Editors: Elias Levy, aleph1@securityfocus.com
Iván Arce, ivan.arce@coresecurity.com

Bad Peripherals

Iván Arce
Core Security Technologies

Modern computer systems benefit from a plethora of peripheral devices that provide such features as removable data storage, wireless network connectivity, multimedia capabilities, high-speed data transfers, and other enhancements for the human–machine interface. The hardware and software industries have made sensible efforts in defining and refining specifications, standards, and APIs that support several types of peripheral devices while fostering new development, which might prove to be a new battleground for sophisticated attackers and defenders. In this installment of Attack Trends, I'll analyze how peripheral devices interact with the operating system and what threats that interaction poses.

Who controls the peripheral device?

A computer peripheral is a hardware device that can be added (or connected) to a host computer to expand its capabilities; essentially, they're optional components that enhance computers' basic data processing and storage functions. The conceptual divide between basic functions and expanded capabilities has diminished with computers' rapid development since the inception of general-purpose PCs, and, more importantly, with the prevalence of the millions of computers used for work and leisure. Several decades ago, monitors and keyboards, now required components for normal operation of a computer system, were once thought of as add-ons that were "nice to have."

Similarly, new types of peripherals that provide additional features and enhance users' experiences today—USB storage devices, memory and smart-card readers, DVD players and burners—might become must-have computer system components in the near future. Network cards, video cards with powerful processing capabilities, and onboard modems and audio devices are examples of such a trend.

Additionally, the concept of a hardware-only peripheral is rapidly approaching obsolescence: as these devices pack in more complex and processor-intensive capabilities, ad hoc hardware and their required firmware are becoming common in the world of peripheral development.

This brief account of peripheral evolution raises some pointed questions: Who controls the peripherals of a computer system? Where is the trust put? What are the threats? Let's examine some plausible answers.

The operating system's view

From the operating system's viewpoint, peripherals are hardware devices controlled by the kernel. Their operation and management are handled by *device drivers*—portions of kernel code responsible for initialization, data transfer to and from peripherals, and overall management of the device's operation—and *interfaces*, portions of kernel code that implement a standardized and generic interface to peripherals that also provide access to the peripheral's functionality to other kernel subsystems and user applications.

The most simple and clear view of peripheral management is at the core of Unix operating system design: all peripherals are either *block* or *character* devices, which are controlled by basic operations that peripheral-specific device drivers must implement. Network cards are a notable variation of this basic design principle in modern Unix systems; they're controlled by *network interface drivers* with certain peculiarities, but the design premise remains invariable.[1]

Proliferation of peripheral manufacturers and a range of new types of peripheral devices have made both device driver and interface code bigger and more complex. Implementing kernel interfaces to support peripherals with a rich set of features and broad functional specifications is a common requirement in modern operating systems running on PCs, leading to an increase in code complexity and size. This is detrimental to software security and quality—and peripheral management code is no exception to the rule.

In "The Kernel Craze" (*IEEE Security & Privacy*, vol. 2, no. 3, 2004, pp. 79–81), I examined kernel bugs and their direct threat to system security and stability. Peripherals that interact directly with buggy kernel code or that have lax security controls can only amplify the possible threats and attack avenues. Nonethe-

less, the answer to our first question appears clear now: peripherals are (or at least seem to be) controlled by the operating system's kernel code.

The computer user's view

Computer users might have a different answer—one that's not based on peripherals' technical aspects. Most users consider peripherals tangible, physical objects that they plug into the computer to obtain expanded functionality. Their physical appearance meets consumer expectations with regard to size, color, overall design, desired functionality, usability, and other requirements; security is often last on the list.

Peripheral and operating systems manufacturers have made long strides toward simplifying device use for the average consumer by moving the complexity into operating system components; they now have greater flexibility and generic interfaces that support a range of devices and feature sets (see www.upnp.org). Fidgeting with hardware or operating system software to make a new peripheral work can negatively impact adoption of a type or brand of peripheral. Therefore, from the user's viewpoint, we arrive at another plausible answer to our first question: computer users control (or think they control) computer peripherals.

However, because this answer is largely based on user perception rather than objective analysis, it suffers from inaccuracies:

- Some peripherals are invisible to the user. Onboard wired and wireless networking cards, modems, and video and audio adapters are transparent and can't be physically disconnected from the system. Software components govern their operations, whereas the system's user interfaces control installation, configuration, activation, and uninstallation.
- A peripheral's physical appearance denotes its functionality and follows design and usability paradigms that might not be aligned with information security best practices. Most computer users feel that a USB device is not a keyboard unless it looks like one.
- Most peripherals' security capabilities aren't easily visible. A wary user can't easily verify that a given peripheral is in fact what it claims to be and that it operates according to specification.
- Most peripherals aren't tamper-resistant. An attacker needs only minimal skills to alter a peripheral without leaving a trace.

Computer users' adoption of popular peripheral devices implies the adoption of a security threat model that's not explicitly defined and one which is largely based on perception. Users, device manufacturers, and software vendors might have different ideas of what that model is and which trusted party controls it.

Subverted trust

Both the operating system's and user's viewpoints of peripheral devices outline a security scenario with blurred lines for defense and no explicit trust model to apply.

The kernel code that manages and operates peripheral devices is a poor single line of defense for today's most popular operating systems. Device drivers and kernel interfaces don't consider peripherals potentially malicious; at most, they're only prepared to handle abnormal conditions due to benign device malfunction. Some vendors are attempting to modify this sad state of affairs (see www.microsoft.com/whdc/driver/security/threatmodel.mspx), but significant results at a relevant scale have yet to be seen.

Additionally, because users view peripherals as benign, friendly devices, they use them according to an implicit trust model that might not match information security expectations and best practices.

A thorough analysis and clear understanding of the underlying implicit trust model associated with the use of computer peripherals can help identify potential threats. Unfortunately, this task is cumbersome and resource intensive for general-purpose computer systems that run

off-the-shelf operating systems and application software, and that use non-security-oriented generic peripheral devices. Security devices and security-oriented operating system architectures might prove useful as references for analysis.

...having a clear picture of the trust model associated with peripheral devices is just the first step toward identifying threats and mitigating factors.

However, having a clear picture of the trust model associated with peripheral devices is just the first step toward identifying threats and mitigating factors. To effectively control threats associated with peripheral devices, we must not only understand the implicit trust model used by both the kernel and the users, but also the implementation details and attack scenarios that could subvert the model.

This might answer our second question: both the operating system and the user are the security trustees of a computer peripheral, but it's unclear what that trust is and to what degree each of them contribute to the overall trust model.

Attack through peripherals

A comprehensive approach to understanding and addressing information security risks associated with using computer peripherals is vital. Such an approach would necessarily touch on topics *IEEE Security & Privacy* magazine covers in regular departments and feature issues: usability, basic security training, secure architectures, hardware and software design, testing, and privacy matters are all variables that must be factored in. Such an analysis should also contemplate past and present telltale signs that hint at a growing attack trend associated with peripherals. To address our third question, let's identify some threats.

Kill the floppy drive

The first account of a peripheral being considered a security threat can be tracked to the mid '80s, when virus proliferation was the prevalent security concern. In that early age of computer networks, virus infections spread via floppy disks. Consequently, floppy drives—the most popular peripheral to expand computer system abilities along with the dot matrix printer—became security conscious individuals' and organizations' immediate focus of attention.

In the absence of more efficient or cost-effective solutions, many organizations choose to physically disconnect popular peripherals from all PCs to mitigate virus threats. Less drastic solutions involved removing floppy drives from the operating system boot sequence to prevent automatic infections due to unattended system reboots or user distraction.

Owned by an iPod

Researcher Maximilian Dornseif at Aachen University of Technology's Laboratory for Dependable and Distributed Systems presented a more modern incarnation of a bad peripheral threat in November 2004 at the Pacific Security conference in Tokyo, Japan (www.pacsec.jp/psj04/psj04-dornseif-e.ppt).

In his presentation, "Owned by an iPod," Dornseif demonstrated several attack scenarios by using the popular IEEE 1394 compliant port found on most modern computer systems (http://standards.ieee.org/reading/ieee/std_public/description/busarch/1394-1995_desc.html) including today's hottest peripheral device, Apple's iPod (www.apple.com/ipod/). The IEEE 1394 specification—which Apple's FireWire or Sony's iLink ports adheres to—lets client devices directly access host memory, bypassing operating system controls that are imposed on user applications. A malicious client device can read and modify sensitive memory, causing privilege escalation, information leakage, and system compromise.

Dornseif's demonstration included privilege escalation attacks on vulnerable systems by directly overwriting kernel memory and interface spoofing attacks by directly manipulating the system's video memory. Interface spoofing attacks and their link to current attack trends were described by Attack Trends department coeditor Elias Levy in "Interface Illusions" (vol. 2, no. 6, 2004, pp. 66–69).

Dornseif pointed out that although the specification describes filtering capabilities that driver developers could use to prevent attacks from malicious devices, in practice, these mechanisms aren't currently implemented or easily configurable.

Kernel code lacking stringent controls (due to inferior security design and implementation) can't handle potentially malicious peripheral devices, which opens the door for these sorts of attacks. This premise is applicable to other types of hardware interfaces, such as those from the Personal Computer Memory Card International Association (PCMCIA, an international standards body and trade association) and CardBus.

USB security

Recent security concerns surrounding USB-compatible hardware focus on the risk of disclosing proprietary information through USB storage devices or attackers' ability to deliver attacks through them in the form of viruses and other malware in ways reminiscent to the floppy-drive scare discussed earlier (www.computerworld.com/printthis/2004/0,4814,94319,00.html).

A different and seemingly overlooked threat lies in the fact that USB devices that implement certain types of functionality, such as storage, might also implement hidden "features" such as keyboard and mouse functionality that can automatically launch attacks when connecting to a host computer. Given the tamper-prone nature of most peripheral devices, such an attack can be difficult to identify and properly mitigate without substantial involvement of security-trained users.

Wireless woes (again)

Wireless network and protocol security is a recurrent topic for *IEEE Security & Privacy*; the May/June 2004 issue was devoted to it, and several articles have appeared on other issues such as Nick Petroni and William Arbaugh's "The Dangers of Mitigating Security Design Flaws: A Wireless Case Study" (vol. 1, no. 1, 2003, pp. 28–36). But, as Bloomberg security researchers Shane Macaulay and Dino Dai Zovi showed in November 2004, wireless security woes aren't limited to networking protocols (www.pacsec.jp/psj04/psj04-macaulay_zovi-e.ppt).

Macaulay and Dai Zovi's research demonstrates how design and implementation decisions at the device driver and user application layer in today's most popular operating systems can be leveraged to launch attacks in ways that users hardly notice. In their attack scenario, a malicious device—a rogue access point—can force wireless clients to disconnect from legitimate networks and connect to hostile ones that automatically deliver attacks. Although their research applies to wireless networks based on IEEE 802.11 specifications, other wireless technologies such as Bluetooth and the upcoming wireless USB (www.nwfusion.com/news/2004/1022inteloutli.html) might suffer from similar problems.

Additionally, researchers at security consultancy Atstake have disclosed security threats posed by Bluetooth-enabled devices (www.atstake.com/research/reports/acrobat/atstake_war_nibbling.pdf).

These examples illustrate that attackers can compromise vulnerable systems by using computer peripherals to deliver malware using benign devices at attack vectors, either directly or by relying on unsuspecting or malicious users.

Computer peripheral security risks are hardly a new topic in the context of the floppy-drive concern of more than 20 years ago, yet many indicators suggest that we're headed back to square one in terms of peripheral device security.

The bad news is that the mid '80s approach of physically disconnecting bad peripherals from the system might not be applicable in the present situation. Although other effective risk mitigation strategies can be devised to achieve effective results from operating system vendors, device driver authors and hardware manufacturers must address the threat of bad peripherals in a consistent and coordinated manner that doesn't exclude the end user from the equation.

Until that happens, information security practitioners will be condemned to remain on the receiving end of an uneven battle between attackers and unhappy users. ▢

Reference

1. M.K. McKusick et al., *The Design and Implementation of the 4.4BSD Operating System*, Addison-Wesley, 1996, pp. 193–205.

Iván Arce *is chief technology officer and cofounder of Core Security Technologies, an information security company based in Boston. Previously, he worked as vice president of research and development for a computer telephony integration company and as information security consultant and software developer for various government agencies and financial and telecommunications companies. Contact him at ivan.arce@coresecurity.com.*