2 How Hackers Hack

The most enjoyable career activity I do is penetration testing (also known as pen testing). Pen testing is hacking in its truest sense. It's a human against a machine in a battle of wits. The human "attacker" can use their own ingenuity and new or existing tools as they probe for weaknesses, whether they be machine- or human-based. In all my years of pen testing, even though I am usually given weeks to conduct a test, I have successfully hacked my target the majority of the time in around one hour. The longest it has ever taken me is three hours. That includes every bank, government site, hospital, and corporate site that has ever hired me to do so.

I'm not even all that good as a pen tester. On a scale 1 to 10, with 10 being the best, I'm about a 6 or a 7. On the defender side, I feel like I'm the best person in the world. But as an attacker, I'm very average. I've been surrounded by awesome pen testers—men and women who think nothing of writing their own testing tools or who don't consider their testing a success unless they did not generate a single event in a log file that could have caused an alert. But even the people I consider to be 10s usually think of themselves as average and admire other pen testers that they think are tens. How good must those hackers be?

But you don't have to be extremely good to be a very successful hacker. You don't even have to actually break in for the customer that hired you (I'm assuming you're being paid for a lawful assignment to pen test) to be happy with your work. In fact, the customer would absolutely be thrilled if you were not successful. They could brag that they hired some hackers and their network withstood the attack. It's a win-win for everyone involved. You get paid the same and they get to brag that they are impenetrable. It's the only job I know where you cannot have a bad outcome. Unfortunately, I know of no pen tester who has *ever* not successfully broken into *all* of their targets. I'm sure there must be hackers who fail, but the vast majority of pen testers "capture their prize."

NOTE If your pen testing doesn't find any weaknesses and soon afterward your client is compromised by real attackers, you aren't going to look good. If this happens several times, word will get around and you'll probably be looking for a new career. The weaknesses are there. Find them.

Usually pen testers will do something extra to impress their target's senior managers, such as taking a clandestine picture of the CEO at his desk using his own computer's camera or embedding the domain administrator's password in the picture of a pirate flag that shows up on the security administrator's screensaver. A picture is worth a thousand words. Never underestimate how much one goofy picture can increase your customer's satisfaction with your job. They'll be talking about the picture (and bragging about you) years after you've finished the job. If you can, always finish with a flourish. I'm giving you "consultant gold" with this recommendation.

The Secret to Hacking

If there is a secret to how hackers hack, it's that there is no secret to how they hack. It's a process of learning the right methods and using the right tools for the job, just like an electrician, plumber, or builder does. There isn't even one way to do it. There is, however, a definitive set of steps that describe the larger, encompassing process, and that includes all the steps that a hacker could possibly have to perform. Not all hackers use all the steps. Some hackers only use one step. But in general, if you follow all the steps, you're likely to be very successful at hacking. You can skip one or more of the steps and still be a successful hacker. Malware and other hacking tools often allow hackers to skip steps, but at least one of the steps, initial penetration foothold, is always required.

Regardless of whether you're going to make a career out of being a (legal) hacker, if you're going to fight malicious hackers, you have to understand the "hacking methodology" or whatever it is being called by the person or document describing it. The models can vary, including the number of steps involved, the names of the steps, and the specific details of each step, but they all contain the same basic components.

The Hacking Methodology

The hacking methodology contains the following progressive steps:

1. Information Gathering
2. Penetration
3. Optional: Guaranteeing Future Easier Access
4. Internal Reconnaissance
5. Optional: Movement
6. Intended Action Execution
7. Optional: Covering Tracks

Information Gathering

Unless a hacker tool is helping the hacker to randomly access any possible vulnerable site, the hacker usually has a destination target in mind. If a hacker wants to penetrate a specific company, the first thing the hacker does is start researching everything they can about the company that might possibly help them break in. At the very least, this means accessible IP addresses, email addresses, and domain names. The hacker finds out how many potential sites and services they can access that are connected to the company. They use the news media and public financial reports to find out who the senior executives are or to find other employee names for social engineering. The hacker looks up news stories to see what big software the target has bought recently, what mergers or divestitures are happening (these are always messy affairs often accompanied by relaxed or missed security), and even what partners they interact with. Many companies have been compromised through a much weaker partner.

Finding out what digital assets a company is connected to is the most important part of information gathering in most hacker attacks. Not only are the main (public) sites and services usually identified, but it's usually more helpful to the attacker to find the less popular connected sites and services, like employee and partner portals. The less popular sites and servers are more likely to have a weakness compared to the main sites that everyone has already beat on for years.

Then any good hacker starts to gather all the software and services hosted on each of those sites, a process generally known as *fingerprinting*. It's very

important to learn what operating systems (OS) are used and what versions. OS versions can tell a hacker what patch levels and which bugs may or may not be present. For example, they might find Windows Server 2012 R2 and Linux Centos 7.3-1611. Then they look for software programs and versions of those software versions (for the same reason) running on each OS. If it's a web server, they might find Internet Information Server 8.5 on the Windows server and Apache 2.4.25 on the Linux server. They do an inventory of each device, OS, application, and version running on each of their intended targets. It's always best to do a complete inventory to get an inclusive picture of the target's landscape, but other times a hacker may find a big vulnerability early on and just jump into the next step. Outside of such a quick exploit, usually the more information the hacker has about what is running, the better. Each additional software and version provides additional possible attack vectors.

NOTE Some hackers call the general, non-technical, information gathering *footprinting* and the OS and software mapping *fingerprinting*.

Sometimes when a hacker connects to the service or site it helpfully responds with very detailed version information so you don't need any tools. When that isn't the case, there are plenty of tools to help with OS and application fingerprinting. By far the number one used hacker fingerprinting tool is Nmap (`https://nmap.org/`). Nmap has been around since 1997. It comes in several versions including Windows and Linux and is a hacker's Swiss Army knife tool. It can perform all sorts of host scanning and testing, and it is a very good OS fingerprinter and an okay application fingerprinter. There are better application fingerprinters, especially when they are focused on a particular type of application fingerprinting, such as web servers, databases, or email servers. For example, Nikto2 (`https://cirt.net/Nikto2`) not only fingerprints web servers better than Nmap, but also performs thousands of penetration tests and lets you know which vulnerabilities are present.

Penetration

This is the step that puts the "hack" in "hacker"—gaining initial foothold access. The success of this step makes or breaks the entire cycle. If the hacker has done their homework in the fingerprinting stage, then this stage really isn't all that hard. In fact, I've never not accomplished this stage. There is always old software being used, always something left unpatched, and almost always something misconfigured in the collection of identified software.

NOTE One of my favorite tricks is attacking the very software and devices that the defenders use to defend their networks. Often these devices are *appliances*, which is simply another word for running a computer with harder-to-update software. Appliances are notorious for being years out of patch compliance.

If by chance all the software and devices are perfectly secured (and they never are), then you can attack the human element, which is always the weakest part of the equation. But without the initial penetrating foothold, all is lost for the hacker. Fortunately for the hacker, there are lots of ways to penetrate a target. Here are the different techniques a hacker can use to break into a target:

- Zero-days
- Unpatched software
- Malware
- Social engineering
- Password issues
- Eavesdropping/MitM
- Data leaks
- Misconfiguration
- Denial of service
- Insider/partner/consultant/vendor/third party
- User error
- Physical access
- Privilege escalation

Zero-days Zero-day (or 0-day) exploits are rarer than every-day vulnerabilities, which vendors have usually long ago patched. A zero-day exploit is one for which the targeted software is not yet patched against and the public (and usually the vendor) isn't aware of. Any computer system using software with a zero-day bug is essentially exploitable at-will, unless the potential victim uninstalls the software or has put in place some sort of other mitigation (for example a firewall, an ACL list, VLAN segmentation, anti-buffer overflow software, and so on).

Zero-days are not as common as known exploits because they can't be widely used by an attacker. If an attacker overused a zero-day, the coveted exploit hole would be discovered and patched by vendors and placed in anti-malware

signatures. These days most vendors can patch new exploits within a few hours to a few days after discovery. When zero-days are used, they are either used very broadly against many targets all at once for maximum exploitation possibility or used "low and slow," which means sparingly, rarely, and only used when needed. The world's best professional hackers usually have collections of zero-days that they use only when all else has failed and even then in such a way that they won't be especially noticed. A zero-day might be used to gain an initial foothold in an especially resistant target, and then all traces of it will be removed and more traditional methods used from that point onward.

Unpatched Software Unpatched software is always among the top reasons why a computer or device is exploited. Each year there are thousands (usually between 5000 and 6000, or 15 per day) of new publicly announced vulnerabilities among all popularly used software. (Check out the stats reported in each issue of Microsoft's *Security Intelligence Report*, `http://microsoft.com/sir`.) Vendors have generally gotten better at writing more secure code and finding their own bugs, but there are an ever-increasing number of programs and billions of lines of code, so the overall number of bugs has stayed relatively stable over the last two decades.

Most vendors do a fairly good job of patching their software in a timely manner, especially after a vulnerability becomes publicly known. Unfortunately, customers are notoriously slow in applying those patches, even often going so far as disabling the vendor's own auto-patching routines. Some moderate percentage of users never patch their system. The user either ignores the multiple patch warnings and sees them as purely annoying or is completely unaware that a patch needs to be applied. (For example, many point-of-sale systems don't notify cashiers that a patch needs to be applied.) Most software exploits happen to software that has not been patched in many, many years.

Even if a particular company or user patches critical vulnerabilities as quickly as they are announced, a persistent, patient hacker can just wait for a patch to be announced that is on their target's fingerprint inventory list and launch the related attack before the defender has time to patch it. (It's relatively easy for a hacker to reverse engineer patches and find out how to exploit a particular vulnerability.)

Both zero-days and regular software vulnerabilities come down to insecure software coding practices. Software vulnerabilities will be covered in Chapter 6.

Malware Malicious programs are known as malware, and the traditional types are known as viruses, Trojan horse programs, and worms, but today's

malware is often a hybrid mixture of multiple types. Malware allows a hacker to use an exploit method to more easily attack victims or to reach a greater number of victims more quickly. When a new exploit method is discovered, defenders know that malware writers will use automated malware to spread the exploit faster in a process known as "weaponization." While any exploit is something to be avoided, it is often the weaponization of the exploit that creates the most risk to end-users and society. Without malware, an attacker is forced to implement an attack one victim at a time. With malware, millions of victims can be exploited in minutes. Malware will be covered in more detail in Chapter 9.

Social Engineering One of the most successful hacking strategies is social engineering. Social engineering, whether accomplished manually by a human adversary or done using automation, is any hacker trick that relies upon tricking an end-user into doing something detrimental to their own computer or security. It can be an email that tricks an end-user into clicking on a malicious web link or running a rogue file attachment. It can be something or someone tricking a user into revealing their private logon information (called *phishing*). Social engineering has long been in the quiver of attacks used by hackers. Long-time whitehat hacker, Kevin Mitnick, used to be one of best examples of malicious social engineers. Mitnick is profiled in Chapter 5, and social engineering is covered in more detail in Chapter 4.

Password Issues Passwords or their internally stored derivations can be guessed or stolen. For a long time, simple password guessing (or social engineering) was one of the most popular methods of gaining initial access to a computer system or network, and it still is. But credential theft and re-use (such as pass-the-hash attacks) has essentially taken over the field of password hacking in a big way over the past half decade. With credential theft attacks, an attacker usually gains administrative access to a computer or device and retrieves one or more logon credentials stored on the system (either in memory or on the hard drive). The stolen credentials are then used to access other systems that accept the same logon credentials. Almost every major corporate attack has involved credential theft attacks as a common exploit component, so much so that traditional password guessing isn't as popular anymore. Password hacks are covered in Chapter 21.

Eavesdropping/MitM Eavesdropping and "man-in-the-middle" (MitM) attacks compromise a legitimate network connection to gain access to or

maliciously participate in the communications. Most eavesdropping occurs due to flaws in network or application protocols, but it can also be accomplished due to human error. These days the biggest eavesdropping attacks occur on wireless networks. Network attacks will be covered in Chapter 33, and wireless attacks will be covered in Chapter 23.

Data Leaks Leaks of private information can be an outcome from one of the other forms of hacking or can result from an unintentional (or intentional) human action. Most data leaks occur because of inadvertent (and underprotected) placement or because some hacker figured out a way to access otherwise private data. But insider attacks where an employee or contractor intentionally steals or uses private information are also a common form of hacking. Several of the chapters in this book apply to preventing data leakages.

Misconfiguration It is also common for computer users and administrators to (sometimes inadvertently) implement very weak security choices. I can't tell you how many times I've gone to a public web site to find that its most critical files are somehow marked with Everyone or World permissions—and those permissions are exactly what they look like. And when you tell the entire world that they can access any file they like, your site or the files stored on it are not going to stay private for very long. Secure operating systems and configurations are covered in Chapter 30.

Denial of Service Even if no one made a single error or had a single piece of unpatched software, it's still possible to take nearly any web site or computer off the Internet. Even if you are perfect, your computers rely on one or more services, not under your control, that are not perfect. Today, huge distributed denial of service (DDoS) attacks can take down or significantly impact nearly any web site or computer connected to the Internet. These attacks often contain billions of malicious packets per second, which overwhelms the targeted site (or its upstream or downstream neighbors). There are dozens of commercial (sometimes illegal) services that anyone can use to both cause and defend against huge DDoS attacks. DDoS attacks are covered in Chapter 28.

Insider/Partner/Consultant/Vendor/Third Party Even if your network and all its computers are perfect (which they aren't), you can be compromised by a flaw in a connected partner's computer or by insider employees. This category is fairly broad and crosses a range of other hacker methods.

User Error This penetration category also crosses a range of other hacker methods. For example, a user can accidentally send private data to an

unauthorized user by putting a single mistyped character in an email address. The user can accidentally miss patching a critical server or can accidentally set the wrong permission. A frequent user error is when someone replies to an email thinking they are replying privately to one person or a smaller list of people but they accidentally are actually replying to the larger list or even to a person they are talking disparagingly about. I point out user error separately here only because sometimes mistakes happen and hackers are ready to take advantage of them.

Physical Access Conventional wisdom says that if an attacker has physical access to an asset, they can just steal the whole thing (poof, your cell phone is gone) and destroy it or eventually bypass all protections to access private data. And this perception has proven pretty accurate so far, even against defenses that are explicitly meant to protect against physical attacks. For example, many disk encryption programs can be defeated by the attacker using an electron microscope to identify the protected secret key by identifying the individual electrons that compose the key. Or RAM can be frozen by canned air to reveal the secret encryption key in plaintext because of a fault in the way memory physically stores data.

Privilege Escalation Each hacker uses one of the various penetration methods described in the previous sections to initially exploit a target system. The only question after gaining access is what type of security access they get. If they exploit a software program or service running in the user's own security context, they initially only have the same access privileges and permissions as the logged on user. Or they may get the Holy Grail on that system and get complete administrative system access. If the attacker only gets regular, non-privileged access permissions, then they generally execute a second, privilege escalation attack to try and obtain higher privileged access. Privilege escalation attacks run the gamut, essentially duplicating the same approaches as for penetration, but they begin with the higher starting point of already having at least some access. Privilege escalation attacks are generally easier to perform than the initial exploits. And since the initial exploits are almost always guaranteed to succeed, the privilege escalation is just that much easier.

Guaranteeing Future Easier Access

Although it's optional, once an attacker has obtained the initial foothold access, most hackers then work on implementing an additional method to ensure that they can more easily access the same asset or software faster the next time

around. For many hackers, this means placing a "listening" backdoor program that they can directly connect to next time. Other times it means cracking passwords or creating new accounts. The attacker can always use the same exploits that worked successfully last time to gain the initial foothold, but usually they want some other method that will work even if the victim fixes the issue that worked the previous time.

Internal Reconnaissance

Once most hackers have penetrated the system, they start executing multiple commands or programs to learn more about the target they have gained access to and what things are connected to it. Usually that means looking in memory, on the hard drive, for network connectivity, and enumerating users, shares, services, and programs. All this information is used to better understand the target and also as a launching point for the next attack.

Movement

It is the rare attacker or malware program that is content to break into one target. Nearly all hackers and malware programs want to spread their range of influence over more and more targets. Once they gain access to the initial target, spreading that influence within the same network or entity is pretty easy. The hacker penetration methods listed in this chapter summarize the various ways they can do it, but comparing it to the initial foothold efforts, the subsequent movement is easier. If the attacker moves to other similar targets with similar uses, it is called lateral movement. If the attacker moves from devices of one privilege to a higher or lower privilege, it's called vertical movement.

Most attackers move from lower to high levels of privilege using vertical movement techniques (again, using the hacker penetration methods described in this chapter). For example, a very common hacker methodology is for the attacker to first compromise a single, regular end-user workstation. They use that initial foothold to search for and download local administrative account passwords. Then, if those local administrative credentials are shared among more machines (which they often are), they then move horizontally and repeat the process until they can capture very privileged account access. Sometimes this is done immediately during the first break-in because the logged on user or system already has very high privileges. They then move to the authentication server and capture every user's logon credentials. This is the standard modus

operandi for most hacker groups these days, and moving from the initial compromise to complete network ownership (or pwning in hacker terminology) can be less than an hour.

In my personal experience, and remember I'm just an average hacker, it usually takes me about one hour to gain the initial foothold and another hour to capture the centralized authentication database. So for me, an average hacker, it takes about two hours to completely own a company. The longest it has taken me is three hours.

Intended Action Execution

After access is guaranteed and asset ownership is taken, hackers then accomplish what they intended to do (unless the action of breaking in revealed something new to do). Every hacker has intent. A legitimate penetration tester has a contractual obligation to do one or more things. A malicious hacker might spread malware, read or steal confidential information, make a malicious modification, or cause damage. The whole reason for the hacker to compromise one or more systems is to do something. In the old days (two or three decades ago), simply showing off that they had hacked a system would have been enough for most hackers. Today, hacking is 99% criminally motivated, and the hacker is going to do something malicious to the target (even if the only damage they do is to remain silently infiltrated for some potential, future action). Unauthorized access without any direct damage is still damage.

Covering Tracks

Some hackers will try to cover their tracks. This used to be what almost all hackers did years ago, but these days computer systems are so complex and in such great numbers that most asset owners don't check for hacker tracks. They don't check the logs, they don't check the firewall, and they don't look for any signs of illegal hacking unless it hits them in the face. Each year, Verizon's *Data Breach Investigations Report* (`http://www.verizonenterprise.com/verizon-insights-lab/dbir/`) reports that most attackers go unnoticed for months to years, and over 80% of the attacks would have been noticed had the defenders bothered to look. Because of these statistics, most hackers don't bother to cover their tracks anymore.

Hackers need to cover their tracks even less these days because they are using methods that will never be detected using traditional hacker-event detection. Or what the hacker uses is so common in the victim's environment that it

is nearly impossible to distinguish between legitimate and illegitimate activity. For example, after breaking in, a hacker usually performs actions in the security context of a legitimate user, often accessing the same servers and services as the legitimate user does. And they use the same tools (such as remote access software and scripting languages) that the admins do. Who can tell what is and isn't malicious? The field of intrusion detection is covered in Chapter 14.

Hacking Is Boringly Successful

If you want to know how hackers hack, there you go. It's all summarized throughout this chapter. The only thing left to do is add tools, curiosity, and persistence. The hacking cycle works so well that many penetration testers, after getting over the initial excitement of being paid to be a professional hacker, get bored and move on to something else after a few years. Could there be a bigger testament to how well the cycle works? And it is within this framework and understanding that defenders need to fight against attackers.

Automated Malware as a Hacking Tool

When malware is involved, the malware can accomplish one or more of the steps, automating everything, or hand over manual control once the target is acquired and pwned. Most hacking groups use a combination of social engineering, automated malware, and human attackers to accomplish their objectives. In larger groups, the individual hackers may have assigned roles and specialties. Malware may execute a single penetration step and be successful without ever trying any of the other steps. For example, the fastest malware program in history, SQL Slammer, was just 376 bytes big. It executed its buffer-overflowing payload against SQL UDP port 1434 regardless of whether the target was running SQL. Since most computers aren't running SQL, you might think it would be very inefficient. Nope, in 10 minutes it changed the world. No malware program has ever come close to infecting as many hosts in as short of a time.

NOTE If I've missed a step in the hacker methodology or missed a penetration method, I apologize. Then again, I told you I was only an average hacker.

Hacking Ethically

I would like to think that my readers are ethical hackers who make sure they have the legal right to conduct hacking on any target they have fixed their sights on. Hacking a site you do not have the predefined and expressed authority to hack is unethical and often illegal. It is even unethical (if not also illegal) to hack a site and let them know of a found vulnerability for no money. It is unethical and often illegal to find a vulnerability and then ask the site to hire you as a pen tester. This latter scenario happens all the time. I'm sorry, there is no way to tell someone that you have found a way to hack their sites or servers and ask for a job or money without it being seen as extortion. I can tell you that almost all sites receiving such an unsolicited request do not think you're being helpful and do not want to hire you. They see you as the enemy, and lawyers are always immediately called.

The rest of this book is dedicated to describing specific types of hacking, particular penetration methods, how defenders fight those methods, and experts in their field at fighting hackers at their own game. If you want to hack for a living or fight hackers, you'll need to understand the hacker methodology. The people profiled in this book are the giants in their field, and you can learn a lot from them. They led the way. A great place to start is with Bruce Schneier, who is profiled in Chapter 3 and is considered by many to be the father of modern computer cryptography.