality which enables the hacker to communicate with the system. As

an example it could open an additional port on which a program listens for connections from the hacker on which he now can enter the system.

### 2.5.3 WLAN Hacking

To gain access to wired networks the attacker first must gain access to a company's building, which is very deterrent and at least should be very difficult. This is not the case with wireless networks (WLANs) which do not need cables but communicate broadcasting the data. As the WLAN's broadcasts often can be received outside the company, hackers can easily access networks. As a result computer manufacturers developed encryption systems, that should restrict hackers from intercepting data directly. 'Wireless Equivalent Privacy' (WEP) was first introduced to protect against sniffing WLAN networks. WEP uses 64 or 128 bit encryption but can be cracked very easily. A lot of powerful scripts exist, which enable a hacker to crack WEP security, so cracking a 128 bit WEP key just takes about twenty minutes.

### 2.5.4 Web Application Vulnerabilities

Very often networks get hacked through their services and programs that a server part of it runs. For example a lot of servers run some kind of web server with a web application. If the application is programmed badly or the web server has some other kind of vulnerability and a hacker is able to misuse it, he can compromise the server. As soon as he owns the server, he is one step further and at least owns a box in the 'Demilitarized Zone' (DMZ). He then will start the information gathering process again to identify new targets and then will attack them until he reaches his aims.

## 2.6 Escalation of Privilege

As soon as the penetration tester has gained access to a system the ultimate aim is to be 'administrator' or 'root'. How this can be reached depends on the type of access he currently has to the system. Often a command shell has been opened for the attacker where he now can send commands to the system. Some buffer overflows exist, which grant access to a Windows computer as 'SYSTEM' which can even do more than 'administrator'. The next step then is to place the following command to the command prompt:

```
net user /add hacker
```

The command creates a user named 'hacker' with normal user rights. This is not what a penetration tester looks for, so he adds this user to the group 'administrator'. This is done with

```
net localgroup administrators hacker /add
```

After that the user owns the box because he has administrator access with the user name 'hacker' and a blank password.

## 2.7 Maintaining Access

After gaining access to a system and escalating the privileges a penetration tester wants to maintain his access to the system. This means he wants to be able to come back to the system even if the vulnerabilities are fixed in the mean time. This is done by installing additional programs on to the system which open additional hidden communication channels controlled by the tester. The most important tool, rootkits, are discussed in the following section.

### 2.7.1 Rootkits

Rootkits are very useful programs for maintaining access to a computer system. It can be differentiated between kernel rootkits and usermode rootkits. In principle both types have the same working principle – they replace system tools with customized versions and hide their additional activities. The user does not recognise any difference as everything works normal. Usermode rootkits are the less dangerous one because it is possible, though very hard, to detect them. Some programs exist which automatically check for them. Kernel rootkits directly manipulate the system at the kernel level which makes them completely invisible and undetectable. This type of kernel can manipulate everything and can, for example hide files or open back doors, without a chance to notice the existence of the rootkit.

### 2.7.2 Physical Security

Physical Security is another very important security issue. If someone has physical access to a machine there is no way to keep him from getting root or administrative access to it. A lot of ways exist for this but the most popular is to boot the computer from a bootable CD-ROM with a malicious program on it. This program enables the hacker to reset the password or even to log in without any credentials. On Linux systems, for example, it is enough to

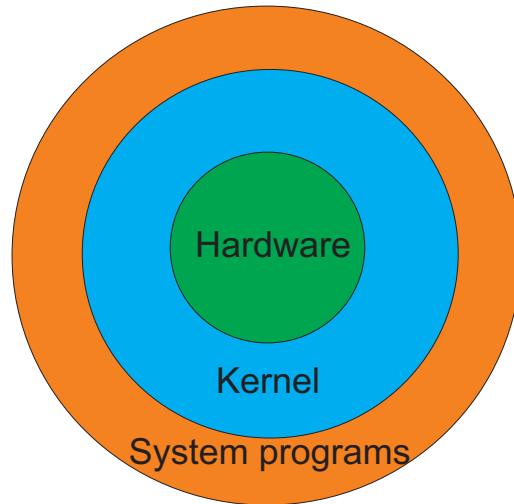


Figure 2.31: Layers of an operating system.

boot a computer with a Linux live distribution and to mount the partition containing the root file '/'. The hacker then open the '/etc/passwd' file and removes the 'x' character from the root user's entry, which is the placeholder to tell the operating system to use shadow passwords. Then he saves the modified file back to hard disk and reboots the computer. He now can login as 'root' without any password.

### 2.7.3 Denial of Service

Denial of Service (DoS) attacks first seem to have nothing to do with hacking a box, because this kind of attack prohibits other users to use services by overloading them with request, for example. But in combination with other attacks DoS can help a hacker to penetrate a system. Some spoofing attacks only work if the host with the real IP does not respond because otherwise this client would see that a package is sent to him without although he has not sent a request and then would reset the connection. So a hacker does the following: He launches a DoS attack against a client, lets say 'X'. Client 'X' now cannot respond to any communication because he is overloaded. Now the attacker starts sniffing all packages intended for 'X', spoofs the IP address, for example, from 'X' and sends packets with the IP address of his victim to, let's say server 'S'. The communications partner receives the packages from the attacker, thinking they come from 'X' and responds to them. Normally 'X' would now reset a connection, because he never tried

to contact 'S' but he is overloaded and cannot do so. Attacker 'A' sniffs these packages and thus has can complete the three way handshake. Now 'A' has a communications channel with server 'S' and server 'S' thinks he is communication with 'B' which he grants several rights. These rights can now be misused by the attacker 'A'.

## 2.8 Covering Tracks and placing Backdoors

The last step a hacker would do is to ensure that nobody knows that the system got hacked and to manipulate the system in a way that he can come back to it whenever he wants. This is done by placing backdoors on the system and manipulating the system log files. Backdoors is a way to log into a system without using the access control mechanisms the administrator of the system set up. For example take the Windows logon mechanism, where the user must enter user name and password to being logged on. If a hacker placed a backdoor, like 'DreampackPL', he can enter some secret command into the password field and now can use the system without any password.

On Unix Systems with shell access a lot of backdoors work in combination with trojan horses or rootkits. Normal system programs such as 'login', 'passwd' or 'sshd' get replaces with modified versions, which have some hidden functionality and enable the login with a special password that only the hacker knows. This password gives back a root shell to the hacker but other users do not recognize a different behaviour of the system, because their login process works just like ever.

Another very popular way is to open an additional communications channel from the inside to the outside with 'netcat'. This is also called a 'reverse shell', because the hacker instructs the victim's computer to open a connection from the inside network to his computer. This often works because most firewalls filter incoming traffic very restrictive but outgoing traffic very gentle.

Rootkits often contain some logic to hide files, the hacker places on the machine. For example the program 'ls' is replaced with a version that does not show files that match some predefined patterns, like all files that start with 'xy'. If the hacker then creates a file 'xyhackertool', the system will not show it when 'ls' is used. The same can be done for processes running on the machine. Certainly the hacker does not want his reverse shell to show up in the process list, so he uses some rootkit or similar software to hide it.

After hacking and manipulating the system, the last step is to modify the log files, that is the hacker covers his tracks. Tons of tools exist which automatically change log files. Logs are protected by the operating system

during run time, so tools are needed to modify them. As soon as they are made writable, all malicious activities are deleted. One noisier method is to flood the log with a lot of information, so the administrator does not find the real actions done in the huge amount of data but he definitely recognizes that something is going on on his system, so the first method should be preferred.

## 2.9 Reports

Reporting is one of the most important parts of a penetration test because it represents all activities done before. Even if the testing process was excellent but the report delivered to the customer was not very good, the work effort will not be honored, because the customer only sees the results. Thus the report must be created very carefully.

No static structures for penetration testing reports exist, but some points are an absolute must. Typically a report start with a page describing who conducted the test, what was tested, which knowledge the testers had, what type of tests were used and how long the test took. During the test it is extremely important for the testers to document every command and every action they do, because if a system gets damaged, it otherwise is hard to proof that the penetration tester did not damage it. The best way is to document each command with a timestamp that states, when it was launched. Some testers even film the whole penetration test.

A report should, at least, contain the following sections:

- Executive Summary
- Risk matrix
- Proof of Concepts
- Remedies and Workarounds
- Best practises
- Final summary

A more complete list can be found in the 'The Open Source Security Testing Methodology Manual' (OSSTMM)<sup>3</sup>. The following list show the most important parts regarding the OSSTMM:

- Date and time of test

---

<sup>3</sup><http://www.isecom.org/>

- Duration of the test
- Testers and analysts involved
- Test type
- Scope
- Index (method of target enumeration)
- Channel tested
- Vector of the test
- Verified tests and metrics calculations of operational protection levels, loss controls, and security limitations
- All tests which have been made, not made, or only partially made and to what extent
- Any issues regarding the test and the validity of the results
- Test error margins,
- The processes which influence the security limitations,
- Any unknowns or anomalies.

## 2.10 VulnTester

One of the most important parts in a Penetration Test is definitely creating the report. Every work and effort done before is worthless, if the report does not communicate the findings to the different stakeholders. This means if all IPs were found scanned and a lot of security flaws were found they must be shown to the management in a concise way. Often several reports with different granularity are needed. This is because the management does not have time and knowledge to review the full technical details revealed during the test. A database administrator on the other hand wants to know all details of possible SQL injections and a system's administrator wants to learn everything about network weaknesses and configuration problems. Generating these reports is very time-consuming and is very annoying because most of the data is already stored digitally somewhere.

To shorten this process I implemented a software program that uses existing tools as basis, takes the output of them and then wraps them into an

Extensible Markup Language (XML) format. Each result of a program is first parsed into its own XML file. After all programs finished their work, one large XML file, containing all the information of the other programs, is created. As this file uses XML, it can be parsed as wanted, meaning the data can be extracted as wanted. This enables the creator of the report to exactly define which information should be integrated into the report which makes it possible to create reports with different information for all stakeholders very easy. The results are then put into an Rich Text File (RTF) and can be adapted with OpenOffice.org<sup>4</sup>, for example, which is freely available.

At the planning phase of the software, several thoughts were in mind. First tools in computer security change very often and, of course, it should be possible to integrate them into the reporting framework. Second, most of the tools are Linux tools and freely available. The excellent Linux Live Distribution BackTrack<sup>5</sup> contains most of the tools needed in a penetration test, which makes it the first choice. Another point of interest was to enable the reporter to generate reports for different stake holders and last but not least, how the software should be implemented so that it can be adapted and expanded easily. This means a flexible and easy to learn programming language is an advantage.

As a consequence the reporting framework was planned modular, using XML as a data structure, building on BackTrack and using Python<sup>6</sup> as programming language. The rest of the document describes the detailed architecture and structure of the reporting tool, how it can be extended and how it can be used. Interesting to know is that the software already got implemented and is used by a large Austrian bank as a tool in their penetration tests.

### 2.10.1 Architecture

In software design it is very important to have a good architecture, because this is the most fundamental part of the whole program. If the architecture does not match the needs or is inflexible, the user later on will suffer. After having talked to the future users of the system I decided to use a modular architecture which only runs locally. A client/server architecture with would have been also possible but then not realized because no real advantages were given but it would have meant additional effort implementing security for data protection. As a result the solution was to logically divide the application into core components.

---

<sup>4</sup>[www.http://de.openoffice.org/](http://de.openoffice.org/)

<sup>5</sup><http://www.remote-exploit.org/index.php/BackTrack>

<sup>6</sup>[www.python.org](http://www.python.org)



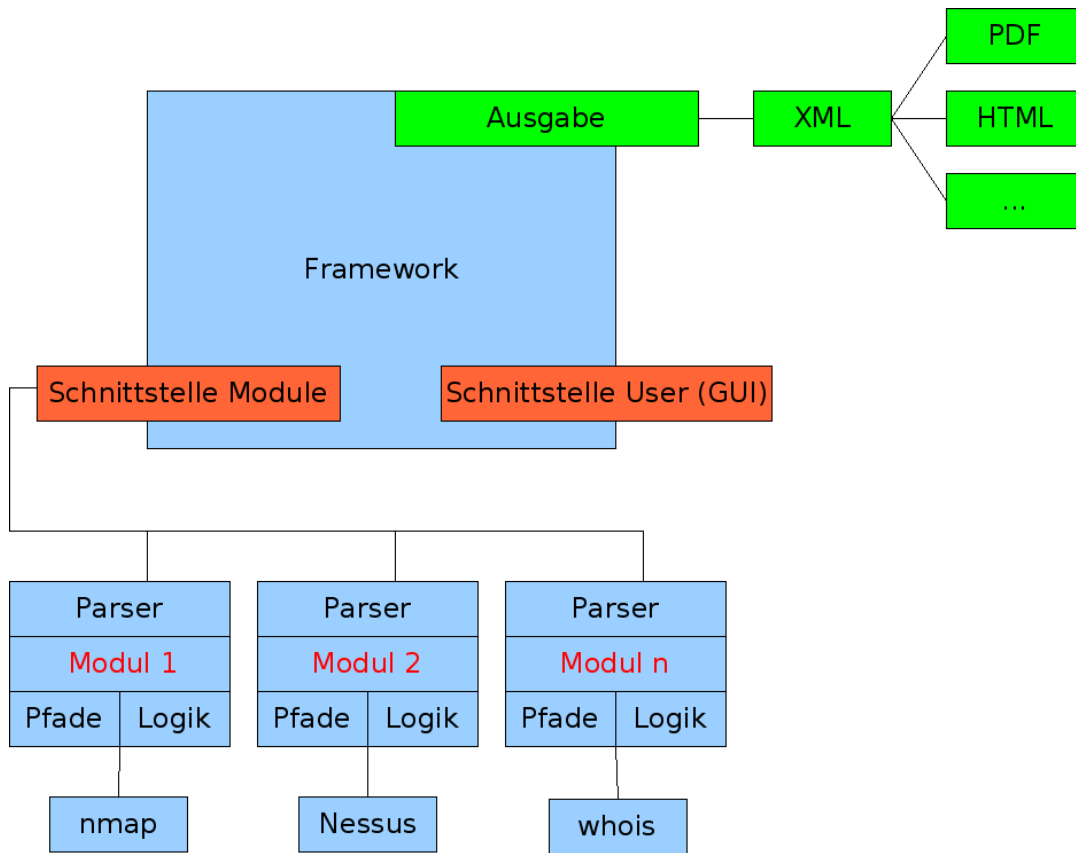


Figure 2.32: Modular architecture of the Reporting Tool.

### Framework

The central part around which all other components are grouped is the reporting framework itself, which takes the XML output of all the other components, processes it, and creates the one main XML files which contains all information. This framework has three specified interfaces for communicating with all the other components: a module interface, an output interface, and an interface for the graphical user interface. The Module interface is the part of the framework where the preprocessed XML documents from the programs used are imported. These files are then aggregated into the main XML document. This main document then gets processed in the framework as needed by the person creating the report. For example if a report for the management should be created the framework extracts predefined tags from the XML file it created before. These information is placed in a separate file containing only the information relevant for the stakeholder 'management'.

Now the output interface comes into effect. It takes this file and creates a RTF document out of it. Now the penetration tester can open this file and modify it with OpenOffice.org Writer, or Microsoft Word<sup>7</sup>.

## Module

The extensibility is very important. It should be easy to integrate new tools into the reporting framework. To solve this problem, each program is thought of as part of a complete own module. A module consists of the program itself, and the parser logic. The parser logic does nothing else then transforming the output of the tool into a XML format. As XML is very popular nowadays most tools provide direct XML output. In this case the output of the tool can be used directly but it showed up that a lot of unnecessary information is contained in the XML output of most tools. It therefore would be a good idea to process the data even it is XML and remove the information not needed afterwards. The parser then creates a new XML file and stores it on the hard disk. Optionally this file can be encrypted using a method implemented in the framework. Here the work of the module ends and the framework can start. The following code example shows the method for encrypting a file.

```
def encrypt(self, passwd, infile, outfile, blocksize=1024, enc=None):
    """
    Encrypt the data in the file handle infile to the file handle
    outfile in blocks of length blocksize

    enc is a EncryptCipher instace. If None, default to
    EncryptCipher(passwd, CIPHER_BLOWFISH, MODE_CBC)
    """
    print "Now encrypting!"
    if enc is None:
        enc = EncryptCipher(passwd, CIPHER_BLOWFISH, MODE_CBC)
    while 1:
        data = infile.read(blocksize)
        if len(data)==0: break
        enc.feed(data)
        outfile.write(enc.data)
    enc.finish()
    outfile.write(enc.data)
    infile.close()
```

---

<sup>7</sup>[www.microsoft.com](http://www.microsoft.com)

```
outfile.close()
```

Here the library 'yawPyCrypto' is used, which is an extension of the 'PyCrypto' library. The code is nearly self-explanatory, it takes a password, the input file and the output file as arguments and encrypts the file with the 'Blowfish' symmetric algorithm and uses the 'chain block cipher' mode.

At the moment the encryption online works with existing files, which means the files are created, written to the hard disk of the computer, encrypted and then deleted. This means that the files are unencrypted on the hard disk for less than a second. A better choice would be to encrypt the data directly in memory, but this is only planned for a later version of the software.

## Output

As soon as the modules have processed the data and have written their files and the framework has processed the data into one file, the next step is to transform the XML data into a format suitable for editing with a 'What you see is what you get' (WYSIWYG) editor. Therefore the output interface uses a library to output the data into the RTF format, but every other format is possible because XML is very flexible. It would also be possible to create a 'Portable Document Format' (PDF) file directly from the XML file. At the moment a method for RTF output is included because this is the most sense able way from the standpoint of the author because this can be edited directly with OpenOffice.org and then this output get be transformed into PDF within OpenOffice.org.

To see, that this can be done very easily an example is given. The following code sections will show how the output from a Nessus scan is imported, the necessary information is extracted and then processed.

```
file=open("nessus.xml")
nessusTree = etree.parse(file)
nessusElem = nessusTree.getroot()
file.close()
print "nessus end"
temp=nessusElem.find("./osname")
nessusOs = temp.text
temp=nessusElem.find("./osvers")
nessusVersion=temp.text
temp=nessusElem.find("./results")
nessusResults=temp
```

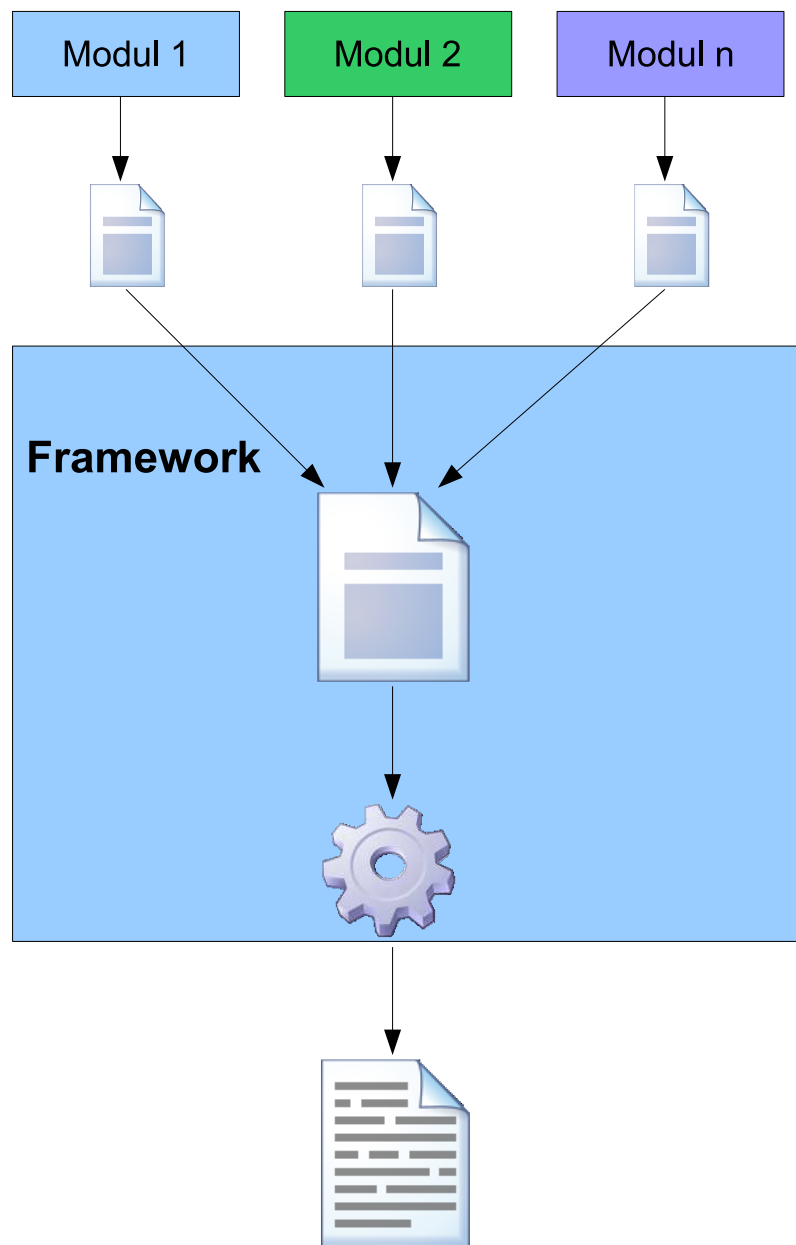


Figure 2.33: Creating a report.

The first four lines open the file, parse the document and save the location of the root element into the variable 'nessusElem'. The file then is closed and the tags with the names 'osname', 'osvers' and 'results' are searched and the values of these tags are saved into variables.

Then a new XML file is generated, and a new root element is created called 'report'. Next a child element of 'report' with the tag name 'dnsElem' is created and appended to the root element.

```

"""creating new root element"""
root=etree.Element("report")
root.append(dnsElem)

"""make tree structure of DOM"""
tree=etree.ElementTree(root)

"""write to file, name=report(year+month+day+hour+minutes).xml"""
filename="/home/dhuemer/Desktop/report"+getTimestamp()+".xml"
tree.write(filename)

```

The last step is to transform the information into an RTF document, which afterwards can be edited with a Text processing program. The following code shows a part of the code for creating this file.

```

self.doc      = Document()
    self.ss      = self.doc.StyleSheet
    self.section = Section()
    self.doc.Sections.append( self.section )

    self.p = Paragraph( self.ss.ParagraphStyles.Heading1 )
    self.p.append( 'Externer Penetrationstest' )
    self.section.append( self.p )

    self.p=Paragraph(self.ss.ParagraphStyles.Heading2)
    self.p.append("Informationsbeschaffung")
    self.section.append(self.p)

    self.p=Paragraph(self.ss.ParagraphStyles.Heading2)
    self.p.append("Subdomains")
    self.section.append(self.p)

    self.p = Paragraph( self.ss.ParagraphStyles.Normal )
    #self.p.append( etree.tostring(self.rootElem))
    self.p.append(self.dtTextDNS)
    self.section.append( self.p )

```

```

self.p = Paragraph( self.ss.ParagraphStyles.Normal )
self.p.append("The following domain was used: "+self.searchDomain)
self.p.append("The following string was used for searching E-mail addresses")
self.section.append( self.p )

self.p = Paragraph( self.ss.ParagraphStyles.Normal )
self.p.append("The following E-Mail addresses were found:")
self.section.append( self.p )

```

## GUI

As it must be easy to operate with the software a graphical user interface (GUI) is obligatory. This is the last interface to the framework. The whole GUI, as the rest of the program, is implemented using the programming language Python. Another good approach is that if the user plans to integrate a new module, the development can be done without touching the framework anywhere. This is because every module is a complete independent python class. This class is programed, interpreted and tested in its own file. Even the execution window in which it is tested is implemented in this class and as soon as the programmer wants to integrate the module into the framework, he just creates an object of that class in the class of the framework, without changing even one line of code. The framework then immediately is able to use this module. Two design principles enable this. First the creation of the test GUI in the module's class looks like the following:

```

if __name__ == '__main__':

    master = Tkinter.Tk()
    master.geometry( '1024x768' )
    d=Dns( master )
    master.mainloop()

,

if __name__ == '__main__':

```

This is the call for the test GUI of the 'DNS' module. The first line enables the flexibility of directly integrating this class into the framework because the code after this line only gets executed, if the class is called directly. If The class is integrated into the framework, this command validates to 'false' and

does not get executed. Technically this can be explained with the evaluation of this line which says if the calling process is the same like the current class then execute the code. This is not the case if the class is called by the framework.

The GUI of the framework uses the 'Python Megawidget' (PMW) library and just creates a window with Notebooks on top of the window. Every module gets its own Notebook with the name of the module. Depending which Notebook is pressed, the window beneath the Notebook pane shows the matching class. Here is where the software takes advantage of the good design because technically just an object of the defined class is created and displayed in this window. As an example see the following code:

```
page = notebook.add('DNS')
        notebook.tab('DNS').focus_set()
        curr= notebook.getcurselection()
        d=DNS.Dns(page)
```

This code creates a new tab called 'DNS', and then sets the focus to this tab. The variable 'curr' stores which tab is selected at the moment. Then with 'd=DNS.Dns(page)' a new object of the class 'DNS' is created and displayed in the tab we created.

### 2.10.2 Data structure (XML)

As mentioned before the application uses the 'Extensible Markup Language' (XML) as its data format. This has several reasons, but the most important one is the flexibility. XML can be read by humans and interpreted by computers, it can be transformed and queried very easily and even ontologies can be used to associate the data with meanings. Here XSLT is used to extract the information from the huge amount of data that is needed for the several stakeholders. As example take the following document:

The management will definitely not be interested in the technical details of this Nessus scan, such as port number or Bugtraq ID, but they would like to know what this flaw is all about. So the programmed tool can extract only the data section for the management report and the full technical details for the systems' administrators. This can be defined as wanted, its just a matter of which tags are extracted from the whole document. This is very easy, because all data is stored in XML.

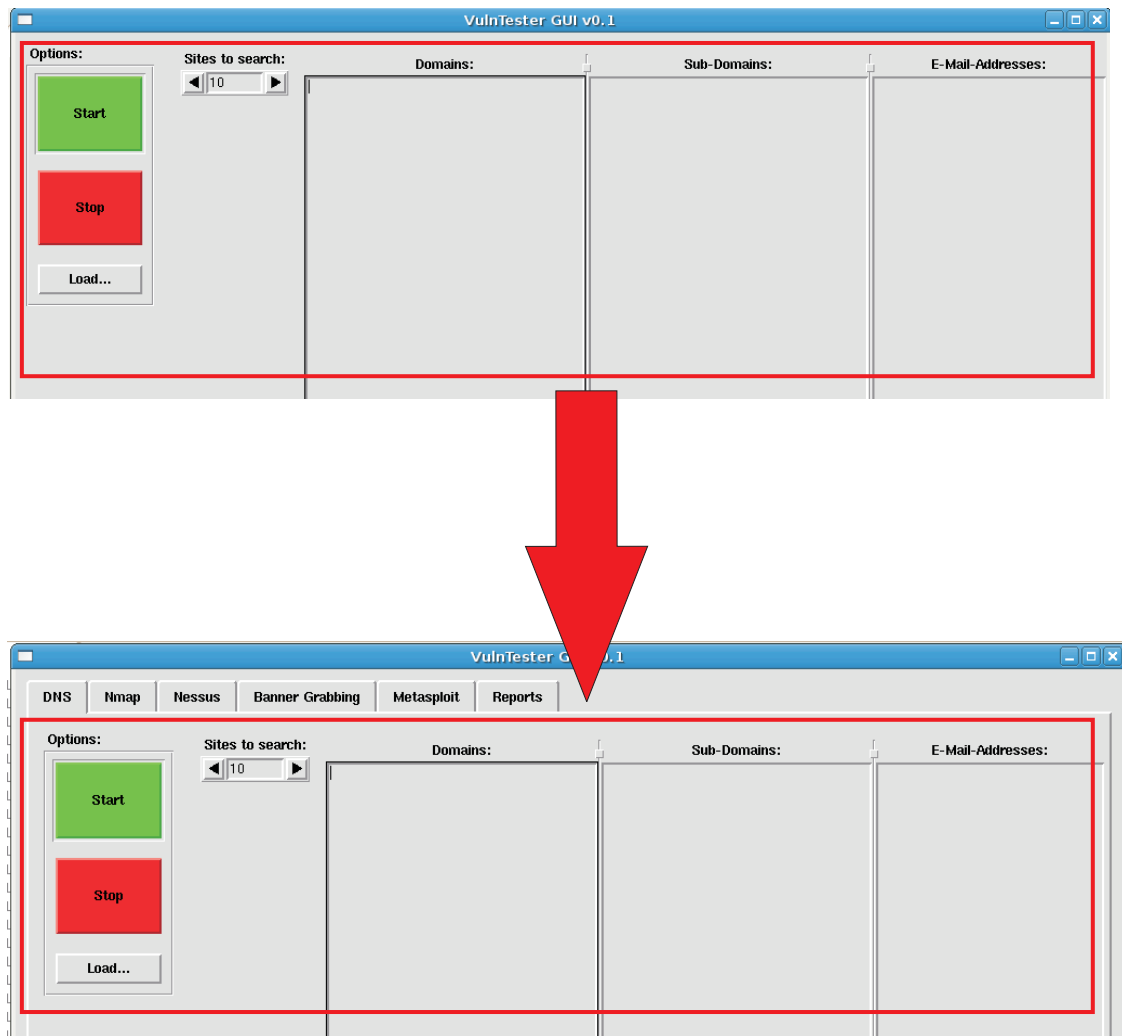


Figure 2.34: The tested module gets integrated into the GUI.

### 2.10.3 Stakeholders

In all companies different stakeholders with different responsibilities and different know how exist. The penetration tester normally gets hired by management to check their security level as this is a very serious task which only can be conducted through management. The problem is, that the penetration tester must afterwards report not only to the management, but the technicians must be informed too. Several other stakeholders exist, for whom the report should be targeted. The best a penetration tester could do is to identify the all the stakeholders. Often 3 kind of reports are created, one



```

<port protocol="tcp" portid="445">
  <service name="microsoft-ds" method="nessus" conf="3" />
  <information>
    <severity>Security Hole</severity>
    <id>11808</id>
    <data>

      Synopsis :

      Arbitrary code can be executed on the remote host.

      Description :

      The remote version of Windows contains a flaw in the function
      RemoteActivation() in its RPC interface which may allow an attacker to
      execute arbitrary code on the remote host with the SYSTEM privileges.

      A series of worms (Blaster) are known to exploit this vulnerability in the
      wild.

      Solution :

      http://www.microsoft.com/technet/security/bulletin/MS03-026.msp

      Risk factor :

      Critical / CVSS Base Score : 10
      (AV:R/AC:L/Au:NR/C:C/A:C/I:C/B:N)
      CVE : CVE-2003-0352
      BID : 8205
      Other references : IAVA:2003-A-0011

    </data>
  </information>

```

Figure 2.35: Part of a Nessus scan result.

for the management, one for the systems' administrators, and one for the databases' administrators but it is worth thinking about the stakeholders before creating the reports.

### 2.10.4 Reports

The reports are generated using the Python library 'PyRTF'. This library even provides support for formatting the document. For example the headings can be defined to be output in bold and using bigger letters. As mentioned in the chapter above, it is extremely important to know which command was executed at what time. If the programs are integrated in this

reporting framework, this is done automatically because it records all activities with a timestamp. Later on the user can exactly see when a special command was executed. This enables the tester and the customers to fully reconstruct the test, which may be important for re-testing. Normally after the vulnerabilities are found it is tried to eliminate them. Then the penetration test could be executed again, because all commands and parameters were automatically included into the report by the framework.

### 2.10.5 Extending the Software

If the reporting framework must be extended this can be done very fast and comfortable. The programmer first checks if the tool, that should be integrated, natively supports XML output. If it does support XML, he creates a new class which calls the external program and sets the calling parameters and the paths which tell the tool where to put its output. Then he adds the graphical elements in the class and tests the tool. If this works, he creates an instance of this class in the reporting framework. Then he just adds one line to tell the framework to include the new XML file of the tool into the one large XML file, which collects all information of all tools. The last step is to program some logic, that tells the program which tags of the XML file should be evaluated when a report for a certain stakeholder is created. Now the tool can be used.

If the tool does not support XML output by itself, the programmer has to add some parser logic, which transforms the native output into XML. This can be a little bit tricky, but should not be a huge problem. Unfortunately some tools support XML output but create non-consistent XML files. That means they use one tag for different kinds of information. This, for example, happened at the XML output of Nessus and has to be dealt with separately. The solution was to parse the XML output of the program before the file was integrated into the framework.

### 2.10.6 Programming Language

Before beginning to program the software, a programming language had to be chosen. As the software should be modular extensible some object orientation would be preferred and if the software. Another requirement was that the language can be learned easily and is supported by a community, as the most valuable help comes from other users, when problems occur. As the software was planned to work under BackTrack it would be great if the language was supported by the live distribution without installing it before.

At the end, I decided to use Python, as it is a very powerful and flexible

language, gets actively developed and has a large community. It even can be run under Linux and Windows platforms, so the tool can be used in both worlds. Unfortunately most tools require Linux, but if someone really wants the software to run under Windows, he could add some logic for calling the needed tools on remote Linux systems very easily. Then the software would only display the results and the GUI in Windows and do the work on some Linux system on the network. Again this can be done fast with python. As this is the first program I wrote using Python I can definitely say, that learning Python is easy and programs can be built very fast. All together this programming language was an excellent choice.

# Chapter 3

## Penetration Testing Tools

At the end I want to list some of the most important tools for a penetration test. Only a short overview of the tools will be given but the interested reader will find more information on various web sites. The list will be structured regarding the several phases of a penetration test.

### 3.1 Reconnaissance

#### 3.1.1 Online query tools

##### BGPlay

BGPlay is a Java application which displays animated graphs of the routing activity of a certain prefix within a specified time interval. Its graphical nature makes it much easier to understand how BGP updates affect the routing of a specific prefix than by analyzing the updates themselves.<sup>1</sup>

##### DNS online tools

The following list contains some online tools, which can be used to query for different DNS information.

#### 3.1.2 Samspade

Samspade includes tools such as ping, nslookup, whois, dig, traceroute, finger, raw HTTP web browser, DNS zone transfer, SMTP relay check, website search, and more. It can be downloaded freely at <http://samspade.org/ssw/>.

---

<sup>1</sup><http://bgplay.routeviews.org/bgplay/>

Tool	URL
Robtex	<a href="http://www.robtex.com/">http://www.robtex.com/</a>
Dig	<a href="http://us.mirror.menandmice.com/knowledgehub/tools/dig">http://us.mirror.menandmice.com/knowledgehub/tools/dig</a>
Dig gateway	<a href="http://www.spacereg.com/a.rpl?m=dig">http://www.spacereg.com/a.rpl?m=dig</a>
DNS digger	<a href="http://www.dnsdigger.com/">http://www.dnsdigger.com/</a>
Demon	<a href="http://www.demon.net/external/">http://www.demon.net/external/</a>
DNS Bajaj	<a href="http://www.zonecut.net/dns/">http://www.zonecut.net/dns/</a>
DNSq	<a href="http://stephan.sugarmotor.org/dnsq.html">http://stephan.sugarmotor.org/dnsq.html</a>
DNSwatch	<a href="http://www.dnswatch.info/">http://www.dnswatch.info/</a>
Netcraft	<a href="http://www.netcraft.com/">www.netcraft.com/</a>

Table 3.1: Online tools for DNS.

## 3.2 Scanning & Enumeration

### 3.2.1 Nmap

Nmap is the most used port scanning tool nowadays. Its scan engine not only can scan ports in various ways but has also built in some functionality for operating system fingerprinting. The homepage describes nmap as following:

Nmap ('Network Mapper') is a free open source utility for network exploration or security auditing. It was designed to rapidly scan large networks, although it works fine against single hosts. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. Nmap runs on most types of computers and both console and graphical versions are available. Nmap is free and open source.<sup>2</sup>

### 3.2.2 Paketto Keiretsu 1.10: Advanced TCP/IP Toolkit

The Paketto Keiretsu is a collection of tools that use new and unusual strategies for manipulating TCP/IP networks. They tap functionality within existing infrastructure and stretch protocols beyond what they were originally intended for. It includes Scanrand, an unusually fast network service and topology discovery system, Minewt, a user space NAT/MAT router, Linkcat, which

---

<sup>2</sup><http://insecure.org/nmap/>

presents a Ethernet link to stdio, Paratrace, which traces network paths without spawning new connections, and Phentropy, which uses OpenQVIS to render arbitrary amounts of entropy from data sources in three dimensional phase space.<sup>3</sup>

### 3.2.3 Amap

This tool is very useful, if open ports are found, but it is not known which application listen on it. It does this by sending trigger packets, and looking up the responses in a list of response strings. The project homepage can be found at <http://www.thc.org/thc-amap/>.

### 3.2.4 fping

Fping is a tool for ping sweeps. It can scan a lot of IP addresses in a very short time.

fping is a ping(1) like program which uses the Internet Control Message Protocol (ICMP) echo request to determine if a host is up. fping is different from ping in that you can specify any number of hosts on the command line, or specify a file containing the lists of hosts to ping. Instead of trying one host until it timeouts or replies, fping will send out a ping packet and move on to the next host in a round-robin fashion. If a host replies, it is noted and removed from the list of hosts to check. If a host does not respond within a certain time limit and/or retry limit it will be considered unreachable.<sup>4</sup>

Unlike ping, fping is meant to be used in scripts and its output is easy to parse.

### 3.2.5 strobe

strobe is a network/security tool that locates and describes all listening tcp ports on a (remote) host or on many hosts in a bandwidth utilisation maximising, and process resource minimizing manner. strobe approximates a parallel finite state machine internally. In non-linear multi-host mode it attempts to apportion bandwidth and sockets among the hosts very efficiently. This can

---

<sup>3</sup><http://www.doxpara.com/read.php/code/paketto.html>

<sup>4</sup><http://www.fping.com/>

reap appreciable gains in speed for multiple distinct hosts/routes. On a machine with a reasonable number of sockets, strobe is fast enough to port scan entire Internet sub domains. It is even possible to survey an entire small country in a reasonable time from a fast machine on the network backbone, provided the machine in question uses dynamic socket allocation or has had its static socket allocation increased very appreciably (check your kernel options). In this very limited application strobe is said to be faster than ISS2.1 (a high quality commercial security scanner by cklaus@iss.net and friends) or PingWare (also comercial).<sup>5</sup>

### 3.2.6 Superscan

SuperScan 4 is a completely-rewritten update of the highly popular Windows port scanning tool, SuperScan. Here are some of the new features in this version.

- Superior scanning speed
- Support for unlimited IP ranges
- Improved host detection using multiple ICMP methods
- TCP SYN scanning
- UDP scanning (two methods)
- IP address import supporting ranges and CIDR formats
- Simple HTML report generation
- Source port scanning
- Fast hostname resolving
- Extensive banner grabbing
- Massive built-in port list description database
- IP and port scan order randomization
- A selection of useful tools (ping, traceroute, Whois etc)
- Extensive Windows host enumeration capability<sup>6</sup>

---

<sup>5</sup><http://linux.maruhn.com/sec/strobe.html>

<sup>6</sup>[www.foundstone.com/resources/proddesc/superscan.htm](http://www.foundstone.com/resources/proddesc/superscan.htm)

### 3.2.7 Nessus

Nessus is the world's most popular vulnerability scanner used in over 75,000 organizations world-wide. Many of the world's largest organizations are realizing significant cost savings by using Nessus to audit business-critical enterprise devices and applications.<sup>7</sup>

### 3.2.8 Pwdump

Pwdump is a Windows 2000/XP/2003 NTLM and LanMan Password Grabber with which password hashes can be directly dumped through the network. The homepage<sup>8</sup> recommends switching to 'fgdump' which has even more functionality.

### 3.2.9 Dsniff

dsniff is a collection of tools for network auditing and penetration testing. dsniff, filesnarf, mailsnarf, msgsnarf, urlsnarf, and web-spy passively monitor a network for interesting data (passwords, e-mail, files, etc.). arpspoof, dnsspoof, and macof facilitate the interception of network traffic normally unavailable to an attacker (e.g, due to layer-2 switching). sshmitm and webmitm implement active monkey-in-the-middle attacks against redirected SSH and HTTPS sessions by exploiting weak bindings in ad-hoc PKI.<sup>9</sup>

## 3.3 Penetration

### 3.3.1 Cain & Able

Cain & Abel is a password recovery tool for Microsoft Operating Systems. It allows easy recovery of various kind of passwords by sniffing the network, cracking encrypted passwords using Dictionary, Brute-Force and Cryptanalysis attacks, recording VoIP conversations, decoding scrambled passwords, recovering wireless network keys, revealing password boxes, uncovering cached passwords and analyzing routing protocols. The program does not exploit any software vulnerabilities or bugs that

---

<sup>7</sup><http://www.nessus.org/about/>

<sup>8</sup><http://www.foofus.net/fizzgig/pwdump/>

<sup>9</sup><http://www.monkey.org/~dugsong/dsniff/>



could not be fixed with little effort. It covers some security aspects/weakness present in protocol's standards, authentication methods and caching mechanisms; its main purpose is the simplified recovery of passwords and credentials from various sources, however it also ships some 'non standard' utilities for Microsoft Windows users.<sup>10</sup>

### 3.3.2 John the Ripper

John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), Windows, DOS, BeOS, and OpenVMS. Its primary purpose is to detect weak Unix passwords. Besides several crypt(3) password hash types most commonly found on various Unix flavors, supported out of the box are Kerberos AFS and Windows NT/2000/XP/2003 LM hashes, plus several more with contributed patches.<sup>11</sup>

### 3.3.3 Metasploit

The Metasploit Framework is an advanced open-source platform for developing, testing, and using exploit code. This project initially started off as a portable network game and has evolved into a powerful tool for penetration testing, exploit development, and vulnerability research [15].

The Framework was written in the Perl scripting language and includes various components written in C, assembler, and Python. The widespread support for the Perl language allows the Framework to run on almost any Unix-like system under its default configuration. A customized Cygwin environment is provided for users of Windows-based operating systems. The project core is dual-licensed under the GPLv2 and Perl Artistic Licenses, allowing it to be used in both open-source and commercial projects.<sup>12</sup>

---

<sup>10</sup><http://www.oxid.it/cain.html>

<sup>11</sup><http://www.openwall.com/john/>

<sup>12</sup><http://www.metasploit.com/projects/Framework/>

## 3.4 Maintaining Access

### 3.4.1 DreamPackPL

With this tool it is possible to log into a Windows account without knowing the password. The homepage describes the software as following:

If You have forgot logon password, then this tool is for You. This is the only soft that allow you to log on into any local account without reset existing passwords. DreamPackPL will just turn off the password validation process. If you don't want to log on into any existing account, then you can execute any application (e.g. Regedit or window with accounts management) at the logon desktop. You may also load the Explorer shell at new desktop and work with admin privileges. Installation is not difficult. Just boot from prepared before CD, copy one file to system32 directory and reboot. In order to see main DreamPackPL window, you should enter command 'dreamon' at the logon desktop in 'User name' or 'Password' edit box.<sup>13</sup>

### 3.4.2 Netcat

Netcat is also called the 'swiss army knife' for network connections, because it is very flexible and powerful. With netcat an attacker could start a reverse shell on the computer he has hacked and maintains access to the system in this way.

### 3.4.3 FU Rootkit

This is a rootkit, that can hide processes and run processes with different user rights such as 'system' or similar. This is excellent, if a program was installed which should not show up in the process list.

### 3.4.4 Netbus

Netbus is a backdoor that can find cached passwords, provides full control over Windows, can capture videos from a video input device, starts a scheduler to run scripts on specified hosts at a certain time, and supports plugins.

---

<sup>13</sup><http://www.d-b.webpark.pl/dreampackpl.en.htm>

# Chapter 4

## Conclusio

As computers get more and more important in our lives and they are connected to each other they need to be protected against malicious users and other kind of attacks. This led to a complete new engineering discipline: the computer security. As part of the computer security penetration testing exists. Here computer experts think like hackers and try to break into computer networks and systems but strictly follow the code of ethical hacking, while doing so. This reveals vulnerabilities that would have not been found otherwise.

At the end of a penetration test, a report for different stakeholders, such as management or system administrator, must be created. This is a very time consuming activity, because each stakeholder gets a report with different granularity. To help a penetration tester in creating these reports, a software was programmed as part of this master's thesis. This software is kind of a reporting framework, which can be extended modularly. It extensively uses XML as its data structure and is therefore very flexible and information can be extracted and transformed as wanted. The reporting framework, at the end, generates a report in the RTF format, that can be used as a basis for the different reports.

Concluding I think computer security will see a real hype in the next years because a lot of assets are stored digitally. These valuable information must be protected against malicious activities and that's where computer security is needed.

# Chapter 5

## Abbreviations

TCP/IP	Transmission Control Protocol / Internet Protocol
UDP	Universal Datagram Protocol
SNMP	Simple Network Management Protocol
MSRPC	Microsoft Remote Procedure Call
MIB	Management Information Base
BGP	Border Gateway Protocol
LDAP	Lightweight Directory Access Protocol
AD	Active Directory
DC	Domain Controller
BGP	Border Gateway Protocol
ASN	Autonomous System Number
RPC	Remote Procedure Call
SQL	Structured Query Language
NFS	Network File System
SMB	Server Message Block
XML	Extensible Markup Language
WYSIWYG	What you see is what you get
PDF	Portable Document Format
PMW	Python Megawidget
ARP	Address Resolution Protocol
WEP	Wireless Equivalent Privacy
WLAN	Wireless LAN
LAN	Local Area Network
DMZ	Demilitarized Zone

# Chapter 6

## Glossary

### **System**

The term system is used to entitle a computer system.

### **Access control**

Mechanism limiting access to a particular object.

### **Assets**

Information, which is valuable for the owner.

### **Covert channel**

A communication channel not planned by the administrator.

### **Confidentiality**

A security principle, that works to ensure, information is not disclosed to unauthorized subjects.

### **ACL**

Access Control List. Manages which role has what kind of access to a certain object.

# Bibliography

- [1] Markus a Campo. *Penetration Testing*. Mitp-Verlag, 2006.
- [2] Markus a Campo. *Penetration Testing*. Mitp-Verlag, 2006.
- [3] Lee Barken, Eric Bermel, John Eder, Matt Fanady, Alan Koebrick, Michael Mee, and Marc Palumbo. *Wireless Hacking: Projects for Wi-Fi Enthusiasts*. Syngress Publishing, 2004.
- [4] Mark Burnett and Dave Kleiman. *Perfect Passwords: Selection, Protection, Authentication*. Syngress Publishing, 2006.
- [5] Daniel Bursch. *IT-Security im Unternehmen. Grundlagen, Strategien, Check-up*. Vdm Verlag Dr. Müller, 2005.
- [6] John Chirillo. *Hack Attacks Denied: A Complete Guide to Network Lockdown for UNIX, Windows, and Linux, Second Edition*. Wiley, 2002.
- [7] John Chirillo. *Hack Attacks Revealed: A Complete Reference for UNIX, Windows, and Linux with Custom Security Toolkit, Second Edition*. Wiley, 2002.
- [8] Eric Cole. *Hackers Beware: The Ultimate Guide to Network Security*. Sams, 2001.
- [9] douglas Chick. *Steel Bolt Hacking*. TheNetworkAdministrator.com, 2004.
- [10] EC-Council. *Ethical Hacking*. Osb Publisher Pte Ltd, 2003.
- [11] Renato Ettisberger. *IT-Security & Hacking*. Books on Demand GmbH, 2006.
- [12] Renato Ettisberger. *IT-Security & Hacking*. Books on Demand GmbH, 2006.

- [13] Ankit Fadia. *The Unofficial Guide to Ethical Hacking (Miscellaneous)*. Course Technology PTR, 2002.
- [14] James C. Foster. *Buffer Overflows*. Mitp-Verlag, 2005.
- [15] James C Foster. *Metasploit Toolkit for Penetration Testing, Exploit Development, and Vulnerability Research*. Syngress Publishing, 2007.
- [16] Walter Fumy and Jörg Sauerbrey. *Enterprise Security: IT Security Solutions: Concepts, Practical Experiences, Technologies*. Publicis Corporate Publishing, 2006.
- [17] Michael Gregg and Stephen Watkins. *Hack the Stack: Using Snort and Ethereal to Master the 8 Layers of an Insecure Network*. Syngress Publishing, 2006.
- [18] Brian Hatch and James Lee. *Hacking Linux Exposed, Second Edition*. McGraw-Hill Osborne Media, 2002.
- [19] Michael Horton and Clinton Mugge. *HackNotes(tm) Network Security Portable Reference*. McGraw-Hill Osborne Media, 2003.
- [20] Ji Hu, Dirk Cordel, and Christoph Meinel. *A virtual machine architecture for creating IT-security labs*. Universitätsverlag Potsdam, 2006.
- [21] Chris Hurley, Russ Rogers, Frank Thornton, Daniel Connelly, and Brian Baker. *Wardriving & Wireless Penetration Testing*. Syngress Publishing, 2006.
- [22] Sverre H. Huseby. *Sicherheitsrisiko Web-Anwendung*. Dpunkt Verlag, 2004.
- [23] Marty Jost. *IIS Security*. McGraw-Hill/OsborneMedia, 2002.
- [24] Kris Kaspersky. *Hacker Disassembling Uncovered (Uncovered series)*. A-List Publishing, 2007.
- [25] Tobias Klein. *Buffer Overflows und Format-String-Schwachstellen. Funktionsweisen, Exploits und Gegenmaßnahmen*. Dpunkt Verlag, 2003.
- [26] Marco Kleiner, Lucas Müller, and Mario Köhler. *IT-Sicherheit - Make or Buy*. Vieweg, 2005.
- [27] T. J. Klevinsky, Scott Laliberte, and Ajay Gupta. *Hack I.T.: Security Through Penetration Testing*. Addison-Wesley Professional, 2002.

- [28] Stefan Krämer. *Konfigurieren und Administrieren von IT-Security Systemen auf der Basis der Microsoft Technologie*. rps consulting GmbH, 2001.
- [29] Wilhelm Kruth. *IT-Grundlagenwissen für Datenschutz- und Security-Management. Kompaktwissen, Informationstechnik*. Datakontext, 2003.
- [30] Kevin Lam, David LeBlanc, and Ben Smith. *Assessing Network Security (Pro-One-Offs)*. Microsoft Press, 2004.
- [31] Cricket Liu and Paul Albitz. *DNS and BIND (5th Edition)*. O'Reilly Media, 2006.
- [32] Andrew Lockhart. *Network Security Hacks*. O'Reilly Media, 2004.
- [33] Johnny Long, Chris Hurley, James C Foster, Mike Petruzzi, Noam Rathaus, SensePost, Mark Wolfgang, and Max Moser. *Penetration Tester's Open Source Toolkit*. Syngress Publishing, 2005.
- [34] Johnny Long, Timothy Mullen, Ryan Russell, and Tim Mullen. *Stealing the Network: How to Own a Shadow (Stealing the Network)*. Syngress Publishing, 2007.
- [35] Johnny Long, Ed Skoudis, and Alrik van Eijkelenborg. *Google Hacking for Penetration Testers*. Syngress Publishing, 2004.
- [36] Stuart McClure, Saumil Shah, and Shreeraj Shah. *Web Hacking: Attacks and Defense*. Addison-Wesley Professional, 2002.
- [37] Thorsten Merz. *XML und IT-Security - Konzeption einer sicheren Web Services-Infrastruktur*. Diplomica, 2003.
- [38] Kevin D. Mitnick and William L. Simon. *The Art of Intrusion: The Real Stories Behind the Exploits of Hackers, Intruders & Deceivers*. John Wiley & Sons, 2005.
- [39] Kevin D. Mitnick and William L. Simon. *Die Kunst der Täuschung. Risikofaktor Mensch*. Mitp-Verlag, 2006.
- [40] Michael O'Dea. *HackNotes(tm) Windows Security Portable Reference*. McGraw-Hill Osborne Media, 2003.
- [41] Angela Orebaugh, Gilbert Ramirez, Josh Burke, Larry Pesce, and et al. *Wireshark & Ethereal Network Protocol Analyzer Toolkit (Jay Beale's Open Source Security)*. Syngress Publishing, 2006.



- [42] Institute Historian T. F. Peterson. *Nightwork: A History of Hacks and Pranks at MIT*. The MIT Press, 2003.
- [43] Charles P. Pfleeger. *Security in Computing, Second Edition*. Prentice Hall, 1996.
- [44] Donald L. Pipkin. *Halting the Hacker: A Practical Guide to Computer Security (With CD-ROM)*. Prentice Hall PTR, 2002.
- [45] David Pollino, Bill Pennington, Tony Bradley, and Himanshu Dwivedi. *Hacker's Challenge 3*. McGraw-Hill Osborne Media, 2006.
- [46] Enno Rey, Michael Thumann, Dominick Baier, and Stephen Fedtke. *Mehr IT-Sicherheit durch Pen-Tests*. Vieweg, 2005.
- [47] Enno Rey, Michael Thumann, Dominick Baier, and Stephen Fedtke. *Mehr IT-Sicherheit durch Pen-Tests*. Vieweg, 2005.
- [48] Russ Rogers and Matthew G Devost. *Hacking a Terror Network: The Silent Threat of Covert Channels*. Syngress Publishing, 2004.
- [49] Ryan Russell, Dan Kaminsky, Rain Forest Puppy, Joe Grand, K2, David Ahmad, Hal Flynn, Ido Dubrawsky, Steve W. Manzuik, and Ryan Perme. *Hack Proofing Your Network (Second Edition)*. Syngress Publishing, 2002.
- [50] Joel Scambray and Stuart McClure. *Windows Server 2003 (Hacking Exposed)*. McGraw-Hill Osborne Media, 2003.
- [51] Georg F. Schröder. *IT-Security Rechtssichere Umsetzung im Unternehmen*. Interest, 2005.
- [52] Michael T. Simpson. *Hands-On Ethical Hacking and Network Defense*. Course Technology, 2005.
- [53] Ed Skoudis. *Counter Hack: A Step-by-Step Guide to Computer Attacks and Effective Defenses*. Prentice Hall PTR, 2001.
- [54] Marion Steiner. *Analyse von Netzwerkangriffen. Einsatz von Simulationen zur Steigerung der Awareness*. Vdm Verlag Dr. Müller, 2006.
- [55] George Kurtz Stuart McClure, Joel Scambray. *Hacking Exposed*. McGraw-Hill, 2005.
- [56] Paul Taylor. *Hackers*. Routledge, 1999.

- [57] James S. Tiller. *The Ethical Hack: A Framework for Business Value Penetration Testing*. AUERBACH, 2004.
- [58] Dr. X and Dr. X. *The Complete Hacker's Handbook : Everything You Need to Know About Hacking in the Age of the Web*. Carlton Books, 2000.
- [59] Susan Young and Dave Aitel. *The Hacker's Handbook: The Strategy Behind Breaking into and Defending Networks*. AUERBACH, 2003.

# List of Figures

2.1	Difference between Red Teaming and Penetration Testing. . .	17
2.2	Black Box testing. . . . .	18
2.3	White Box testing. . . . .	18
2.4	Gray box testing. . . . .	19
2.5	Data encapsulation. . . . .	20
2.6	OSI versus TCP/IP. . . . .	21
2.7	Layer and program communicate using an API. . . . .	21
2.8	Dialog management of the Session layer. . . . .	22
2.9	Data gets transformed into a stream by the Transport layer. .	23
2.10	Data link layer. . . . .	25
2.11	TCP 3-way handshake. . . . .	26
2.12	TCP and UDP packet format. . . . .	27
2.13	Data structures of TCP and UDP. . . . .	28
2.14	Overview of Penetration Testing. . . . .	30
2.15	DNS zones. . . . .	31
2.16	Whois overview. . . . .	36
2.17	Scanning phases. . . . .	40
2.18	The connect Scan. . . . .	40
2.19	The SYN scan. . . . .	41
2.20	The FIN scan when the port is closed. . . . .	41
2.21	The FIN scan when the port is open. . . . .	42
2.22	Banner grabbing with Telnet. . . . .	45
2.23	Banner grabbing with Netcat. . . . .	45
2.24	LM Hash creation. . . . .	49
2.25	NTLM Hash creation. . . . .	49
2.26	Link manipulation. . . . .	52
2.27	ARP poisoning. . . . .	53
2.28	IP address spoofing. . . . .	55
2.29	Structure of a buffer. . . . .	55
2.30	Buffer gets overflowed. . . . .	55
2.31	Layers of an operating system. . . . .	59

---

2.32	Modular architecture of the Reporting Tool. . . . .	64
2.33	Creating a report. . . . .	67
2.34	The tested module gets integrated into the GUI. . . . .	71
2.35	Part of a Nessus scan result. . . . .	72

# List of Tables

2.1	TCP flags. . . . .	26
2.2	Differences between TCP and UDP. . . . .	27
2.3	DNS records . . . . .	33
2.4	Growing of possible passwords with increasing key space. [30] .	51
3.1	Online tools for DNS. . . . .	76