Proxying: The Next Step in Reducing IT Energy Use

Bruce Nordman, *Lawrence Berkeley National Laboratory*
Ken Christensen, *University of South Florida*

Proxying is a simple and effective means of allowing network hosts to sleep while maintaining network presence.

At the core of Green IT is reducing the energy use of the electronic devices that help us to acquire, store, process, and display information. The energy use of PCs, set-top boxes, and even most servers is mostly driven by their operating patterns and low utilization. This results in annual energy use being dominated by the power required simply to make the devices be present and available, not by the incremental energy for actual useful computations. Thus, the greatest savings can be attained by reducing the power needed to maintain functional presence when idle, not by making the active operation more efficient.

PCs in the US use about $7 billion of electricity per year (plus several billion dollars more for displays). Most of this energy use occurs when no one is present and the PC is idle. A major and increasing reason for PCs to be left on is to be continuously available for use on the network—that is, to maintain "network presence." A host that fails to do this will not be reachable or addressable on the network, will not be manageable, and will lose some application state. While today's PCs have a reliable sleep mode with quick wake-up, they lose network connectivity while asleep. The key is to combine the compelling energy-saving characteristic of sleep while not sacrificing network presence.

NETWORK PRESENCE

Presence for communications is not new. Telephones maintain presence on the network between calls, to awaken any time a call comes in, and televisions listen for a remote-control signal to wake up. In these cases, a device that did not respond appropriately would be understood as failing a basic function.

PCs are increasingly used in ways that call for continuous presence on the network. This is part of the industry-wide transition from a two- to a three-state power model. At one time, electronic products were simply on or off. Beginning several decades ago, some devices added a third basic power state: sleep. Maintaining state and network connectivity are becoming the defining characteristics of sleep. Responsiveness to network activity will in the future likely be the key differentiation between sleep and off. A typical sleeping PC consumes less than 10 percent of its power when on (even if idle).

Presence on the network can be understood as having three "layers" (these are not intended to be the Open System Interconnection layers, but rather abstractions):

- link-layer presence for maintaining local network connectivity—for example, Wi-Fi or Ethernet;
- network-layer presence for supporting end-to-end transport; and
- application-layer presence for functionality.

Maintaining an Ethernet link while asleep (or even off) has been common in PCs for years; maintaining a Wi-Fi link in sleep is not currently done. Basic network connectivity requires active participation in numerous core Internet protocols, such as the Address Resolution Protocol (ARP) to maintain reachability. Application requirements vary: Some include activity for a system while it is asleep, but all require awareness of when the host needs to wake up for that application.

PROXYING

Proxying is the use of a low-power entity to maintain presence on the network for a high-power device like a PC. The component that does this, the proxy, can be internal to a device (for example, in the network interface chip within a PC), in an immediately adjacent network switch or router, or even another PC on the subnet. The proxy enables a host

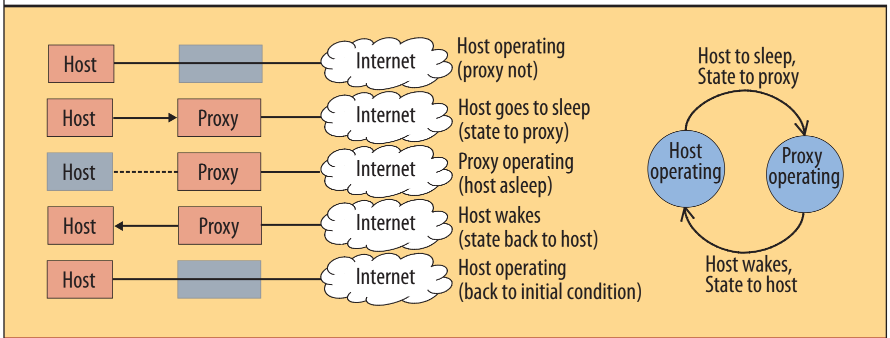

Figure 1. Proxy operation. A proxy enables a host to transition into and out of sleep transparently to the network. When the host is operating normally, the proxy function is not engaged.

to transition into and out of sleep transparently to the network. Use of a proxy requires no infrastructure changes such as changing existing protocols, or maintaining state in routers or switches.

As Figure 1 shows, a PC going to sleep signals the proxy that it needs to begin operation and then passes key information about its network state to the proxy. The host ceases to receive network traffic, and the proxy does the following on behalf of the host:

- maintain the network link,
- respond to packets,
- maintain presence state, and
- generate packets

Most incoming packets require no action and are thus simply ignored. The proxy can handle the majority of the remaining packets with routine responses. When the proxy finds network traffic that it cannot handle (that needs the host's attention), it wakes the host. As the host wakes, it might receive some information from the proxy—for example, the packet that triggered the wake.

Once the mechanics of proxying are in place, the actual operations that the proxy undertakes are not necessarily complicated, but failure to carry them out is catastrophic for network presence.

Systems that are presently left on for many hours while not in use (often 24/7) can save large amounts of energy by instead being asleep most of the time and waking only when needed to execute required tasks. However, hosts that are today powered down when not in active use will still benefit; doing so will not save energy, but the devices will gain additional functionality through network presence they previously lacked when in sleep. Thus, the addition of proxying functionality can benefit all PCs.

STANDARDIZATION

Functions that act on a network usually require standardization to enable interoperability. Proxying involves coordination among operating systems, applications, and the proxy itself. The EPA Energy Star program recognized the value of proxying in its Version 5.0 Specification for Computers but stipulated that it should be defined by a standard. Then, major PC companies, including AMD, Apple, Intel, Microsoft, and Sony came together under the auspices of the Ecma International standards organization to create such a standard, which they completed in November 2009.

The Ecma proxying committee, TC38-TG4 (www.ecma-international.org/memento/TC38-TG4.htm), considered various typical consumer and enterprise usages, covering all three of the aspects of network presence: link (Ethernet and Wi-Fi), network (IPv4 and IPv6), and several applications.

For network connectivity for IPv4 networks, the standard includes key protocols such as ARP, the Dynamic Host Configuration Protocol (DHCP), and the Internet Group Management Protocol (IGMP). Application-level proxying needs vary: Some require only basic link and network presence; some need only basic wake-up mechanisms—for example, waking on a Transmission Control Protocol (TCP) SYN packet; and for others, the proxy must undertake specific actions.

The Ecma standard addresses several application-oriented features, including the Simple Network Management Protocol (SNMP), remote wake-up, and remote access for IPv6 with Teredo. The standard also covers service discovery with multicast DNS (mDNS) and link-local multicast name resolution (LLMNR).

IMPLEMENTATION

Several research efforts have focused on proxying. Most notable is the Somniloquy project at the University of California, San Diego, and Microsoft Research; it is a proxy residing on USB-attached Gumstix hardware with both Ethernet and Wi-Fi connectivity. Somniloquy specifically explored the needs of applications for proxying.

Intel Research Berkeley conducted extensive trace analysis to uncover what a proxy needs to do, and it also implemented a simple proxy not requiring coordination with the host. Intel Labs conducted a technical session at Intel Developer Forum 2009 on proxying and also demonstrated Wi-Fi and remote wake-up (using the Session Initiation Protocol) capabilities.

Proxying is also making its way into various products. In mid-2009, Apple introduced proxying functionality for its PCs, with the proxy residing in Apple-brand access points or in a nonsleeping Mac on the same subnet. More recently, Apple has begun shipping systems with the proxy functional internal to the PC. The Apple implementation focuses on Bonjour (mDNS), which enables media sharing among other functions.

Microsoft's Windows 7 includes support for network interface chips that can respond to ARP and neighbor discovery (ND) queries.

FUTURE CHALLENGES

As noted above, the proxy can be either internal to the sleeping device or external to the sleeping host. In the long run, an internal proxy offers ease of coordination with the host operating system and ensures that the device can reliably proxy within complex configurations. In the near term, however, external proxying can utilize existing network equipment and proxy for existing PCs and so be useful and save energy much more quickly—a single piece of network equipment might contain proxies for many attached hosts. Thus, both approaches are necessary.

The next step for proxying is wide deployment in enterprises. For general proxying, deployment of external proxies in commercial businesses first has several advantages:

- it can be implemented with existing equipment, and a single IT person can enable the functionality for many PCs;
- a business that saves energy for dozens or hundreds of PCs will see economic savings that are much larger than a household would; and
- many business network environments and applications are more uniform than those of households in general are, reducing the number of complications or problems in implementation.

Proxying should also be useful for battery-powered devices to reduce the amount of communications, computation, and required on-time, therefore extending battery life.

Proxying raises the topic of what network and application protocols and behaviors are truly critical for a modern host to support. This same question has arisen in Internet Engineering Task Force discussions of what very simple hosts—such as lights, communicating thermostats, and so on—need to support to enable good operation and interoperability. This may provide an extra push for simplicity in the network realm.

For applications that are not compatible with the proxying standard, deployment will provide incentives to update the application to accommodate proxying. The biggest frontier for proxying is likely to be the handling of specific applications and application-oriented protocols.

Several promising directions for proxying to help reduce the energy use of IT equipment include:

- Proxying for security-related protocols such as Internet Protocol Security (IPSec). The challenge here is how to transfer security-related state (passwords and so on) to and from a proxy and how the proxy can be trusted.
- Proxying P2P protocols to allow hosts running them to power down when not actively downloading, uploading, or streaming content. Currently, P2P hosts must remain fully powered on all the time. P2P hosts already use significant energy and this use is expected to grow, making this an area of urgent interest.
- Adding more application-level functionality into a proxy—for example, the ability to store and share content.
- Enabling proxies to coordinate activities and thus be able to consolidate IT services to further reduce energy use.
- Extending proxying to non-IT equipment such as residential appliances that are rapidly being connected to the Internet as network hosts.

Network connectivity proxying offers a chance to both save large amounts of energy and add functionality, at very low cost. As implementation requires coordination of numerous entities and organizations, basing it on an industry standard greatly increases the chances that it will succeed and be widely deployed sooner. Proxying offers the opportunity for hosts and applications to expose their power state when desired, or to hide low-power states from the network when that is more appropriate. This brings power states and power management to higher-layer protocols, which historically have lacked them. ■

Bruce Nordman *is a researcher in the Energy Analysis Department at Lawrence Berkeley National Laboratory. Contact him at bnordman@lbl.gov.*

Ken Christensen *is a professor and director of the undergraduate program in the Department of Computer Science and Engineering at the University of South Florida. Contact him at christen@cse.usf.edu.*

Editor: Kirk W. Cameron, Dept. of Computer Science, Virginia Tech; cameron@cs.vt.edu