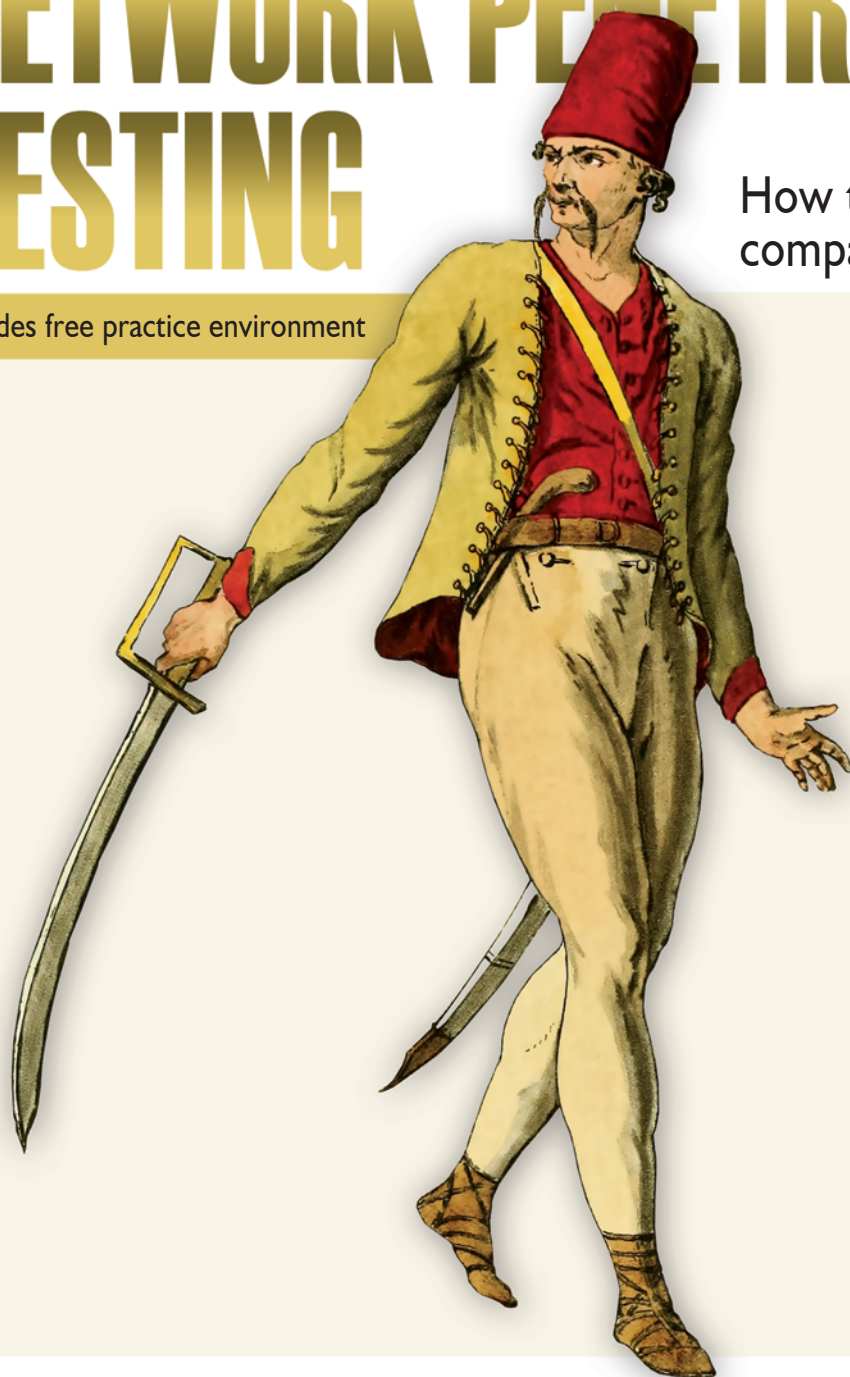


the art of

# NETWORK PENETRATION TESTING

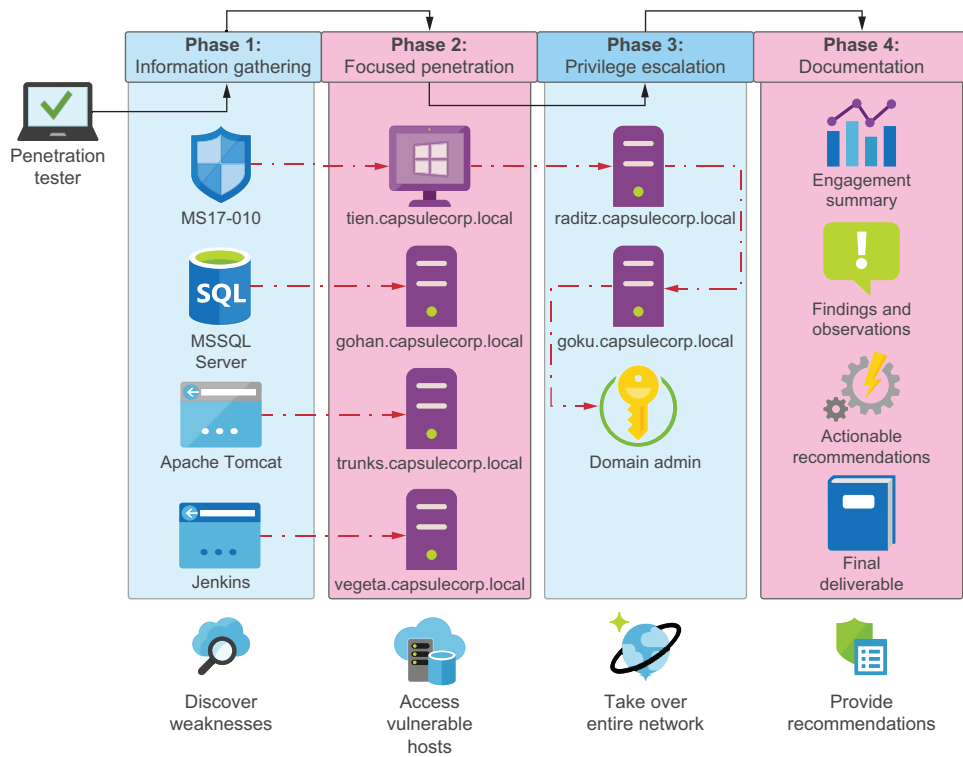
How to take over any company in the world

Includes free practice environment





Capsulecorp Inc. Internal Network Penetration Test  
LAN: 172.28.128.0/24  
Active Directory: capsulecorp.local



*The Art of Network  
Penetration Testing*

HOW TO TAKE OVER ANY COMPANY IN THE WORLD

ROYCE DAVIS



MANNING  
SHELTER ISLAND

For online information and ordering of this and other Manning books, please visit [www.manning.com](http://www.manning.com). The publisher offers discounts on this book when ordered in quantity. For more information, please contact


Special Sales Department  
Manning Publications Co.  
20 Baldwin Road  
PO Box 761  
Shelter Island, NY 11964  
Email: [orders@manning.com](mailto:orders@manning.com)

©2020 by Manning Publications Co. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by means electronic, mechanical, photocopying, or otherwise, without prior written permission of the publisher.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in the book, and Manning Publications was aware of a trademark claim, the designations have been printed in initial caps or all caps.

© Recognizing the importance of preserving what has been written, it is Manning's policy to have the books we publish printed on acid-free paper, and we exert our best efforts to that end. Recognizing also our responsibility to conserve the resources of our planet, Manning books are printed on paper that is at least 15 percent recycled and processed without the use of elemental chlorine.

 Manning Publications Co.  
20 Baldwin Road  
PO Box 761  
Shelter Island, NY 11964

Development editor: Toni Arritola  
Technical development editor: Karsten Strøbæk  
Review editor: Mihaela Batinic  
Production editor: Lori Weidert  
Copy editor: Tiffany Taylor  
Proofreader: Melody Dolab  
Technical proofreader: Giampeiro Granatella  
Typesetter: Gordan Salinovic  
Cover designer: Marija Tudor

ISBN 9781617296826  
Printed in the United States of America

## *about the author*

---

**ROYCE DAVIS** is a professional hacker specializing in network penetration testing and enterprise adversarial attack emulation. He has been helping clients secure their network environments for more than a decade and has presented research, techniques, and tools at security conferences all over the United States. He has contributed to open source security testing tools and frameworks and is the co-founder of [PentestGeek.com](https://pentestgeek.com), an ethical hacking training and education online resource.



# 1

## *Network penetration testing*

---

### ***This chapter covers***

- Corporate data breaches
- Adversarial attack simulations
- When organizations don't need a penetration test
- The four phases of an internal network penetration test

Everything today exists digitally within networked computer systems in the cloud. Your tax returns; pictures of your kids that you take with a cellphone; the locations, dates, and times of all the places you've navigated to using your GPS—they're all there, ripe for the picking by an attacker who is dedicated and skilled enough.

The average enterprise corporation has 10 times (at least) as many connected devices running on its network as it does employees who use those devices to conduct normal business operations. This probably doesn't seem alarming to you at first, considering how deeply integrated computer systems have become in our society, our existence, and our survival.

Assuming that you live on planet Earth—and I have it on good authority that you do—there’s a better than average chance you have the following:

- An email account (or four)
- A social media account (or seven)
- At least two dozen username/password combinations you’re required to manage and securely keep track of so that you can log in and out of the various websites, mobile apps, and cloud services that are essential in order for you to function productively every day.

Whether you’re paying bills, shopping for groceries, booking a hotel room, or doing just about anything online, you’re required to create a user account profile containing at the very least a username, a legal name, and an email address. Often, you’re asked to provide additional personal information, such as the following:

- Mailing address
- Phone number
- Mother’s maiden name
- Bank account and routing number
- Credit card details

We’ve all become jaded about this reality. We don’t even bother to read the legal notices that pop up, telling us precisely what companies plan to do with the information we’re giving them. We simply click “I Agree” and move on to the page we’re trying to reach—the one with the viral cat video or the order form to purchase an adorable coffee mug with a sarcastic joke on the side about how tired you feel all the time.

Nobody has time to read all that legal mumbo jumbo, especially when the free shipping offer expires in just 10 minutes. (Wait—what’s that? They’re offering a rewards program! I just have to create a new account really fast.) Perhaps even more alarming than the frequency with which we give random internet companies our private information is the fact that most of us naively assume that the corporations we’re interacting with are taking the proper precautions to house and keep track of our sensitive information securely and reliably. We couldn’t be more wrong.

## **1.1 Corporate data breaches**

If you haven’t been hiding under a rock, then I’m guessing you’ve heard a great deal about corporate data breaches. There were 943 disclosed breaches in the first half of 2018 alone, according to Breach Level Index, a report from Gemalto (<http://mng.bz/YxRz>).

From a media-coverage perspective, most breaches tend to go something like this: Global Conglomerate XYZ has just disclosed that an unknown number of confidential customer records have been stolen by an unknown group of malicious hackers who managed to penetrate the company’s restricted network perimeter using an unknown vulnerability or attack vector. The full extent of the breach, including everything the hackers made off with, is—you guessed it—unknown. Cue the tumbling stock price, a



flood of angry tweets, doomsday headlines in the newspapers, and a letter of resignation from the CEO as well as several advisory board members. The CEO assures us this has nothing to do with the breach; they've been planning to step down for months now. Of course, somebody has to take the official blame, which means the Chief Information Security Officer (CISO) who's given many years to the company doesn't get to resign; instead, they're fired and publicly stoned to death on social media, ensuring that—as movie directors used to say in Hollywood—they'll never work in this town again.

## **1.2 How hackers break in**

Why does this happen so often? Are companies just that bad at doing the right things when it comes to information security and protecting our data? Well, yes and no.

The inconvenient truth of the matter is that the proverbial deck happens to be stacked disproportionately in favor of cyber-attackers. Remember my earlier remark about the number of networked devices that enterprises have connected to their infrastructure at all times? This significantly increases a company's attack surface or *threat landscape*.

### **1.2.1 The defender role**

Allow me to elaborate. Suppose it's your job to defend an organization from cyber-threats. You need to identify every single laptop, desktop, smartphone, physical server, virtual server, router, switch, and Keurig or fancy coffee machine that's connected to your network.

Then you have to make sure every application running on those devices is properly restricted using strong passwords (preferably with two-factor authentication) and hardened to conform to the current standards and best practices for each respective device. Also, you need to make sure you apply every security patch and hotfix issued by the individual software vendors as soon as they become available. Before you can do any of that, though, you have to triple-check that the patches don't break any of your business's day-to-day operations, or people will get mad at you for trying to protect the company from hackers.

You need to do all of this all of the time for every single computer system with an IP address on your network. Sounds easy, right?

### **1.2.2 The attacker role**

Now for the flip side of the coin. Suppose your job is to break into the company—to compromise the network in some way and gain unauthorized access to restricted systems or information. You need to find only a single system that has slipped through the cracks; just one device that missed a patch or contains a default or easily guessable password; a single nonstandard deployment that was spun up in a hurry to meet an impossible business deadline driven by profit targets, so an insecure configuration setting (which shipped that way by default from the vendor) was left on. That's all it takes to get in, even if the target did an impeccable job of keeping track of every node on

the network. New systems are stood up daily by teams who need to get something done fast.

If you're thinking to yourself that this isn't fair, or that it's too hard for defenders and too easy for attackers, then you get the point: that's exactly how it is. So, what should organizations do to avoid being hacked? This is where penetration testing comes in.

### **1.3 Adversarial attack simulation: Penetration testing**

One of the most effective ways for a company to identify security weaknesses *before* they lead to a breach is to hire a professional adversary or *penetration tester* to simulate an attack on the company's infrastructure. The adversary should take every available action at their disposal to mimic a real attacker, in some cases acting almost entirely in secret, undetected by the organization's IT and internal security departments until it's time to issue their final report. Throughout this book, I'll refer to this type of offensive-security exercise simply as a *penetration test*.

The specific scope and execution of a penetration test can vary quite a bit depending on the motivations of the organization purchasing the assessment (the client) as well as the capabilities and service offerings of the consulting firm performing the test. Engagements can focus on web and mobile applications, network infrastructure, wireless implementations, physical offices, and anything else you can think of to attack. Emphasis can be placed on stealth while trying to remain undetected or on gathering vulnerability information about as many hosts as possible in a short time. Attackers can use human hacking (social engineering), custom-exploit code, or even dig through the client's dumpster looking for passwords to gain access. It all depends on the scope of the engagement. The most common type of engagement, however, is one that I have performed for hundreds of companies over the past decade. I call it an *internal network penetration test* (INPT). This type of engagement simulates the most dangerous type of *threat actor* for any organization: a malicious or otherwise compromised insider.

**DEFINITION** *Threat actor* is a fancy way of saying attacker. It refers to anyone attempting to harm an organization's information technology assets.

During an INPT, you assume that the attacker was able to successfully gain physical entry into a corporate office or perhaps was able to obtain remote access to an employee's workstation through email phishing. It is also possible that the attacker visited an office after hours, posing as a custodial worker, or during the day, posing as a vendor or flower delivery person. Maybe the attacker is an actual employee and used a badge to walk in the front door.

There are countless ways to gain physical entry to a business, which can be easily demonstrated. For many businesses, an attacker simply needs to walk through the main entrance and wander around while smiling politely at anyone who passes, appearing to have a purpose or talking on a cell phone until they identify an unused area where they can plug into a data port. Professional companies offering high-caliber penetration

testing (pentest) services typically bill anywhere from \$150 to \$500 per hour. As a result, it's often cheaper for the client purchasing the penetration test to skip this part and place the attacker on the internal subnet from the beginning.

Either way, the attacker has managed to get access to the internal network. Now, what can they do? What can they see? A typical engagement assumes that the attacker knows nothing about the internal network and has no special access or credentials. All they have is access to the network—and coincidentally, that's usually all they need.

### 1.3.1 Typical INPT workflow

A typical INPT consists of four phases executed in order, as depicted in figure 1.1. The individual names of each phase are not written in stone, nor should they be. One pentest company might use the term *reconnaissance* in place of *information gathering*. Another company might use the term *delivery* in place of *documentation*. Regardless of what each phase is called, most people in the industry agree on what the penetration tester should do during each phase.

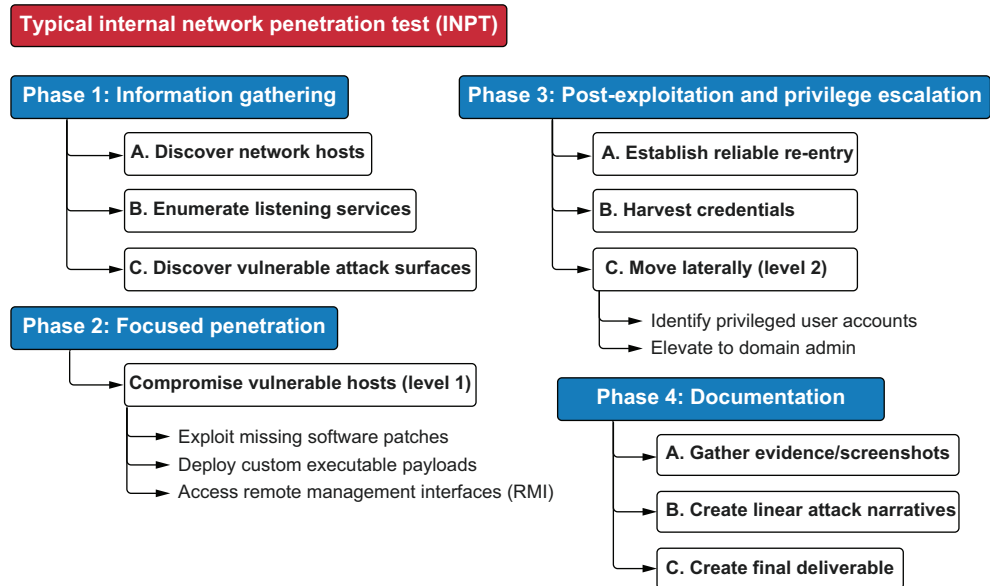


Figure 1.1 The four phases of a network penetration test

- *Phase 1*—Information gathering
  - a Map out the network.
  - b Identify possible targets.
  - c Enumerate weaknesses in the services running on those targets.
- *Phase 2*—Focused penetration
  - a Compromise vulnerable services (gain unauthorized access to them).

- *Phase 3*—Post-exploitation; privilege escalation
  - a Identify information on compromised systems that can be used to further access (*pivoting*).
  - b Elevate privileges to the highest level of access on the network, effectively becoming the company’s system administrator.
- *Phase 4*—Documentation
  - a Gather evidence.
  - b Create the final deliverable.

Once the testing portion of the engagement has concluded, the penetration tester now makes a mental shift from that of an adversary and transitions into a consultant. They spend the rest of the engagement creating as detailed a report as possible. That report contains the specific explanation of all the ways they were able to breach the network and bypass security controls as well as the detailed steps the company can take to close these identified gaps and ensure that they can no longer be exploited by anyone. In 9 out of 10 cases, this process takes about 40 hours on average, but the time required can vary depending on the size of the organization.

## **1.4** *When a penetration test is least effective*

You may have heard the familiar saying, “To a hammer, every problem looks like a nail.” Turns out you can apply this saying to just about any profession. A surgeon wants to cut, a pharmacist wants to prescribe a pill, and a penetration tester wants to hack into your network. But does every organization truly need a penetration test?

The answer is that it depends on the level of maturity within a company’s information security program. I can’t tell you how many times I’ve been able to take over a company’s internal network on the first day of a penetration test, but the number is in the hundreds. Of course, I would love to tell you that this is because of my *super leet hacker skillz* or that I’m just that good, but that would be a gross exaggeration of the truth.

It has a lot more to do with an exceedingly common scenario: an immature organization that isn’t even doing the basics is sold an advanced-level penetration test when it should be starting with a simple vulnerability assessment or a high-level threat model and analysis gig. There is no point in conducting a thorough penetration test of all your defense capabilities if there are gaping holes in your infrastructure security that even a novice can spot.

### **1.4.1** *Low-hanging fruit*

Attackers often seek out the path of least resistance and try to find easy ways into an environment before breaking out the big guns and reverse-engineering proprietary software or developing custom zero-day exploit code. Truth be told, your average penetration tester doesn’t know how to do something that complex, because it’s never been a skill they’ve needed to learn. No need to go that route when easy ways in are

widespread throughout most corporations. We call these easy ways in *low-hanging fruit* (LHF). Some examples include the following:

- Default passwords/configurations
- Shared credentials across multiple systems
- All users having local administrator rights
- Missing patches with publicly available exploits

There are many more, but these four are extremely common and extremely dangerous. On a positive note, though, most LHF attack vectors are the easiest to remediate. Make sure you're doing a good job with basic security concepts before hiring a professional hacker to attack your network infrastructure.

Organizations with significant numbers of LHF systems on their network shouldn't bother paying for a "go-all-out" penetration test. It would be a better use of their time and money to focus on basic security concepts like strong credentials everywhere, regular software patching, system hardening and deployment, and asset cataloging.

#### **1.4.2 When does a company really need a penetration test?**

If a company is wondering whether it should do a penetration test, I advise answering the following questions honestly. Start with simple yes/no answers. Then, for every yes answer, the company should see if it can back up that answer with, "Yes, *because* of internal process/procedure/application XYZ, which is maintained by employee ABC":

- 1 Is there an up-to-date record of every IP address and DNS name on the network?
- 2 Is there a routine patching program for all operating systems and third-party applications running on the network?
- 3 Do we use a commercial vulnerability scan engine/vendor to perform routine scans of the network?
- 4 Have we removed local administrator privileges on employee laptops?
- 5 Do we require and enforce strong passwords on all accounts on all systems?
- 6 Are we utilizing multi-factor authentication everywhere?

If your company can't answer a solid yes to all of these questions, then a decent penetration tester would probably have little to no trouble breaking in and finding your organization's crown jewels. I'm not saying you absolutely shouldn't buy a penetration test, just that you should expect painful results.

It may be fun for the penetration tester; they may even brag to their friends or colleagues about how easily they penetrated your network. But I am of the opinion that this provides very little value to your organization. It's analogous to a person never exercising or eating a healthy diet and then hiring a fitness coach to look at their body and say, "You're out of shape. That'll be \$10,000, please."

## 1.5 **Executing a network penetration test**

So, you've gone through all the questions and determined that your organization needs a network penetration test. Good! What's next? Up to now, I've discussed penetration testing as a service that you would typically pay a third-party consultant to conduct on your behalf. However, more and more organizations are building internal *red teams* to conduct these types of exercises on a routine basis.

**DEFINITION** *Red team*—A specialized subset of an organization's internal security department, focused entirely on offensive security and adversarial attack-simulation exercises. Additionally, the term *red team* is often used to describe a specific type of engagement that is considered as realistic as possible, simulating advanced attackers and using a goal-oriented, opportunistic approach rather than a scope-driven methodology

I'm going to make an assumption from here on that you've been or you're hoping to be placed in a role that would require you to perform a penetration test for the company you work for. Maybe you have even done a handful of penetration tests already but feel like you could benefit from some additional guidance and direction.

My intention in writing this book is to provide you with a "start-to-finish" methodology that you can use to conduct a thorough INPT, targeting your company or any other organization from which you receive written authorization to do so.

You'll learn the same methodology that I have matured over a decades-long career and used to successfully and safely execute hundreds of network penetration tests targeting many of the largest companies in the world. This process for executing controlled, simulated cyber-attacks that mimic real-world internal breach scenarios has proved successful in uncovering critical weaknesses in modern enterprise networks across all vertices. After reading this book and working through the companion exercises, you should have the confidence to execute an INPT, regardless of the size or industry of the business you're attacking. You will work through the four phases of my INPT methodology using the virtual Capsulecorp Pentest network that I have set up as a companion to this book. Each of the four phases is broken into several chapters demonstrating different tools, techniques, and attack vectors that penetration testers use frequently during real engagements.

### 1.5.1 **Phase 1: Information gathering**

Imagine the engineers who designed the entire corporate network sitting down with you and going over a massive diagram, explaining all the zones and subnets, where everything is, and why they did it that way. Your job during phase 1, the information-gathering phase of a penetration test, is to come as close as you can to that level of understanding without the network engineers' help (figure 1.2). The more information you gain, the better your chances of identifying a weakness.

Throughout the first few chapters of this book, I'll teach you how to gather all of the information about the target network that is necessary for you to break in. You'll

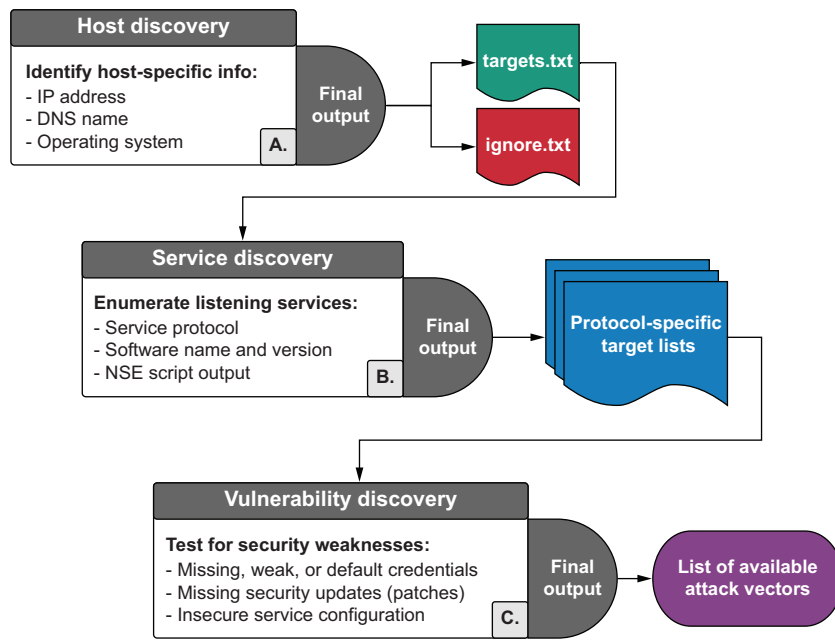


Figure 1.2 The information-gathering phase

learn how to perform network mapping using Nmap and discover live hosts within a given IP address range. You'll also discover listening services that are running on network ports bound to those hosts. Then you'll learn to interrogate these individual services for specific information, including but not limited to the following:

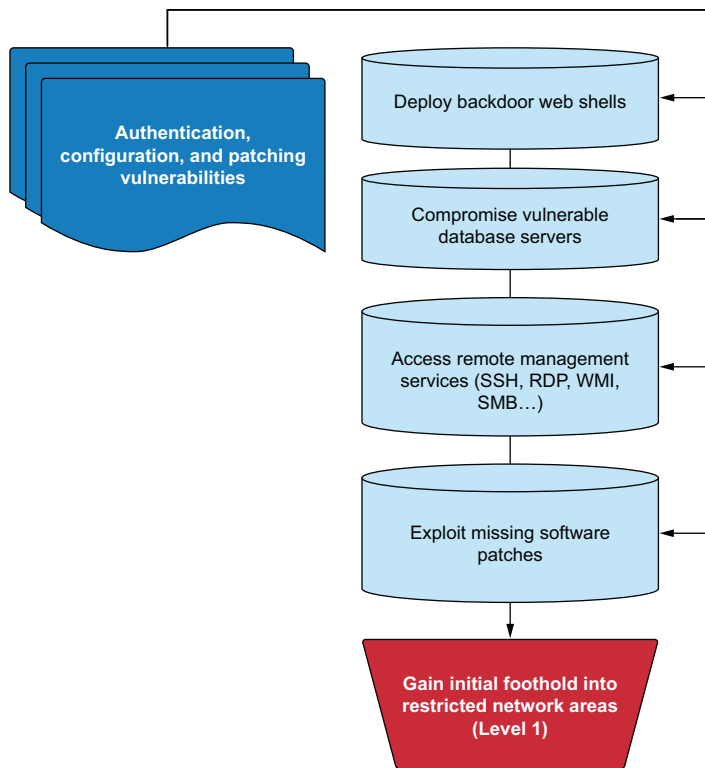
- Software name and version number
- Current patch and configuration settings
- Service banners and HTTP headers
- Authentication mechanisms

In addition to using Nmap, you'll also learn how to use other powerful open source pentest tools such as the Metasploit framework CrackMapExec (CME), Impacket, and many others to further enumerate information about network targets, services, and vulnerabilities that you can take advantage of to gain unauthorized access to restricted areas of the target network.

### 1.5.2 Phase 2: Focused penetration

Let the fun begin! The second phase of an INPT is where all the seeds planted during the previous phase begin to bear fruit (figure 1.3). Now that you have identified vulnerable attack vectors throughout the environment, it's time to compromise those hosts and start to take control of the network from the inside.

During this section of the book, you'll learn several types of attack vectors that will result in some form of remote code execution (RCE) on vulnerable targets. RCE means



**Figure 1.3** The focused penetration phase

you can connect to a remote command prompt and type commands to your compromised victim that will be executed and will send output back to you at your prompt.

I'll also teach you how to deploy custom web shells using vulnerable web applications. By the time you're finished with this phase of the book, you'll have successfully compromised and taken control over database servers, web servers, file shares, workstations, and servers residing on Windows and Linux operating systems.

### 1.5.3 Phase 3: Post-exploitation and privilege escalation

One of my favorite security blogs is written and maintained by a respected penetration tester named Carlos Perez (@Carlos\_Perez). The heading at the top of his page (<https://www.darkoperator.com>) absolutely fits for this section of the book: "Shell is only the beginning."

After you've learned how to compromise several vulnerable hosts within your target environment, it's time to take things to the next level (figure 1.4). I like to refer to these initial hosts that are accessible via a direct access vulnerability as *level-1 hosts*. This phase of the engagement is all about getting to level-2.

*Level-2 hosts* are targets that were not initially accessible during the focused penetration phase because you couldn't identify any direct weaknesses within their listening services. But after you gained access to level-1 targets, you found information or



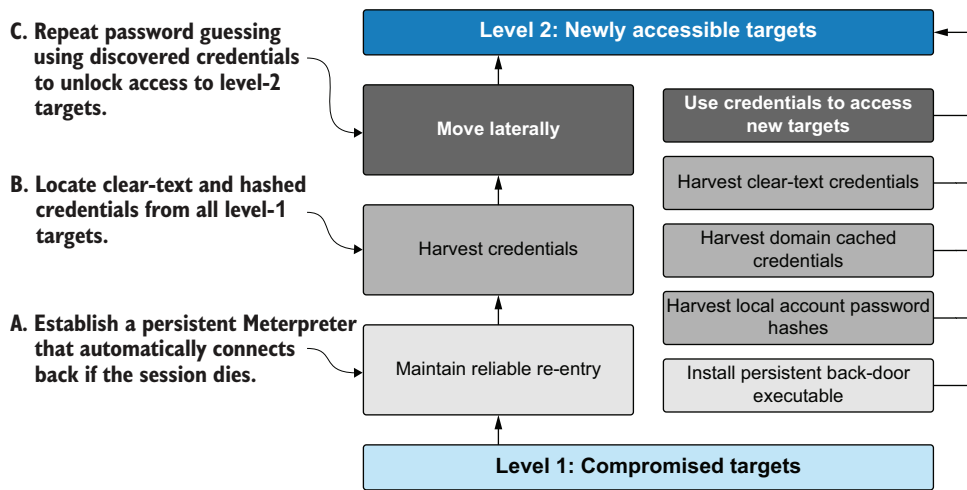


Figure 1.4 The privilege escalation phase

vectors previously unavailable to you, which allowed you to compromise a newly accessible level-2 system. This is referred to as pivoting.

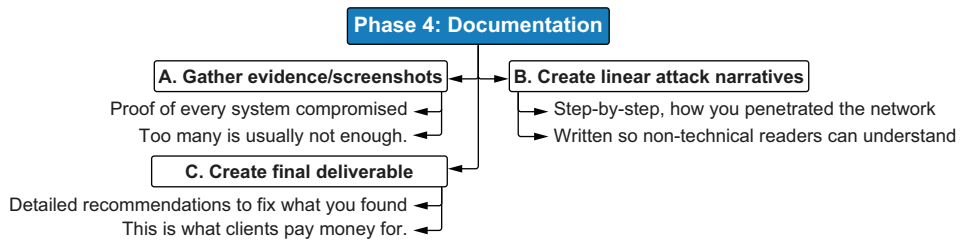
In this section, you'll learn post-exploitation techniques for both Windows- and Linux-based operating systems. These techniques include harvesting clear-text and hashed account credentials to pivot to adjacent targets. You'll practice elevating non-administrative users to admin-level privileges on compromised hosts. I'll also teach you some useful tricks I've picked up over the years for searching passwords inside hidden files and folders, which are notorious for storing sensitive information. Additionally, you'll learn several different methods of obtaining a domain admin account (a superuser on a Windows Active Directory network).

By the time you've finished with this section of the book, you'll understand exactly why we say in this industry that it takes only a single compromised host for you to spread through a network like wildfire and eventually capture the keys to the kingdom.

#### 1.5.4 Phase 4: Documentation

I realized early in my career that hiring a professional consulting firm to execute a network penetration test is kind of like buying a \$20,000 PDF document. Without the report, the penetration test means nothing. You broke into the network, found a bunch of holes in their security, and elevated your initial access as high as it could go. How does that benefit the target organization? Truth be told, it doesn't, unless you can provide detailed documentation illustrating exactly how you were able to do it and what the organization should do to ensure that you (or someone else) can't do it again (figure 1.5).

I've written hundreds of pentest deliverables, and I've had to learn—sometimes the hard way—what clients want to see in a report. I've also come to the realization



**Figure 1.5** The documentation phase

that they're the ones paying thousands of dollars to read the report, so it's probably a good idea to make sure they're impressed.

In addition to showing you exactly what to put in an engagement deliverable, I'll also share some efficiency habits I've learned over the years that have saved thousands of production hours of my time—time I was then able to spend doing things I enjoy, like breaking into corporate networks (rather than staring at a Word document editor).

### **What makes this book different from other penetration testing books?**

Looking at this book's table of contents, you may be wondering why topics you've seen covered in other penetration testing books are missing: social engineering, evading antivirus software, wireless hacking, mobile and web application testing, lock picking—I could go on, but you get the point. In reality, all of these topics deserve their own books, and covering them in a single chapter doesn't do justice to the breadth of information that's available on each one.

The purpose of this book is to arm you with the tools necessary to conduct a typical internal network penetration test (INTP). This engagement is sold by every pentesting firm out there and is the most common type of engagement you will perform, should you end up in a career as a professional penetration tester.

During typical INTPs (where you will spend at least 80% of your time), you will not be asked (or even allowed) to touch your client's wireless infrastructure or send email phishing messages to the company's employees or try to tailgate into its physical datacenters. You won't have the time or resources to properly build custom payloads designed to bypass the organization's specific EDR solution.

Rather than gloss over subjects that are interesting and definitely have value in other engagements, this book chooses to focus solely on the topic at hand.

## **1.6 Setting up your lab environment**

The topic of network penetration testing is one that should be learned by doing. I have written this book in a format that assumes you, the reader, have access to an enterprise network and authorization to perform basic penetration testing activities against it. I understand that some of you may not have such access. Therefore I have created an open source project called the Capsulecorp Pentest, which will serve as a

lab environment that you can use to work through the entire INPT process you will learn throughout the remaining chapters.

### 1.6.1 The Capsulecorp Pentest project

The Capsulecorp Pentest environment is a virtual network set up using VirtualBox, Vagrant, and Ansible. In addition to the vulnerable enterprise systems, it also comes with a preconfigured Ubuntu Linux system for you to use as your attacking machine. You should download the repository from the book's website (<https://www.manning.com/books/the-art-of-network-penetration-testing>) or GitHub (<https://github.com/r3dy/capsulecorp-pentest>) and follow the setup documentation before moving forward to the next chapter.

## 1.7 Building your own virtual pentest platform

Some of you may prefer to roll your own setup from the ground up. I completely understand this mentality. If you want to create your own pentest system, I urge you to consider a couple of things before choosing an operating system platform to start with.

### 1.7.1 Begin with Linux

Like most professional penetration testers, I prefer to use the Linux operating system to conduct the technical portions of an engagement. This is primarily due to a chicken and egg kind of phenomenon, which I will try to explain.

Most penetration testers use Linux. When an individual develops a tool to make their job easier, they share it with the world, usually via GitHub. It's likely the tool was developed on Linux and coincidentally works best when run from a Linux system. At the very least, it requires fewer headaches and dependency battles to get it working on Linux. Therefore, more and more people are basing and conducting their penetration testing from a Linux platform so they can use the latest and best available tools. So, you see, you could make the argument that *Linux is the most popular choice among penetration testers because it is the most popular choice among penetration testers*—and thus my chicken-and-egg comparison.

There is a good reason why this occurs, though. Until the introduction of Microsoft's PowerShell scripting language, Linux/UNIX-based operating systems were the only ones that shipped with native support for programming and scripting automated workflows. You didn't have to download and install a big, bulky IDE if you wanted to write a program. All you had to do was open a blank file in Vim or Vi (the most powerful text editors on the planet), write some code, and then run it from your terminal. If you're wondering what the connection is between penetration testing and writing code, it's simple: laziness. Just like developers, pentesters can be lazy, and consequently loath doing repetitive tasks; thus we write code to automate whatever we can.

There are other somewhat political reasons for using Linux, which I won't cover in detail because I'm not a political person. I will say, though, that most pentesters fancy

themselves as hackers. Hackers—at least traditionally—tend to prefer open source software, which can be freely obtained and customized, as opposed to closed source commercial applications developed by corporations trying to make a buck. Who knows what those big, bad companies have hidden in their products? Information should be free, fight the man, hack the planet . . . you get the point.

**TIP** Linux is the operating system preferred by most penetration testers. Some of these pentesters have written really powerful tools that work best on a Linux platform. If you want to do pentesting, you should use Linux, too.

### 1.7.2 *The Ubuntu project*

This is where my personal preference begins to enter the monologue: I am most comfortable pentesting from Ubuntu Linux, which is a derivative of the much older Debian Linux. My reason is not an elitist opinion battle between *mine* and *theirs*. Ubuntu is simply the best-performing platform of the dozen or so distributions I've experimented with over the years. I won't discourage you from choosing a different distribution, especially if you are already comfortable with something else. But I encourage you to choose a project that is extremely well-documented and supported by a vast community of educated users. Ubuntu certainly meets and exceeds these criteria.

Choosing a Linux distribution is a lot like choosing a programming language. You'll find no shortage of die-hard supporters with their feet buried deep in the sand, screaming at the top of their lungs all the reasons why their camp is superior to the others. But these debates are pointless because the best programming language is usually the one you know the best and can therefore be the most productive with. That is also true with Linux distributions.

#### **What is a Linux distribution?**

Unlike commercial operating systems such as Microsoft Windows, Linux is open source and freely customizable to your heart's content. As a direct result, hundreds of different versions of Linux have been created by individuals or groups or even companies that have their own perspective on how Linux should look and feel. These versions are called *distributions*, *distros*, or sometimes *flavors*, depending on who you're chatting with.

The core of the Linux operating system is called the *kernel*, which most versions leave untouched. The rest of the operating system, though, is totally up for grabs: the window manager, package manager, shell environment, you name it.

### 1.7.3 *Why not use a pentest distribution?*

You may have heard about Kali Linux, Black Arch, or some other custom Linux distribution marketed for pentesting and ethical hacking. Wouldn't it be easier to just download one of those instead of building a platform from scratch? Well, yes and no.

Although the grab-and-go factor is undoubtedly appealing, what you'll find when you work in this field long enough is that these preconfigured pentest platforms tend to be a little bloated with unnecessary tools that never get used. It's kind of like starting a new DIY home project. A big hardware store like Home Depot has absolutely everything you could ever need, but the individual project you are working on, no matter how complex it is, requires only a dozen or so tools. I want to go on record stating that I respect and admire the hard work that's put in by the various developers and maintainers of these distros.

At some point, though, you'll inevitably Google "How to do XYZ in Linux" while on an active engagement and find a really great article or tutorial with just four simple commands that work on Ubuntu but not Kali, even though Kali is based on Ubuntu! Sure, you can go digging into the problem, which, of course, has a simple solution once you find out what it is; but I've had to do this so many times that I simply run Ubuntu and install what I need—and only what I need and that works best for me. That's my philosophy, right or wrong.

Last, I'll say this. I place a great deal of importance on building out your own environment—not just for your competency and skill progression, but also so that you can have the confidence to look your client in the eye and tell them everything that's running on your system if they ask you. Clients are often scared of penetration testing because they don't have much experience with it, so it's not uncommon for them to be cautious when allowing a third party to plug an unmanaged device into their network. I've been asked many times to provide a write-up of every tool I use and links to the documentation.

**NOTE** Maybe you're thinking "I still want to use Kali." That's completely fine. Most of the tools covered in this book are natively available within Kali Linux. Depending on your skill level, it may be easier to go that route. Keep in mind that all of the exercises and demonstrations in the book are done using the custom-built Ubuntu machine covered in appendix A. I expect that you can follow along with this book using Kali Linux if that is your preference.

All that being said, if you prefer to create your own system from scratch, you can take a look at appendix A, where I have outlined a complete setup and configuration. Otherwise, if you simply want to get started learning how to conduct an INPT, you can download and set up the Capsulecorp Pentest environment from the GitHub link in section 1.6.1. Either way, make your choice, set up your lab environment, and then get started conducting your first penetration test in chapter 2.

## Summary

- The world as we know it is operated by networked computer systems.
- It is increasingly difficult for companies to manage the security of their computer systems.

- Attackers need to find only a single hole in a network to blow the doors wide open.
- Adversarial attack simulation exercises, or penetration tests, are an active approach to identifying security weaknesses in an organization before hackers can find and exploit them.
- The most common type of attack simulation is an internal network penetration test, which simulates threats from a malicious or compromised insider.
- A typical INPT can be executed within a 40-hour work week and consists of four phases:
  - 1 Information gathering
  - 2 Focused penetration
  - 3 Post-exploitation and privilege escalation
  - 4 Documentation