Editors: Richard Ford, rford@se.fit.edu | Deborah A. Frincke, debfrincke@gmail.com

Protecting Me

Richard Ford and Marco Carvalho | Florida Institute of Technology

Tip-tapping away on the keyboard of a laptop as we write this article, it's easy to forget just how many layers of security are in place on a home machine to protect us from security ills. In fact, the amount of security-related technologies on a typical computer is immense, and ranges from the simple (such as firewalls) to the arcane (file reputation systems—and beyond). Whether you use a Mac, a PC, or a tablet, a great deal of technology is devoted to doing one task: protecting you from online attacks.

In this column, we give a quick overview of the many lines of protection you find on a typical home machine, which in turn helps us illustrate the complexity and challenges involved in securing a standard computer system. We focus on Windows here simply because it's widespread and widely attacked; many of the same techniques can be found on other platforms. As such, if you're a Unix user or a Mac lover, or even if you do all your work on a tablet, these techniques—and others that are very similar to them—can be found on your device, even if their presence isn't immediately obvious in the user interface. To be clear, we're using Windows as an example, but it isn't unique in its multifaceted protection mechanisms.

The way we've cited this installment is a bit different from previous columns: because we outline so many different technologies here, putting in a reference for each one would be quite painful. So for Windows, we refer the reader to MSDN as a starting point.

From the Ground Up

One of the many things that makes defense interesting is that no single silver bullet handles everything. Instead, the defenses deployed are layered, providing defense in depth. As there are very few (if any!) foolproof defenses, defenders have instead opted to continually raise the bar: if an attack defeats one preventive, another barrier is still present.

The first layer of protection is so commonplace that we don't even see it: a typical home machine's processor actually has many features that help provide security. While early DOS machines had just one mode of operation—any program could do anything—a modern CPU has multiple modes and offers different privilege rings for the system and applications. The goal is to provide isolation, so that a mistake or attack in a user-mode application (Ring 3) doesn't open the entire machine to compromise (Ring 0). New features add to this, allowing for virtualization of the host machine within a hypervisor; as such, the hardware itself is integral to protecting you. It isn't just the CPU; memory is also virtualized with the help of hardware, allowing fast operation of paging and virtualization of memory addresses, and adding to the isolation that the OS can provide quickly and inexpensively. Without these hardware features, security would be even more problematic than it is today (as hard to believe as that is).

In terms of hardware assistance, modern CPUs also allow the discrimination of data from code. This is most apparent when attempting to exploit a stack-based buffer overflow. Historically, when a programmer failed to adequately bounds-check a buffer—typically, a string buffer stored on the stack—the return value popped off the stack when an RET instruction was executed could have been corrupted. This could allow an attacker to transfer code to a location of his or her own choosing. Furthermore, as the computer chip didn't discriminate between code and data, the very same corrupted buffer on the stack was often the location to which the attacker would jump. This specific attack was made much harder by the addition of the "NX bit," which allowed the processor to mark certain regions of memory as not executable.

Not to be outdone by such a detail, attackers quickly developed techniques such as return-to-libc attacks, which further evolved to "return-oriented programming." In such an attack, small gadgets are identified within legitimate code segments. These gadgets are chained together to carry out the attacker's desires. Such techniques are feasible but tricky, as they depend on the attacker knowing

Copublished by the IEEE Computer and Reliability Societies

where these small code snippets are. In response, modern operating systems now randomize their memory layout; in Windows, for example, ASLR (Address Space Layout Randomization) randomizes the layout of the heap, stack, and other parts of the environment, making it hard to determine where a gadget may be.

The stack is also protected by the use of stack cookies, countering various stack-smashing attacks. Here, an unpredictable value is placed between the return address stored on the stack and local variables. Thus, if the stack is overrun, this stack "canary" will be destroyed. All the program therefore needs to do is to check that the canary is still in place (just like miners are warned about the presence of poisonous gases by the death of their bird) before carrying out a return.

Other countermeasures at these low levels are the BIOS and the TPM. The BIOS can step in before the machine has even really begun booting the OS and prevent an attacker from modifying startup options or even prevent the machine from booting without a password. Similarly, the TPM can be used to provide secure cryptographic services, providing high-quality random numbers or storing and generating cryptographic keys securely. The TPM is probably best known for its assistance in providing whole hard drive encryption via Bitlocker, Microsoft's full disk encryption software.

Access Control

Although most nontechnical people pay little to no attention to it, the Windows Access Control model is incredibly rich. Once again, users might be forgiven for not even noticing that they've logged on to a system, but desktop machines typically require a login, identifying users and their permissions. Even when a user is operating with high privileges, Windows uses User Account Control to switch between a lower-privileged user account and a higher-privileged administrator from the GUI, without requiring a switch of users. This potentially limits the privileges of the day-to-day user, although the ease and frequency of switching privileges have led to complaints.

Once a particular user is identified, Windows has highly granular access control lists. In addition to simple read, write, and execute privileges, the user has complete control over what can be done with any particular object. These are controlled by a DACL (Discretionary Access Control List) and an SACL (System Access Control List), providing features such as the ability to read and write a file or to list and traverse directories. While most users never modify these rights, they're still there, and by default, they're configured to provide some limited protection.

On top of this, Windows also provides a feature called Mandatory Integrity Control, which assigns integrity levels to different processes, such that lower integrity levels can't write or delete objects with higher integrity levels. For process objects, higher-integrity level processes can't even be read by lower-integrity processes. Finally, Windows ACLs can specify the minimum integrity level required for access to named objects (or any object with an ACL). This is particularly helpful to Internet-facing processes, which can be marked as low integrity, providing some level of OS-assisted sandboxing.

The Network

If OS security seems problematic, it gets much more difficult when the machine is connected to a network. To this end, the typical home machine has several features designed to improve the security of both using the network and being attached to it.

First, most machines now have a hardened network interface, in that a firewall is in place to limit the visibility and accessibility of services on the machine. This firewall can be a complex after-market addition or simply the one built in by Microsoft. Either way, the goal is to limit the type and number of connections made to the machine. Often, different profiles are provided for public (that is, untrusted) networks, work, and home.

But even with a firewall in place, most home machines aren't directly attached to the Internet. Instead, the home router typically employs a security technique known as network address translation (NAT) to hide machines from external attackers. NAT is very simple and in practice means that it's difficult for an attacker to "start" a network conversation with a host inside the protected (translated) region. Instead, conversations must be initiated from the host behind the NAT boundary.

Finally, encryption can be used to protect data in transit. It's very common now for encryption services such as Secure Sockets Layer (SSL) to provide end-to-end encryption of communication over the Internet. By making use of certificate authorities, SSL can also ensure that your machine is communicating with the machine you think it is—this is probably best known in the context of HTTPS, where the browser can identify a particular endpoint. At the lower level of the network stack, Wi-Fi can (and should) be configured to use WPA2 (Wi-Fi Protected Access). This effectively prevents an attacker from snooping on the contents of wireless packets as they pass by in the ether.

Antimalware Software

Despite the many layers of security found within the OS, most users also deploy some form of antimalware. This software has slowly grown in footprint and function,

and might now more correctly be called a security suite. Products are fairly tightly integrated into the browser and email clients.

Historically, antimalware software focused on stopping malware from running on the machine. This was achieved primarily by looking for particular malware "signatures"—perhaps a particular byte construction or sequence. As malware got more sophisticated, products likewise improved their detection methods, and an uneasy standoff existed between attackers and defenders. Recently, however, truly ubiquitous connectivity has allowed attackers to focus on malware that doesn't self-replicate and is "server-side polymorphic"—that is, where the samples change frequently, sometimes with every download. This approach makes it impractical to follow simple signature-based techniques, so defenders have been forced to adopt file reputation systems, where the cloud is used to pull information on a particular file, providing answers to questions, such as, "Is the file known good or bad?," and "How frequently is such a file encountered?"

In addition to just looking at files, client-side security suites often look at link reputation and content before allowing the browser to connect and also attempt to detect entire classes of exploits generically. A great deal of innovation and research by defenders goes into protecting the client from rogue software and exploits.

The Browser

Because the browser is the machine's "window to the world," attackers and defenders have focused on it. Unsurprisingly, a large number of technologies aim to secure the browser.

One of the most common attacks a user might expect online is a phishing attempt, which has any number of preventives. For example, built into Internet Explorer is a filter that uses a blacklisting approach to block access to sites known to be phishing attack endpoints. In addition, an analysis compares site content to known phishing sites. The phishing filter can also send information back to Microsoft, allowing real-time comparison of the site URL and IP address to a continuously updated list of attack sites.

Another common attack is cross-site scripting (XSS), in which the attacker attempts to capture the cookie that uniquely identifies a browser or individual session with a third-party website. Such attacks can be either persistent (such as embedded in a message in a poorly filtered message board) or included in the browser request and sent via a link. Either way, the attacker can use the captured credentials to impersonate the user.

Many browsers attempt to mitigate XSS. It's now standard to be able to mark cookies as "HTTPOnly," making them unavailable (in theory) to a script running on the client side. Another approach is to examine the code presented to the browser to look for signs of XSS. While this technique can be quite good at detecting an attack, it can also suffer from false positives, which limits its utility.

Of course, it isn't just the base browser that's vulnerable: attackers (and unscrupulous advertisers) also take advantage of plugins to accomplish their goals. Plugin vulnerabilities are distressingly common and can range from privacy issues (such as the cookies that can be stored and recovered through the Flash plugin) to code injection. An overview of the security techniques used to prevent (or otherwise mitigate the damage) from these attacks could easily fill an entire article, but sandboxing, code signing, and mandatory integrity control are often employed.

We've just given a very quick tour of some of the many different technologies in place on a typical Windows machine. As a conscious choice, we decided not to dig too deeply but instead wanted to show the breadth and diversity of defensive techniques that go into protecting a typical home setup. There's a real smorgasbord of approaches here, and aspiring security wizards should be aware of the range, even if the deeper technical details escape them.

Although each technology has its limitations, it's amazing to see just how many different security features a current-generation OS has in place. Unfortunately, this is representative of the coevolution that takes place between the attacker and the defender: each improvement in robustness has an attacker response. Furthermore, the asymmetry of the situation means that attackers, as always, need to find only one way in, whereas the technologies shown here have to stop every possible attack. As it turns out, protecting you—or, selfishly, us!—is hard work indeed. ■

Richard Ford is co-director of the Harris Institute for Assured Information at the Florida Institute of Technology. His research interests include malicious code, novel exploit techniques, and low-level security. Contact him at rford@fit.edu.

Marco Carvalho is co-director of the Harris Institute for Assured Information at Florida Institute of Technology. His research interests include adaptive and coordinated defense, cyber resilience, and tactical network security. Contact him at mcarvalho@fit.edu.