CHAPTER 2
Understanding Adversaries

By Heather Adkins and David Huska
with Jen Barnason

In August 1986, Clifford Stoll, a systems administrator at Lawrence Livermore Laboratory, stumbled upon a seemingly benign accounting error that led to a 10-month search for someone stealing government secrets from the United States.[1] Largely considered to be the first public example of its kind, Stoll spearheaded an investigation that laid bare the specific tactics, techniques, and procedures (TTPs) the adversary used to achieve their goals. Through careful study, the investigation team was able to construct a picture of how the attacker targeted and siphoned data out of protected systems. Many system designers have incorporated lessons that arose from Stoll's seminal article describing the team's efforts, "Stalking the Wily Hacker."

In March 2012, Google responded to an unusual power outage at one of its Belgian datacenters that ultimately led to local data corruption. Investigation revealed that a cat had damaged a nearby external power supply, triggering a series of cascading failures in the building's power systems. By studying how complex systems fail in similar ways, Google has been able to adopt resilient design practices when suspending, burying, and submerging cables around the world.

Understanding a system's adversaries is critical to building resilience and survivability for a wide variety of catastrophes. In the reliability context, adversaries usually operate with benign intent and take abstract form. They might exist as routine hardware failures or cases of overwhelming user interest (so-called "success disas-

[1] Stoll documented the attack in an article in *Communications of the ACM*, "Stalking the Wily Hacker", and the book *The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage* (Gallery Books). Both are good resources for anyone designing secure and reliable systems, as their findings are still relevant today.

ters”). They could also be configuration changes that cause systems to behave in unexpected ways, or fishing vessels that accidentally sever undersea fiber-optic cables. By contrast, adversaries in the security context are human; their actions are calculated to affect the target system in an undesirable way. Despite these contrasting intents and methods, studying reliability and security adversaries is important for understanding how to design and implement resilient systems. Without this knowledge, anticipating the actions of a Wily Hacker or a Curious Cat would be quite challenging.

In this chapter, we deep dive on security adversaries to help specialists in diverse fields develop an adversarial mindset. It may be tempting to think of security adversaries through the lens of popular stereotypes: attackers in dark basements with clever nicknames and potentially shady behaviors. While such colorful characters certainly exist, anyone with time, knowledge, or money can undermine the security of a system. For a small fee, anyone can purchase software that enables them to take over a computer or mobile phone to which they have physical access. Governments routinely buy or build software to compromise the systems of their targets. Researchers often probe the safety mechanisms of systems to understand how they work. Therefore, we encourage you to maintain an objective perspective about who is attacking a system.

No two attacks—or attackers—are the same. We recommend taking a look at Chapter 21 for a discussion of the cultural aspects of dealing with adversaries. Predicting future security catastrophes is mostly a guessing game, even for knowledgeable security experts. In the following sections, we present three frameworks to understand attackers that we’ve found helpful over the years, exploring the potential motives of people attacking systems, some common attacker profiles, and how to think about attackers’ methods. We also provide illustrative (and hopefully entertaining) examples within the three frameworks.

Attacker Motivations

Security adversaries are first and foremost human (at least for the time being). Therefore, we can consider the purpose of attacks through the eyes of the people who carry them out. Doing so may better equip us to understand how we should respond, both proactively (during system design) and reactively (during incidents). Consider the following attack motivations:

Fun
: To undermine the security of a system for the sheer joy of knowing it can be done.

Fame
: To gain notoriety for showing off technical skills.

Activism
: To make a point or broadcast a message—typically, a political viewpoint—widely.

Financial gain
: To make money.

Coercion
: To get a victim to knowingly do something they don't want to do.

Manipulation
: To create an intended outcome or change behavior—for example, by publishing false data (misinformation).

Espionage
: To gain information that might be valuable (spying, including industrial espionage). These attacks are often performed by intelligence agencies.

Destruction
: To sabotage a system, destroy its data, or just take it offline.

An attacker might be a financially motivated vulnerability researcher, government espionage agent, and criminal actor all at the same time! For example, in June 2018 the US Department of Justice indicted Park Jin Hyok, a North Korean citizen accused of participating in a wide variety of activities on behalf of his government, including creating the infamous 2017 WannaCry Ransomware (used for financial gain), the 2014 compromise of Sony Pictures (intended to coerce Sony into not releasing a controversial movie, and ultimately harming the company's infrastructure), and the compromise of electric utilities (presumably for espionage or destructive purposes). Researchers have also observed government attackers using the same malware leveraged in nation-state attacks to pilfer electronic money in video games for personal gain.

When designing systems, it's important to keep these diverse motivations in mind. Consider an organization that is processing money transfers on behalf of its customers. If we understand why an attacker might be interested in this system, we can design the system more securely. A good example of possible motivations in this case can be seen in the activities of a group of North Korean government attackers (including Park) who allegedly attempted to steal millions of dollars by breaking into banking systems and exploiting the SWIFT transaction system to transfer money out of customer accounts.

Attacker Profiles

We can better understand attacker motivations by taking the people themselves into account: who they are, whether they perform attacks for themselves or for someone else, and their general interests. In this section, we outline some *profiles* of attackers, indicating how they relate to a system designer and including a few tips for protecting your systems from these types of attackers. For the sake of brevity, we've taken some liberties with generalizations, but remember: no two attacks or attackers are the same. This information is meant to be illustrative rather than definitive.

Early Hacking

MIT is considered to be the birthplace of the term *hacking*, which dates back to the 1950s, when this activity spawned from innocent pranks. These benign origins have led some to differentiate "hacking" and "attacking" into separate notions of nonmalicious and malicious behavior. We continue this tradition throughout this book. The MIT hacker community today operates by a loose set of ethics documented in the MIT Mind and Hand Book.

Hobbyists

The first computer hackers were *hobbyists*—curious technologists who wanted to understand how systems worked. In the process of taking computers apart or debugging their programs, these "hackers" discovered flaws that the original system designers hadn't noticed. Generally speaking, hobbyists are motivated by their thirst for knowledge; they hack for fun, and can be allies to developers looking to build resilience into a system. More often than not, hobbyists abide by personal ethics about not harming systems and don't cross boundaries into criminal behavior. By leveraging insight into how these hackers think about problems, you can make your systems more secure.

Vulnerability Researchers

Vulnerability researchers use their security expertise professionally. They enjoy finding security flaws as full-time employees, part-time freelancers, or even accidentally as average users who stumble across bugs. Many researchers participate in Vulnerability Reward Programs, also known as *bug bounties* (see Chapter 20).

Vulnerability researchers are typically motivated to make systems better, and can be important allies to organizations seeking to secure their systems. They tend to operate within a set of predictable disclosure norms that set expectations between system owners and researchers about how vulnerabilities are discovered, reported, fixed, and discussed. Researchers operating under these norms avoid inappropriately accessing data, causing harm, or breaking the law. Typically, operating outside these norms invalidates the possibility of getting a reward and may qualify as criminal behavior.

Relatedly, *Red Teams* and penetration testers attack targets with the permission of the system owner, and may be hired explicitly for these exercises. Like researchers, they look for ways to defeat system security with a focus on improving security and operate within a set of ethical guidelines. For more discussion on Red Teams, see Chapter 20.

Governments and Law Enforcement

Government organizations (for example, law enforcement agencies and intelligence agencies) may hire security experts to gather intelligence, police domestic crime, commit economic espionage, or complement military operations. By now, most national governments have invested in fostering security expertise for these purposes. In some cases, governments may turn to talented students fresh out of school, reformed attackers who have spent time in jail, or notable luminaries in the security industry. While we can't cover these types of attackers extensively here, we provide a few examples of their most common activities.

Intelligence gathering

Intelligence gathering is probably the most publicly discussed government activity that employs people who know how to break into systems. In the past few decades, traditional spying techniques, including signals intelligence (SIGINT) and human intelligence (HUMINT), have modernized with the advent of the internet. In one famous example from 2011, the security company RSA was compromised by an adversary many experts associate with China's intelligence apparatus. The attackers compromised RSA to steal cryptographic seeds for their popular two-factor authentication tokens. Once they had these seeds, the attackers didn't need physical tokens to generate one-time authentication credentials to log in to the systems of Lockheed Martin, a defense contractor that builds technology for the US military. Once upon a time, breaking into a company like Lockheed would have been performed by human operatives onsite—for example, by bribing an employee or having a spy hired at the firm. The advent of systems intrusion, however, has enabled attackers to use more sophisticated electronic techniques to obtain secrets in new ways.

Military purposes

Governments may break into systems for military purposes—what specialists often refer to as *cyber warfare* or *information warfare*. Imagine that a government wants to invade another country. Could they somehow attack the target's air defense systems and trick them into not recognizing an inbound air force? Could they shut down their power, water, or banking systems?[2] Alternatively, imagine that a government wants to prevent another country from building or obtaining a weapon. Could they remotely and stealthily disrupt their progress? This scenario supposedly happened in Iran in the late 2000s, when attackers illicitly introduced a modularized piece of software onto the control systems of centrifuges used to enrich uranium. Dubbed *Stuxnet* by researchers, this operation reportedly intended to destroy the centrifuges and halt Iran's nuclear program.

Policing domestic activity

Governments may also break into systems to police domestic activity. In a recent example, NSO Group, a cybersecurity contractor, sold software to various governments that allowed private surveillance of communications between people without their knowledge (through the remote monitoring of mobile phone calls). Reportedly, this software was intended to surveil terrorists and criminals—relatively noncontroversial targets. Unfortunately, some of NSO Group's government customers have also used the software to listen in on journalists and activists, in some cases leading to harassment, arrest, and even possibly death.[3] The ethics of governments using these capabilities against their own people is a hotly debated topic, especially in countries without strong legal frameworks and proper oversight.

Protecting your systems from nation-state actors

System designers should carefully consider whether they could be the target of a nation-state actor. To this end, you need to understand activities carried out by your organization that may be attractive to these actors. Consider a technology company that builds and sells microprocessor technology to a military branch of the government. It's possible that other governments would also be interested in having those chips, and may resort to stealing their designs via electronic means.

Your service may also have data that a government wants but that is otherwise difficult for it to obtain. Generally speaking, intelligence agencies and law enforcement value personal communications, location data, and similar types of sensitive personal information. In January 2010, Google announced it had witnessed a sophisticated targeted attack from China (dubbed "Operation Aurora" by researchers) against its corporate infrastructure that is now widely understood to have been aimed at long-term access to Gmail accounts. Storing the personal information of customers,

[2] As an example of how complicated this space can be, not all attackers in such conflicts are part of an organized military. For example, Dutch attackers reportedly compromised the US military during the Persian Gulf War (1991) and offered stolen information to the Iraqi government.

[3] NSO Group's activities have been researched and documented by The CitizenLab, a research and policy laboratory based at the Munk School of Global Affairs & Public Policy, University of Toronto. For an example, see *https://oreil.ly/IqDN_*.

especially private communications, can raise the risk that an intelligence or law enforcement agency would be interested in your systems.

Sometimes you might be a target without realizing it. Operation Aurora wasn't limited to large tech companies—it affected at least 20 victims in a variety of finance, technology, media, and chemical sectors. These organizations were both large and small, and many did not consider themselves at risk of a nation-state attack.

Consider, for example, an app that aims to provide athletes with data tracking analytics, including where they cycle or run. Would this data be an attractive target to an intelligence agency? Analysts looking at a public heatmap created by the fitness tracking company Strava considered this exact question in 2018 when they noticed that the locations of secret military bases in Syria were revealed when US troops used the service to track their workouts.

System designers should also be aware that governments can typically deploy significant resources to obtain access to data that they're interested in. Mounting a defense against a government that's interested in your data might require far and above the resources your organization can dedicate to implementing security solutions. We recommend that organizations take the long view with regard to building security defenses by investing early in protecting their most sensitive assets, and by having a continued rigorous program that can apply new layers of protections over time. An ideal outcome is forcing an adversary to expend a significant amount of their resources to target you—increasing their risk of being caught—so that their activities can be revealed to other possible victims and government authorities.

Activists

Hacktivism is the act of using technology to call for social change. This term is loosely applied to a wide variety of online political activities, from the subversion of government surveillance to the malicious disruption of systems.[4] For the purpose of thinking about how to design systems, we consider the latter case here.

Hacktivists have been known to *deface* websites—that is, replace normal content with a political message. In one example from 2015, the Syrian Electronic Army—a collective of malicious actors operating in support of the regime of Bashar al-Assad—took over a content distribution network (CDN) that served web traffic for *www.army.mil*. The attackers were then able to insert a pro-Assad message subsequently seen by visitors to the website. This kind of attack can be very embarrassing for website owners and can undermine user trust in the site.

[4] There is some debate about who coined this term and what it means, but it became widely used after 1996 when it was adopted by Hacktivismo, a group associated with the Cult of the Dead Cow (cDc).

Other hacktivist attacks may be far more destructive. For example, in November 2012 the decentralized, international hacktivist group Anonymous[5] took numerous Israeli websites offline through denial-of-service attacks. As a result, anyone visiting the affected websites experienced slow service or an error. Distributed denial-of-service attacks of this nature send the victim a flood of traffic from thousands of compromised machines distributed across the world. Brokers of these so-called botnets often provide this sort of capability for purchase online, making the attacks common and easy to carry out. On the more serious end of the spectrum, attackers may even threaten to destroy or sabotage systems entirely, inspiring some researchers to label them cyberterrorists.

Unlike other types of attackers, hacktivists are usually vocal about their activity and often take credit publicly. This can manifest itself in numerous ways, including posting on social media or destroying systems. Activists involved in such attacks may not even be very technically savvy. This can make predicting or defending against hacktivism difficult.

Protecting your systems from hacktivists

We recommend thinking about whether your business or project is involved in controversial topics that may draw the attention of activists. For example, does your website allow users to host their own content, like blogs or videos? Does your project involve a politically oriented issue like animal rights? Do activists use any of your products, such as a messaging service? If the answer to any of these questions is "yes," you may need to consider very robust, layered security controls that ensure your systems are patched against vulnerabilities and resilient to DoS attacks, and that your backups can restore a system and its data quickly.

Criminal Actors

Attack techniques are used to carry out crimes that closely resemble their nondigital cousins—for example, committing identity fraud, stealing money, and blackmail. *Criminal actors* have a wide range of technical abilities. Some may be sophisticated and write their own tools. Others may purchase or borrow tools that other people build, relying on their easy, click-to-attack interfaces. In fact, *social engineering*—the act of tricking a victim into aiding you in the attack—is highly effective despite being at the lowest end of difficulty. The only barriers to entry for most criminal actors are a bit of time, a computer, and a little cash.

Presenting a full catalog of the kinds of criminal activities that occur in the digital realm would be impossible, but we provide a few illustrative examples here. For

[5] Anonymous is a moniker that a wide variety of people use for hacktivist (and other) activities. It may (or may not) refer to a single person or a collective of related persons, depending on the situation.

example, imagine that you wanted to predict merger and acquisition activities so you could time certain stock trades accordingly. Three criminal actors in China had this exact idea in 2014–2015 and made a few million dollars by stealing sensitive information from unsuspecting law firms.

In the past 10 years, attackers have also realized that victims will hand over money when their sensitive data is threatened. *Ransomware* is software that holds a system or its information hostage (usually by encrypting it) until the victim makes a payment to the attacker. Commonly, attackers infect victim machines with this software (which is often packaged and sold to attackers as a toolkit) by exploiting vulnerabilities, by packaging the ransomware with legitimate software, or by tricking the user into installing it themselves.

Criminal activity does not always manifest as overt attempts to steal money. *Stalkerware*—spying software that's often sold for as little as $20—aims to gather information about another person without their knowledge. The malicious software is introduced onto a victim's computer or mobile phone either by tricking the victim into installing it or via direct installation by an attacker with access to the device. Once in place, the software can record video and audio. Since stalkerware is often used by people close to the victim, such as a spouse, this kind of trust exploitation can be devastatingly effective.

Not all criminal actors work for themselves. Companies, law firms, political campaigns, cartels, gangs, and other organizations hire malicious actors for their own purposes. For example, a Colombian attacker claimed he was hired to assist a candidate in the 2012 presidential race in Mexico and other elections throughout Latin America by stealing opposition information and spreading misinformation. In a stunning case from Liberia, an employee of Cellcom, a mobile phone service provider, reportedly hired an attacker to degrade the network of its rival cellular service provider, Lonestar. These attacks disrupted Lonestar's ability to serve its customers, causing the company to lose significant amounts of revenue.

Protecting your systems from criminal actors

When designing systems to be resilient against criminal actors, keep in mind that these actors tend to gravitate toward the easiest way to meet their goals with the least up-front cost and effort. If you can make your system resilient enough, they may shift their focus to another victim. Therefore, consider which systems they might target, and how to make their attacks expensive. The evolution of Completely Automated Public Turing test (CAPTCHA) systems is a good example of how to increase the cost of attacks over time. CAPTCHAs are used to determine whether a human or an automated bot is interacting with a website—for example, during a login. Bots are often a sign of malicious activity, so being able to determine if the user is human can be an important signal. Early CAPTCHA systems asked humans to validate slightly distorted letters or numbers that bots had a difficult time recognizing. As the bots became more sophisticated, CAPTCHA implementers began using distortion pictures and object recognition. These tactics aimed to significantly increase the cost of attacking CAPTCHAs over time.[6]

Automation and Artificial Intelligence

In 2015, the US Defense Advanced Research Projects Agency (DARPA) announced the Cyber Grand Challenge contest to design a cyber-reasoning system that could self-learn and operate without human intervention to find flaws in software, develop ways to exploit these flaws, and then patch against the exploitations. Seven teams participated in a live "final event" and watched their fully independent reasoning systems attack each other from the comfort of a large ballroom. The first-place team succeeded in developing such a self-learning system!

The success of the Cyber Grand Challenge suggests that it's likely at least some attacks in the future could be executed without humans directly at the controls. Scientists and ethicists ponder whether fully sentient machines might be capable enough to learn how to attack each other. The notion of autonomous attack platforms is also prompting the need for increasingly automated defenses, which we predict will be an important area of research for future system designers.

Protecting your systems from automated attacks

To withstand the onslaught of automated attacks, developers need to consider resilient system design by default, and be able to automatically iterate the security posture of their systems. We cover many of these topics in this book, such as automated configuration distribution and access justifications in Chapter 5; automated build, test, and deployment of code in Chapter 14; and handling DoS attacks in Chapter 8.

Insiders

Every organization has *insiders*: current or former employees who are trusted with internal access to systems or proprietary knowledge. *Insider risk* is the threat posed by such individuals. A person becomes an *insider threat* when they are able to perform actions that cover a wide range of malicious, negligent, or accidental scenarios that could result in harm to the organization. Insider risk is a large topic that could fill the

[6] The race to increase the effectiveness of CAPTCHA techniques continues, with newer advancements using behavioral analysis of users as they interact with the CAPTCHA. reCAPTCHA is a free service you can use on your website. For a relatively recent overview of the research literature, see Chow Yang-Wei, Willy Susilo, and Pairat Thorncharoensri. 2019. "CAPTCHA Design and Security Issues." In *Advances in Cyber Security: Principles, Techniques, and Applications*, edited by Kuan-Ching Li, Xiaofeng Chen, and Willy Susilo, 69–92. Singapore: Springer.

pages of several books. To help system designers, we cover the topic briefly here by considering three general categories, as outlined in Table 2-1.

Table 2-1. General categories of insiders and examples

First-party insiders	Third-party insiders	Related insiders
Employees	Third-party app developers	Friends
Interns	Open source contributors	Family
Executives	Trusted content contributors	Roommates
Board directors	Commercial partners	
	Contractors	
	Vendors	
	Auditors	

Intersection of Reliability and Security: Effects of Insiders

When it comes to protecting against adversaries, reliability and security intersect most when you're designing systems to be resilient against insiders. This intersection is largely due to the privileged access insiders have to your systems. Most reliability incidents stem from actions taken by an insider who often doesn't realize how they're impacting the system—for example, by introducing faulty code or an errant configuration change. On the security side, if an attacker can take over an employee's account, then the attacker can act maliciously against your systems as if they were that insider. Any permissions or privileges you assign to your insiders become available to an attacker.

When designing systems to be both reliable and secure, it's best practice to consider both well-intended insiders who might make mistakes and attackers who might take over an employee account. For example, if you have a database with sensitive customer information that's critical to your business, you likely want to prevent employees from accidentally deleting the database while performing maintenance work. You also want to protect database information from an attacker that hijacks an employee's account. Techniques for least privilege, outlined in Chapter 5, protect against both reliability and security risks.

First-party insiders

First-party insiders are people brought into the fold for a specific purpose—usually to participate directly in meeting business objectives. This category includes employees who directly work for the company, executives, and members of the board who make critical company decisions. You can probably think of other people who fall into the category too. Insiders with first-party access to sensitive data and systems make up the majority of news stories about insider risk. Take the case of the engineer working for General Electric who was indicted in April 2019 on charges of stealing proprietary files, embedding them into photos using steganographic software (in order to conceal their theft), and sending them to his personal email account. Prosecutors allege that his goal was to enable him and his business partner in China to produce low-cost versions of GE's turbomachines and sell them to the Chinese government. Stories like this are prevalent throughout high-tech firms that produce next-generation technology.

Access to personal data can also be tempting to insiders with voyeuristic tendencies, people who want to seem important for having privileged access, and even people who want to sell such information. In an infamous case from 2008, several hospital workers were fired from UCLA Medical Center after inappropriately looking at the files of patients, including high-profile celebrities. As more and more consumers sign up for social networking, messaging, and banking services, protecting their data from inappropriate employee access is more important than ever.

Some of the most radical stories of insider risk involve disgruntled insiders. In January 2019, a man who had been fired for poor performance was convicted of deleting 23 of his former employer's virtual servers. The incident lost the company key contracts and significant revenue. Almost any company that's been around for a while has similar stories. Because of the dynamics of employment relationships, this risk is unavoidable.

The preceding examples cover scenarios in which someone with malicious intent affects the security of systems and information. However, as some examples earlier in the book illustrate, first-party insiders can also impact the reliability of systems. For example, the previous chapter discusses a string of unfortunate insider actions in the design, operation, and maintenance of a password storage system that prevented SREs from accessing credentials in an emergency. As we'll see, anticipating the mistakes that insiders can introduce is vital to guaranteeing system integrity.

Third-party insiders

With the rise of open source software and open platforms, it's increasingly likely that an insider threat may be someone whom few people (or no one) in your organization have ever met. Consider the following scenario: your company has developed a new library that's helpful for processing images. You decide to open source the library and accept code change lists from the public. In addition to company employees, you now have to consider open source contributors as insiders. After all, if an open source contributor on the other side of the world whom you've never met submits a malicious change list, they can harm people using your library.

Similarly, open source developers rarely have the ability to test their code in all environments where it might be deployed. Additions to the codebase might introduce unpredictable reliability issues, such as unanticipated performance degradations or hardware compatibility issues. In this scenario, you'd want to implement controls ensuring that all submitted code is thoroughly reviewed and tested. For more details on best practices in this area, see Chapters 13 and 14.

You should also think carefully about how you extend a product's functionality via application programming interfaces (APIs). Suppose your organization develops a human resources platform with a third-party developer API so companies can easily extend the functionality of your software. If the third-party developer has privileged or special access to the data, they may now be an insider threat. Carefully consider the access you're providing through the API, and what the third party can do once they have access. Can you limit the impact these extended insiders have on system reliability and security?

Related insiders

It's not uncommon to implicitly trust the people we live with, but these relationships are often overlooked by system designers when designing secure systems.[7] Consider a situation in which an employee takes their laptop home over the weekend. Who has access to that device when it's unlocked on the kitchen table, and what impact could they have, either maliciously or unintended? Telecommuting, working from home, and late-night pager duty are increasingly common for technology workers. When considering your insider risk threat model, be sure to use a broad definition of "workplace" that also includes the home. The person behind the keyboard may not always be the "typical" insider.

Determining Insider Intent

If a system goes offline because of the actions of an insider, and they claim their actions were an accident, do you believe them? The answer to this question can be difficult to determine, and in extreme cases of negligence, it may be impossible to conclusively confirm or rule out. Such cases often require working with expert investigators, such as your organization's legal department, human resources, and perhaps even law enforcement. First and foremost, when designing, running, and maintaining your systems, plan for both malicious and unintended actions, and assume you may not always know the difference.

[7] For an example of considering how security and privacy features are impacted by domestic partner abuse, see Matthews, Tara et al. 2017. "Stories from Survivors: Privacy & Security Practices When Coping with Intimate Partner Abuse." *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*: 2189–2201. *https://ai.google/research/pubs/pub46080*.

Threat modeling insider risk

Numerous frameworks exist for modeling insider risk, ranging from simple to highly topic-specific, sophisticated, and detailed. If your organization needs a simple model to get started, we have successfully used the framework in Table 2-2. This model is also adaptable to a quick brainstorming session or fun card game.

Table 2-2. Framework for modeling insider risk

Actor/Role	Motive	Actions	Target
Engineering	Accidental	Data access	User data
Operations	Negligent	Exfiltration (theft)	Source code
Sales	Compromised	Deletions	Documents
Legal	Financial	Modifications	Logs
Marketing	Ideological	Injections	Infrastructure
Executives	Retaliatory	Leak to press	Services
	Vanity		Financials

First, establish a list of *actors/roles* present in your organization. Attempt to think of all the *actions* that may cause harm (including accidents) and potential *targets* (data, systems, etc.). You can combine items from each category to create many scenarios. Here are some examples to get you started:

- An *engineer* with access to *source code* is unsatisfied with their performance review and *retaliates* by injecting a malicious backdoor into production that steals *user data*.
- An *SRE* with access to the website's SSL *encryption keys* is approached by a stranger and is *strongly encouraged* (for example, via threats to their family) to hand over sensitive material.
- A *financial analyst* preparing the *company financials* is working overtime and *accidentally* modifies the *final yearly revenue numbers* by a factor of 1,000%.
- An *SRE's child* uses their parent's laptop at home and installs a game bundled with *malware* that locks the computer and *prevents the SRE from responding* to a serious outage.

Threat Modeling Mistakes

Sometimes people just make mistakes—to err is human. For around 40 minutes on January 31, 2009, Google Search displayed an ominous warning—"This site may harm your computer"—to every user, for every search! This warning is normally reserved for search results that link to a website that's either compromised or hosting malware. The root cause of this issue was very simple: a "/" had been implicitly (and accidentally!) added to the system's list of sites known to install malicious software in the background, which matched every website on the planet.

Given sufficient time working with systems, everyone is likely to encounter some version of this horror story. These mistakes can be caused by working late at night without enough sleep, typos, or simply encountering unforeseen functionality in the system. When designing secure and reliable systems, remember that humans make mistakes, and consider how to prevent them. An automated check on whether "/" was added to the configuration would have prevented the aforementioned outage!

Designing for insider risk

This book presents many design strategies for security that are applicable to protecting against insider risk and malicious "outside" attackers. When designing systems, you must consider that whoever has access to a system or its data could be any of the attacker types outlined in this chapter. Therefore, the strategies for detecting and mitigating both types of risk are similar.

We have found a few concepts to be particularly effective when thinking about insider risk:

Least privilege
: Granting the fewest privileges necessary to perform job duties, both in terms of scope and duration of access. See Chapter 5.

Zero trust
: Designing automated or proxy mechanisms for managing systems so that insiders don't have broad access that allows them to cause harm. See Chapter 3.

Multi-party authorization
: Using technical controls to require more than one person to authorize sensitive actions. See Chapter 5.

Business justifications
: Requiring employees to formally document their reason for accessing sensitive data or systems. See Chapter 5.

Auditing and detection
Reviewing all access logs and justifications to make sure they're appropriate. See Chapter 15.

Recoverability
The ability to recover systems after a destructive action, like a disgruntled employee deleting critical files or systems. See Chapter 9.

Attacker Methods

How do the threat actors we've described carry out their attacks? Knowing the answer to this question is critical for understanding how someone might compromise your systems and, in turn, how you can protect them. Understanding how attackers operate can feel like complex magic. Trying to predict what any particular attacker might do on any given day is unfeasible because of the variety of attack methods available. There is no way for us to present every possible method here, but thankfully, developers and system designers can leverage an increasingly large repository of examples and frameworks to wrap their heads around this problem. In this section, we discuss a few frameworks for studying attacker methods: threat intelligence, cyber kill chains, and TTPs.

Threat Intelligence

Many security firms produce detailed descriptions of attacks they've seen in the wild. This *threat intelligence* can help system defenders understand how real attackers are working every day and how to repel them. Threat intelligence comes in multiple forms, each serving a different purpose:

- *Written reports* describe how attacks occurred and are especially useful for learning about the progression and intent of an attacker. Such reports are often generated as a result of hands-on response activities and may vary in quality depending on the expertise of the researchers.
- *Indicators of compromise* (IOCs) are typically finite attributes of an attack, such as the IP address where an attacker hosted a phishing website or the SHA256 checksum of a malicious binary. IOCs are often structured using a common format[8] and obtained through automated feeds so they can be used to programmatically configure detection systems.

[8] For example, many tools are incorporating the Structure Threat Information eXpression (STIX) language to standardize the documentation of IOCs that can be traded between systems using services like the Trusted Automated eXchange of Indicator Information (TAXII) project.

- *Malware reports* provide insight into the capabilities of attacker tools and can be a source of IOCs. These reports are generated by experts in *reverse engineering* binaries, usually using standard tools of the trade such as IDA Pro or Ghidra. Malware researchers also use these studies to cross-correlate unrelated attacks according to their common software attributes.

Acquiring threat intelligence from a reputable security firm—preferably one with customer references—can help you better understand the observed activities of attackers, including attacks affecting peer organizations in your industry. Knowing what kinds of attacks organizations similar to yours are facing can provide an early warning of what you might face someday. Many threat intelligence firms also publicly release yearly summary and trend reports for free.[9]

Cyber Kill Chains™

One way of preparing for attacks is to lay out all the possible steps that an attacker may have to take to achieve their goals. Some security researchers use formalized frameworks like the Cyber Kill Chain[10] to analyze attacks this way. These kinds of frameworks can help you plot the formal progression of an attack alongside defensive controls to consider. Table 2-3 shows the stages of a hypothetical attack relative to some defensive layers.

Table 2-3. Cyber Kill Chain of a hypothetical attack

Attack stage	Attack example	Example defenses
Reconnaissance: Surveilling a target victim to understand their weak points.	Attacker uses a search engine to find the email addresses of employees at a target organization.	Educate employees about online safety.
Entry: Gaining access to the network, systems, or accounts necessary to carry out the attack.	Attacker sends phishing emails to employees that lead to compromised account credentials. The attacker then signs in to the organization's virtual private network (VPN) service using those credentials.	Use two-factor authentication (such as security keys) for the VPN service. Only permit VPN connections from organization-managed systems.
Lateral movement: Moving between systems or accounts to gain additional access.	Attacker remotely logs in to other systems using the compromised credentials.	Permit employees to log in to only their own systems. Require two-factor authentication for login to multiuser systems.

[9] Notable examples include the annual Verizon Databreach Investigations Report and CrowdStrike's annual Global Threat Report.

[10] The Cyber Kill Chain, conceived (and trademarked) by Lockheed Martin, is an adaptation of traditional military attack structures. It defines seven stages of cyberattacks, but we've found this can be adapted; some researchers simplify it to four or five key stages, as we've done here.

Attack stage	Attack example	Example defenses
Persistence: Ensuring ongoing access to compromised assets.	Attacker installs a backdoor on the newly compromised systems that provides them with remote access.	Use application whitelisting that permits only authorized software to run.
Goals: Taking action on attack goals.	Attacker steals documents from the network and uses the remote access backdoor to exfiltrate them.	Enable least privileged access to sensitive data and monitoring of employee accounts.

Tactics, Techniques, and Procedures

Methodically categorizing attacker TTPs is an increasingly common way of cataloging attack methods. Recently, MITRE has developed the ATT&CK framework to instrument this idea more thoroughly. In short, the framework expands each stage of the cyber kill chain into detailed steps and provides formal descriptions of how an attacker could carry out each stage of an attack. For example, in the Credential Access stage, ATT&CK describes how a user's *.bash_history* may contain accidentally typed passwords that an attacker could obtain by simply reading the file. The ATT&CK framework lays out hundreds (potentially thousands) of ways attackers can operate so that defenders can build defenses against each attack method.

Risk Assessment Considerations

Understanding potential adversaries, who they are, and which methods they might use can be complex and nuanced. We have found the following considerations important when assessing the risk posed by various attackers:

You may not realize you're a target.
: It may not be immediately obvious that your company, organization, or project is a potential target. Many organizations, despite being small or not involved in handling sensitive information, can be leveraged to carry out attacks. In September 2012, Adobe—a company best known for software that enables content creators—disclosed that attackers had penetrated its networks with the express intent to digitally sign their malware using the company's official software signing certificate. This enabled the attackers to deploy malware that appeared legitimate to antivirus and other security protection software. Consider whether your organization has assets that an attacker would be interested in, either for direct gain or as part of a larger attack on someone else.

Attack sophistication is not a true predictor of success.
: Even if an attacker has a lot of resources and skills, don't assume that they'll always choose the most difficult, expensive, or esoteric means to achieve their goals. Generally speaking, attackers choose the simplest and most cost-effective methods of compromising a system that meet their goals. For example, some of the most prominent and impactful intelligence gathering operations rely on basic

phishing—tricking a user into handing over their password. For this reason, when designing your systems, be sure to cover the simple basics of security (like using two-factor authentication) before worrying about esoteric and exotic attacks (like firmware backdoors).

Don't underestimate your adversary.

Don't assume that an adversary can't procure the resources to carry out an expensive or difficult attack. Consider carefully how much your adversary is willing to spend. The extraordinary tale of the NSA implanting backdoors in Cisco hardware by intercepting shipments en route to customers illustrates the lengths that well-funded and talented attackers will go to achieve their goals.[11] However, keep in mind that these types of cases are very much the exception rather than the norm.

Attribution is hard.

In March 2016, researchers uncovered a new type of ransomware—a malicious program that renders data or systems unavailable until the victim pays a ransom —which they named Petya. Petya appeared to be financially motivated. A year later, researchers discovered a new piece of malware that shared many elements of the original Petya program. Dubbed NotPetya, the new malware spread globally very quickly, but was primarily found on systems in Ukraine on the eve of a Ukrainian holiday. To deliver NotPetya, attackers compromised a company that made products explicitly for the Ukrainian market and abused their software distribution mechanism to infect victims. Some researchers believe that this attack was carried out by a Russian state-sponsored actor in order to target Ukraine.

This example shows that motivated attackers can hide their motives and identity in creative ways—in this case, by disguising themselves as something potentially more benign. Since the identity and intent of attackers may not always be well understood, we recommend that you focus on how attackers work (their TTPs) before worrying about who they are specifically.

Attackers aren't always afraid of being caught.

Even if you manage to track an attacker's location and identity, the criminal system (especially internationally) may make it difficult to hold them legally accountable for their actions. This is especially true for nation-state actors working directly for a government that may be unwilling to extradite them for criminal prosecution.

[11] Greenwald, Glenn. 2014. *No Place to Hide: Edward Snowden, the NSA, and the U.S. Surveillance State*. New York: Metropolitan Books, 149.

Conclusion

All security attacks can be traced back to a motivated person. We've covered some common attacker profiles to help you identify who may want to target your services and why, allowing you to prioritize your defenses accordingly.

Assess who might want to target you. What are your assets? Who buys your products or services? Could your users or their actions motivate attackers? How do your defensive resources compare to the offensive resources of your potential adversaries? Even when facing a well-funded attacker, the information in the rest of this book can help make you a more expensive target, possibly removing the economic incentive for an attack. Don't overlook the smaller, less conspicuous adversary—anonymity, location, ample time, and the difficulty of prosecution can all be advantages to an attacker, allowing them to cause you disproportionately large amounts of damage. Consider your insider risk, as all organizations face both malicious and nonmalicious potential threats from insiders. The elevated access granted to insiders allows them to inflict significant damage.

Stay current on the threat intelligence issued by security firms. While a multistep attack methodology can be effective, it also provides multiple contact points where you can detect and prevent an attack. Be mindful of complex attack strategies, but don't forget that simple, unsophisticated attacks like phishing can be painfully effective. Don't underestimate your adversaries or your own value as a target.